

ІНФОРМАТИКА

Підручник для 8 класу
загальноосвітніх навчальних закладів

Рекомендовано Міністерством освіти і науки України

Львів
Видавництво “Світ”
2016

УДК 004(075.3)
ББК 32.973-018я721.6
I-74

Авторський колектив:

А.М. Гуржій, Л.А. Карташова, В.В. Лапінський, В.Д. Руденко

*Рекомендовано Міністерством освіти і науки України
(наказ МОН України від 10.05.2016 р. № 491)*

Експерти, які здійснили експертизу підручника під час проведення конкурсного відбору проектів підручників для учнів 8 класу загальноосвітніх навчальних закладів і дійшли висновку про доцільність надання підручнику грифа “Рекомендовано Міністерством освіти і науки України”:
Рубаненко-Крюкова М.Ю., методист Науково-методичного педагогічного центру Департаменту освіти Харківської міської ради Харківської області;
Сафонова С.О., доцент кафедри комп’ютерної інженерії Східноукраїнського національного університету імені В. Даля, кандидат технічних наук.

Інформатика : підруч. для 8 кл. загальноосвіт.
I-74 навч. закл. / А.М. Гуржій, Л.А. Карташова, В.В. Лапінський, В.Д. Руденко. – Львів : Світ, 2016. – 256 с. : іл., табл.

ISBN 978-966-603-996-8

Підручник призначений для навчання інформатики у 8 класі загальноосвітніх навчальних закладів. Зміст підручника повністю відповідає навчальній програмі “Інформатика. 5–9 класи”, рекомендованій Міністерством освіти і науки України (наказ від 06.06.2012 р. № 664).

УДК 004(075.3)
ББК 32.973-018я721.6

ISBN 978-966-603-996-8

© Гуржій А.М., Карташова Л.А.,
Лапінський В.В., Руденко В.Д., 2016
© Видавництво «Світ», 2016

Дорогі учні!

Ви навчаєтесь інформатики вже четвертий рік, у повсякденному житті щораз більше й більше використовуєте засоби інформаційних технологій (мобільний телефон, планшет, електронну книгу та інші сучасні гаджети). Ви їздите на транспортних засобах, більшість із яких так чи інакше має в своєму складі комп'ютери – у автомобілях, автобусах, залізничних локомотивах комп'ютери керують роботою двигунів, визначають місце на маршруті потягів метро, маршрутних автобусів. Телефонуючи, навіть зі стаціонарного телефону на інший стаціонарний телефон, ви користуєтесь не менше ніж двома комп'ютерами – один з них міститься в телефоні (якщо це не телефон з набірним диском, яким дуже багато з вас, напевне, вже й не знають як користуватися), а інший – на телефонній станції, встановлює з'єднання з абонентом, перемикає режими роботи, вмикає голосові повідомлення.

Ми перестаємо помічати комп'ютери, приймаючи як належне їх непомітну присутність. Але вони дедалі частіше починають керувати пристроями, які є необхідною частиною середовища, в якому живе і навчається, працює сучасна людина. Звичайно, можна не знати, як відбувається керування пранням у сучасній пральній машині – треба лише натиснути кнопку вибору відповідної програми. Але людина стала людиною розумною тому, що не просто користувалася знаряддями, а намагалася їх удосконалити, придумати нові.

Отже, для того щоб ефективно використовувати блага цивілізації, придумані й створені для нас, потрібно, хоча б у загальних рисах знати, як вони працюють. Ці знання, зокрема, допоможуть визначити, чого можна вимагати від певного гаджета, а чого – ні, визначити, чи правильно він працює. А в майбутньому – усвідомлено обрати вид діяльності, пов'язаний зі створенням нових засобів.

У галузі інформаційних технологій, як і в усіх галузях науки і техніки, мистецтва, є знання, які не застарівають. У інформатиці – це знання про принципи програмного управління, на основі яких побудована робота всіх комп'ютерів. Ви докладніше ознайомитеся з конструкцією комп'ютера, почнете освоювати створення програм уже в середовищі, наближеному до того, яким користуються програмісти, і зрозумієте, що це легко!

Ви продовжуватимете на практиці засвоювати правила безпеки життєдіяльності під час виконання робіт з використанням комп'ютера та інших засобів інформаційних технологій – це теж необхідно.

Успіхів вам у навчанні й житті!

Автори









ЯК ПРАЦЮВАТИ З ЦІЄЮ КНИГОЮ

Для користування цим підручником обов'язковою є наявність у вашому розпорядженні персонального комп'ютера. На ньому має бути встановлено комплект програмних засобів, які описано в підручнику. Бажано підключити до цього комп'ютера описані в підручнику пристрої.




Навчальний матеріал поділено на дев'ять розділів. Перших сім складаються з параграфів, у яких коротко викладено теоретичний матеріал, описи виконання дій з програмними продуктами, які ви вивчаєте, наведено запитання для перевірки засвоєння матеріалу і вправи для виконання. Останні два розділи містять приклади задач і завдань навчальних проектів, розв'язування і виконання яких вимагатиме знань не тільки інформатики, але й інших предметів.

Нові терміни надруковано **жирним шрифтом**.

У підручнику використано такі підзаголовки і позначення:

	“Це ви вже знаєте” – короткий виклад знань, необхідних для засвоєння матеріалу розділу або підрозділу
	“Що вивчатимемо” – короткий виклад змісту розділу (підрозділу)
	“Важливе положення. Бажано запам'ятати”
	“Зверніть особливу увагу”
	“Для допитливих” – додаткові відомості
	“Словничок” – трактування термінів, які використовуються в розділі
	Рекомендується виконати (обговорювати) в колективі
	Рекомендується виконати вдома

Рівні складності завдань і запитань позначено таким чином:

-  перший
-  другий
-  третій

РОЗДІЛ 1. КОДУВАННЯ ДАНИХ



Будь-який реальний або уявний об'єкт і дії (процеси), що виконуються ним або над ним, можна описати, створивши повідомлення. Повідомлення може складатися з тексту і числових даних.

Для того щоб отримати із повідомлення інформацію, його слід створювати за певними правилами.



Отримання й опрацювання даних як інформаційний процес. Кодування та декодування повідомлень. Двійкове кодування. Одиниці вимірювання довжини двійкового коду. Кодування символів.

1.1. Отримання, кодування і декодування даних

Перш ніж описати якийсь об'єкт або процес, передати відомості про нього таким чином, щоб створене повідомлення було зрозумілим для того, хто його отримуватиме, слід сформулювати певні правила. Ці правила мають описувати процес **отримання відомостей** про об'єкт, їх запису та відтворення. Для того щоб подати значення властивості об'єкта у вигляді числа (даних), потрібно її **виміряти**.



Вимірюваність властивостей об'єктів визначається через досвід людини і суспільства, через поняття “більше” й “менше”.

У найпростішому випадку, для прямої, властивість вимірюваності полягає в тому, що обмежені двома точками частини різних прямих (відрізки) можна порівнювати між собою, визначивши для них властивість “довжина”. Для цього слід уважати за можливе порівняння відрізків, які належать різним прямим. Це порівняння виконується з використанням різних засобів вимірювання – мірної стрічки, лінійки, штангенциркуля тощо.

Для інших властивостей об'єктів також створюються засоби вимірювання, що відтворюють значення властивостей об'єктів – величину кута (транспортир – кут між його основою і напрямком на певну поділку транспортира), значення маси (набір гирь – маса гирі або їх набору на пальці терезів), значення часу (годинник – інтервал часу між двома положеннями стрілок на циферблаті) тощо.

Після цього маємо зберегти отримані дані, але таким чином, щоб вони були доступними як для нас, так і для будь-кого іншого. Отже, отримані значення слід записати словами, подати у вигляді чисел, креслень тощо, тобто **створити повідомлення**.

Повідомлення має бути передане або збережене як деякий фізичний об'єкт зі зміненими властивостями. Наприклад, це може бути аркуш паперу з написаними на ньому цифрами і літерами, ієрогліфами, оптичний диск із дзеркальними і недзеркальними мікроскопічними ділянками. Повідомлення має бути створене за певними правилами.

✓ *Створення повідомлення за певними, наперед визначеними правилами, називається **кодуванням**.*

Властивості повідомлення як об'єкта, що містить інформацію, теж потрібно **вимірювати**. Для визначення способу і засобів вимірювання властивостей повідомлення необхідно спочатку проаналізувати, як воно створюється і прочитується.

Нехай у повідомленні описуємо колір деякого об'єкта. Використовуватимемо лише три фарби: червону (R – red), зелену (G – green) і синю (B – blue), які змішуватимемо для отримання потрібного кольору. Кожну фарбу братимемо лише в певній кількості або не братимемо взагалі. Тоді колір об'єкта можна описати значеннями трьох змінних – R, G, B, кожна з яких може набувати значення 0 – фарби цього кольору немає, 1 – взяти 1 одиницю фарби. Тоді повідомлення, в яких буде **закодовано** колір, можуть мати такий вигляд: 111 – білий; 100 – червоний; 010 – зелений; 001 – синій. Можливими будуть і повідомлення 110; 011; 101; 000. Разом – вісім різних повідомлень і відповідно вісім різних кольорів, отриманих трьома фарбами.

Зауважмо, що для запису цих повідомлень використано лише **дві цифри** – 0 і 1.

✓ *Запис числа, в якому використовують лише дві цифри, називають **двійковим**.*

Застосовуючи для повідомлення **двійковий запис** числа довжиною три цифри (розряди), можна записати вісім значень кольорів – $2 \cdot 2 \cdot 2 = 2^3 = 8$.

Нехай для позначення кількості кожної фарби можемо використувати не один, а два розряди. За такого кодування код 00 означає відсутність фарби, коди 01, 10 і 11 – 1/3, 2/3 і 3/3 одиниці кількості відповідної фарби – чотири значення для кожної фарби: $4 = 2 \cdot 2 = 2^2$. Двійковий запис кольору, отриманого змішуванням трьох фарб, матиме вже шість розрядів ($6 = 2 \cdot 3$), а кількість кольорів, які можна буде описати, становитиме $2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^6 = 64$.

Для того щоб ще точніше поінформувати про колір об'єкта, можна використати ще більше розрядів, тобто повідомлення більшої довжини. Приблизно так само можна подати й інші повідомлення – більшої точності подання властивості об'єкта можна досягти, збільшуючи кількість символів (розрядів числового подання) у повідомленні.

У обчислювальній техніці повідомлення передають і зберігають у пристроях, які можна уявити як набори вимикачів, кожен із яких може бути замкненим або розімкненим. Один вимикач відповідає одному розряду двійкового числа. Вимикачі реалізуються як фізичні об'єкти зі зміненими властивостями: чорний або білий; рухається або перебуває в стані спокою; намагнічений або ні; кристалічний або ні; має електричний заряд певного знака або ні; проводить електричний струм або ні тощо. На лазерних дисках нулям і одиницям відповідають дзеркальні та недзеркальні піти (дуже маленькі ділянки диска), як показано на рис. 1.1.



Рис. 1.1. Піти на компакт-диску – об'єкти з двома можливими станами

- ✓ Об'єкт, у якому зберігається один розряд двійкового числа, називається **бістабільним**; цей об'єкт (пристрій) може перебувати в одному з двох можливих станів – “так” або “ні”, 0 або 1.
- ✓ Форма запису повідомлення у вигляді послідовності нулів і одиниць називається **двійковим кодом** (повідомлення, даних, команд).

Зрозуміло, що найменшою можливою довжиною двійкового коду повідомлення є вміст одного бістабільного пристрою, або **1 біт** (англ. *bit* – *binary digit* – цифра, однорозрядне число). Внутрішній запам'ятовуючий пристрій (ВЗП) більшості сучасних комп'ютерів складається з комірок пам'яті, які мають по 32 або 64 бістабільних пристрої, тобто можуть зберігати 32-розрядне або 64-розрядне двійкове число.

Одиниці розміру файлів, з якими вам уже доводилось працювати, походять (є похідними одиницями) від **біта**.

Вісім бітів становлять 1 байт (1 Б).

1024 байти становлять 1 кілобайт (1 КБ).

1024 кілобайти становлять 1 Мегабайт (1 МБ).


1024 Мегабайти становлять 1 Гігабайт (1 ГБ).

1024 Гігабайти становлять 1 Терабайт (1 ТБ).

Ім'я	Дата змінення	Тип	Розмір
Білл Гейтс виступив з допо...	21.01.2016 15:25	Папка фой...	
Kosmos	02.11.2012 7:37	Архів ZIP ...	10 137 КБ
Nasouy	02.11.2012 7:38	Архів ZIP ...	295 КБ
електромагніти	25.03.2013 17:19	Архів ZIP ...	485 КБ
Оптика гра 7 клас	02.11.2012 7:41	Архів ZIP ...	5 126 КБ
кросворди	09.04.2012 10:17	Документ ...	193 КБ
Урок вода	08.04.2012 19:00	Документ ...	18 КБ
молекули	02.11.2012 7:39	Документ ...	53 КБ
прості механізми	12.03.2014 12:37	Документ ...	85 КБ
струм у середовищах	02.11.2012 7:45	Документ ...	75 КБ
урок гра оптика	03.05.2012 18:48	Документ ...	271 КБ
густина	02.11.2012 13:23	Презента...	1 257 КБ
Фестиваль Прир-Матем на...	25.02.2013 15:13	Презента...	3 984 КБ
двійковий запис числа	21.01.2016 15:34	Текстовий...	3 КБ
Білл Гейтс виступив з допо...	29.03.2008 14:01	Текстовий...	41 КБ

Білл Гейтс виступив з доповід...

Текстовий документ OpenOffice.org 1.1



Дата змінення: 29.03.2008 14:01
 Розмір: 402 КБ
 Дата створенн...: 21.01.2016 15:25
 Доступність: Доступно в автономн...

Рис. 1.2. Уміст папки з електронними документами

❗ Для одиниць байт, кілобайт, мегабайт і т. д. прийнято використовувати кратність не 1000, як у Міжнародній системі одиниць фізичних величин (SI), а 1024, оскільки це основа двійкової системи числення у степені 10 (2^{10}).

✓ *Повідомлення може бути вимірне довжиною двійкового коду, яким воно записане (рис. 1.2).*

Читаючи повідомлення (лист, цей текст, телеграму, опис алгоритму тощо), ми здійснюємо процес, який називається декодуванням.

✓ *Декодування – відновлення інформації, закодованої в повідомленні.*

Для людини, яка не знає певної мови, лист, стаття в газеті тощо, подані незнайомою мовою, може не містити жодної інформації, крім того, що це текст. Але це жодною мірою не означає, що у повідомленні відсутня інформація.

Зручно вважати, що отримувач повідомлення завжди знає, як його сприйняти, опрацювати, отримати всю інформацію, яку воно містить. У такому випадку стає можливим порівняння кількості інформації в різних повідомленнях.







✓ *Для порівняння кількості інформації у повідомленнях використовують довжину їх подання у двійковому записі (двійковому коді), яка вимірюється у бітах, байтах і похідних від них одиницях.*

Таке трактування інформаційної місткості повідомлення використовується для визначення розміру місця, необхідного для його розміщення на фізичному носії – магнітному або оптичному диску, флешці.

Форми та способи подання інформації у повідомленнях, як і подання самих повідомлень, можуть бути різними: рисунки, ієрогліфи, піктограми, текст і дані, сформовані в документи на різних матеріальних носіях. Окремим, нині дуже важливим, видом документа є **електронний документ**.

Зважаючи на важливість для суспільства таких явищ, як **електронний документ і електронний документообіг**, їх означення та властивості, форми використання, основні інформаційні процеси, що відбуваються при їх створенні й застосуванні, означені нормативно – на рівні закону (закон України “Про інформацію” // Відомості Верховної Ради України. 1992. № 48).

Перевіряємо себе

1. Що необхідно виконати для того, щоб отримати числове значення властивості об'єкта? ✨
2. Як називається інформаційний процес, результатом якого є повідомлення зі значеннями температури в кількох різних приміщеннях? ✨
3. У чому полягає сутність кодування даних у комп'ютері? ✨
4. Назвіть основні особливості кодування даних у комп'ютерних системах. ★
5. Що називається декодуванням даних у комп'ютерних системах? ★
6.  Яким чином в Україні регламентується здійснення суспільно значущих інформаційних процесів?
7.  Чому двійкова система є основною в обчислювальній техніці? ★
8. Як нумеруються розряди числа? ✨
9. Що називається бістабільним пристроєм? Наведіть приклади. ✨
10. Чому в 1 кілобайті 1024 байти, а не 1000? ✨
11. У яких одиницях вимірюється ємність накопичувачів на магнітних дисках (вінчестерів)? ✨
12. Скільки бістабільних пристроїв використовується для зберігання файла “двійковий запис числа.txt” (рис. 1.2)? ✨
13.   Ви вже знаєте, що у комп'ютері використовуються різні типи даних. На вашу думку, для зберігання якого числа потрібно більше бістабільних пристроїв – цілого чи такого, що має дробову частину? Поясніть свою думку. ✨
14.   Чи завжди доцільно для порівняння кількості інформації у повідомленнях використовувати довжину їх подання у двійковому записі? ★

Виконуємо

1. Визначте ємності пристроїв зовнішньої пам'яті комп'ютера (накопичувачів на магнітних дисках, вінчестерів), за яким ви працюєте. Яка частина їх уже заповнена? ✦
2. Визначте на рис. 1.2 найбільші та найменші файли серед файлів одного типу. ✦
3. Запишіть довжини всіх файлів кожного типу в мегабайтах. Знайдіть загальну довжину файлів кожного типу. ✦
4. У скільки разів файл “двійковий запис числа.txt” менший за файл “прості механізми.docx” (рис. 1.2)? ✦
5. Обчисліть, скільки потрібно двобайтових комірок пам'яті, щоб зберегти двійкове число 1010110101.
6. Обчисліть, скільки різних станів об'єкта можна закодувати, використовуючи три канали подання сигналів 0 або 1. ▲
7. Латинський алфавіт має 26 літер. Обчисліть, скільки потрібно мати розрядів у комірці пам'яті, щоб закодувати ці літери (великі й малі), шість розділових знаків і цифри. ✦

1.2. Текстові повідомлення та їх кодування

Тексти вводяться в комп'ютер зазвичай за допомогою клавіатури. На будь-якій сучасній клавіатурі комп'ютера розміщені символи англійського і національного алфавітів, синтаксичні та деякі спеціальні знаки, десяткові цифри, функціональні клавіші й клавіші керування (Ctrl, Enter та інші). Натиснення на клавішу або комбінацію клавіш передається в комп'ютер для опрацювання.

Створений текстовий електронний документ зберігається спочатку в оперативному запам'ятовуючому пристрої (ОЗП) комп'ютера, а потім – на зовнішньому запам'ятовуючому пристрої (ЗЗП) або передається іншому користувачеві. Текст зберігається і передається у вигляді двійкового коду. Для того щоб правильно виконати декодування, тобто передати користувачеві електронний документ, усі літери, внесені в документ при його створенні, мають бути на сторінці там, де їх розмістили, і бути такими, які ввів користувач, що створював документ. Відповідність між літерою та її числовим кодом має бути однаковою для всіх комп'ютерів. Таке узгодження здійснюється прийняттям так званої таблиці кодування, тобто таблиці, в якій кожному символу (літері, цифрі, розділовому знаку, пробілу) і несимвольним командам (кінець рядка, кінець абзацу та іншим) поставлено у відповідність певне число.

Укладання відповідних стандартів виконує Міжнародна організація зі стандартизації – International Organization for Standardization (ISO).

Нині найбільш поширена таблиця кодування ASCII (American Standard Code for Information Interchange – американський стандарт коду для обміну інформацією). У основному варіанті коду ASCII для будь-якого символу клавіатури використовується один байт.

Перша, або нижня, половина кодової таблиці, тобто коди, подані молодшими сімома бітами (коди 0–127), в усьому світі застосовуються для кодування стандартного набору символів (символів латинського алфавіту, цифр, знаків арифметичних і логічних операцій та деяких інших). Перші 32 комбінації (коди 0–31) призначені для команд керування комп’ютером і зовнішніми пристроями (наприклад, принтером). У табл. 1.1 подано нижню половину кодової таблиці ASCII за стандартом ANSI X3.4-1977, яку змінювати не можна.

✓ Цю таблицю часто називають основною, або базовою, таблицею ASCII.

Таблиця 1.1

Основна, або базова, таблиця ASCII

Номери 7→ розрядів 4↓				0	0	0	0	1	1	1	1
			6→	0	0	1	1	0	0	1	1
		5→	0	1	0	1	0	1	0	1	0
	3↓	2↓	1↓								
0	0	0	0	NUT	DLE	Пробіл	0	@	P	.	P
0	0	0	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	STX	DC2	“	2	B	R	b	r
0	0	1	1	ETX	DC3	#	3	C	S	c	s
0	1	0	0	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	ACK	SYN	&	6	F	V	f	v
0	1	1	1	BEL	ETB	'	7	G	W	g	w
1	0	0	0	BS	CAN	(8	H	X	h	x
1	0	0	1	HT	EM)	9	I	Y	i	y
1	0	1	0	LF	SUB	*	:	J	Z	j	z
1	0	1	1	VT	ESC	+	;	K	[k	{
1	1	0	0	FF	FS	,	<	L	\	l	
1	1	0	1	CR	GS	-	=	M]	m	}
1	1	1	0	SO	RS	.	>	N	^	n	-
1	1	1	1	SI	US	/	?	O	C	o	DEL

Другу (верхню) половину таблиці (коди 128–255) використовують для кодування символів національних алфавітів, а також символів псевдографіки.

Для того щоб не використовувати довгі двійкові коди, для нумерування символів прийнято використовувати шістнадцяткову систему числення, у якій до десяти арабських цифр 0, 2...9 додано ще шість цифр A – 10, B – 11, C – 12, D – 13, E – 14, F – 15. Із табл. 1.1 бачимо, наприклад, що велика літера F має код 1000110 (у шістнадцятковій системі 46_{16}), символ 5 – код 0110101 (35_{16}), символ h – 1101000 (68_{16}).

Великі й малі літери відрізняються лише значенням 6-го розряду в двійковому поданні (нагадуємо – розряди цілої частини нумеруються справа наліво). Наприклад, символ A має код 1000001 (41_{16}), а символ a – 1100001 (61_{16}).

Перші 32 кодові комбінації таблиці, тобто комбінації від 0000000 (00_{16}) до 0011111 ($1F_{16}$), відведені під символи керування. Вони не виводяться на екран (не друкуються), а використовуються для спеціальних цілей, зокрема, для передавання команд периферійним пристроям, наприклад, для управління принтером.

Базова таблиця ASCII була застосована в комп'ютерах IBM PC для внутрішнього подання символів. Але 7-розрядний код не забезпечував повноцінного кодування національних абеток. Тому стандартом ISO 646 уведено нову, 8-розрядну, версію коду ASCII. Восьмий розряд надав ще 128 кодових комбінацій, які могли використовуватися в різних цілях, у тому числі для подання національної абетки.

Фірма Microsoft запропонувала власний варіант кодування кирилических символів алфавіту, так звану кодову таблицю Windows 1251. Ця кодова таблиця нині більш популярна, ніж кодування, рекомендоване ISO. Тому в деяких випадках, при відтворенні текстів із сайтів, виникає необхідність переналагодження браузера.

Оскільки персональні комп'ютери поширені по всьому світу, то, безумовно, бажано мати єдину систему кодування символів мов народів світу. Для розв'язання цієї проблеми група комп'ютерних компаній у 1991 році розробила міжнародний стандарт 16-розрядного кодування ISO 10646 під назвою Unicode (Юнікод), який забезпечує вже 65536 кодових комбінацій.

Складниками стандарту є UCS (Universal Character Set – універсальна таблиця символів) і UTF (Unicode Transformation Format – формат перетворення Юнікоду). UTF широко застосовується для передавання символів через Інтернет (чат, електронна пошта тощо).

Коди для символів у системі Юнікод поділено на області. Зокрема, коди від 0000_{16} до $007F_{16}$ – символи набору ASCII, а коди від 0400_{16} до $052F_{16}$ відведені для символів кирилиці. Стандарт постійно вдосконалюється, але це відбувається без зміни кодувань, які вже існують.

Для визначення шістнадцяткового коду символу в системі Unicode, а також в інших системах кодування, наприклад, у “ASCII, кирилиця”

необхідно у MS Word на вкладці **Вставлення** натиснути кнопку **Символ**, потім – **Інші символи**. Відкриється вікно **Символ** (рис. 1.3). У нижній частині цього вікна в шістнадцятковій системі числення виводиться код вибраного символу.

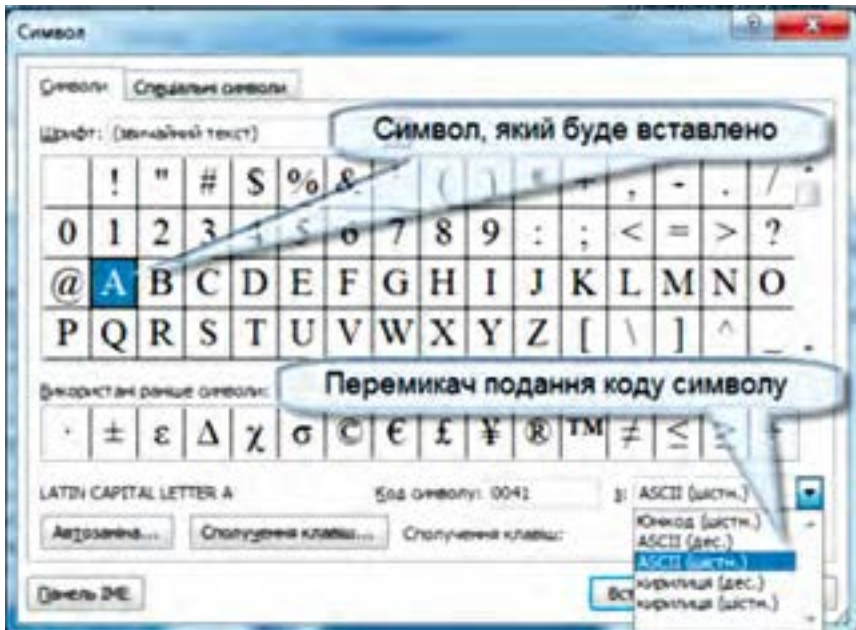





Рис. 1.3. Вікно вставлення символів редактора MS Word



Перші кодові комбінації кодової таблиці UNICODE (від 0000_{16} до $00FF_{16}$) відведені під код ASCII (базова таблиця і таблиця з англійськими літерами зі штрихами, так званий набір Latin-1). Ця таблиця поділена на блоки по 16 кодів кожний. Декілька блоків утворюють зону. Зона може мати різну кількість блоків і закріплюється за певною мовою.

Перевіряємо себе

1. Що називають кодовою таблицею? ▲
2. Чому було прийняте кодування символів 8-розрядними двійковими числами? ▲
3. Де розташовані літери кириличних абеток у кодовій таблиці ASCII? ★
4. Скільки символів можна було закодувати, використовуючи 7-розрядне кодування? ★
5.   Чому інколи замість літер кирилиці на деяких сайтах мережі Інтернет видно незрозумілі символи? ★
6. У якій кодовій таблиці можна закодувати більшу кількість символів – ASCII чи UCS? ▲

7. Чому при впорядкуванні за абеткою прізвищ, поданих українською мовою, деякі програмні засоби виводять спочатку прізвища, що починаються не з А, а з інших літер? Знайдіть з яких. ★

8.  У списку є рядки, що починаються з символів кирилиці (великих і малих літер), латиниці (великих і малих літер), цифр. У якій послідовності програмний засіб, де використано кодову таблицю ASCII, виводитиме рядки? ✦

9.   У комп'ютерах, що випускалися у 80-х роках минулого століття в СРСР, використовувався стандарт кодування КОІ-7. Які проблеми з відтворенням кириличного тексту могли виникати? ✦

Виконуємо


1. Заповніть таблицю шістнадцятковими кодами наведених у верхньому рядку символів; для цього завантажте MS Word і відкрийте вікно **Символи** (рис. 1.3). ▲


Символи	1	2	А	В	+	?	>	:
Юнікод								
ASCII								


Проаналізуйте отриману таблицю. Чим відрізняються коди наведених символів? ✦



2. У редакторі Word відкрийте вікно **Символи**. У віконці **Шрифт** встановіть **звичайний шрифт** і виберіть таблицю **кирилиця (шістн.)**. Використовуючи повзунок смуги прокручування, знайдіть символи кирилиці. Запишіть шістнадцяткові коди великих і малих літер В, Г, Д, Е, Ж. ✦


3. У книжці 450 сторінок. На кожній сторінці 30 рядків по 68 символів. Визначте обсяг пам'яті, потрібний для її збереження. ▲

4.  Знайдіть в Інтернеті таблицю кодування KOI8-U, яка є стандартом для української Інтернет-спільноти, і таблицю KOI8-R. Визначте їх спільні ознаки та відмінності. ✦

5.  Прийнято кодове повідомлення в системі кодування кирилиця (шістнадцяткове подання): 00С7 00Е0 00Е2 00F2 00F0 00F0 00Е0 00F8 00F2 00ЕЕ 00F0 00Е0. Розшифруйте отриманий код. ★

6.  Книжка зберігається в комп'ютері і займає 420 кБ. Кожна сторінка книжки містить 28 рядків по 75 символів у кодах ASCII. Визначте кількість сторінок у книжці. ★

7.   У цифровому пристрої з обсягом пам'яті 600 байт зберігається текстове повідомлення довжиною 1200 символів. Визначте можливу кількість різних символів у цьому повідомленні. ✦

8.  Прочитайте у підрозділі “Для допитливих” описи кодування за методом Гауса і кодування за методом Бодо. Що спільного в цих методах? Чим вони відрізняються? ★

Підказка: зверніть увагу на кількість можливих комбінацій станів у повідомленні, яке відповідає одному символу.

Практична робота № 1

Тема:	Розв’язування задач на визначення довжини двійкового коду текстових даних
Мета:	Набути практичних навичок визначення розміру текстових файлів

1. У цифровому пристрої необхідно зберігати повідомлення “ЛАБОРАТОРНА РОБОТА”.

Обчислити кількість бітів, необхідних для кодування символів цього повідомлення (розглянути можливі варіанти кодувань), і обсяг пам’яті, потрібний для його збереження.

2. Символами кодової таблиці ASCII передано повідомлення: МАТЧ ШАХТАР – ДИНАМО ЗАВЕРШИВСЯ З РАХУНКОМ 2:2. Обчислити обсяг пам’яті, необхідний для його збереження.

3. Завантажте текстовий редактор MS Word. Створіть новий файл і введіть речення: “Зараз будемо перевіряти, як кодуються літери української абетки і де вони містяться, і в якій кодовій таблиці вони є, чи ґрунтовно я знаю правила кодування?”. Використовуючи засіб “Вікно вставлення символів”, визначте коди літер, які відсутні в російській абетці, але є в українській. Зробіть висновок щодо кодової таблиці, яка використовується.

Збережіть файл у форматах *.doc, *.rtf, *.txt, порівняйте розміри файлів. Поясніть, чому розміри файлів відрізняються.



ДЛЯ ДОПИТЛИВИХ

Розвиток систем кодування символів фактично **розпочався зі спроб використання електрики** для передавання повідомлень.

У 1833 році математик Фредерік Гаус запропонував метод кодування 25 символів (букви І та J були об’єднані) за допомогою матриці 5*5. Ідея полягала в такому: по одному дроту передавався електричний сигнал п’яти рівнів, від якого стрілка відхилялася на певне значення вправо, а потім – електричний сигнал також п’яти рівнів, від якого стрілка відхилялася вліво. Перше значення визначало номер рядка в матриці, а друге – номер стовпця. Наприклад, якщо перше відхилення стрілки було зафіксоване на значенні 3, а друге – на значенні 2, це означало, що була передана літера М (рис. 1.4).

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I, J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Рис. 1.4. Кодування символів за методом Гауса

У XIX столітті американський художник і винахідник Семюел Морзе запропонував систему кодування символів за допомогою коротких і довгих сигналів (крапка, тире). Буква А кодується коротким і довгим сигналом (• –), буква S – трьома короткими сигналами (• • •), цифра 1 – коротким і чотирма довгими сигналами (• – – – –). Зазначмо, що код Морзе має суттєвий недолік. У ньому використовується різна довжина кодових комбінацій. Фактично код Морзе не є двійковим, оскільки для кодування використовується не тільки тривалість натиснення ключа (тире і точка), а й тривалість пауз між натисненнями. Нині код Морзе застосовують переважно радіоаматори.

Спосіб кодування, яким донині користуються в телеграфії, запропонував у 1870 році французький інженер Еміль Бодо, який обмежив кількість сигналів у кодовій комбінації п'ятьма. Сигнал мав два стани: “увімкнено” і “вимкнено”, тобто кодування було дійсно двійкове. Це давало можливість мати 32 кодові комбінації ($2^5=32$), що недостатньо для 26 літер, 10 десяткових цифр і знаків синтаксису. Тому Е. Бодо використав 26 комбінацій для літер і 6 комбінацій для керуючих символів. Зокрема, кодова комбінація 11111 використовувалася для перемикавання на реєстр літер (LTRS), а кодова комбінація 11011 – для перемикавання на реєстр цифр (FIGS). Зазначмо, що коди керуючих символів LTRS і FIGS, як і інші керуючі символи, завжди інтерпретуються однаково, незалежно від того, в якому реєстрі знаходиться приймальний пристрій. Фрагмент кодової таблиці Бодо подано в табл. 1.2.

Таблиця 1.2

Фрагмент таблиці коду Бодо

Сигнали коду					У реєстрі	У реєстрі
1	2	3	4	5	LTRS	FIRS
•	•				A	C
•			•	•	B	?
	•	•	•		C	:
•	•	•	•	•	LTRS	
•	•		•	•	FIGS	
		•			Проміжок	
		•		•	Повернення каретки	
	•	•		•	Перехід на новий рядок	

Код Бодо став основою для розроблення стартстопного телеграфного апарата, в якому кожна комбінація починалася сигналом СТАРТ і закінчувалася сигналом СТОП. Телеграфні апарати на основі коду Бодо використовувалися понад 50 років. Для того щоб розрізняти великі й малі букви, пізніше використовувалося й 6-розрядне кодування.

Сьогодні, в епоху розвитку комп'ютерної техніки, існують різні коди для передавання символів. Найпоширенішими є такі:

- міжнародний алфавіт № 2 ССІЕЕ – простий 5-розрядний код, який використовується для передавання телекських повідомлень;
- код EBCDIC – використовується головним чином у системах зв'язку з великими ЕОМ;
- код ASCII (American Standard Code Information Interchange);
- у країнах колишнього Радянського Союзу використовувалися 7-розрядний Код Обміну Інформацією (КОІ-7), 8-розрядний Код Обміну Інформацією (КОІ-8) і Двійковий Код для Обробки Інформації (ДКОІ). Вони використовуються й тепер, зокрема, в Інтернеті.



СЛОВНИЧОК

Двійкова система числення – система числення з основою два.

Двійковий код – подання повідомлення або даних у вигляді послідовності чисел у двійковій системі числення.

Декодування – перетворення даних із коду на форму, яку сприймає людина або може опрацювати певна програма.

Десяткова система числення – система числення з основою десять (прийнята для повсякденного застосування система запису чисел).

Код – подання повідомлення (даних) після кодування.

Кодування – процес створення на основі повідомлення або даних послідовності символів, який виконується за певним правилом (алгоритмом кодування).

Кодова таблиця – таблиця, у якій кожному можливому значенню (слова, величини, символу) в одному поданні поставлено у відповідність це ж значення в іншому поданні.

Кодування (у обчислювальній техніці) – перетворення даних (повідомлення) на подання, доступне для опрацювання комп'ютером, зазвичай – двійковий код.

Шістнадцяткова система числення – система числення з основою шістнадцять (цифри для чисел від 10 до 15 (десяткових) позначаються латинськими літерами від А до F).



РОЗДІЛ 2. АПАРАТНО-ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРА



Комп'ютер складається з апаратного і програмного забезпечення. Основними і обов'язковими частинами кожного комп'ютера є запам'ятовуючий пристрій (пам'ять), процесор і пристрій управління. Пам'ять, у якій дані й команди можуть зберігатися без споживання електроенергії, називається постійним запам'ятовуючим пристроєм. Пам'ять, вміст якої зберігається за умови наявності електричного живлення, називається оперативним запам'ятовуючим пристроєм.

Програма – це послідовність команд, яка описує процес опрацювання даних, частина з яких може міститись у самій програмі, а частина – надходити з пристроїв уведення, або зовнішніх запам'ятовуючих пристроїв.

Програмне забезпечення поділяється на системне, службове та прикладне.

2.1. Історія засобів опрацювання інформаційних об'єктів

Історія розвитку електронних обчислювальних машин (ЕОМ) як засобів опрацювання інформаційних об'єктів нерозривно пов'язана з розвитком механічних і електромеханічних обчислювальних машин. Першу механічну машину для виконання обчислень створив у 1642 році французький математик Блез Паскаль, якому на той час було 19 років. Вона виконувала операції додавання і віднімання десяткових чисел. На честь розробника цієї машини названа мова програмування Pascal.

У 1673 році німецький математик Готфрід Вільгельм Лейбніц сконструював другу механічну машину, що виконувала чотири арифметичні дії. Ця машина стала прообразом арифмометра, який з'явився у 1820 році.

Одним із перших програмно-керованих автоматів промислового застосування був створений у 1808 році Жозефом Марі Жаккаром ткацький верстат, програма для якого записувалась на перфокартах. Перфоровані карти зшивалися в стрічку. Ця стрічка з перфокарт могла займати два поверхи. Одній перфокарті відповідав один прокид човника з ниткою піткання.

У 1833 році професор Кембриджського університету Чарльз Беббідж сконструював першу механічну обчислювальну машину з програмним керуванням. Тогочасна технологічна база не дозволила повністю реалізувати проект. Програми для цієї машини розробляла Ада Августа Лавлейс, донька видатного англійського поета лорда Байрона. На честь леді Лавлейс названа мова програмування Ada.

Через 19 років після смерті Беббіджа один із принципів його аналітичної машини – використання перфокарт – отримав практичне

застосування у пристрої, який назвали “статистичний табулятор”. Цей пристрій побудував американський інженер Герман Холлеріт з метою прискорити опрацювання результатів перепису населення, що проводився у США в 1890 році. Спосіб збирання даних був досить простим: відомості про людину кодувались на одній перфокарті, яка мала 240 позицій, з яких пробитими виявлялись 18–20. Оскільки дані опрацьовували вже за допомогою електричних сигналів, можна було знаходити відповіді на запитання, які потребували порівняння кількох даних, наприклад: “Скільки чоловіків віком більше 55 років живе у певному регіоні?”

У 1936 році німецький інженер Конрад Цузе створив діючу обчислювальну машину на електромагнітних реле, яка мала назву Z1. Пізніше він розробив ще дві машини на електромагнітних реле (Z2 і Z3). Але всі вони були знищені в 1944 році під час Другої світової війни.

На початку 40-х років ХХ століття американський учений Джон Атанасов, болгарин за походженням, розробив проект обчислювальної машини з використанням двійкової системи числення. Американський дослідник Говард Айкен у 1944 році розробив діючу обчислювальну машину MARK-1, у якій були реалізовані ідеї Ч. Беббіджа, тільки замість зубчатих коліщат використовувалися електромагнітні реле. Програма вводилася з перфострічки. Команди над 23-розрядними десятковими числами виконувалися за шість секунд.

Перша електронна обчислювальна машина (ЕОМ) COLOSSUS у режимі надзвичайної таємності була розроблена у 1943 році в Англії під керівництвом Макса Ньюмана. У розробленні машини брав участь знаменитий англійський математик Алан Тюрінг. Із цим комп’ютером пов’язане дешифрування повідомлень, якими обмінювались морські та сухопутні гітлерівські війська під час Другої світової війни. Уважалося, що код повідомлень неможливо розшифрувати, але з використанням ЕОМ COLOSSUS це зроблено, про що гітлерівське командування не знало. Таємниця створення й існування ЕОМ COLOSSUS зберігалася ще багато років, тому ідеї її побудови суттєво не вплинули на історію розвитку ЕОМ.

Перша ЕОМ з назвою ENIAC (Electronic Numerical Integrator and Computer – електронний числовий інтегратор і обчислювач), яка стала відома спільноті, була побудована в США у 1945 році під керівництвом Джона Моучлі й Джона Екертта. Вона містила 18000 електронних ламп, 1500 реле, споживала 140 кВт електроенергії і мала масу тридцять тон. У роботі над проектом машини брав участь американський математик угорського походження Джон фон Нейман. У 1945 році він сформулював принципи побудови ЕОМ, які увійшли в історію як базові принципи роботи програмно-керованого автомата, а запропонована ним конструкція ЕОМ отримала назву “фоннейманівська архітектура”.

Фундаментальний внесок у розвиток ЕОМ зробив академік С. О. Лебедев. За його керівництва у 1950 році в Києві була побудована Мала Електронна Обчислювальна Машина (МЕОМ, МЭСМ). Ця ЕОМ – перша обчислювальна машина на електронних лампах, у якій програма зберігалась у оперативному запам'ятовуючому пристрої (ОЗП), побудована в континентальній Європі. ЕОМ на електронних вакуумних та іонних приладах отримали назву “**ЕОМ першого покоління**” і розвивалися до кінця 50-х років ХХ століття.

Поштовхом до виникнення другого покоління ЕОМ стало винайдення у 1956 році напівпровідникового трьохелектродного підсилюючого пристрою – **транзистора**. Транзистор став основним елементом ЕОМ наступних поколінь.

Перша експериментальна ЕОМ другого покоління (на транзисторах) була розроблена в 1957 році, а перша серійна ЕОМ PDP-1 – у 1961 році під керівництвом Кеннета Ольсена у США. Комп'ютер PDP-1 мав ОЗП ємністю 4096 слів по 18 біт і швидкодію 200 тисяч операцій за секунду. Він уже мав дисплей 512*512 пікселів. Перша в Європі серійна **ЕОМ другого покоління** була розроблена в Києві у 1961 році. Вона отримала назву “Дніпро”. Цю ЕОМ уперше стали практично використовувати в промисловості, зокрема, для керування виплавлянням сталі на металургійному комбінаті у Дніпродзержинську, розкroюванням листів сталі на суднобудівному заводі в Миколаєві. Комплекс із двох комп'ютерів “Дніпро” керував великим екраном, на якому відображався космос і, зокрема, стикування кораблів за програмою “Союз” – “Аполлон”.

Великим попитом користувалися машини серії МІР (машини інженерних розрахунків), розроблені в Інституті кібернетики НАН України, керівником якого був видатний український учений академік В. М. Глушков. Машина МІР-1 фактично була попередником майбутніх персональних комп'ютерів. Комп'ютери МІР згодом широко використовувалися для навчання, їх встановлювали навіть у студентських гуртожитках, зокрема, Київського державного університету імені Тараса Шевченка. Для цих комп'ютерів розроблено мову програмування Аналітик, яка була **першою в світі мовою описання задач обчислювального характеру для непрограмістів**.

У 1958 році розроблено **інтегральні мікросхеми**, які стали основою ЕОМ **третього покоління**. З цього моменту починається розроблення програмно-сумісних комп'ютерів. У середині 60-х років ХХ століття з'явилася перша сім'я комп'ютерів System/360 компанії ІВМ (моделі 30, 40, 50 та інші). Кожна “старша” модель була продуктивнішою за “молодшу”, але програми, розроблені для “молодших” машин, могли бути перенесені на “старші”.

На початку 80-х років ХХ століття виготовлено перші надвеликі (не за розміром, а за кількістю елементів!) мікросхеми, на яких розміщувалися окремі пристрої (процесор, модулі пам'яті та інші). З них починається ера персональних комп'ютерів (ПК), які належать до **четвертого покоління**.

У 1981 році фірма IBM розробила комп'ютер IBM PC і опублікувала на нього повну документацію, тобто зробила **архітектуру** комп'ютера відкритою. Спільно з компанією Microsoft для нього була розроблена операційна система (ОС) MS DOS, а пізніше — ОС Windows. Надалі персональні комп'ютери розвивалися здебільшого на основі процесорів компаній Intel і AMD.

В Україні використовувалася лінійка єдиної системи ЕС ЕОМ і розроблялася власна лінійка малих комп'ютерів СМ ЕОМ (Система Малих ЕОМ). У Києві, зокрема, розроблено одну з перших у світі серійних систем збирання даних на базі персональної ЕОМ типу ДВК-1 (Диалоговий Вычислительный Комплекс), яка широко використовувалася в наукових дослідженнях. Серійно випускалися персональні 16-розрядні ЕОМ ДВК 1–3, Електроніка.

Об'єднання "Електронмаш" (м. Київ) розробило і серійно випускало персональну електронно-обчислювальну машину (ПЕОМ) Пошук-1, яка була першою в СРСР 16-розрядною, сумісною з IBM PC системою, призначеною для персонального (домашнього) використання.









Нинішні ЕОМ, у тому числі персональні (ПЕОМ), здебільшого є втіленням ідей, які виникли під час створення ЕОМ четвертого покоління. Елементною базою їх є великі й надвеликі мікросхеми, на 1 см² кристалів яких розміщуються тисячі й мільйони транзисторів, а значення ємності запам'ятовуючих пристроїв сягають десятків гігабайтів для оперативної пам'яті (ВЗП) і терабайтів – для зовнішньої пам'яті. Швидкодія сучасних ЕОМ сягає сотень мільярдів елементарних операцій за секунду.

Комп'ютери **п'ятого покоління** планувалося створити в Японії наприкінці 80-х років минулого століття. Вони мали б відрізнитися від комп'ютерів попередніх поколінь уже не елементною базою (електронні лампи – транзистори – інтегральні мікросхеми – великі та надвеликі мікросхеми), а принципами опрацювання і зберігання даних. Очікувалося, що вони матимуть так звану **нефоннейманівську** архітектуру, будуть багатопроцесорними, подібними до людського мозку з паралельно працюючими нейронами, з реальним виконанням паралельних обчислень. Як відомо, у **фоннейманівських** комп'ютерах виконання кількох задач здійснюється перемиканням між ними.


У Інституті кібернетики АН України під керівництвом В. М. Глушкова у 1987 році також розроблено нефоннейманівську ЕОМ, пробна експлуатація якої показала перспективність цього напрямку розвитку. Цей комп'ютер було спроектовано таким чином, щоб забезпечувати паралельну роботу до 256 процесорів.


Величезний прогрес у створенні мікросхем, суттєве (у мільйони разів) збільшення швидкодії роботи процесорів створили можливість побудувати обчислювальні системи на основі фоннейманівської архітектури, які за швидкістю перевершили запроєктовані для нефоннейманівських ЕОМ показники. Напрацювання вчених у галузі паралельних обчислень використовуються для організації роботи сучасних систем з багатоядерними процесорами.

Перевіряємо себе

-  Коли і з якою метою створено перший програмно-керований автомат, який використовувався у промисловості? Чому його не можна назвати обчислювальною системою? ★
-   Назвіть приклади програмно-керуваних автоматів, які не є комп'ютерами. ★
-   З якою метою створено першу ЕОМ? Чому її створення і результати застосування тривалий час були суворо засекреченими? ★
-   Назвіть відомі вам побутові пристрої, в яких є вбудовані електронні цифрові системи. Завдяки чому це стало можливим? ★
5. Чим відрізняються покоління ЕОМ? ★
6. Де і коли створено першу в континентальній Європі ЕОМ? ★
7. До якого покоління ЕОМ можна зарахувати комп'ютер, за яким ви зараз працюєте? Чому? ★
-  Порівняйте обсяги пам'яті звичайного комп'ютерного оптичного диска, DVD диска, картки пам'яті для телефону або фотоапарата, флеш-пам'яті. Яку закономірність можна помітити? ★

Виконуємо

1. Знайдіть у мережі Інтернет відомості про комп'ютери, які розроблені й випускалися в Україні (“Промінь”, “Дніпро”, МІР, СМ1420, СМ1425, “Пошук”). Збережіть отримані дані для використання під час виконання навчального проекту. ★
2. Знайдіть у мережі Інтернет відомості про фірму Епл, її засновників. Збережіть отримані дані для використання під час виконання навчального проекту. ★
3.  Знайдіть у мережі Інтернет відомості про Катерину Логвинівну Ющенко. Збережіть отримані дані для використання під час виконання навчального проекту. ★

4.  Знайдіть у мережі Інтернет відомості про проект “Союз” – “Аполлон” та використання в ньому ЕОМ “Дніпро”. Збережіть отримані дані для використання під час виконання навчального проекту. ★
5. Знайдіть у мережі Інтернет відомості про перші бортові ЕОМ ракетно-космічних комплексів. Збережіть отримані дані для використання під час виконання навчального проекту. ★
6. Знайдіть у мережі Інтернет відомості про Олександра Миколайовича Щукарьова. Збережіть отримані дані для використання під час виконання навчального проекту. ★
7. Знайдіть у мережі Інтернет відомості про Грейс Мюррей Хоппер. Збережіть отримані дані для використання під час виконання навчального проекту. ★
8. Знайдіть у мережі Інтернет відомості про Миколу Амосова. Збережіть отримані дані для використання під час виконання навчального проекту. ★

2.2. Види сучасних комп'ютерів та їх застосування



Комп'ютер, або програмно-керований автомат, виник з необхідності автоматизації виробничих процесів і виконання обчислювальних робіт, потреба в яких на початку ХХ століття значно збільшилася.



Слід пам'ятати, що сучасні засоби бездротової комунікації випромінюють електромагнітні хвилі надвисокої частоти, тому всі пристрої, які використовують Wi-Fi, а також мобільні телефони, є **небезпечними для здоров'я**, якщо зловживати їх використанням. Тому точку доступу Wi-Fi (роутер) доцільно розташовувати в нежитловому приміщенні (коридорі) або закріпити якомога вище (під стелею).

Не слід **тривалий час розмовляти** по мобільному телефону, навіть використовуючи для цього блутус-гарнітуру. Краще застосовувати звичайну дротову гарнітуру, розташувавши телефон подалі від себе.

Працюючи за комп'ютером, **слід робити фізкультпаузи**, обов'язково **виходити на свіже повітря**.

Нині комп'ютери використовуються у багатьох сферах людської діяльності – від мобільних телефонів та інших побутових приладів до систем управління виробництвом. Особливе місце серед них належить персональним комп'ютерам. Персональні комп'ютери поділяються на два основних типи: стаціонарні (настільні) та мобільні (рис. 2.1).



Рис. 2.1. Основні типи комп'ютерів

Мобільними називаємо комп'ютери, якими можна користуватися у будь-яких умовах: на відпочинку, у дорозі. Залежно від характеристик і конструктивних особливостей вони поділяються на ноутбуки, нетбуки, планшетні та кишенькові комп'ютери (рис. 2.2).



Рис. 2.2. Основні типи мобільних комп'ютерів

Стационарні персональні комп'ютери нині використовуються менше, ніж 10 років тому, але все ж є одним із основних застосувань обчислювальної техніки. Їх конструкція також зазнала змін – з'явилися так звані моноблоки, в яких персональний комп'ютер виконано в одному корпусі з дисплеєм, інколи дисплей є пристроєм тактильного введення команд (має сенсорний екран, як більшість смартфонів). Вони більш компактні, ніж персональні комп'ютери традиційного компоновання, мають меншу вартість.

Набувають щораз більшого поширення телевізори, які, крім приймання телевізійного сигналу зі звичайної антени, супутникової антени, мережі кабельного телебачення, можуть забезпечувати вхід до мережі Інтернет, запис на запам'ятовуючі пристрої (вбудовані й зовнішні) статичних зображень, відео і звуку, мають вбудовані відеокамеру і мікрофон, тобто можуть використовуватися як повноцінні мережеві комунікатори. Такі телевізори називають Smart TV, тобто “розумними телевізорами”. Більшість показаних на рис. 2.3 пристроїв, які раніше були додатковими до стационарного персонального комп'ютера, вже є вбудованими в цей і подібні пристрої.







Рис. 2.3. Склад сучасної персональної обчислювальної системи


Більшість сучасних мобільних телефонів перестали нині бути тільки апаратами для розмов на віддалі. Наявність у них процесорів і пам'яті з характеристиками, які суттєво перевищують характеристики навіть деяких стаціонарних комп'ютерів, роблять їх пристроями, які забезпечують повноцінний вихід в Інтернет, фотографування, знімання і відтворення відео, відеозв'язок з використанням програм, подібних до Skype, спілкування з використанням засобів, подібних до Viber, тощо.

Завдяки поширенню засобів бездротового зв'язку і спрощенню доступу до мережі, виникають нові можливості комунікації. Прийшовши в гості до друзів, не потрібно навіть налагоджувати передавання принесених фотографій, відеозаписів на комп'ютер. Слід "дозволити" своєму телефону або планшету увійти до локальної мережі, до якої може входити і Smart TV. Поширені також електронні книги – рідери, у пам'ять яких можна ввести вміст всіх шкільних підручників і ще багатьох інших книжок.







Основні ж принципи роботи обов'язкового складника всіх сучасних цифрових пристроїв – комп'ютера спільні для всіх комп'ютерів, незалежно від того, керує цей комп'ютер ядерним реактором, автомобілем або телевизором.

Перевіряємо себе

1.  Що таке Skype і Viber? З якою метою їх можна використовувати? 
2. Що таке моноблок? Чим ви можете пояснити його меншу, порівняно зі стаціонарними комп'ютерами традиційного компонування, вартість? 
3. Чому рідери часто комплектуються ліхтариками, які можуть кріпитися на корпусі пристрою і під'єднуватися до USB порту? 

4.  Яка, орієнтовно, тривалість роботи без підзаряджання ноутбука, нетбука, мобільного телефону, рідера? Чим це зумовлено? ✦
5. Чим відрізняється зазвичай нетбук від ноутбука? ✦
6. Які фактори шкідливі для людини, що працює за персональним комп'ютером? ✦
7. Які шкідливі для здоров'я фактори, спільні для мобільного телефону і персонального комп'ютера, що працює в бездротовій мережі? Як уникнути дії цих факторів? ✦
8. Навіщо потрібно виходити на свіже повітря, робити фізичні вправи, дивитися на небо? ★
9. Що краще для вашого організму – пограти 30 хвилин у комп'ютерну гру чи цей же час пограти з друзями в квача або погуляти на свіжому повітрі? Чому? ★

Виконуємо

1. Знайдіть на рис. 2.3 зовнішні пристрої комп'ютера, які вбудовані в сучасний мобільний телефон. ✦
2.  Чим відрізняється планшетний комп'ютер від нетбука? Знайдіть спільні ознаки й відмінності. ✦
3.  Запишіть окремо назви пристроїв введення й пристроїв виведення, подані на рис. 2.3. Які сучасні пристрої, відомі вам, на рисунку не показано? ★
4.   У пошуковій системі Google (google.com.ua) здійсніть пошук за зразком “типи принтерів”. На знайденій сторінці Вікіпедії ознайомтеся з особливостями матричного, лазерного та струминного принтерів. Коротко занотуйте. ★
5.   В чому полягає небезпека для людини, яка працює за комп'ютером? Запропонуйте вправи для фізкультпаузи. Виконайте їх. ✦

2.3. Архітектура комп'ютера



Будова і алгоритм роботи. Процесор, його будова та призначення. Пам'ять комп'ютера, її види. Зовнішні та внутрішні запам'ятовуючі пристрої. Визначення властивостей комп'ютера.

Комп'ютер – пристрій, створений як універсальний інструмент опрацювання даних через виконання певної послідовності команд. Для управління комп'ютером задля виконання різних алгоритмів створюються програми, які зберігаються в запам'ятовуючих пристроях. Для виконання вони завантажуються до внутрішнього запам'ятовуючого пристрою (розміщуються у ВЗП).

Незалежно від розмірів, зовнішнього вигляду, призначення майже всі сучасні комп'ютери мають однаковий принцип дії та приблизно однакову загальну структуру.

✓ *Загальний опис побудови і функціонування комп'ютера як складної системи, включаючи зв'язок між процесором і пам'яттю, операції введення – виведення тощо, називають його архітектурою.*

У 40-х роках ХХ століття Джон фон Нейман запропонував правила побудови ЕОМ, які є основою створення вже чотирьох поколінь комп'ютерів.

1. Програма і дані зберігаються в одній загальній пам'яті (внутрішній пам'яті).

2. Кожна комірка внутрішньої пам'яті позначається номером, який називається її адресою.

3. Уміст комірок пам'яті (команди та дані) різняться за способом використання, але не за способом кодування або способом подання в пам'яті.

4. Кожна програма виконується послідовно, починаючи з першої команди. Для зміни цієї послідовності використовуються команди передавання управління.

✓ *Внутрішня пам'ять комп'ютера є єдиною і лінійною:*

- “єдина” означає, що програма і дані зберігаються в одній пам'яті, одна й та сама комірка пам'яті для однієї задачі може зберігати команду, а для іншої – дані;

- “лінійна” означає, що всі комірки пам'яті послідовно пронумеровані, номер комірки є її **адресою**.

Центральний процесор (ЦП) сучасного комп'ютера виконаний у вигляді великої мікросхеми, яка сама є маленьким комп'ютером. Центральний процесор містить арифметично-логічний пристрій (АЛП), у якому виконуються арифметичні та логічні дії над даними, регістри загального призначення (РЗП), над вмістом яких виконуються ці дії.

Важливими частинами ЦП є акумулятор і лічильник адреси. Акумулятор – оперативний запам'ятовуючий пристрій (ОЗП), через який здійснюється передавання даних і команд із ВЗП до ЦП, і навпаки.

Програми опрацювання даних у ЦП (у цьому випадку вони називаються мікропрограмами) містяться у внутрішньому постійному запам'ятовуючому пристрої процесора (ВПЗП) і забезпечують виконання зовнішніх команд. Керує роботою ЦП внутрішній пристрій управління (ВПУ).

Лічильник адреси (ЛА) також є оперативним запам'ятовуючим пристроєм, але в ньому зберігаються не дані й команди, а номер комірки ВЗП, з якою в даний момент часу з'єднано ЦП.

Апаратна частина кожного комп'ютера містить такі складові (рис. 2.4):

- 1) центральний процесор, який виконує арифметичні операції й логічні (порівняння) операції;
- 2) пристрій управління (ПУ), який забезпечує управління виконанням програм;
- 3) внутрішню пам'ять (ВП), призначену для збереження програм і даних у процесі виконання програми;
- 4) пристрої для забезпечення введення та виведення команд і даних (У/В).

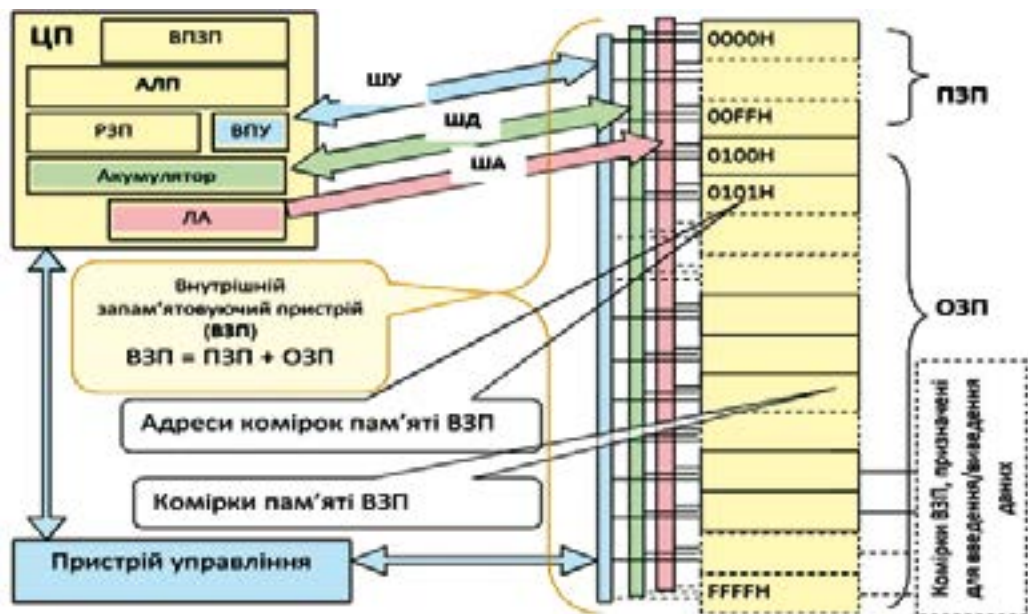


Рис. 2.4. Спрощена схема комп'ютера

На рис. 2.4 позначено:

ЦП – центральний процесор (CPU – Central Processing Unit);

ОЗП – оперативний запам'ятовуючий пристрій – (RAM – Random Access Memory – пам'ять довільного доступу);

ПЗП – постійний запам'ятовуючий пристрій (ROM – Read Only Memory – пам'ять тільки для читання);

Пристрої У/В – пристрої, призначені для введення/виведення даних (I/O – Input/Output – введення/виведення).

Внутрішня пам'ять комп'ютера (ВП), або внутрішній запам'ятовуючий пристрій (ВЗП), складається із постійного запам'ятовуючого пристрою і оперативного запам'ятовуючого пристрою.

Внутрішня пам'ять з'єднана з центральним процесором трьома системами провідників, які називаються **шинами** і разом утворюють **магістраль**. Така будова комп'ютера набула назви **магістрально-модульної архітектури**.

✓ *Магістрально-модульна архітектура – будова комп'ютера, за якої окремі частини комп'ютера з'єднуються між собою провідниками, призначення кожного з яких визначене для всіх комп'ютерів певного типу.*

Шина даних (ШД) слугує для передавання даних між центральним процесором і внутрішньою пам'яттю.

Шина управління (ШУ) слугує для передавання команд від процесора до внутрішньої пам'яті. Ці команди подаються тоді, коли процесор приймає дані з певної комірки пам'яті (кажуть – “зчитує дані”), або коли записує їх у пам'ять.

Шина адреси (ША) слугує для того, щоб з'єднати з ЦП певну комірку; на її провідники подаються сигнали, які у двійковому поданні відповідають номеру комірки ВЗП.

Дуже спрощено алгоритм роботи програмно-керованого автомата (комп'ютера), побудованого за принципом фон Неймана (**фоннейманівського типу**), можна описати таким чином:

1. Лічильник адреси (ЛА) встановлюється у стан, який відповідає адресі першої команди програми, що має виконуватись; цей же код подається на шину адреси (ША).

2. На шину управління (ШУ) подається сигнал “читаю”. З комірки ВЗП, адреса якої подана на ША, через шину даних (ШД) код (команда або дані) подаються до ЦП.

3. ЦП прийнятий код порівнюється з кодами, що містяться у внутрішньому постійному запам'ятовуючому пристрої ЦП (ВПЗП). Якщо це не код зупинки, то запускається на виконання відповідна мікропрограма, ЛА встановлюється в стан, що відповідає адресі комірки ВЗП, в якій містяться необхідні дані, інакше – виконання програми зупиняється.

4. На шину управління (ШУ) подається сигнал “читаю”, дані через ШД передаються у ЦП і опрацьовуються.

5. ЛА встановлюється у стан, що відповідає адресі комірки ВЗП, у яку має бути записано результат. На шину управління (ШУ) подається сигнал “записую”, за яким дані розміщуються у комірці пам'яті.

6. ЛА встановлюється у стан, що відповідає адресі комірки ВЗП, в якій розміщено код наступної команди, цей же код подається на ША.

7. Перейти до кроку 2.

Завдяки застосуванню магістрально-модульної архітектури стало можливим будувати дуже складні пристрої з більш простих “цеглинок” – модулів. На рис. 2.5 показано деякі модулі персонального комп'ютера і роз'єми, в які вони вставляються.

Перші персональні комп'ютери мали внутрішню пам'ять, кожна комірка якої складалась із восьми бістабільних пристроїв. Комірки пам'яті сучасних комп'ютерів складаються з 64 бістабільних пристроїв.

✓ *Кількість бістабільних пристроїв у одній комірці пам'яті називається розрядністю пам'яті.*

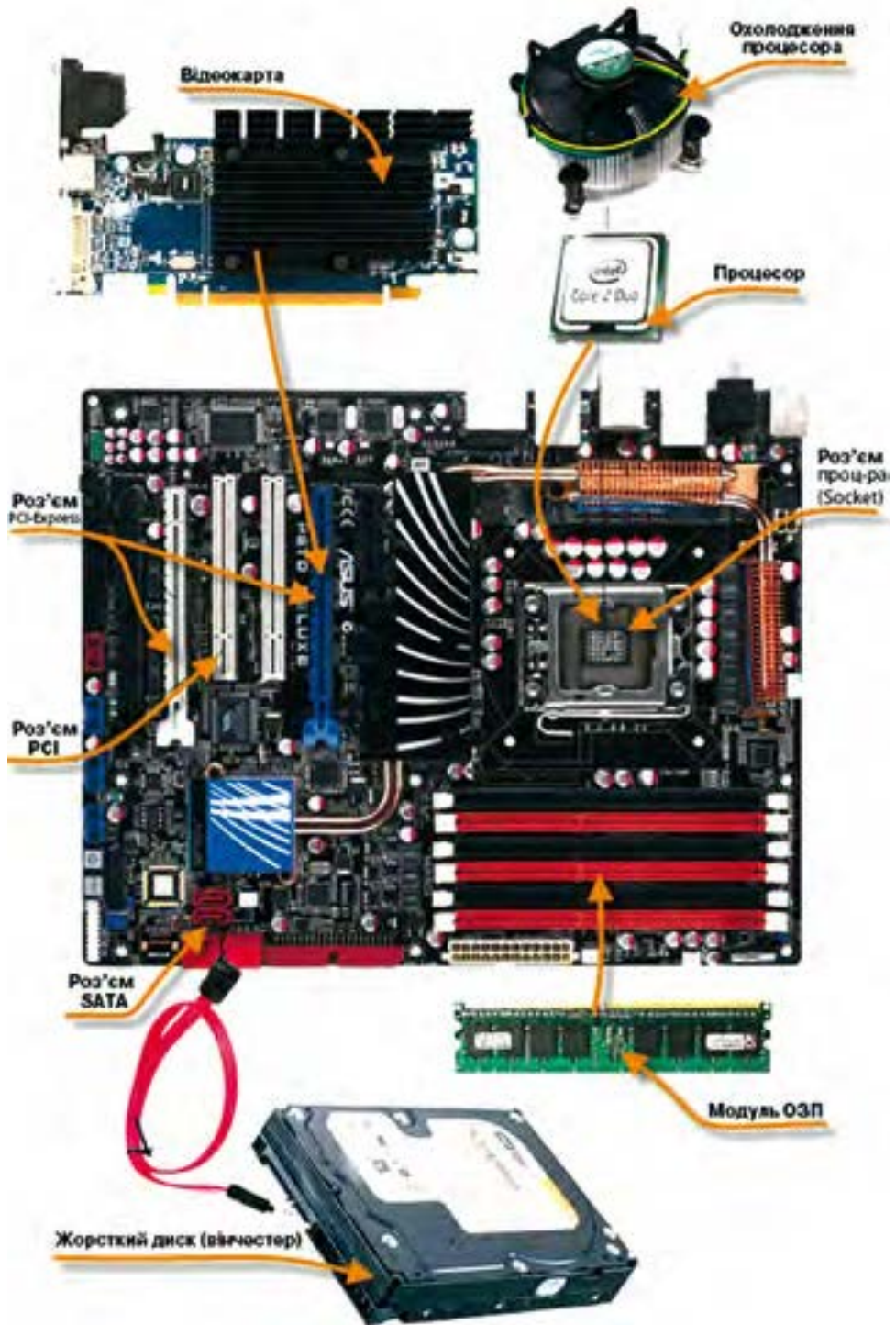


Рис. 2.5. Материнська плата і пристрої, що розміщуються на ній і приєднуються до неї

Як уже зазначалося, для того щоб бути виконаною центральним процесором, послідовність команд і даних (програма) має бути розміщена в комірках ВЗП.

До складу ВЗП (див. рис. 2.4) входять як комірки ОЗП, так і комірки ПЗП, тобто пристрою пам'яті, що не потребує напруги живлення для збереження даних. У цих комірках зберігаються команди й дані, потрібні одразу ж після ввімкнення комп'ютера. Інші програми для виконання завантажуються до ВЗП із зовнішніх запам'ятовуючих пристроїв (ЗЗП) – вінчестера, флеш-пам'яті, компакт-диска.

Спільне використання програмами апаратного обладнання і програмного забезпечення вимагає застосування системи управління програмними й апаратними засобами. Управління пристроями здійснюється різними програмами, але має бути узгоджене у часі. Принтер, процесор та інші пристрої не можуть одночасно виконувати більш ніж одну команду.

✓ *Одночасне звернення кількох програм або кількох команд, що належать одній програмі, до одного пристрою називають **конфліктом** (конфліктом доступу).*

Для уникнення конфліктів обчислювальна система розглядається як сукупність об'єктів, кожен із яких володіє певним ресурсом.

Ресурсами вважають:

- час процесора, який розподіляється між програмами, якщо виконується більше ніж одна програма;
- оперативну пам'ять, яку керуюча система виділяє кожній програмі у вигляді обмеженої початковою і кінцевою адресою ділянки внутрішнього запам'ятовуючого пристрою;
- час периферійних пристроїв, який керуюча система також розподіляє таким чином, щоб уникнути конфліктів доступу;
- дані, які зберігаються у ЗЗП, і вільну його частину.

✓ *Використанням ресурсів керує сукупність програм, яка називається **операційною системою**.*

Для універсальних комп'ютерів наявність операційної системи є обов'язковою, оскільки її складники забезпечують виконання операцій із завантаження до ОЗП програм і даних, роботу із зовнішніми пристроями пам'яті, управління апаратними складниками комп'ютера.

Операції, пов'язані з використанням однакового обладнання, виконуються однаково. Такими операціями є введення даних із клавіатури, виведення результатів на екран або принтер тощо. Зазначені операції, незалежно від того, якими програмами використовуються пристрої, виконуються зі застосуванням стандартних програм, що містяться у постійному запам'ятовуючому пристрої або завантажуються до ОЗП із ЗЗП.

Одним із основних способів забезпечення злагодженої роботи програм і зовнішніх пристроїв, уникнення конфліктів є використання переривань, які викликаються подіями.

✓ **Подією** називається поява сигналу (повідомлення) від зовнішнього пристрою (рух миші, натиснення клавіші на миші або клавіатури, вставлення флеш-картки у відповідний роз'єм тощо) або отримання цього сигналу в процесі виконання програми.









✓ **Переривання** – це спеціальний сигнал, або команда, що подається для перемикавання процесора на програму опрацювання події (переривання).

Опрацювання події (часто кажуть “опрацювання переривання”) означає, що деякі дані, які виникли внаслідок обчислень або отримані від апаратного забезпечення, мають бути програмно опрацьовані задля подальшого виконання необхідних для отримання результату дій.










Наприклад, після вставлення флеш-карти ми спостерігаємо на екрані відповідне повідомлення, рух миші по поверхні килимка супроводжується переміщенням курсора по екрану тощо.

✓ **Опрацювання подій** ви використовували, розробляючи програми в середовищі Скреч.

Виконуємо

-  Оцініть кількість можливих команд процесора і, відповідно, мікропрограм процесора, за 8-розрядного і 16-розрядного кодування їх номерів. ✦
- Наведіть приклади подій, що зумовлюють переривання виконання програми комп'ютером. ✦
-  Опишіть, що відбувається після натиснення клавіші PrintScreen на клавіатурі. ★
-  Від чого залежить швидкість опрацювання даних комп'ютером? Порівняйте дані, отримані для кількох комп'ютерів. ✦
-   Визначте властивості апаратного забезпечення комп'ютера робочого місця учня. Порівняйте з властивостями комп'ютера SM1425 (дані знайдіть у мережі Інтернет). ✦
-  Перші процесори для персональних комп'ютерів мали 8-розрядні комірки ВЗП і 16-розрядну шину адреси. З яким обсягом пам'яті вони могли працювати? ✦
-  На рис. 2.5 знайдіть, де містяться ША, ШД і ШУ. ✦
-  У середині системного блока комп'ютера розміщено пристрій пам'яті, який називається “вінчестер”. Чи можна назвати вінчестер внутрішнім запам'ятовуючим пристроєм? Поясніть відповідь. ✦

Перевіряємо себе

1.  Вміст якої комірки ВЗП передається до ЦП? 
2.  Сучасні операційні системи забезпечують або 32-розрядне, або 64-розрядне адресування команд і даних. Під управлінням якої ОС комп'ютер працюватиме швидше? Чому? 
3. Які переваги має шинно-модульна архітектура комп'ютера? 
4.  Що таке ША, ШД, ШУ? 
5.  Після ввімкнення живлення комп'ютера першою виконується програма самотестування комп'ютера (PROST – Power On Self Test). У якій пам'яті вона має зберігатися? Обґрунтуйте. 



ДЛЯ ДОПИТЛИВИХ

Готова до виконання програма (програма у машинних кодах) розміщується на ВЗП, з якого завантажується до ВЗП.

Існують спеціальні програмні засоби, які дають можливість переглядати вміст внутрішньої пам'яті комп'ютера, спостерігати за процесами виконання програм, “розшифровувати” програми, подані в машинних кодах. На рис. 2.6 подано результат роботи однієї з таких програм.

Адреси комірок пам'яті й коди команд подано в шістнадцятковому кодуванні.



The screenshot shows a debugger window titled "Hiew: KEYRUS.COM". The main area displays assembly code in hexadecimal and mnemonic format. Annotations with callouts identify specific parts of the code:

- значення операндів** (operand values): Points to the right column of the code, such as "ax, 04052" and "0000000E".
- команди (мовою Макроасемблера)** (assembly commands): Points to the mnemonic column, such as "jap", "push", "pop", "mov", "cmp", "jnz", "add", "sub", "push", "pop", "japs", "jz", "and", "japs".
- коди команд і даних** (command and data codes): Points to the left column of the code, such as "E9FB32", "305248", "7405", "EAD4010C15", "1E", "0E", "1F", "A0C", "80F", "758", "B082", "B107", "B303", "1E", "07", "E823", "74902", "7412", "80F96", "7504", "24FC", "E809".
- адреси комірок** (memory addresses): Points to the leftmost column of the code, such as "00000000", "00000003", "00000004", "00000007", "00000009", "0000000E", "00000010", "00000011", "00000014", "00000017", "00000019", "0000001B", "0000001D", "0000001F", "00000020", "00000021", "00000023", "00000026", "00000028", "0000002B", "0000002D", "0000002F".

Рис. 2.6. Програма, готова до завантаження до ВЗП комп'ютера

2.4. Мультимедійні пристрої введення та виведення



Управління комп'ютером відбувається через пристрої введення/виведення, які забезпечують відображення повідомлень від комп'ютера у зрозумілій користувачеві формі (текстовій, графічній, звуковій) та введення до комп'ютера даних і команд. Під час виконання програми процесор періодично опитує пристрої введення/виведення. Виконання програми може перериватись сигналами пристроїв керування, які генеруються як реакція на події – запити зовнішніх пристроїв на обслуговування, команди користувача тощо.



Пристрої, що входять до складу мультимедійного обладнання. Технічні характеристики зображення і мультимедійного обладнання.

Для правильного використання пристроїв введення/виведення потрібно знати, яким чином характеризують їх якість, одиниці фізичних величин, які застосовуються для порівняння якості різних засобів відтворення зображень і звуку.

Загальною назвою засобу, на якому відтворюється зображення, є назва “екран”. Залежно від фізичної реалізації розрізняють екрани, які відбивають світло, й такі, що його випромінюють. Зображення може відтворюватись на екрані електронно-променевої трубки, екранах, утворених із сукупності окремих елементів, які або самі випромінюють світло (електронно-променеві трубки, люмінесцентні екрани або плазмові панелі), або змінюють свою прозорість, пропускаючи різну кількість світла від джерела (дзеркальної поверхні, люмінесцентних ламп, світлодіодів).

Зображення, незалежно від способу його відтворення, характеризується такими основними показниками: **яскравість, контрастність, чіткість, розміри.**

Одним із основних методів утворення зображення, яке може спостерігати велика група людей, є світлова проекція.

✓ **Світлова проекція** – одержання на екрані зображення певного об'єкта з використанням потужного джерела світла й оптичної системи.

Засоби світлової проекції характеризуються такими основними показниками якості: **світловий потік і яскравість.**

✓ **Світловий потік** характеризує потужність світлового випромінювання проектора і вимірюється в люменах (лм).

У цих же одиницях виражається світловий потік, який випромінюють освітлювальні лампи. Знаючи величину світлового потоку проектора,

визначимо, чи можна використовувати засіб без затемнення у певному приміщенні.

Яскравість характеризує зображення, отримане на екрані. Яскравість зображення залежить від коефіцієнта відбиття поверхні екрана, тобто від того, яка частина світла, що падає на екран, відбивається. Сучасні матеріали екранів мають коефіцієнти відбивання приблизно 0,85...0,99.

✓ *Яскравість вимірюється в канделах на квадратний метр ($\text{кд}/\text{м}^2$).*

Орієнтовно можна вважати, що яскравість зображення, більша за $120 \text{ кд}/\text{м}^2$, достатня для його спостереження за умов нормальної освітленості.

Яскравість зображення, створюваного сучасними проекційними засобами зі світловим потоком 1000...1600 лм на сучасних екранах, становить не менше $120...180 \text{ кд}/\text{м}^2$. Для порівняння – яскравість зображення, створюваного рідинно-кристалічними моніторами з діагоналлю екрана 19 дюймів (46 см), становить приблизно $300 \text{ кд}/\text{м}^2$.

✓ *Контрастність зображення визначається відношенням яскравості найбільш світлих ділянок зображення на екрані до яскравості найбільш темних.*

Досить якісне за контрастністю зображення характеризується значенням 100:1. Реальні об'єкти мають контрастність від 10:1 (портрет білявої людини при прямому освітленні) до 10000:1. Сучасні проекційні засоби відтворення зображення забезпечують контрастність до 2000:1 і більше.

✓ *Роздільна здатність зображення є його об'єктивною характеристикою, яка визначається кількістю точкових елементів (пікселів, англ.: picture cell, комірка зображення), які утворюють це зображення.*

Найменшою роздільною здатністю засобу відтворення зображення, за якої зображення розмірами приблизно 16×24 см може вважатись достатньо якісним для спостереження на віддалі найкращого зору (20...30 см), є така, що забезпечується кількістю елементів зображення $640 \times 480 = 307200$. Якісніше зображення створюється за роздільної здатності $800 \times 600 = 480000$ елементів. Більшість сучасних цифрових засобів відтворення зображення забезпечує роздільні здатності $1024 \times 768 = 786432$ елементів та 1152×864 , 1280×720 , 1280×768 , 1280×1024 і більші.

Відомості про яскравість і колір кожного елемента (пікселя) зображення зберігаються у одному або кількох бістабільних пристроях пам'яті. Кількість таких пристроїв, призначених для зберігання даних про один піксель, називається **глибиною кольору**.

✓ *Глибина кольору вимірюється у бітах.*

Для цифрових засобів відтворення зображення роздільна здатність і глибина кольору визначаються обсягом пам'яті, призначеної для зберігання зображення. На пристроях зовнішньої пам'яті зображення зберігаються у вигляді файлів.

✓ Для того щоб оцінити (у байтах) розмір файла, в якому зберігається зображення без його стиснення, потрібно: перемножити розміри зображення у пікселях (визначити кількість пікселів у зображенні), отримане число помножити на глибину кольору і поділити на вісім.

Наприклад, зображення розміром 640×480 пікселів і глибиною кольору 8 біт, матиме розмір $(640 \cdot 480 \cdot 8) / 8 = 307200(\text{Б}) = 307200 / 1024 = 300$ (кБ). Зображення розміром 1280×1024 пікселів, з глибиною кольору 24 біти зберігатиметься у файлі з розширенням *.bmp розміром $1280 \cdot 1024 \cdot 24 / 8 / 1024 / 1024 = 3,75$ (МБ).

✓ Чіткість зображення визначається можливістю відтворення дрібних деталей зображення на екрані.

Зображення на папері, світлинні, вважається достатньо чітким, якщо можна розрізнити 50 ліній на 1 см зображення. Така чіткість вимагається від зображень, які людина з нормальним зором спостерігає з віддалі найкращого зору. Для зображень, призначених для спостереження на іншій віддалі, враховуються кутові розміри найменшого елемента зображення, тобто кут при вершині рівнобедреного трикутника, основою якого є елемент зображення, а у вершині розташоване око спостерігача.

Людське око не розрізняє кольорів дрібних деталей, тому чіткість зображення визначається для одноколірного (монохромного) зображення. Чіткість зображення, таким чином, є частково його суб'єктивною характеристикою.

У сучасних проєкційних засобах прозора оптична матриця, з використанням якої створюється зображення, встановлюється між потужним джерелом світла і проєкційним об'єктивом. Від співвідношення між розміром цієї матриці (панелі), фокусною відстанню об'єктива і віддаллю від проєктора до екрана, залежить розмір зображення.

Сучасним пристроєм уведення, який використовується в багатьох пристроях, є сенсорний, або тактильний, екран.

✓ Сенсорним екраном називають поверхню, на якій розташовані чутливі елементи, дотик до яких пальцем або спеціальним пристроєм перетворюється на сигнал, що передається в комп'ютер.

Одним із перших застосувань подібних засобів уведення команд у комп'ютер була система "електронне перо", яка використовувалась на вітчизняних ЕОМ МІР (Машина Інженерних Розрахунків), що серійно випускались в Україні в 60-х–70-х роках минулого століття.

✓ Звук відтворюється акустичною системою, яка характеризується потужністю (вимірюється у ватах, Вт) і частотним діапазоном (вимірюється в герцах, Гц).

Прийнятними для якісного відтворення звуку є потужність одного каналу (звукової колонки), не менше 3 Вт, і частотний діапазон системи від 50 до 12500 Гц.

За технічними засобами, які поєднують у собі систему відтворення зображення, систему відтворення звуку і систему управління, закріпилась назва “мультимедійна система” (рис. 2.7).

Взаємодію складників мультимедійної системи можна описати таким чином:

– комп’ютер передає зображення до проектора, звук – до акустичної системи;

– сенсорний екран (або інший пристрій уведення команд) приймає команди від користувача і передає їх у комп’ютер;

– комп’ютер опрацьовує ці сигнали і виконує запуск певних програм, додатків тощо.

Проектор способом світлової проекції відтворює зображення на екрані.

Програмними складниками таких комплексів можуть бути програми Microsoft Office PowerPoint, StarOffice Impress, Photodex ProShow, Producer 4.1, Camtasia Studio та інші.

Мультимедійне апаратне забезпечення – це обладнання, необхідне для створення, зберігання та відтворення мультимедійного програмного забезпечення. До нього належать: звукова карта, дисковод для відтворення і запису оптичних дисків, звукові колонки. Таке устаткування називають також базовим мультимедійним комплектом.

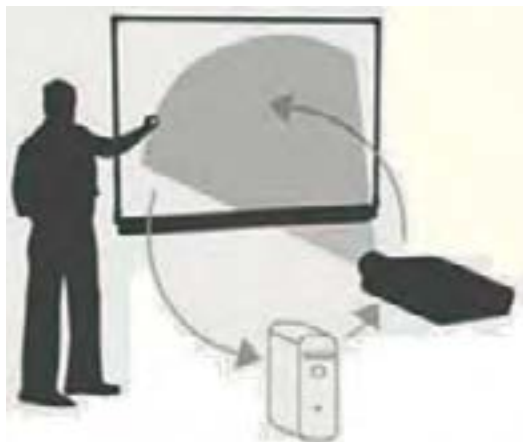






Рис. 2.7. Комплекс апаратних засобів, призначений для відтворення зображення

Перевіряємо себе

1. Які апаратні засоби і пристосування потрібні для відтворення зображення методом світлової проекції? ▲
2. Чому для проекційних екранів використовують матеріали з найбільшим можливим коефіцієнтом відбивання світла? ▲
3.  Які фізичні величини характеризують зображення? ✦

4. Коли доцільно використовувати управління демонстрацією з клавіатури, а коли – з сенсорного екрана? Наведіть приклади. ✦
5. У яких сучасних пристроях використовуються сенсорні поверхні? Наведіть приклади. ✦
6.  У якому випадку дисплей може бути тільки пристроєм виведення, а в якому – забезпечуватиме й уведення команд? ✦

Виконуємо

1. Обчисліть розмір файла, в якому розміщено (без стиснення) копію зображення, створеного на екрані розміром 800×600 пікселів з роздільною здатністю 800×600 пікселів і глибиною кольору 24 біти. ✦
2.  Яким чином можна обчислити розмір файла, в якому розміщено зображення, що займає прямокутну область певного розміру на екрані, якщо задано роздільну здатність пристрою і розміри (у сантиметрах) цього зображення? Що додатково потрібно знати при цьому про пристрій для відтворення зображення? ★
3.  Виконайте креслення, яке б проілюструвало текст: “Для зображень, призначених для спостереження на іншій віддалі, враховуються кутові розміри найменшого елемента зображення, тобто кут при вершині рівнобедреного трикутника, основою якого є елемент зображення, а у вершині розташоване око спостерігача”. ★

Підказка: скористатися подібністю двох рівнобедрених трикутників, в основі яких лежить найменший елемент зображення, який можна розрізнити, і побудованих для зображення, розташованого на віддалі найкращого зору і на більшій віддалі.



ДЛЯ ДОПИТЛИВИХ

Про світло, екологію та історію людства

Величина світлового потоку та світлова температура нині вказуються на всіх сучасних джерелах світла. Навіщо?

Протягом багатьох століть людство не замислювалося над проблемою економії природних ресурсів, над тим, що ці ресурси не є невичерпними. Зокрема, для освітлення використовували переважно лампи розжарення, коефіцієнт корисної дії яких приблизно такий, як у паровоза, тобто лише близько 5% електроенергії у них перетворюється на світло, а решта – на тепло. Вони конструктивно дуже прості й недорогі у виробництві, не вимагають спеціальних заходів з утилізації, але мають невеликий термін експлуатації, і, найголовніше, споживають дуже багато електроенергії.

Найбільшим досягненням технологій є світлодіодні лампи, коефіцієнт корисної дії яких у десять разів і більше перевищує коефіцієнт корисної дії ламп розжарення. Термін їх експлуатації вимірюється десятками років, вони **не містять ртуті**.

Для того щоб переконати споживачів купувати нові освітлювальні прилади, на упакуванні ламп вказують не просто споживану потужність, а й величину світлового потоку в люменах і так звану колірну температуру в кельвінах.

Колірна температура, або колірність, є параметром джерела світла, який використовується для позначення відтінку світла від джерела або зображення на екрані. Проектори і більшість сучасних дисплеїв також мають можливість регулювати цей параметр.

Слід мати на увазі, що збільшення колірної температури забарвлює зображення у “холодні” кольори (блакитний, синій), а зменшення – у “теплі” (помаранчевий, червоний). Чому? Первісна людина стикалася лише з низькотемпературними (“червоними”) джерелами світла – вогнищем, лісовою пожежею тощо. Для неї тепло пов’язувалося лише з цими джерелами. Джерела світла, якими послуговувалися художники середніх віків, були такими ж – свічки, вогонь у каміні. Зараз ми знаємо, що найгарячіші джерела світла – електрична дуга, плазмовий різак – мають синюватий, а то й фіолетовий відтінок, але з багатовіковим досвідом людства не сперечаємося.

Про звук і вуха

Людське вухо – дуже складний орган, який перетворює коливання повітря, тобто звук, на нервові сигнали. Подібним чином діє й мікрофон, який перетворює звукові коливання на електричний сигнал – зміни електричного струму.

У людини, як і в більшості тварин, є два вуха. **Навіщо?**

Поширюючись від джерела, звук змінюється, зменшується його гучність, інколи звуки від кількох різних джерел послаблюються по-різному. Два вуха надають можливість людині визначити напрям на джерело звуку. Слухаючи “наживо” в залі великий оркестр, ми можемо, навіть заплющивши очі, розпізнати, як на сцені розташовані інструменти, як рухається по сцені соліст.

Для того щоб створити у слухача відчуття присутності в залі, при записі музики використовують щонайменше два мікрофони, сигнали від яких записуються окремо й відтворюються двома пристроями – навушниками, акустичними колонками (стереозапис).

2.5. Класифікація та загальні характеристики програмного забезпечення



Основним програмним забезпеченням комп'ютера є комплект програм, які називаються операційною системою. Для виконання програма завантажується із зовнішнього пристрою пам'яті у внутрішню пам'ять.




Основні функції операційних систем. Поняття сумісності програмного забезпечення. Ліцензії на програмне забезпечення, їх типи. Інсталяція та деінсталяція програмного забезпечення.

Програмне забезпечення (ПЗ) персонального комп'ютера можна поділити на три основні складові: системне, прикладне і системи програмування.

До системного програмного забезпечення зараховують два типи програм: **операційні системи (ОС)** й **утиліти**. Інколи утиліти також відносять до ОС. Найголовніші функції ОС полягають у забезпеченні роботи прикладних програм і пристроїв, а також у підтримці інтерфейсу користувача.

Нині найпоширенішими операційними системами для персональних комп'ютерів є ОС Windows (версії 8 і 10), досить поширеними є Mac OS, Linux. Для портативних пристроїв (смартфонів, планшетних комп'ютерів, електронних книг, цифрових програвачів, наручних годинників, ігрових приставок, нетбуків, смартбуків) використовуються ОС Android (створена на основі Linux), Windows CE і iPhone OS.

 **Операційна система** – це комплекс взаємозв'язаних програм, призначений для управління обчислювальними процесами, апаратними засобами, ефективного розподілу ресурсів комп'ютера між обчислювальними процесами, а також організації взаємодії користувача з комп'ютером.

 Операційні системи **класифікуються** за багатьма ознаками. Найголовніші з них:

1. За типом комп'ютера, для якого призначена ОС – для великих ЕОМ, для ПК, для смартфонів та інших мобільних пристроїв.

2. За кількістю процесів, що одночасно виконуються, **однозадачні** та **багатозадачні**.

3. За кількістю одночасно обслуговуваних процесорів – **однопроцесорні** та **багатопроцесорні**.

4. За типом інтерфейсу користувача – з графічним інтерфейсом і з інтерфейсом командного рядка.

5. За кількістю двійкових розрядів даних, що одночасно опрацьовуються, – 16-розрядні, 32-розрядні і 64-розрядні.

6. За кількістю користувачів – однокористувацькі й багатокористувацькі.

7. За методом використання ресурсів – локальні та мережеві.

8. За методом доступу користувача до ресурсів комп'ютера: розподілу часу, реального часу і пакетного опрацювання.

Однозадачні операційні системи використовуються тоді, коли комп'ютер призначено для одночасного виконання тільки однієї задачі – керування деяким пристроєм (двигуном, автоматичною лінією на виробництві, атомним реактором), в ігрових приставках.

Багатозадачні ОС використовуються здебільшого у випадках, коли у користувача виникає необхідність паралельного виконання кількох задач – створювати текстовий документ і шукати для нього відомості в Інтернеті, виконувати обчислення в електронних таблицях. Багатозадачні ОС використовуються й у багатофункціональних пристроях – телевізорах, медіа центрах тощо.

✓ *ОС Windows, Mac OS, Linux є багатозадачними.*

Більшість ОС призначені для завантаження із ЗЗП комп'ютера, але зараз набувають популярності ОС з віддаленим завантаженням і так звані хмарні ОС, наприклад, Google Chrome OS, iCloud. Особливостями цих ОС є забезпечення доступу користувача до ресурсів незалежно від його розташування. Таким чином використовуються ресурси, які можуть належати кільком комп'ютерам, розташованим навіть на різних континентах.

Значна частина ОС передбачає можливість авторизації користувача. Такі ОС називаються *багатокористувацькими*. Створення для користувача власного реєстраційного запису (англ. *account*, рахунок) передбачає зберігання для нього налаштувань програмних і апаратних засобів, створення простору для зберігання даних, надання певних прав на управління обчислювальною системою. Авторизація, тобто вхід до системи, передбачає зазвичай два кроки – введення імені користувача (login) і пароля (password). Процедура авторизації обов'язкова для роботи в усіх системах з великою кількістю користувачів – поштових сервісах, хмарних сервісах, іграх.

✓ *Операційні системи сучасних ПК містять такі основні частини: ядро ОС; драйвери; файлову систему; інтерфейс користувача; бібліотеку системних функцій.*

✓ *Ядро – центральна частина ОС, що керує процесом виконання програм і їх доступом до ресурсів комп'ютера.*

Ядро ОС визначає, коли і яку програму слід запустити процесору, які ресурси слід надати програмі, якій програмі необхідно у даний момент дозволити доступ до того чи іншого апаратного засобу. Під час запуску

програми ядро ОС виділяє програмі частину внутрішньої пам'яті. Важливою частиною ядра ОС є командний процесор, який отримує команди користувача і виконує їх. Ядро після завантаження ОС постійно зберігається в оперативній пам'яті, тобто є **резидентною частиною ОС**.

Сучасні багатозадачні ОС включають ще один обов'язковий складник – механізм підтримки багатозадачності, або планувальник процесів. Для того щоб можна було одночасно виконувати декілька програм (задач), ядро ОС під час їх завантаження до ВЗП розподіляє їх на процеси, тобто частини програми, які допускають незалежне виконання (рис. 2.8).

✓ **Драйвер** – це програма, що керує конкретним пристроєм комп'ютера.

Кожний пристрій має власні драйвери для різних ОС (драйвер монітора, драйвер клавіатури та інші). Драйвери постачаються разом з пристроєм або їх можна завантажити із сайту розробника (рис. 2.9). Драйвер перетворює команди ОС на команди, доступні конкретному пристрою.

Наприклад, за допомогою графічного редактора можна створити і надрукувати на принтері малюнок. Але графічний редактор є загальним для будь-якого типу принтера, він не враховує особливості, наприклад, лазерного чи матричного принтера. Графічний редактор повідомляє драйверу, де й що друкувати, а все інше виконує драйвер підключеного принтера.

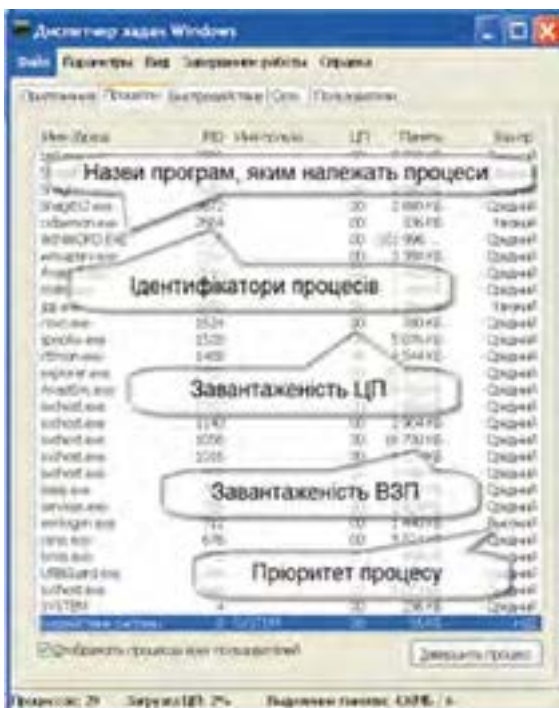


Рис. 2.8. Виконання програм обчислювальною системою під управлінням ОС Windows XP

Файлова система призначена для розміщення і читання із зовнішніх запам'ятовуючих пристроїв програм і даних відповідно до визначених правил. У сучасних пристроях пам'яті використовуються файлові системи FAT32, NTFS та інші. Користувач працює з файловою системою на рівні файлів і папок. Під час запису або читання файла його ім'я повідомляється драйверу файлової системи. Усі інші операції виконуються файловою системою. Файл є найменшою неподільною одиницею даних, з якою працює користувач засобами ОС.

✓ **Файл** – це ділянка пам'яті з присвоєним їй ім'ям, у якій зберігаються програма або дані.

Для групування і систематизації файли розподіляються по каталогах (папках). Папка – це частина файлової системи, яка має ім'я й яка може містити інші папки та файли.

✓ **Інтерфейс користувача** – сукупність апаратних і програмних засобів, які забезпечують взаємодію користувача і комп'ютера.



Рис. 2.9. Компакт-диски з програмним забезпеченням

У сучасних ОС використовуються два типи інтерфейсу: інтерфейс командного рядка і графічний інтерфейс. У інтерфейсі командного рядка команди вводяться у спеціальний рядок за допомогою клавіатури. Графічний інтерфейс дозволяє вводити команди за допомогою миші

обранням відповідних об'єктів (наприклад, піктограм) або команд меню, розташованих на екрані монітора. Інтерфейси користувача мають як операційна система, так і прикладні програми.

Частиною ОС є й бібліотеки системних функцій.

✓ **Бібліотека системних функцій** – сукупність програм, що використовуються багатьма користувачами.

Їх не потрібно розробляти користувачу і програмісту щоразу, коли в цьому виникає потреба, а досить звернутися до відповідної системної функції бібліотеки. Прикладами таких функцій є отримання даних про дисковий простір, створення файла та інші, якими ви вмієте користуватися.

Операційна система позбавляє користувача і програміста необхідності детального знання будови апаратних засобів комп'ютера й управління ними. Наприклад, для запису даних на жорсткий магнітний диск потрібно вказати номер сектору, доріжки і розмір потрібного дискового простору. Але користувачеві не потрібно їх вказувати. Усе це зробить операційна система, а користувачу досить вказати імена диска і файла.

Додаткові зручності для користувача створюють програмні засоби, які називаються **утилітами**.

✓ **Утиліти розширюють можливості, які надаються ОС, і реалізують функції службового ПЗ.**

За допомогою утиліт здійснюють архівування даних, захист комп'ютера від вірусів, очищення жорсткого магнітного диска тощо.

Системи програмування призначені для розроблення програм, у тому числі системних і прикладних. З однією з них ви вже знайомі – це навчальна система програмування Скреч. Відомими професійними системами програмування є: C++, Java, Visual Basic, Turbo Pascal та інші.

✓ **Прикладне програмне забезпечення (ПЗ) використовується у різних сферах людської діяльності.**

Воно забезпечує виконання конкретних завдань, наприклад, малювати, розраховувати прибуток фірми, вводити й редагувати текст, відтворювати музику, здійснювати пошук необхідної інформації.

Прикладне ПЗ поділяється на дві групи: загального і спеціального призначення. До ПЗ загального призначення належать текстові редактори, табличні процесори, графічні редактори, системи керування базами даних, програми для роботи в Інтернеті, програми для підготовки електронних презентацій та деякі інші. Такими програмами користуються фахівці різних спеціальностей. Вони об'єднуються в пакети офісних програм, одним із найпопулярніших з яких є Microsoft Office, до складу якого входять програми Word, Excel, Access, PowerPoint та інші.

До програм спеціального призначення належать вузькоспеціалізовані програми, призначені для фахівців певної спеціальності, наприклад, тренажери для підготовки пілотів, програми для дополіграфічної підготовки документів, програми з Web-дизайну.



Комп'ютерні програми є **об'єктами авторського права** й охороняються законом.

В Україні комп'ютерна програма стала об'єктом авторського права з набуттям у 1993 році чинності закону України “Про авторське право і суміжні права”, який у 2001 році був удосконалений. Авторське право на програмний засіб доводять до відома загалу документом, який називається ліцензією.



Ліцензія – правовий документ, що визначає порядок використання і розповсюдження ПЗ.

Ліцензія захищає авторські права і містить гарантії прав користувачів.

З точки зору прав на використання і розповсюдження програмного забезпечення воно поділяється на **вільне** (відкрите) і **комерційне** (пропріетарне) ПЗ. Ліцензії існують як на вільне ПЗ, так і на пропріетарне.

З вільним ПЗ користувач може працювати без додаткового погодження дій, але він має діяти в межах ліцензії.

Ліцензія на пропріетарне ПЗ дає дозвіл використовувати одну або кілька копій програм, однак її власником залишається розробник (автор, виробник). Користувачеві надається обмежений набір прав, які перелічуються в ліцензії. У ліцензії подається також перелік дій, виконувати які клієнту заборонено.



Для всіх комп'ютерів існує проблема програмної сумісності, під якою розуміють можливість виконувати програми, розроблені для одного комп'ютера, на іншому.

Різні моделі однієї сім'ї ПК мають зазвичай сумісність знизу вгору. Це визначає, що ПК “старших” моделей, які є потужнішими від попередніх, можуть виконувати програми, розроблені для “молодших” моделей ПК, але не навпаки.

Є кілька рівнів сумісності: сумісність на рівні ОС і на рівні базової системи введення/виведення (англ. Base Input Output System – BIOS). У першому випадку досягається повна програмна сумісність, у другому – неповна.

Для використання ПЗ на комп'ютері потрібно виконати його інсталяцію або встановлення (рис. 2.10–2.12). Найчастіше інсталяція програмних продуктів виконується за допомогою програм інсталяції, що входять до складу майже кожної програми.



Рис. 2.10. Перший крок встановлення графічного редактора

Інсталяцію програм дозволяється виконувати тільки користувачу з правами адміністратора. Як ми вже знаємо, цей процес відбувається за кілька етапів, протягом яких необхідно виконувати певні дії, відповідати на запитання програми, вказувати за потреби особливості встановлення програм.




Рис. 2.11. Ознайомлення з ліцензійною угодою

Обслуговування пристроїв комп'ютера виконують програми – драйвери, які можуть бути різними на різних комп'ютерах. Для того щоб

програмний засіб запрацював правильно, необхідно створити для нього певні умови, надати адреси, за якими розташовано потрібні програми, до яких він звертатиметься.

Дуже важливо й те, що користувач обов'язково має ознайомитись із ліцензійною угодою, погодитися з умовами, на яких він користуватиметься певним програмним засобом.

У ОС родини Windows дані щодо встановлених програм і їх налаштувань зберігаються в спеціальному текстовому файлі, який називається **реєстр**.

 Реєстр Windows має чітко визначену структуру. Будь-яке її порушення призводить до порушення роботи ОС, інколи – до повної її відмови.

Основна частина реєстру – це ключі (або параметри), в яких і зберігається вся інформація щодо налагодження ОС та програм користувача. Кожен параметр реєстру Windows відповідає за певну властивість системи. Ключі з даними про налаштування комп'ютера і програм об'єднані в розділи, що своєю чергою є підрозділами більших розділів. У реєстрі, зокрема, зберігаються дані, що вказують, файли яких типів опрацьовуються певною програмою, інші налаштування.

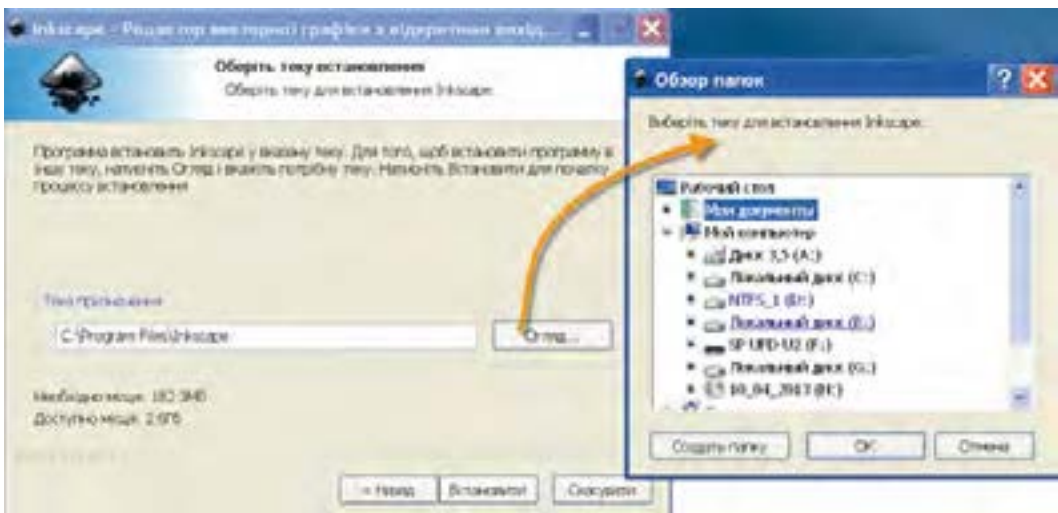


Рис. 2.12. Налаштування місця встановлення програмного засобу

У файл реєстра під час встановлення записуються налаштування для кожної програми.

Операційні системи також встановлюються з використанням спеціального програмного забезпечення. Оскільки ОС є дуже складними програмними комплексами, які постійно вдосконалюються, більшість з них мають у своєму складі засоби налаштування й оновлення.

Панель керування ОС Windows 7 показано на рис. 2.13. Користувачу доводиться виконувати такі основні налаштування: створення облікових записів (account) користувачів; оформлення робочого стола і встановлення роздільної здатності екрана, розмірів шрифту у вікнах; визначення часу, дати, мов уведення і мови графічного інтерфейсу; вилучення програм; оновлення ОС (рис. 2.14).

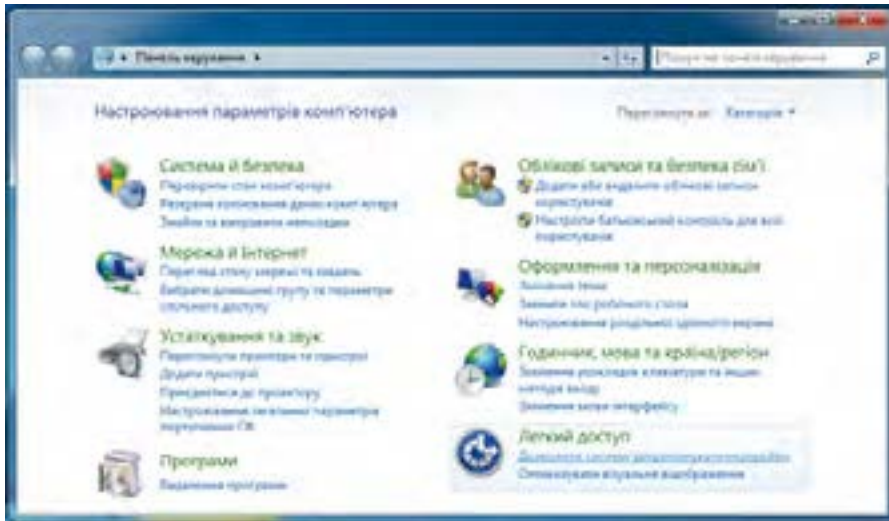


Рис. 2.13. Панель керування ОС Windows 7

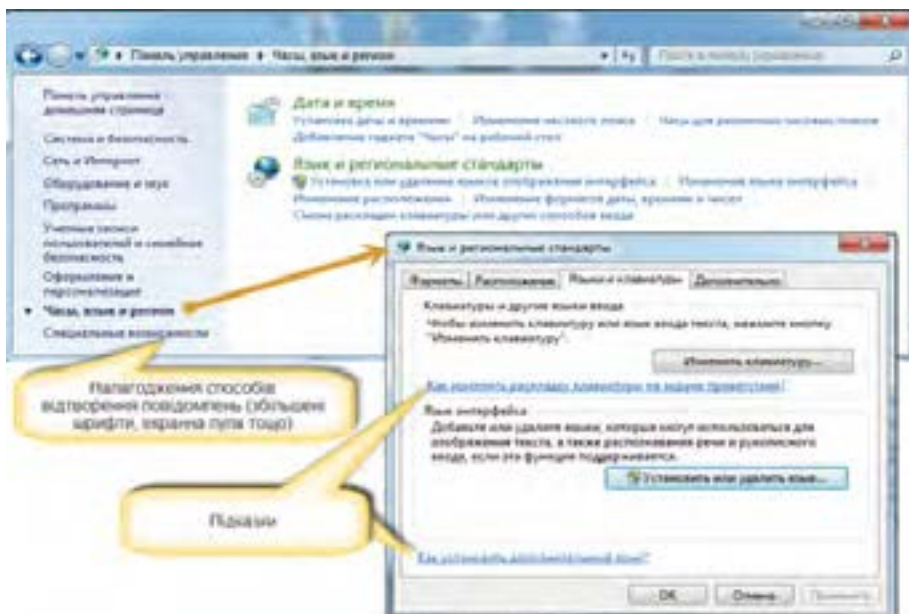



















Рис. 2.14. Вихід на налаштування регіональних і мовних параметрів інтерфейсу ОС Windows 7

Якщо певний програмний продукт не потрібен, виконується його деінсталяція (видалення) засобами ОС або додатковими утилітами.






Оновлення складників сучасних ОС (Android, Windows 8, 10, Linux Ubuntu та інших) відбувається автоматично. Необхідною умовою для цього є наявність підключення комп'ютера до мережі Інтернет.

Для встановлення необхідних параметрів ОС на панелі керування слід обрати відповідну піктограму і перейти до налаштування опцій. На рис. 2.14 показано вихід на налаштування годинника, календаря, регіональних стандартів і мовних параметрів інтерфейсу для ОС Windows 7.

Перевіряємо себе

1. Що таке файл? 
2. Які типи інтерфейсу користувача застосовуються в сучасних ПК? 
3.  За якими основними ознаками класифікуються ОС ПК? 
4. Як поділяються ОС за кількістю одночасно виконуваних процесів? 
5. Які основні частини містить ОС ПК? 
6. Для чого призначені драйвери? 
7. Поясніть основні функції ядра ОС. 
8. Поясніть необхідність файлової системи. 
9. Для чого застосовується бібліотека системних функцій? 
10. Чи можна створити комп'ютер, який би не потребував ОС? 
11. Які дії виконує ядро ОС для запуску на виконання програми користувача? 
12. Які програмні засоби використовуються для розроблення програм користувача? 
13.  Скільки може бути варіантів подання команди копіювання файла в ОС Windows 7? 
14.   Чим відрізняється вміст дисків 3 і 4 (рис. 2.9) від вмісту інших?

Виконуємо

1.   Опишіть призначення і орієнтовний вміст кожного з компакт-дисків, зображених на рис. 2.9. Можна використати результати пошуку необхідних даних у мережі Інтернет. Обговоріть у класі результати ваших пошуків. 
2. Викличте (командою Ctrl+Alt+Del) Диспетчер задач Windows. Який з процесів виконується завжди? Поясніть чому.  

3. Викличте (командою Ctrl+Alt+Del) Диспетчер задач Windows. Запишіть кількість процесів, які виконуються. Запустіть на виконання кілька програмних засобів. Поясніть, чому збільшилась кількість виконуваних процесів. ✦

4. Визначте, які процеси належать запущеним вами на виконання програмам. ✦

**Практична
робота № 2**

Тема:	Конфігурування комп'ютера під потреби користувача
Мета:	Набути практичних навичок вибору апаратного і програмного забезпечення та його налагодження

Вказівки. Конкретні рекомендації можна навести лише щодо вибору монітора. Його слід обирати з найкращими характеристиками. Це має бути монітор рідинно-кристалічного типу з діагоналлю не менше 19 дюймів. Інші параметри залежать від наявності коштів.

Щодо інших пристроїв та їх характеристик можна навести лише загальні рекомендації. Насамперед слід визначити мету придбання комп'ютера. Зазвичай розрізняють таку мету використання ПК: для офісної роботи; для дому; для ігор; для дизайну.

1. У офісі комп'ютер застосовується переважно для роботи з текстом, таблицями, бухгалтерської і фінансової діяльності, для електронного листування та перегляду сторінок в Інтернеті. Для таких робіт вистачає бюджетного варіанта ПК з двоядерним процесором частотою близько 1,5 ГГц. Досить оперативної пам'яті 2 ГБ і жорсткого диска ємністю 200–440 ГБ. Достатня потужність блоку живлення 300 Вт. Може бути використана вбудована відеоплата. Для офісних робіт часто застосовують принтер і сканер.

2. Удома комп'ютер використовується для роботи в Інтернеті, нескладних комп'ютерних ігор, перегляду телевізійних передач і фільмів, роботи з мультимедіа. Тому потрібен більш потужний комп'ютер: з 2–4-ядерним процесором частотою близько 2 ГГц, оперативною пам'яттю 2–4 ГБ, жорстким диском 500–700 ГБ. Обов'язково потрібні привід CD або DVD-типу з можливістю запису, блок живлення не менше 400 Вт.

3. Для ігор бажано мати 4-ядерний процесор із частотою близько 3 ГГц, оперативною пам'яттю 4 ГБ, а жорсткий диск ємністю 1 ТБ. Комп'ютер має мати відеокарту досить високого рівня і звукову систему. Блок живлення потужністю не менше 450 Вт. Обов'язково необхідно мати привід DVD або Blu-Ray із можливістю запису.

4. Для дизайнерської роботи потрібен найпотужніший комп'ютер. Він має бути орієнтований на використання 64-бітних ОС. Процесор бажано мати частотою 3,3 Гц, а кількість ядер – не менше 4. Оперативна пам'ять – не менше 8 ГБ, вінчестер – 1 ТБ.

Виконання роботи

Учні класу об'єднуються в три-чотири групи, а також призначається журі у складі трьох-чотирьох учнів. Кожна група має обговорити і прийняти рішення: для чого можуть використовуватися комп'ютери, основні дані яких наведені в таблиці. За 5 хвилин протягом 15 хвилин по одному представнику від кожної групи обґрунтовують своє рішення, а члени журі оцінюють якість обґрунтування.

Характеристики комп'ютерів

Пристрої	Комп'ютери			
	№ 1	№ 2	№ 3	№ 4
Процесор	2,0 ГГц, 2 Cores, Cashe 1 MB	2,4 ГГц, 2 Cores, Cashe 2 MB	3,0 ГГц, 4 Cores, Cashe 4 MB	3,6 ГГц, 4 Cores, Cashe 8 MB
ОЗП	1 ГБ	2 ГБ	4 ГБ	16 ГБ
Вінчестер	300 ГБ	500 ГБ	1 ТБ	1 ТБ
Відеокарта	Вбудована		1 ГБ	2 ГБ
Привід	–	DVD+DW	DVD+RW	DVD+RW

Команди і журі залишаються в тому самому складі. Учитель пропонує кожній команді визначити склад і параметри комп'ютера учня і комп'ютера вчителя. За 5 хвилин протягом 15 хвилин представник кожної команди обґрунтовує свій варіант. Журі оцінює їхні доповіді й за підсумками двох виступів оголошує кращу команду. Остаточні підсумки підводить учитель.

2.6. Службове програмне забезпечення



Архівування даних. Операції над архівами. Форматування носіїв даних.

У сучасних персональних комп'ютерах можуть зберігатися різноманітні типи даних (графічні, звукові та інші), які потребують величезних обсягів пам'яті. З метою економії обсягу пам'яті зовнішніх пристроїв, на яких зберігаються дані, здійснюється їх стиснення.



Стиснення даних – це процес перекодування початкового файлу даних у новий файл меншого розміру, із якого можна відновити початковий.

Існують різні алгоритми стиснення даних. Один із найпростіших полягає у тому, що для кодування символів, які часто використовуються (наприклад, а, о, к), відводиться менше бітів, ніж для кодування символів, частота використання яких невелика. Алгоритми стиснення даних поділяються на алгоритми з **втратами** і **без втрат**.

- ✓ *Стиснення без втрат* – таке стиснення, за якого файл, відновлений зі стиснутого, повністю відповідає оригіналу.
- ✓ *Стиснення з втратами* – таке стиснення, коли у відновленому файлі є спотворення.

Стиснення без втрат застосовується для текстових і числових даних, програм, тому що відсутність навіть одного символу може призвести до неправильного тлумачення слова, збою виконання програми, а стиснення з втратами – для графічних, звукових і відеоданих.

- ✓ *Стиснення даних характеризується двома основними параметрами.*

1. Коефіцієнтом стиснення, який визначається відношенням обсягу стиснутого файлу до обсягу початкового файлу. Значення цього коефіцієнта міститься в межах від 1 (стиснення вже стиснутих даних) до 10 і більше.

2. Швидкістю стиснення, яка визначає тривалість стиснення і наступного відновлення даних з архіву.

Стиснення даних здійснюється за допомогою спеціальних програм. У ОС Windows XP і старших версіях для цього існують вбудовані програмні засоби. Вони виконують стиснення на тих зовнішніх пристроях, які підтримуються файловою системою NTFS.

- ✓ *Стиснення даних в ОС Windows 7 можна виконати так.*

1. У програмі **Провідник** виокремити об'єкти, які потрібно стиснути. Виокремимо, наприклад, на диску **F:** папку **СТАТТЯ_ОС** і файл **Стаття_БЕЗПЕКА**. Потім слід відкрити контекстне меню виділених об'єктів і виконати команду **Властивості**.

2. Відкриється вікно **Властивості**, зміст якого на вкладці **Загальні** подано на рис. 2.15.

Як бачимо з рис. 2.15, виокремлені об'єкти мають 2 папки і 39 файлів різних типів. На диску вони займають 33,4 МБ. У цьому вікні слід натиснути кнопку **Додатково...**. Відкриється вікно **Додаткові атрибути**, зображене на рис. 2.16.

У цьому вікні увімкнемо прапорці **Дозволити індексування вмісту цих файлів...** і **Стиснути вміст для заощадження місця на диску** й натиснемо кнопку **ОК**. У результаті відкриється вікно **Підтвердження змінення атрибутів**, зображене на рис. 2.17 (якщо воно не відкриється, то у вікні, зображеному на рис. 2.16, слід натиснути кнопку **ОК**, після чого воно має відкритися).

3. Увімкнути в цьому вікні перемикач для вкладених папок і файлів і натиснути кнопку **ОК**. Після цього починається процес стиснення, який залежно від обсягу виділених об'єктів може виконуватися від кількох секунд до десятків хвилин.

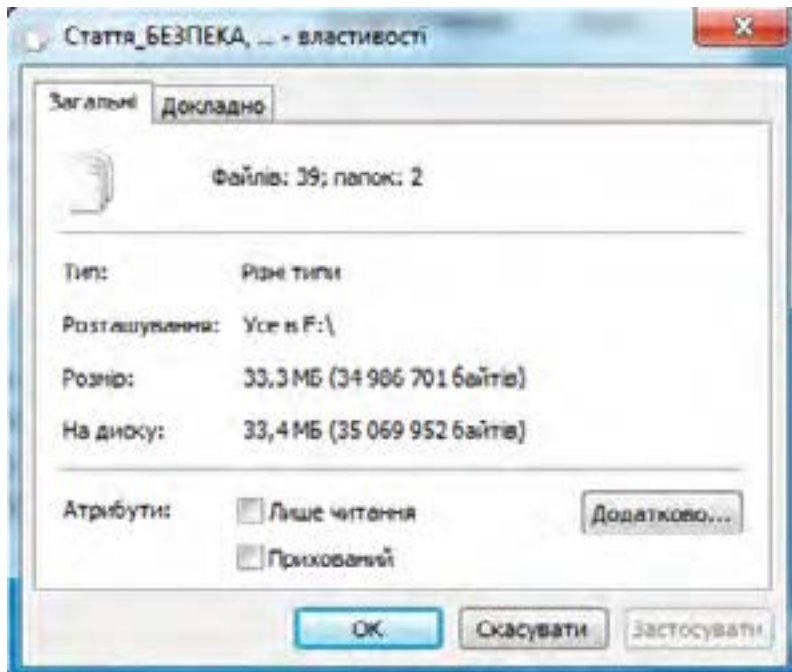


Рис. 2.15. Вікно властивостей виокремлених об'єктів

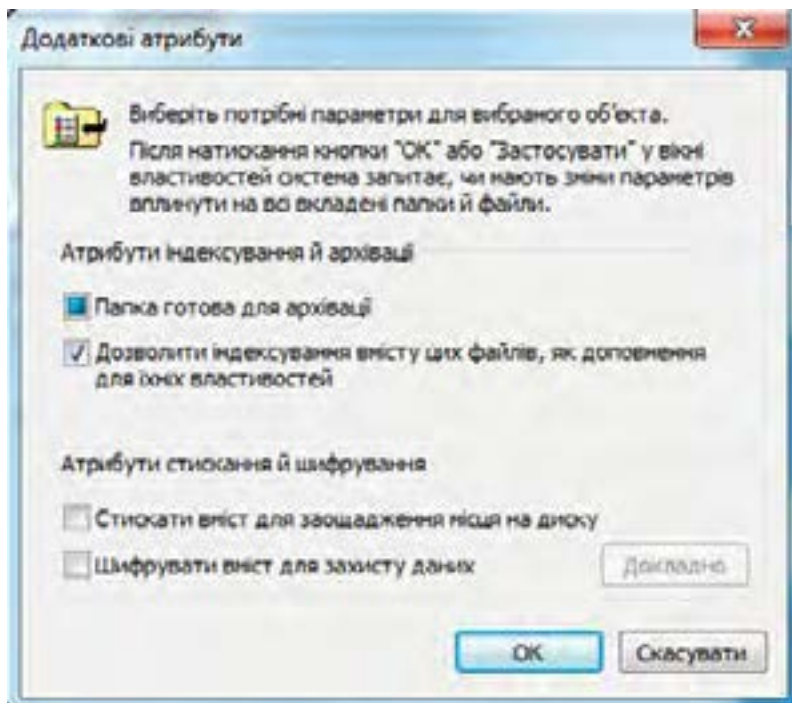


Рис. 2.16. Вікно Додаткові атрибути

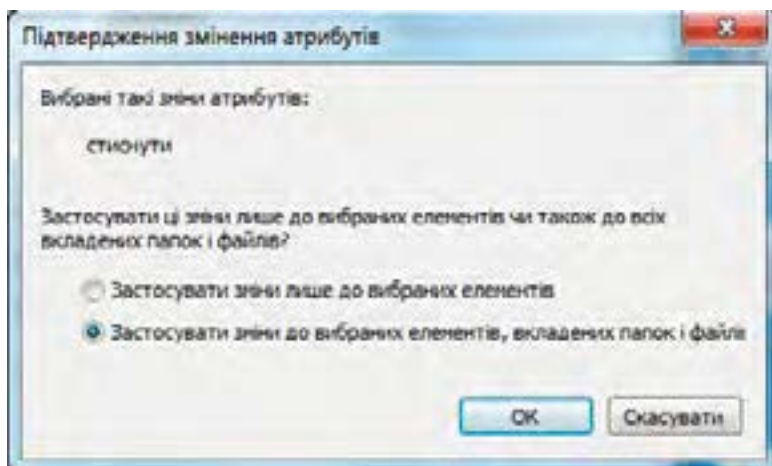


Рис. 2.17. Вікно для підтвердження змінення атрибутів

Зверніть увагу, що стиснуті об'єкти у вікні програми **Провідник** помічені іншим кольором. Для перегляду результатів стиснення слід виділити відповідні об'єкти, відкрити їх контекстне меню і виконати команду **Властивості** (зверніть увагу на те, що стиснуті об'єкти на диску займають 8,44 МБ, тобто вони стиснуті майже в чотири рази).

✓ Для скасування стиснення папок і файлів необхідно виокремити їх, відкрити вікно **Додаткові атрибути** (рис. 2.16), вимкнути в ньому прапорець **Стискати вміст для заощадження місця на диску** і натиснути кнопку **ОК**.

Окремим випадком стиснення файлів є їх архівування.

🔴 **Архівування** – це стиснення файлів і, якщо їх два і більше, об'єднання їх в один архівний файл з метою довготривалого його зберігання, а також для передавання даних комп'ютерними мережами.

Архівний файл має зміст, звідки можна дізнатися, які файли він містить. На кожний файл у архіві є дані про його ім'я, розмір, ступінь стиснення та інші.

Програми-архіватори виконують як стиснення, так і відновлення оригіналу файла з архіву. Останній процес називається **розархівуванням**. Тепер популярні архіватори WinRAR, WinZip, 7-Zip. Архіватори працюють не лише з файлами власного формату, але й з файлами інших форматів. Наприклад, архіватор WinRAR працює з файлами власного формату і з файлами формату ZIP.

Існують архіви, що саморозпаковуються. Для розпакування таких файлів не потрібно жодних додаткових програм. Ці архіви мають розширення **exe**. Їх доцільно застосовувати для пересилання мережею Інтернет, якщо невідомо, чи має кореспондент відповідну програму-архіватор.

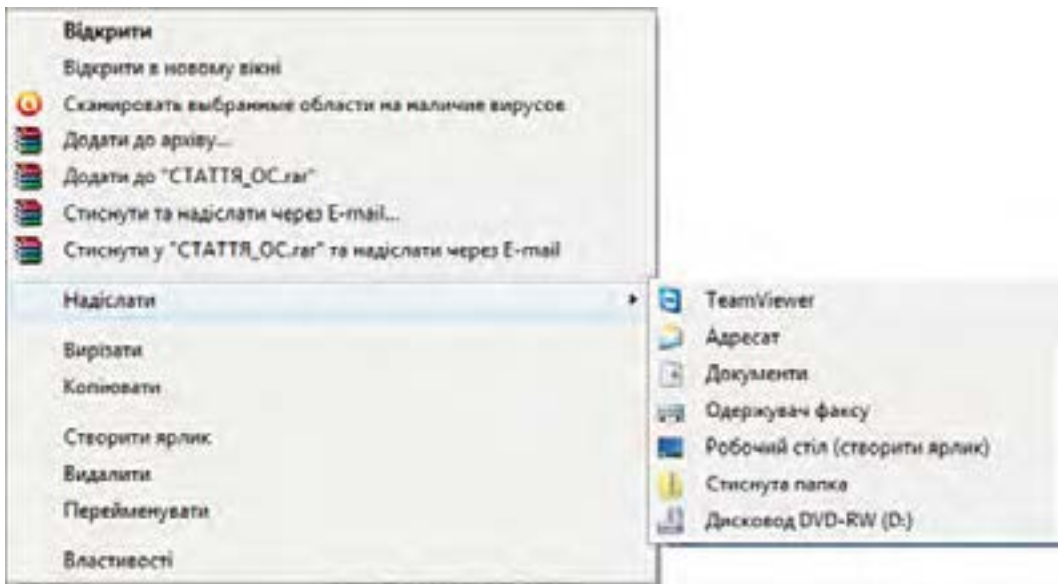


Рис. 2.18. Контекстне меню виокремлених об'єктів

✓ У ОС Windows 7 є вбудовані засоби архівування папок і файлів формату ZIP. Один із найпростіших методів створення архіву цими засобами наведений нижче.

1. У вікні програми **Провідник** виокремити папки та файли, які слід архівувати. Виокремимо, наприклад, на диску **F:** папку **СТАТТЯ_ОС** і файл **Стаття_Безпека**.

2. Відкрити контекстне меню виокремлених об'єктів, встановити курсор на пункт **Надіслати** й у підменю, що відкриється, виконати команду **Стиснута папка** (рис. 2.18). На диску **F:** з'явиться папка, назва якої збігається з назвою одного з об'єктів, виділених для архівування (для прикладу, який ми розглядаємо, ця папка має назву **Стаття_БЕЗПЕКА**).

У створену архівну папку можна додавати папки і файли, які під час копіювання стискаються. З архівної папки об'єкти можна копіювати у звичайну папку. Під час їх копіювання здійснюється їх розпакування. Порядок вилучення архівних папок практично не відрізняється від вилучення звичайних папок.

Існують також **багатотомні** архіви, тобто архіви, що зберігаються не в одному, а в кількох файлах. Така потреба виникає для передавання архівів електронною поштою, якщо вони завеликі.

Архіватор WinRAR має зручний інтерфейс, забезпечує високий ступінь стиснення і має добрі функціональні можливості. Для запуску програми WinRAR слід натиснути кнопку **Запустити** і виконати команди **Усі програми** → **WinRAR**. Відкриється вікно програми, фрагмент якого зображено на рис. 2.19.

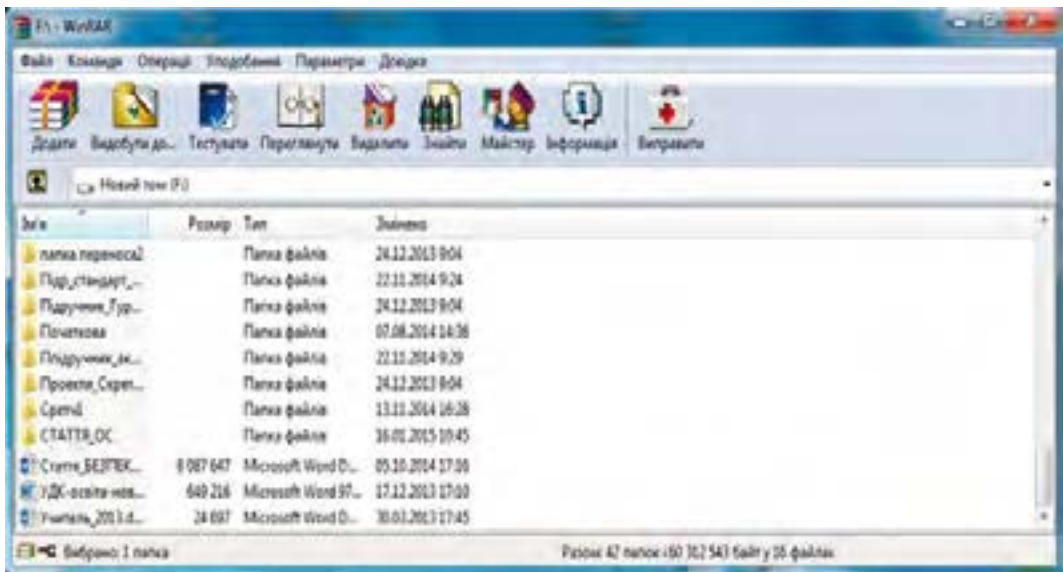


Рис. 2.19. Вікно архіватора WinRAR

Команди меню **Файл** призначені для керування папками, файлами, буфером обміну, а також для встановлення пароля. Команди меню **Команди** забезпечують додавання файлів до архіву, видобування файлів до зазначеної папки, видалення й перейменування файлів та виконання інших операцій. Меню **Операції** призначене для перевірки архіву на віруси, виправлення архіву, створення звіту тощо.

Для створення нового архіву потрібно у вікні програми WinRAR за допомогою команди **Змінити диск** меню **Файл** перейти до диска, в якому містяться об'єкти для архівування. Перейдемо, наприклад, до диска **F:**, у якому виділимо папку **СТАТТЯ_OC** і файл **Стаття_БЕЗПЕКА**, які будемо архівувати. Після цього слід натиснути кнопку **Додати** або в меню **Команди** виконати команду **Додати файли до архіву**. Відкриється вікно **Ім'я та параметри архіву** (рис. 2.20).

У цьому вікні можна ввести нове ім'я архіву або погодитися із запропонованим, вибрати формат архіву, метод стиснення (звичайний, найкращий тощо) та інші параметри. Уведемо, наприклад, ім'я *Перший*, а інші параметри залишимо за замовчуванням.

Після встановлення параметрів слід натиснути кнопку **ОК**, у результаті почнеться процес архівування. Після його завершення на диску **F:** з'явиться архівна папка *Перший.rar*.

Видобути файли з архіву можна різними способами. Наприклад, у вікні програми WinRAR знайти і відкрити архівну папку, для чого слід двічі клацнути її ім'я. Якщо відкрити папку *Перший.rar*, відкриється вікно, фрагмент якого зображено на рис. 2.21.

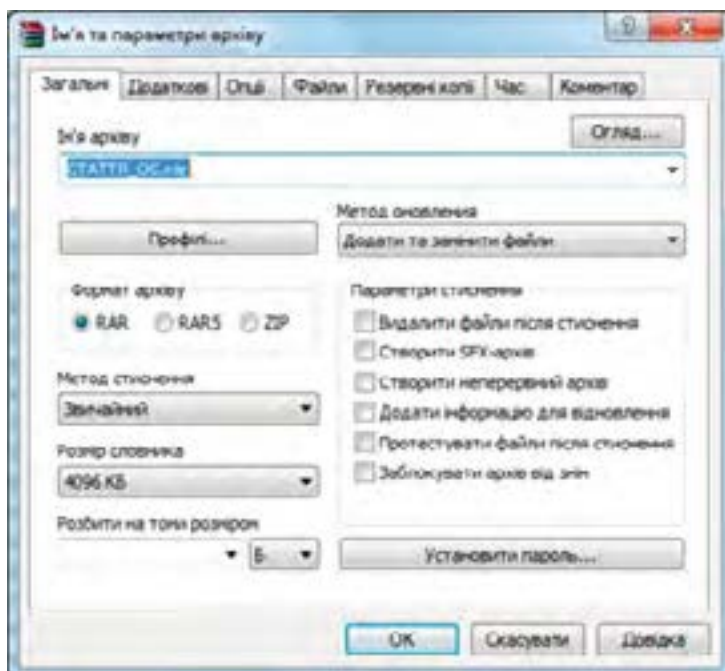


Рис. 2.20. Вікно Ім'я та параметри архіву



Рис. 2.21. Вікно архівної папки *Перший.rar*

У цьому вікні виділимо папки та файли, які необхідно видобути (виділимо, наприклад, папку СТАТТЯ_ОС) і натиснемо кнопку **Видобути до...** або в меню **Команди** виконаємо команду **Видобути файли до зазначеної папки**. У результаті відкриється вікно **Шлях та параметри видобування**, зображене на рис. 2.22.

У цьому вікні вводимо ім'я папки, до якої слід записати об'єкти з архіву, і натискаємо кнопку **ОК** (уведемо, наприклад, ім'я *Початкова* у кореневій папці диска **F:**).

Якщо тепер відкрити папку *Початкова*, то побачимо в ній розархівовану папку СТАТТЯ_ОС. Якщо видобування не виконається, то буде видано відповідне діагностичне повідомлення.

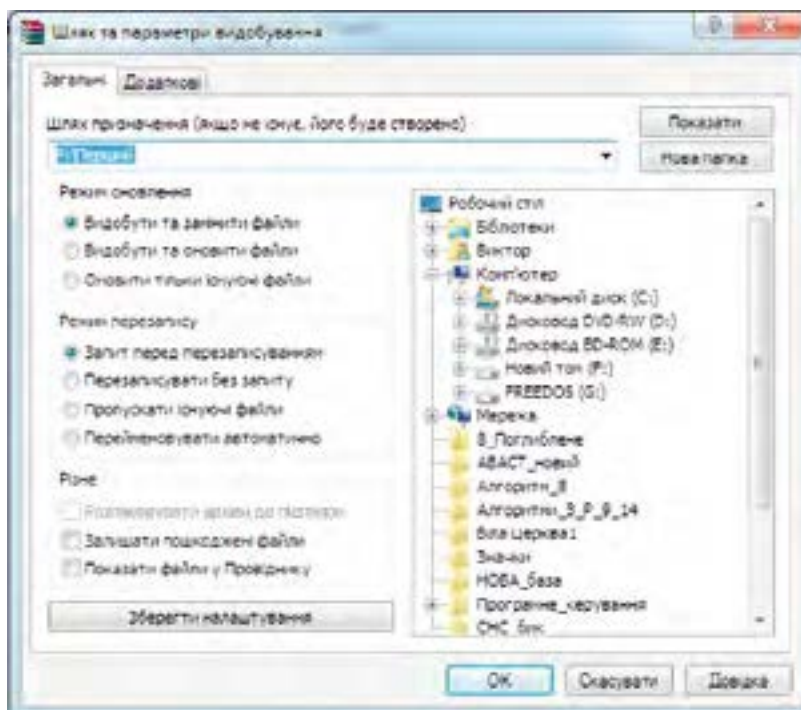




Рис. 2.22. Вікно Шлях та параметри видобування

Для створення саморозпаковуваного архіву треба у вікні **Ім'я та параметри архіву** програми WinRAR (див. рис. 2.20) увімкнути прапорець **Створити SFX-архів**.

Для створення багатотомного архіву слід у вікні **Ім'я та параметри архіву** вказати обсяг тому в полі **Розбити на томи розміром**. За замовчуванням кожний том отримує ім'я **ім'я_тому.partNNN**, де NNN – номер тому. Усі томи мають зберігатися в одній папці, а розпаковувати їх слід починаючи з першого.

 Для того щоб користуватися зовнішнім запам'ятовуючим пристроєм з файловою структурою (наприклад – записати на нього архів), потрібно цю структуру на ньому створити.

 *Процес підготовки носія до запису на нього файлів називається **форматуванням**.*

Форматування здійснюється спеціальною утилітою ОС або спеціалізованою програмою. Форматують розділи на жорсткому диску, флеш-накопичувачі та оптичні диски.

Розрізняють два типи оптичних дисків:

R – диски одноразового запису. Записані на них дані видалити неможливо;

RW – диски багаторазового запису. На них можна неодноразово видаляти й записувати нові дані.

Обсяги дисків CD – близько 700 МБ, дисків DVD – від 4,7 ГБ до 18,8 ГБ, дисків BD – від 15 ГБ до 100 ГБ.

✓ *Нові диски типу R і RW перед використанням потрібно обов'язково форматувати.*

ОС Windows, починаючи з версії XP, мають вбудовані програмні засоби, які забезпечують весь цикл робіт з дисками (форматування, копіювання, вилучення та інші операції).

Оптичні диски можуть бути відформатовані або для запису файлів даних, або для запису відео і звуку. Якщо диск відформатовано за першим варіантом, то потім його можна використовувати як звичайну флешку, тобто копіювати на нього файли, вилучати їх та виконувати інші операції за допомогою програми **Провідник**.

Далі описано порядок форматування оптичних дисків типу R і RW в ОС Windows 7 для запису на них файлів даних.

Після встановлення нового диска в дисковод має відкритися вікно **Автовідтворення**, зображене на рис. 2.23 зліва. Якщо воно не відкриється, то у вікні програми **Провідник** необхідно знайти піктограму диска і двічі клацнути на ній лівою кнопкою миші.

У цьому вікні натиснемо кнопку **Записати файли на диск**. Відкриється вікно **Записати диск**, зображене на рис. 2.23 справа. У цьому вікні вмикаємо прапорець **Як USB флеш-пам'ять**, вводимо ім'я диска, наприклад R1357, і натискаємо кнопку **Далі**. Після цього почнеться форматування диска, яке може тривати більше 10 хвилин.



Рис. 2.23. Початок підготовки диска до запису

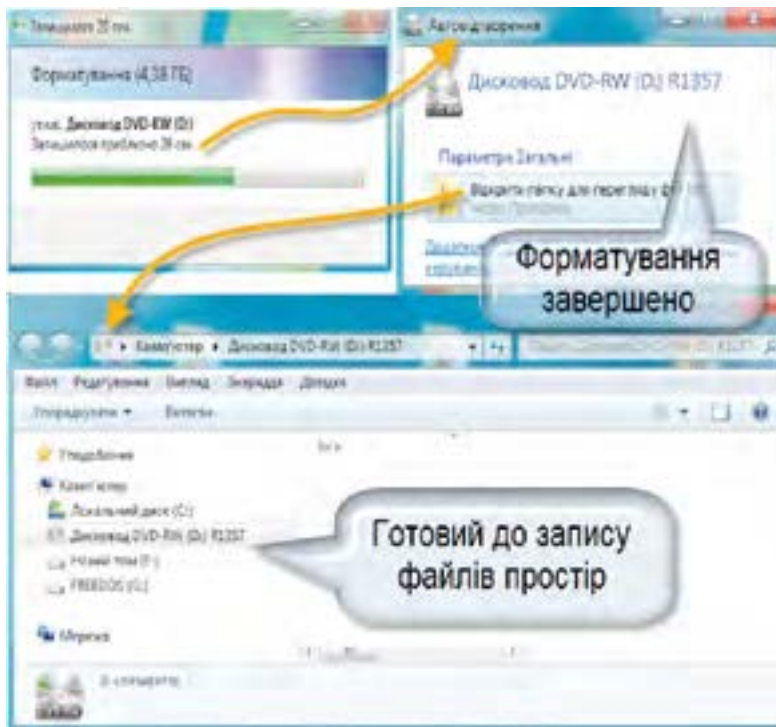


Рис. 2.24. Відображення процесу форматування диска

- ✓ Під час форматування не можна виймати диск і виконувати над ним інші операції.

Перебіг процесу форматування відображається на екрані. Після завершення форматування диска відкриється вікно, зображене на рис. 2.24 справа.

Якщо потрібно одразу після форматування записати на цей диск папки і файли, слід відкрити вікно програми **Провідник**, для чого можна натиснути кнопку **Відкрити папку для перегляду файлів**.

Від цього моменту виконання на диску операцій копіювання, переміщення, перейменування та видалення файлів і папок виконується за допомогою програми **Провідник** і принципово не відрізняється від таких самих операцій на жорсткому магнітному диску або флешці.

Якщо у попередньому сеансі роботи диск був відформатований, але запису файлів на нього не було, то після його вставлення у дисковод відкриється вікно, зображене на рис. 2.24 внизу. Після цього на диск можна записувати дані.

Форматування нового диска для запису на нього файлів даних можна виконати й таким способом. Після вставлення диска в дисковод у вікні програми **Провідник** відкрити контекстне меню диска, в якому виконати команду **Форматувати...** Відкриється вікно, зображене на рис. 2.25.


У цьому вікні вибираємо файлову систему за замовчуванням, стандартний розмір кластера, вводим назву диска, наприклад R1357, і натискаємо кнопку **Почати**. Після завершення форматування з'явиться вікно відповідного повідомлення, у якому слід натиснути кнопку **ОК**.

Для повного очищення оптичного диска RW слід у вікні програми **Провідник** відкрити його контекстне меню і виконати команду **Стерти диск**. Відкриється вікно Майстра очищення диска, в якому потрібно натиснути кнопку **Далі**. На екрані відобразиться процес очищення диска. Для повторного використання цього диска його слід відформатувати.



Рис. 2.25. Вікно форматування диска

Перевіряємо себе

1. Як можна перевірити результат стиснення файлів у ОС Windows 7? ▲
2. Як можна скасувати стиснення файлів засобами ОС Windows 7? ▲
3. З якою метою виконується архівування даних? ▲
4. Які архіватори нині популярні? ▲
5. Які архіви називають багатотомними? ▲
6. Які існують типи алгоритмів стиснення даних? ◆
7. Поясніть методику стиснення даних засобами Windows 7. ◆
8. Коли доцільно застосовувати архіви, що саморозпаковуються? ◆
9. Поясніть порядок створення архіву в ОС Windows 7. ◆
10. Поясніть сутність стиснення даних. ★
11. Назвіть основні параметри алгоритмів стиснення даних. ★
12.  Поясніть, як створюється архів у програмі WinRAR. ★
13. Як можна видобути файли з архіву WinRAR? ★
14. Які існують типи оптичних дисків? ▲
15. Чи можна на дисководі типу R працювати з дисками RW? ▲
16. Які обсяги пам'яті мають оптичні диски? ◆
17. Які операції можна виконувати на оптичних дисках? ◆

**Практична
робота № 3**

Тема: Архівування та розархівування даних

Мета: Набути практичних навичок використання архівів

1. У особистій папці створіть папку “Резервні копії”.
2. Скористайтеся послугою Пошук та знайдіть на комп’ютері три довільних малюнки формату *.jpg й три довільних малюнки формату *.bmp. Скопіюйте їх з вікна пошуку в папку “Малюнки” (ця папка має бути вже створена у вашій папці).
3. Заархівуйте папку “Малюнки” в папку “Резервні копії” (архів назвіть pictures), використовуючи максимальний метод стиснення.
4. Запишіть у зошит розмір архівованої та неархівованої копій. За допомогою калькулятора знайдіть різницю й запишіть її у зошит.
5. Створіть у папці з вашим прізвищем текстовий документ “Різниця”, в який переписіть результат із зошита.
6. Додайте до архіву pictures текстовий файл “Різниця”.
7. Створіть у своїй папці папку Резерв2.
8. Видаліть папку “Малюнки”. Розархівуйте з архіву pictures текстовий файл “різниця” і будь-який малюнок у папку Резерв2.
9. Знайдіть на комп’ютері та заархівуйте до архіву з назвою “Мікст” один файл формату *.mp3, два файли формату *.doc, три файли формату *.gif так, щоб:
 - А) архів не потребував для розкриття наявності архіватора;
 - Б) була додана інформація про відновлення;
 - В) архів можна було розкрити, тільки вказавши пароль (пароль обов’язково запишіть у зошит).Архів має бути розміщений у папці “Резервні копії”.
- Примітка:** для пошуку файлів доцільно використати пошукову систему та маски (* – будь-яке ім’я файла чи тип, або ?). Знайдені файли можна скопіювати у вашу папку й після цього вже архівувати.
10. Додайте до архіву pictures коментар “Це архів малюнків... (ваше прізвище, ім’я)”. Перевірте архів на помилки.
11. Заархівуйте всю свою папку в архів під назвою “Резервна папка Прізвище ім’я” на диск D:, при цьому не забудьте додати інформацію для відновлення.



СЛОВНИЧОК

Архівування – процес стиснення даних, що належать одному або декільком файлам.

Архітектура комп’ютера – спосіб побудови комп’ютера, з’єднання його окремих частин.

Блочно-модульна архітектура – спосіб побудови комп'ютера з виокремленням його частин таким чином, щоб можна було їх замінити.

Мікропрограма – програма, записана в постійному запам'ятовуючому пристрої процесора.

Операційна система – комплекс взаємозв'язаних програм, призначений для управління обчислювальними процесами (програмами), апаратними засобами, ефективного розподілу ресурсів комп'ютера між обчислювальними процесами, а також організації взаємодії користувача з комп'ютером.

Стиснення даних – процес перетворення даних зі зменшенням довжини коду.

Шина – набір провідників, об'єднаних певним призначенням.

Шина адреси – шина, подання двійкового коду на провідники якої викликає приєднання до шини даних комірки пам'яті з адресою, що відповідає двійковому коду.

Шина даних – шина, по якій з комірок внутрішньої пам'яті передаються коди даних і команд до центрального процесора і навпаки – від процесора до комірок внутрішньої пам'яті.

Шина управління – шина, по якій до внутрішньої пам'яті передаються команди від процесора і пристрою управління.



РОЗДІЛ 3. ОПРАЦЮВАННЯ ТЕКСТОВИХ ДОКУМЕНТІВ



Текстові документи, як і будь-які інші, зберігаються в електронному вигляді у файлах. Спосіб відтворення відомостей, що містяться в файлі, залежить від того, якими програмними засобами це робитиметься.



Формати файлів текстових документів. Створення, редагування та форматування списків, таблиць, колонок, символів, формул у текстовому документі. Недруковані знаки. Створення, редагування та форматування графічних об'єктів у текстовому документі. Стильове оформлення абзаців. Шаблони документів. Структура документа. Колонтитули. Розділи. Посилання. Автоматизоване створення змісту та покажчиків. Алгоритм опрацювання складного текстового документа. Робота з кількома документами.

3.1. Формати файлів текстових документів. Форматування символів і абзаців



Формати файлів, що містять текстові документи.

Сучасні текстові процесори, зокрема, LibreOffice Writer і Microsoft Word, дають можливість відкривати й зберігати документи в різних форматах. У діалоговому вікні **Збереження документа** є розкривний список **Тип файла**, де можна вибрати відповідний формат (рис. 3.1).

Як відомо, є універсальні формати текстових файлів, що можуть бути прочитані більшістю текстових редакторів, і оригінальні формати, які використовуються окремими текстовими редакторами.

✓ *Шаблон імені файла прийнято позначати так: *.*, де першою зірочкою позначається ім'я файла, присвоєне користувачем; друга зірочка вказує на тип файла.*

✓ *Найпоширеніші формати файлів текстових документів:*

Документ Word (*.doc) – основний формат текстового процесора Word (до версії 2003 включно). У цьому форматі документи зберігаються за замовчуванням.

Word (*.docx) – основний формат текстового процесора Word (починаючи з версії 2007). У цьому форматі документи зберігаються за замовчуванням.

Документ (*.odt) у відкритому форматі зберігання й обміну офісними документами, доступними для редагування (OpenDocument, ODF, скоро-

чено від Open Document Format for Office Application – відкритий формат документів для офісних застосунків). З травня 2006 року прийнятий як міжнародний стандарт ISO/IEC 26300.

Шаблон документа (*.dot) – документ, на якому можуть базуватись інші документи.

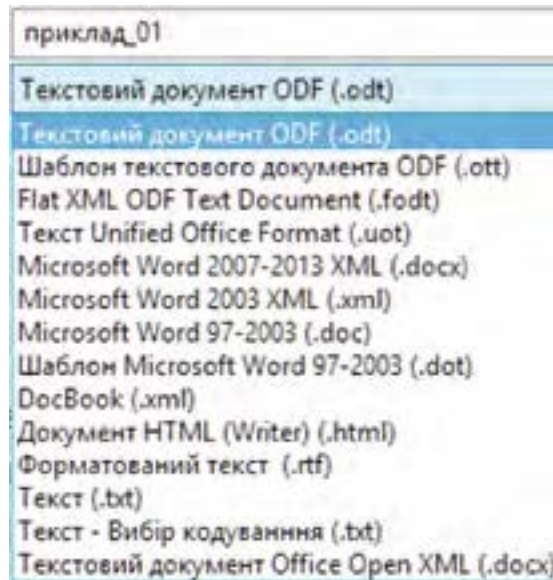


Рис. 3.1. Діалогове вікно Збереження документа з розкритим списком Тип файла

Текст у форматі RTF (*.rtf) – формат RTF (Rich Text Format – розширений текстовий формат), універсальний формат, який зберігає все форматування, перетворює коди керування на команди, які можуть бути прочитані та інтерпретовані у багатьох застосунках, водночас обсяг файла суттєво зростає.

Звичайний текст (*.txt) – найбільш універсальний формат, що зберігає текст без форматування; у текст вставляються тільки символи кінця абзацу. Цей формат використовують для збереження документів, які можуть бути прочитані в застосунках, що працюють у різних операційних системах.

Веб-сторінка (*.htm; *.html) (англ. *Hyper Text Markup Language* – ГіперТекстова Мова Розмічання) – формат Веб-сторінки; його використовують у разі розміщення документа в Інтернеті.

Формат PDF (*.pdf) (англ. *Portable Document Format*) – це електронний формат файла, створений на основі формату Post Script (формат, призначений для виведення документів на папір), який зберігає форматування документа. Використання формату PDF забезпечує незмінність вигляду документа під час перегляду файла в мережі або виведення на

друк. Тобто дані файла не можуть бути випадково змінені. Аналогічним є і формат XPS (англ. *XML Paper Specification*).

Для перетворення файлу текстового документа з одного формату на інший використовують спеціальні програми – конвертори. У текстових процесорах такі конвертори входять до складу програмного засобу.

Для надання текстовому документу вигляду, який покращує його сприйняття, використовують **форматування**.

Форматування тексту передбачає вибір виду, розміру та кольору символів, розміру абзацного відступу, відстані між рядками і абзацями, положення і вирівнювання абзаців відносно меж; нумерацію та маркування абзаців і рядків; встановлення чи заборону переносу слів; встановлення параметрів розрідження (ущільнення) та зміщення елементів тексту; встановлення параметрів колонок тощо. Всі основні засоби форматування розміщені в групах **Шрифт** і **Абзац** меню **Основне**.

✓ Для редагування формату символів потрібно виокремити фрагмент тексту (чи весь текст документа) та почергово (не знімаючи виділення) в меню **Основне** обрати за вимогами всі параметри форматування (розмір, тип, вид, колір, абзацний відступ та інші).

Форматування за зразком (MS Word 2007–2010).

1. Виокремити один із фрагментів тексту, який буде вважатися зразком.

2. Двічі клацнути кнопку **Формат за зразком** на панелі **Основне**.

3. Не натискаючи клавіші миші, перевести курсор на текст документа (він набуде вигляду щіточки, що означає запам'ятовування параметрів форматування).

4. Відшукати в тексті документа фрагмент, до якого мають бути застосовані такі ж параметри форматування, як і до першого зразка.

5. Встановити (не натискаючи кнопку миші) курсор на початок (або кінець) цього фрагмента.

6. Натиснувши й утримуючи натиснутою ліву кнопку миші, провести курсором по тексту фрагмента.

7. Відпустити ліву клавішу миші – фрагмент набуде такого ж вигляду, як перший відформатований фрагмент. Після виконання цих дій курсор все ще матиме вигляд щіточки – це означає, що засоби форматування можна застосувати і до інших фрагментів (рис. 3.2).

Форматування інших фрагментів виконується за схемою, описаною вище, але з тою різницею, що вже немає необхідності щоразу клацати кнопку **Формат за зразком**.

Після форматування фрагментів тексту за зразком першого, команда **Формат за зразком** слід відмовити натисненням кнопки **Формат за зразком**.

Рис. 3.2. Форматування фрагментів тексту за зразком

Якщо необхідно продовжити форматування фрагментів за зразком першого, курсор миші слід, не клацаючи, переміщати між фрагментами!

Форматування інших фрагментів виконується за схемою, описаною вище, але з тією різницею, що вже немає потреби щоразу натискати кнопку **Формат за зразком**.

Після форматування фрагментів тексту за зразком першого, команду **Формат за зразком** слід відмінити натисненням кнопки **Формат за зразком** або кнопки **Esc** на клавіатурі.

Однчасне форматування кількох фрагментів.

1. Виокремити один із фрагментів тексту.
2. Притримуючи клавішу клавіатури **Ctrl**, виокремити інший фрагмент.

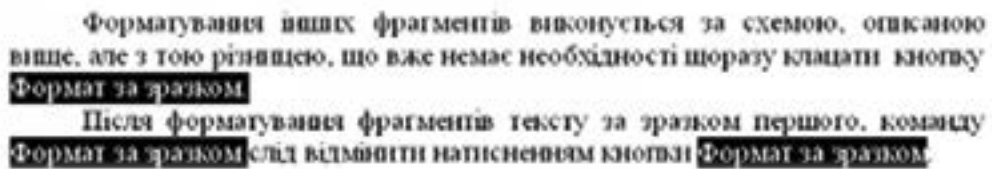




Рис. 3.3. Однчасне форматування кількох виокремлених фрагментів

3. Відпустити клавішу **Ctrl** та розшукати наступний фрагмент, виокремити його як і попередній (рис. 4.3).





4. Подібним чином виокремити всі фрагменти тексту документа, які мають одні й ті ж параметри форматування.

5. Перейти в меню **Основне** й обрати потрібні параметри.

Перевіряємо себе


1. Назвіть найпростіший формат текстового файлу. ▲
2. Що називають **Шаблоном документа**? ◆
3. Які кнопки стрічкового меню **Основне** або опції контекстного меню використовуються для швидкого надання значення міжрядковому інтервалу; абзацному відступу? ◆
4. Як називаються програми, що використовуються для перетворення файлу текстового документа з одного формату на інший? ★
5. У чому полягає форматування символів тексту? ▲
6. Назвіть способи форматування символів тексту. ◆
7.  Як можна виконати форматування фрагментів тексту за зразком? ◆
8.  Які формати текстового файлу доцільно використовувати для передавання документів, якщо невідомо, яким текстовим редактором користуватимуться для його розкриття і наступного редагування? ◆
9. Що називають **Шаблоном документа**? ◆

Виконуємо

1.  Створіть текстовий документ. ▲
2.  Збережіть створений документ у форматі Документ Word (*.docx) ▲
3. Збережіть створений документ у форматі Документ Word 97–2003 (*.doc). ▲
4. Збережіть створений документ у форматі RTF (*.rtf). ▲
5. Збережіть створений документ у форматі Звичайний текст (*.txt). ▲
6.   Порівняйте розміри отриманих файлів і вигляд документів, прочитаних із них. Зробіть висновки. ★
7. Збережіть документ у вигляді шаблону. Які частини документа доцільно зберігати повністю? ✨

3.2. Списки, таблиці, символи і формули

Особливим видом форматування абзаців є оформлення їх у вигляді списків.

 **Списком** називають сукупність взаємозв'язаних абзаців (елементів списку), які позначені спеціальними **маркерами** або **номерами**.

Списки бувають **марковані**, **нумеровані** та **багаторівневі**.

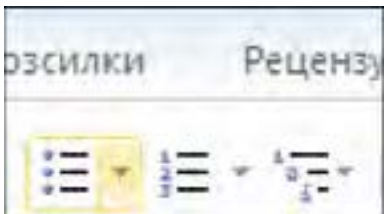


Рис. 3.4. Кнопки: **Маркери**, **Нумерація**, **Багаторівневий список**

Для швидкого нумерування або маркірування користуйтеся кнопками **Маркери**, **Нумерація**, **Багаторівневий список** вкладки **Абзац** стрічкового меню **Основне** або контекстного меню, яке виводиться після виділення тексту й натиснення правої кнопки миші (рис. 3.4).

Марковані списки (кожен абзац на початку помічається спеціальним символом-маркером ♣ ♦ ● ▪ ☺ -) застосовують для опису основних положень доповіді, дій користувача, переліку подій, властивостей об'єкта тощо (рис. 3.5).

У нумерованому списку на початку кожного абзацу зазначений його номер.

Порядковий номер абзацу в списку може бути заданий числом (наприклад 1, 2, 3) або літерою (наприклад а, б, в) (рис. 3.6). Такий список використовують для представлення інформації, якщо важлива послідовність елементів. Наприклад, опис покрокових процедур або список придбаних товарів у накладній.

У багаторівневому списку абзаци пронумеровані за їх ієрархічною структурою. Максимальна кількість вкладень елементів списку – дев'ять рівнів.

Для створення списку необхідно виділити абзаци, які потрібно зробити елементами списку, або зробити поточним абзац, із якого починається список.

Таблиці зазвичай використовують для впорядкування та наочного представлення даних. Дані в таблиці мають компактний вигляд і зручні для сприйняття.

Таблиця складається зі стовпців і рядків, на перетині яких містяться комірки. Комірки можуть містити текст, графічні зображення, числові значення, посилання на дані з інших документів.

Нестандартне застосування таблиць. За допомогою таблиці можна форматовувати документи, наприклад, розташовувати абзаци в кількох рядках, суміщати рисунок із текстовим підписом тощо. Таблиці використовують з метою особливого розташування тексту деяких документів, наприклад, під час створення бланків, буклетів, Веб-сайтів. Межі такої таблиці інколи роблять невидимими.

У текстовому процесорі є можливість:

створити порожню таблицю, а потім заповнити комірки;

перетворити наявний текст на таблицю.

Для розташування таблиці в тексті потрібно: вибрати меню

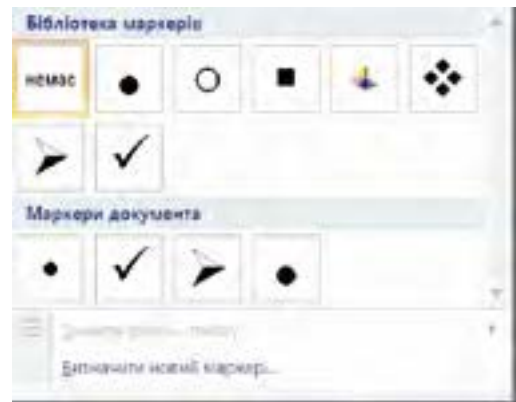
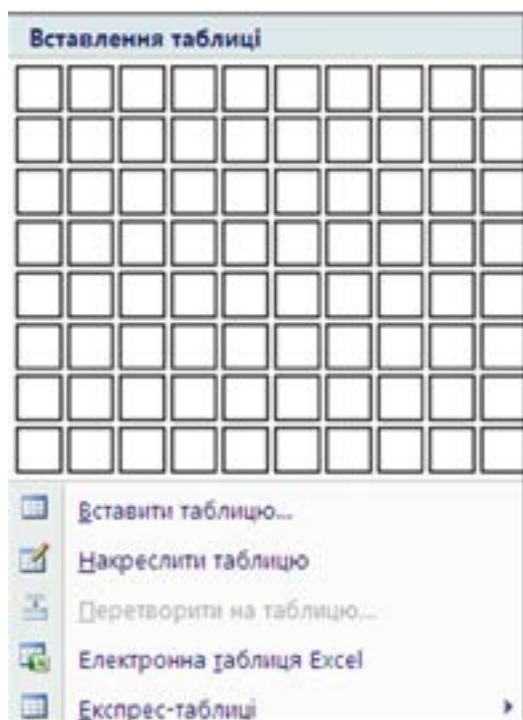


Рис. 3.5. Вікно **Бібліотека маркерів**



Рис. 3.6. Вікно **Бібліотека номерованих списків**



Вставлення, де обрати групу **Таблиці**. У списку, який відкрився, вибрати потрібну команду: **Вставити таблицю**, **Накреслити таблицю**, **Експрес-таблиці**, **Електронна таблиця Excel** (рис. 3.7).

Виконання команди **Вставити таблицю** розпочинається з діалогового вікна, в якому можна вказати кількість стовпців і рядків майбутньої таблиці або вибрати інший варіант додавання таблиці в документ.

Електронна таблиця Excel. Команда передбачає вставлення таблиці як об'єкта з програми Excel.

Рис. 3.7. Список **Вставлення таблиці**

Експрес-таблиці. Передбачається використання шаблону таблиці з колекції програми. Дані шаблону надалі можна замінити на потрібні (рис. 3.8).



Рис. 3.8. Вставлення експрес-таблиці

Засоби переміщення в таблиці та введення даних подано в таблиці.

Засоби переміщення по таблиці та введення даних

Результат	Дія
Переміщення в наступну комірку	Натисніть клавішу TAB. Якщо курсор міститься в останній комірці таблиці, натискання цієї клавіші додає до таблиці новий рядок
Переміщення в попередню комірку	Натисніть клавіші SHIFT+TAB
Переміщення в попередній або наступний рядок	Натисніть клавішу СТРІЛКА УГОРУ або СТРІЛКА УНИЗ
Початок нового абзацу	Натисніть клавішу ENTER
Додавання нового рядка в кінці таблиці	Натисніть клавішу TAB у кінці останнього рядка

Після вставлення таблиці (або в разі актуалізації таблиці, що вже існує) на екрані з'явиться меню **Табличні знаряддя**, де розташовані дві вкладки **Конструктор** і **Макет** (рис. 3.9).

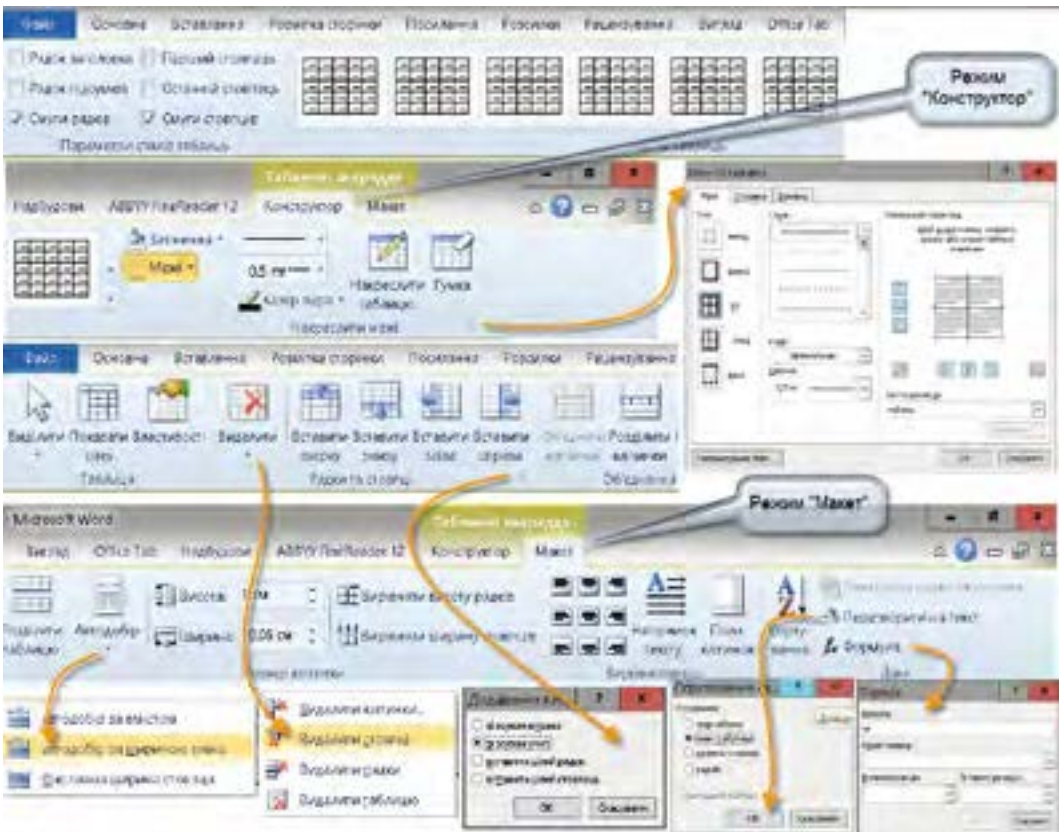


Рис. 3.9. Контекстні вкладки **Табличні знаряддя**

Застосування засобів вкладок **Конструктор** і **Макет** дає можливість змінювати вигляд таблиці.

Конструктор містить параметри стилів таблиці, накреслення таблиці за допомогою олівця та гумки. Такий спосіб зручний для створення складних таблиць.

Наприклад, **Накреслити таблицю**: на вкладці **Конструктор** необхідно обрати тип лінії, її товщину, колір; встановити курсор-олівець у документі; накреслити зовнішні межі таблиці; накреслити інші межі.

Для того щоб вилучити лінію (межу), слід вибрати гумку.

Макет містить засоби редагування таблиці (розташування на сторінці, обтікання текстом, встановлення меж та заливки) роботи зі стовпцями та рядками (об'єднання, розділення, розмір комірок або автодобрі за змістом, напрямок на вирівнювання тексту, сортування тексту за алфавітом або сортування числових даних).

Наприклад, **Перетворення таблиці на текст**: виділити таблицю та перейти на вкладку **Макет**; у групі **Дані** клацнути команду **Перетворити на текст**; відкриється вікно **Перетворення на текст**, де вказати знак розділювача, яким будуть позначені межі стовпців у тексті (рядки будуть розділені знаком абзацу).

✓ Раніше набраний текст можна перетворити на таблицю, якщо відкрити меню **Вставлення**, перейти до групи **Таблиця** та в меню, що спливає, вибрати опцію **Перетворити на таблицю**. Далі вказати в діалоговому вікні, які символи є розділювачами клітинок (знак абзацу, знак табуляції, крапка з комою або інший).

У деяких випадках текст документа необхідно розмістити в декілька колонок.

✓ Щоб перетворити готовий текст на таку форму, потрібно виділити необхідні абзаци документа і в меню **Розмітка сторінки** клацнути **Стовпці**, у списку, який відкрився, вказати кількість стовпців.

Якщо треба змінити параметри стовпців у цьому списку, обрати **Додаткові стовпці**. У діалоговому вікні, що відкрилося, можна обрати одну з п'яти стандартних конфігурацій стовпців, задати число стовпців, їх ширину і відстань між ними.

Уведення стандартних математичних формул або побудову власних формул можна здійснити за допомогою бібліотеки математичних символів у меню **Вставка** групи **Символи**, застосовувати **Формула** (рис. 3.10).

Для використання найбільш уживаних формул потрібно клацнути стрілку поряд із написом **Формула** і вибрати потрібну формулу. У документі з'явиться поле введення формули: "Місце для формули" і відповідне меню.

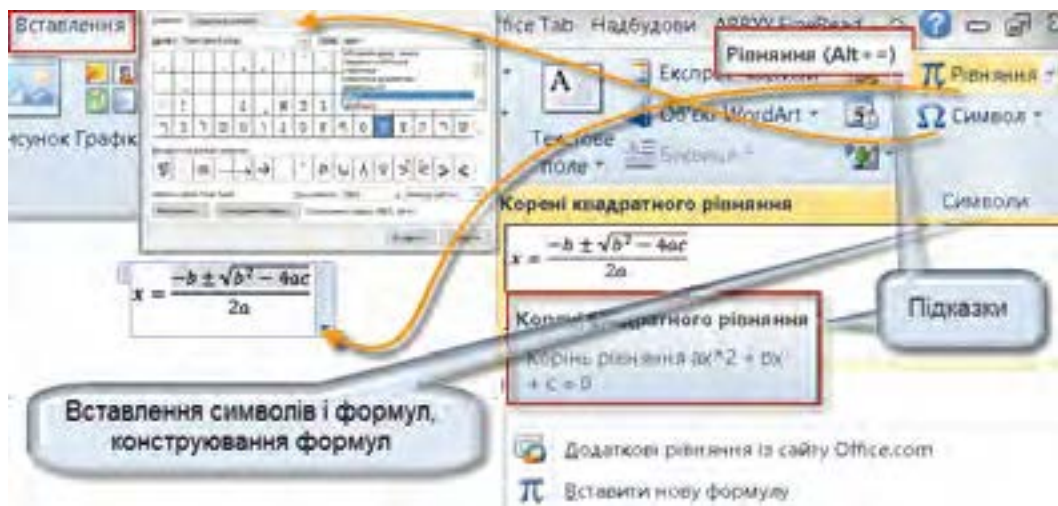


Рис. 3.10. Група Символи

Для створення формули, яка відсутня серед найбільш уживаних, слід перейти на пункт **Формула (Вставити нову формулу)**, після чого відкриється меню **Конструктор (Робота з формулами)**. Для редагування формули досить двічі клацнути її в тексті документа, після чого відкриється меню **Конструктор**, за допомогою якого можна внести зміни в формулу.

Слід враховувати, що для зміни типу, розміру, кольору та інших параметрів символів формули можна використовувати всі відомі засоби форматування та редагування текстового редактора.

Під час набору тексту ми не бачимо певних символів. До таких символів належать пробіл, перенесення, варіант перенесення, розрив рядка, кінець абзацу тощо.

✓ Щоб побачити всі приховані символи, слід здійснити необхідне налаштування: в меню **Основне** в групі **Абзац** натиснути **Показати всі знаки** (рис. 3.11).

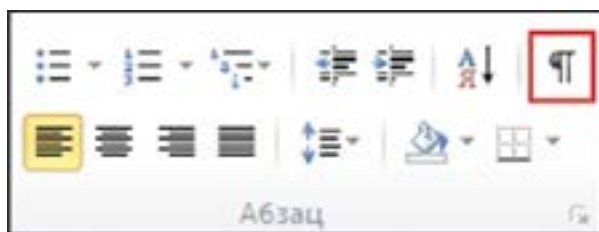





Рис. 3.11. Група Абзац

Перевіряємо себе

1. Що називають **Списком**? ▲
2. Вкажіть кнопки стрічкового меню **Основне** або контекстного меню, які використовуються для швидкого нумерування або маркірування. ▲
3. Як змінити нумерацію абзаців списку? ✦

4.  Як змінити рівень списку? ✦
5. Коли з'являється меню **Табличні знаряддя**? ▲
6. Що потрібно зробити для розташування таблиці у тексті? ✦
7. Що необхідно зробити, щоб перетворити готовий текст на колонки? ✦
8. Що передбачає створення символів тексту? ▲
9. Назвіть способи форматування символів тексту. ✦
10. Як можна виконати форматування фрагментів тексту за зразком? ▲
11. Як можна перетворити частину тексту на таблицю? ✦
12. Назвіть три різні способи розташування таблиці на сторінці. Як їх застосувати до вже створеної і заповненої таблиці? ★

Виконуємо

1.  Створіть документ “Що я знаю про комп'ютер”. ▲
2. Перетворіть частину тексту на таблицю. ✦
3. Частину тексту документа розмістіть у три колонки. ★
4. Частину тексту документа розмістіть у дві колонки. ✦
5. Одну з колонок перетворіть на список. ▲
6.  Виконайте пошук в Інтернеті матеріалів за запитом “Створення маркованого або нумерованого списку — Word”. Знайдений текст скопіюйте у новий документ. Виконайте його форматування (Основний текст – шрифт Times New Roman, 14, чорний, заголовки не форматовувати) ✦. Збережіть текст у файлі для наступного редагування.

3.3. Створення та редагування графічних об'єктів у текстовому документі

Графічний об'єкт – це рисунок, світлина, креслення, що зберігається на диску в графічному файлі. Word може використовувати графічні файли, створені різними програмами. Крім цього, під час інсталяції текстового процесора Word завантажується невелика бібліотека картинок, які можна вставляти у свої документи.

Крім стандартного набору картинок, які можна вставити в документ, Word дає змогу створювати власні графічні об'єкти: малюнки, окремі лінії, фігури тощо.

Створення власного графічного об'єкта складається з трьох основних дій:

1. Вставлення готових графічних об'єктів (ліній, стрілок, фігур і написів) у документ.
2. Пересування створених об'єктів по документу, зміна їх розмірів і пропорцій.

3. Зміна вигляду рисованих об'єктів: товщини лінії, кольору тексту, типу курсора стрілки та інших.

Щоб розпочати створення графічного об'єкта, потрібно в меню **Вставка** натиснути **Фігури**. Відкриється меню **Фігури**, в якому містяться кнопки графічних об'єктів: ліній, стрілок, просторових об'єктів та інших. За їх допомогою легко зображати лінії, стрілки, еліпси, прямокутники, кола, дуги, сектори та різні криві. Після створення графічного об'єкта його можна залити кольором або візерунком, змінити колір і тип ліній, збільшити або зменшити, перемістити, повернути або дзеркально відобразити об'єкти.

Редагування графічних об'єктів. Можна згрупувати декілька об'єктів так, щоб Word розглядав їх як єдине ціле. Наприклад, можна задати режими обтікання текстом як єдиного цілого згрупованих об'єктів (малюнків) або малюнка і підпису до нього. Для цього потрібно:

Виокремити декілька об'єктів, утримуючи затисненою клавішу Shift. Клацнути правою кнопкою миші виділені об'єкти й вибрати в контекстному меню команду **Групування**, а потім **Групувати**.

Після клацання графічного об'єкта правою кнопкою миші з'являється контекстне меню, яке містить основні команди, що забезпечують редагування, форматування, додавання тексту.

✓ *Імпорт графічних об'єктів здійснюється у двох варіантах.*

1. У вигляді цілого графічного файлу.
2. У вигляді частини графічного зображення, збереженого як фрагмент у іншому файлі.

✓ *Щоб вставити рисунок у документ, треба виконати такі дії.*

1. Установити курсор у місце документа, куди потрібно помістити рисунок.
2. Вибрати меню **Вставка** та натиснути **Рисунок**. Відкриється діалогове вікно **Додати рисунок**.
3. У разі потреби вибрати папку з потрібним файлом.
4. Натиснути на імені файла в списку або ввести ім'я файла в текстовому полі **Ім'я файла**.

Перш ніж працювати з об'єктом рисунок у документі, його слід актуалізувати. Для цього треба клацнути на рисунку ліву кнопку миші. Навколо виділеного об'єкта з'явиться вісім маленьких чорних квадратів, що називаються **маркерами розмірів**.

✓ *Для того щоб змінити розміри рисунка, слід виконати такі дії.*

1. Актуалізувати (виокремити) рисунок.
2. Встановити курсор на одному з маркерів розмірів. Курсор набуде вигляду стрілки, спрямованої в два боки.

3. Для зміни розмірів натиснути ліву кнопку миші і перетягнути маркер розміру доти, поки вид картинки не задовольнить Вас. Ви можете як збільшувати, так і зменшувати рисунки. Відпустіть кнопку миші.

4. Змінити розміри рисунка зі збереженням пропорцій.

5. Виділити рисунок.

6. Встановити курсор на кутовий маркер розміру та, утримуючи кнопку миші, переміщувати його до одержання потрібного розміру рисунка.

Перевіряємо себе

1. Що називають графічним об'єктом? ▲
2. Для чого використовується падаюче меню **Фігури**? ★
3. Назвіть основні дії створення власного графічного об'єкта. ★
4. Як здійснити редагування графічних об'єктів? ★
5. Як виконати додавання графічного об'єкта з зовнішнього файлу? ★
6. Як змінити розміри рисунка, імпортованого в документ? ★

Виконуємо

1. Дослідіть меню **Вставка**, з'ясуйте, які засоби призначені для вставлення малюнків та фігур. ▲
2. Вставте рисунок, який міститься в зовнішньому файлі. ★
3. Створіть власноруч графічний об'єкт, який складається з трьох об'єктів. ★
4. Згрупуйте створені об'єкти так, щоб їх можна було пересувати як єдине ціле. ★

3.4. Стильове оформлення абзаців

Під час підготовки великих за обсягом документів виникає низка завдань технічного характеру (форматування тексту, створення змісту, літературного покажчика тощо), на розв'язання яких доводиться витрачати багато часу.

✓ Для швидкого оформлення абзаців рекомендується використовувати стилі тексту.

❗ **Стиль тексту** – це набір значень властивостей тексту (в тому числі – рівень контуру, що використовується для автоматичної побудови документа), який має певну назву.

Текстовий процесор Word має певний набір стилів для форматування, який називають **бібліотекою стилів**.

Переглянути наявні в бібліотеці стилі можна на вкладці **Основне** ⇒ група **Стилі** (рис. 3.12).

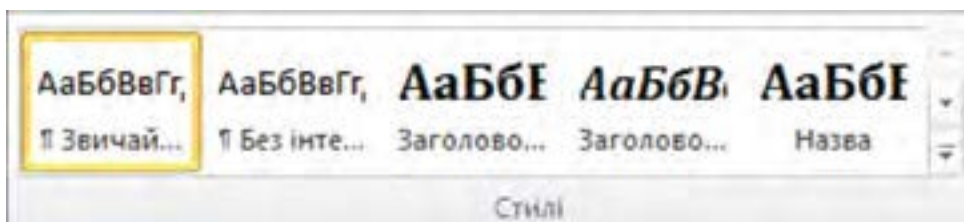


Рис. 3.12. Група Стили

✓ Розрізняють такі групи стилів:

– стиль символів задає формат символів (шрифт, розмір символів, накреслення, ефекти, колір тощо);

– стиль абзацу задає формат абзацу (спосіб вирівнювання тексту, позиції табуляції, міжрядковий інтервал, інтервал перед і після, може містити формат символів та інші);

– стиль таблиці задає формат таблиці (вигляд меж, заливки, вирівнювання тексту, шрифти тощо);

– стиль списку задає формат списків (спосіб вирівнювання, знаки нумерації або маркери, шрифти тощо).

Використання стилів дає змогу однією дією змінити значення кількох властивостей тексту. За збереження документа з ним автоматично зберігаються і застосовані стилі, тобто за подальших відкриттів документа навіть на інших комп'ютерах зовнішній вигляд документа не змінюватиметься.

Стили можна вилучати, перейменовувати, модифікувати. Досить внести зміни до певного стилю і весь документ відповідно автоматично переформатується (звичайно, якщо цей стиль у ньому застосовано).

Для застосування стилю символу, таблиці, списку потрібно виокремити цей текстовий об'єкт, а для застосування стилю абзацу досить встановити курсор у потрібний абзац



Рис. 3.13. Діалогове вікно Стили

тексту, переглянути список запропонованих експрес-стилів і вибрати потрібний.

Поряд із назвою стилю є значок, який показує тип цього стилю (стиль абзацу, стиль символів, стиль таблиці, стиль списків, пов'язаний стиль абзац і символ).

✓ Для того щоб побачити, як відобразиться текст після застосування до нього конкретного стилю, затримайте курсор на кнопці з зображенням цього стилю.

Якщо вибраний стиль влаштовує, потрібно клацнути на ньому мишею і зміни будуть збережені.

Бібліотеку стилів можна доповнювати власними стилями, створюючи їх на основі вже наявних.

Для цього виконати в меню **Основне**: клацнути групу **Стилі** ⇒ вибрати кнопку із зображенням стрілки, яка викличе діалогове вікно **Стилі** (рис. 3.13).

У вікні, яке відкриється, клацнути **Створити стиль**. З'явиться діалогове вікно **Створення стилю за допомогою форматування** (рис. 3.14).

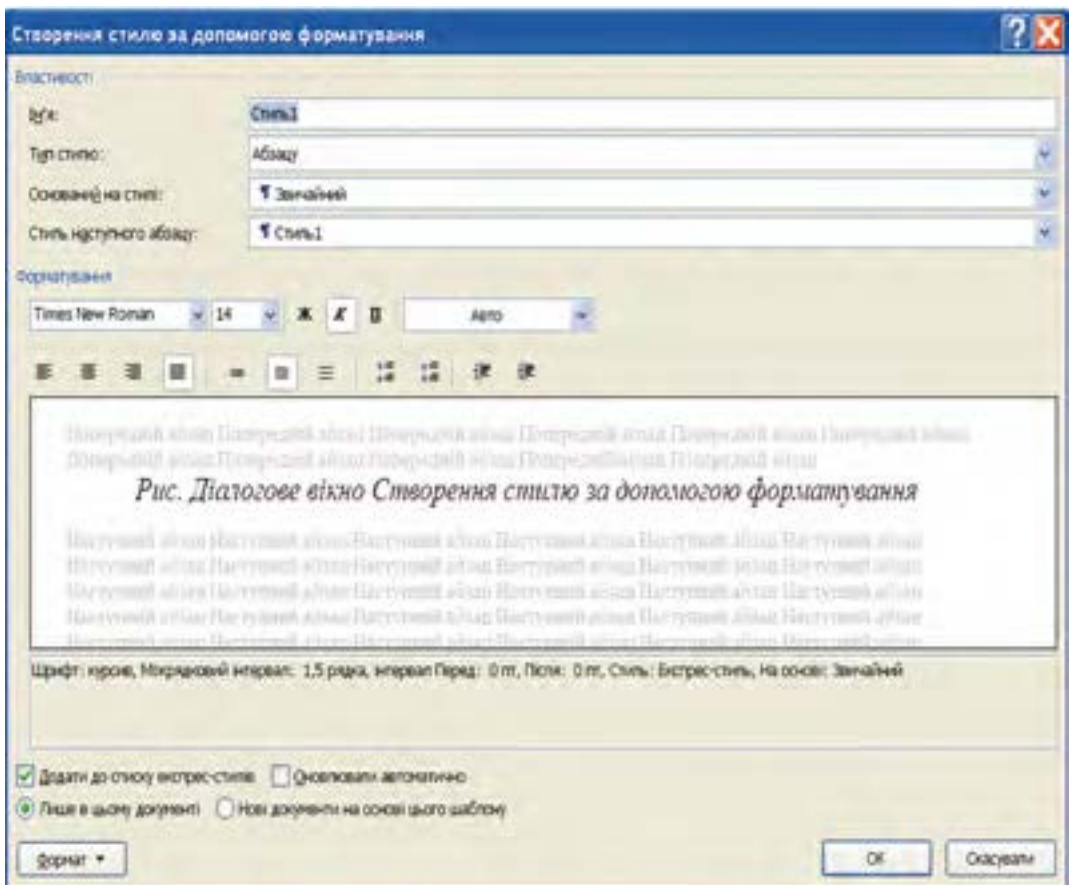


Рис. 3.14. Діалогове вікно Створення стилю за допомогою форматування

У відповідних полях діалогового вікна **Створення стилю за допомогою форматування** потрібно ввести назву нового стилю й вибрати його тип. У полі **Основою на стилі** вибрати наявний стиль, на основі якого буде створено новий. Для встановлення формату нового стилю потрібно

розкрити список **Формат** та вибрати об'єкти, властивості яких потрібно змінити (рис. 3.15).

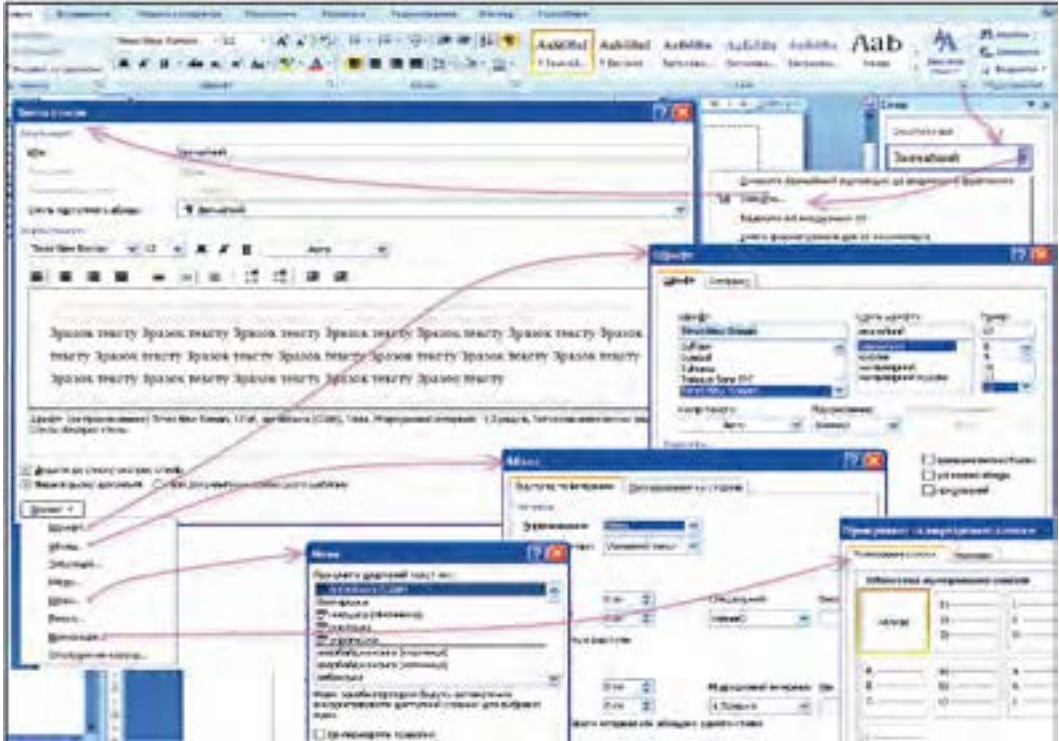


Рис. 3.15. Можливі дії з налагодження стилів

Якщо встановити позначку прапорця Додати до шаблону, то новий стиль діятиме не тільки в активному вікні, а й в усіх документах, які створені на основі цього шаблону. Після створення новий стиль буде доданий до бібліотеки стилів.

У подальшому його можна використовувати як і будь-який, що є вбудованим у програму.

Створення документа починається з вибору шаблону.

✓ **Шаблон документа** – це відформатований певним чином документ-заготовка, що використовується як зразок для створення нових документів.

Більшість сучасних текстових процесорів має бібліотеку шаблонів, за якими можна оформити звичайний документ, діловий лист, звіт, заяву, резюме тощо. Користувач може обрати потрібний шаблон і створити на його основі власний документ.

Для того щоб створити документ на основі шаблону, який міститься в списку шаблонів, слід у меню **Файл** натиснути **Створити**, далі зі списку шаблонів обрати тип шаблону.



Якщо наявні шаблони не задовольняють вимоги користувача, можна створити власний шаблон або модифікувати (змінити) наявний. Після внесення змін до шаблону його слід зберегти з новою назвою як **шаблон**. За замовчуванням документи зберігаються у папці, вказаній у налаштуваннях програми.

Параметри, що зберігаються у шаблоні, визначають характеристики документів, що ґрунтуються на цьому шаблоні.


✓ *Шаблон документа може містити такі елементи:*

- написи та рисунки, які мають з'являтися у кожному документі (колоннитули, поля для вставлення дати, часу та відомостей про автора, назва документа, стереотипний текст і емблеми компаній);
- поля сторінки та інші параметри макета сторінки;
- стилі;
- макроси WordBasic;
- текст і рисунки, збережені як елементи списку автотексту;
- спеціальні панелі інструментів, меню та сполучень клавіш.

Перевіряємо себе

1. Що називають стилем тексту? ▲
2. Текстовий процесор Word має певний набір стилів для форматування. Що називають бібліотекою стилів? ▲
3. Яким чином можна переглянути наявні в бібліотеці стилі? ▲
4. Які розрізняють групи стилів? ▲
5. Що потрібно зробити, щоб побачити, як відобразатиметься текст після застосування до нього конкретного стилю? ★
6.  Чи можна доповнювати **Бібліотеку стилів** власними стилями? Як це зробити? ▲
7.  Що треба зробити, щоб новий стиль діяв не тільки в активному вікні, а й в усіх документах, створених на основі цього шаблону? ★
8. Що називають шаблоном документа? ▲
9. Які дії необхідно виконати, щоб створити документ на основі шаблону, що міститься в списку шаблонів? ★


Виконуємо

1.  Наберіть текст документа з назвою “Що я знаю про комп’ютер”, або використайте вже створений. ★
2. Для заголовка тексту створіть власний стиль, використовуючи діалогове вікно **Створення стилю**: розмір 16, Cambria, підкреслення, розташування по лівому полю, всі символи прописні. ★

3. Для основного тексту створіть стиль: розмір 12, Cambria, курсив, розташування по ширині, перший рядок має відступ – 2. ★
4. За рекомендацією вчителя дайте назву новому стилю. ▲
5. Використовуючи позначку прапорця **Додати до шаблону**, залучіть новий стиль не тільки для активного вікна, а й для всіх документів, які будуть створені на основі цього шаблону. ★
6. Збережіть створений документ як шаблон документа. ★


3.5. Структура документа. Розділи. Колонтитули


Навчаючись працювати в середовищі текстового процесора, ви зрозуміли, що будь-який текстовий документ складається з відповідних частин – розділів, які своєю чергою можуть бути поділені на менші частини – підрозділи або параграфи. Найпростіший документ складається з різних розділів.


 **Розділом** називатимемо частину документа, об'єднану за функціональним змістом.

Розуміння структури документа дає можливість грамотно його оформити і переформатувати в разі потреби, а уявлення про властивості елементів текстового документа і є основою форматування.


Оскільки одні розділи є частинами інших (документ складається з параграфів, параграфи складаються з пунктів), то розділи розрізняють за рівнями. Розділ, що входить до складу іншого, є нижчим від нього на рівень. Проводячи узагальнення, можна сказати, що весь текстовий документ – це розділ 1-го рівня, розділи, з яких він складається, – розділи 2-го рівня і так далі.

 Назви розділів прийнято називати заголовками відповідно до їх рівня: **Заголовок 1-го рівня**, **Заголовок 2-го рівня** і так далі.

 *Побудова структури спрощує подальшу роботу з документом – від створення змісту до підготовки його до будь-якої публікації.*

 **Зміст** документа складається із назв розділів різного рівня – заголовків.

Структура текстового документа формується не тільки за розділами. Кожний абзац можна класифікувати за тим змістом, який він відображає.

 *Розрізняють три типи функціональних одиниць, або структурних елементів текстового документа: **розділи**, **абзаці** та **символьні структурні елементи**.*

Символьними структурними елементами називають речення, які не є абзацами, і навіть окремі слова.

Виокремлення й іменування структурних елементів тексту забезпечує структурування документа, що допомагає краще його сприймати.

Для правильної структуризації документа всі структурні елементи тексту мають бути чітко визначені. Кожен з них має власне форматкування (стиль), яке описується лише один раз.

Крім цього, формування структури дає змогу легше орієнтуватись у великому документі. У структурованому документі можна досить швидко пересуватися між розділами. Це має назву **навігація структурою**.

Для позначення структурних елементів у списку стилів є спеціальні стилі, які мають назву структурних елементів.


 **Стилі:**


– Заголовок 1 використовується для позначення заголовків до основних розділів;

– Заголовок 2 – позначення заголовків другого рівня;


– Заголовок 3 – позначення заголовків третього рівня;

– Звичайний – позначення основного тексту документа.

 *Правильне структурування документів важливе не тільки для користувачів, які безпосередньо працюють з ними, а й для людей, що користуються пошуковими системами та системами автоматичного обліку електронної документації.*

 *Якщо структурований документ буде розташований у мережі Інтернет, то пошуковий робот може більш точно його класифікувати і внести до каталогу обліку ресурсів коректну інформацію.*

Важливою частиною будь-якого документа є **колонтитул**.

 **Колонтитул** – це текст і/або зображення, що розташовується внизу або вгорі кожної сторінки документа.

Залежно від місця розташування (на верхньому або на нижньому полі сторінки) колонтитули бувають верхніми та нижніми. Верхня або нижня частина сторінки називається зоною колонтитула і використовується для введення однакового тексту (назви книжки, імені автора, номера сторінки, логотипа тощо) на кожній сторінці (або парних/непарних сторінках) документа. Відстань від краю сторінки до колонтитула можна змінювати.

Допускається створювати особливий колонтитул або взагалі його не створювати для першої сторінки документа чи певного розділу. Можна також створювати різні колонтитули для парних і непарних сторінок розділів або документа.

 **Для створення колонтитула:**

– виконайте подвійне натиснення лівої кнопки миші у верхньому або нижньому полі сторінки, тобто в місці, де має бути розташований колонтитул;

- наберіть текст колонтитула або вставте рисунок;
- для повернення до роботи з основним текстом документа двічі клацніть лівою кнопкою миші на робочому полі документа.

✓ *Поки на екрані видно зону колонтитула, основний текст документа редагувати неможливо.*

Колонтитул може містити номер сторінки.

✓ *Вставлення номерів сторінок здійснюється зі стрічкового меню:*

клацнути вкладку **Вставлення** та обрати групу **Колонтитули**; клацнути команду (кнопку) **Номер сторінки**, в списку, що відкрився, обрати вигляд номера сторінки. Із цієї ж групи можна також вибрати готові колонтитули (кнопки **Верхній колонтитул** і **Нижній колонтитул**) та виконати їх налагодження і редагування.

✓ *Для видалення колонтитула:*

виберіть розділ, який містить колонтитул, що підлягає видаленню, та подвійним клацанням лівої кнопки миші перейдіть в область колонтитула; після цього виконайте видалення тексту колонтитула відомими вам способами.

Перевіряємо себе

1. Що ви розумієте під “структурою” текстового документа? ▲
2. Що називають розділом? ▲
3. Як прийнято називати назви розділів? ◆
4. Із чого складається зміст документа? ★
5. Які розрізняють типи функціональних одиниць або структурних елементів текстового документа? ★
6. Яку частину тексту називають “символьними структурними елементами”? ★
7. Який елемент стрічкового меню **Вигляд** містить прапорець **Область переходів**? ◆
8. Навіщо використовується поле введення **Пошук у документі** засобу **Навігація**? ★
9. Який засіб має назву “Навігація”? ▲
10. Як називають текст та/або зображення, що розташовується внизу або вгорі кожної сторінки документа? ▲
11. Як можна створити колонтитул? ★
12. Які дії потрібно виконати для видалення колонтитулів? ▲

Виконуємо

1. Відкрийте раніше створений документ “Що я знаю про комп’ютер”. ▲
2. Виокремте його частини, які вважатимуться назвами розділів різних рівнів. ✦
3. Позначте їх відповідними стилями з меню Стили. ★
4. Перегляньте структуру документа, використовуючи **Область навігації**. ▲
5. Вставте нижній колонтитул – своє ім’я та прізвище. ✦
6. Вставте верхній колонтитул – номер школи та населений пункт. ✦
7. Збережіть документ у Документ Word (*.doc). ▲

3.6. Посилання. Автоматизоване створення змісту та покажчиків



Кожний багатосторінковий документ обов’язково має спеціальний список, який містить назви структурних елементів, – зміст.



Зміст документа – впорядкований перелік заголовків і підзаголовків. Для автоматичного створення змісту документа слід попередньо застосувати вбудовані стилі під час форматування до заголовків усіх рівнів.



Посилання – об’єкт у документі, за допомогою якого можна перейти до іншого фрагмента цього ж документа або до іншого документа.

Для побудови змісту: меню **Посилання** клацнути **Зміст** і вибрати потрібний стиль змісту. Після цього вказати потрібні значення властивостей (кількість рівнів, заповнювач тощо) (рис. 3.16).



Рис. 3.16. Вбудовані стилі для створення змісту

✓ Якщо стилі заголовків за рівнями в тексті не були визначені, то слід спочатку здійснити структурування в меню **Основне** в групі **Стилі**. Інакше зміст не буде побудовано.

Автоматизоване створення змісту та покажчиків. Предметний покажчик – це список слів (термінів), які вживаються в текстовому документі. Такий список для зручності пошуку слів вбудовується зазвичай у алфавітному порядку в кінці чи на початку документа. Покажчик містить номери сторінок, на яких ці слова згадуються в тексті документа.

✓ Для побудови **Предметного покажчика** потрібно.

1. Встановити курсор у місце документа, де буде розташовуватися **Предметний покажчик** ⇒ обрати меню **Посилання**, у вікні, яке відкривається, клацнути **Позначити**.

2. Відкриється вікно **Визначення елемента** покажчика, де в рядку **Основний** набрати ключове слово (чи кілька слів), яке є в тексті документа.

3. У рядку **Додатковий** набрати додатковий текст (це можуть бути ключові слова в різних відмінках).

4. Натиснути **Позначити** та, не закриваючи це вікно, почергово внести всі ключові та додаткові фрагменти тексту.

5. Після закінчення натиснути **Закрити** (з'явиться замість **Скасувати**).

6. Усі позначені елементи **Предметного покажчика** вставляються в документ як поля **ХЕ** (елементи предметного покажчика), вони оформляються як приховані знаки. Для того щоб їх приховати, потрібно натиснути кнопку **Недруковані знаки ¶** в меню **Основне**.

7. Обрати меню **Посилання**. У вікні, яке відкриється при потребі, обрати формат **Предметного покажчика** та клацнути **ОК**.

8. Предметний покажчик буде вставлено за положенням курсора.

Перевіряємо себе


1. Що називають **Посиланням**? ▲
2. Що називають **Змістом документа**? ✦
3. Які дії слід виконати для автоматичного створення змісту документа? ★
4. Що називають **Предметним покажчиком**? ▲
5. Послідовність яких дій необхідно виконати для побудови **Предметного покажчика**? ★

Виконуємо


1. Відкрийте документ “Що я знаю про комп’ютер”. ▲
2. У меню **Посилання** клацніть **Зміст** і виберіть стиль змісту через засіб **Вставити таблицю змісту**. ✦

3. У вікні, що відкрилося, виберіть параметри змісту. ▲
4. Перегляньте структуру документа, використовуючи **Область навігації**. ▲
5. У кінці документа, користуючись вказівками, які є в параграфі, створіть предметний покажчик із 10 слів. ★
6. Збережіть документ. ▲

3.7. Опрацювання складного текстового документа

 Документ, який розподілено на декілька розділів або параграфів, можна вважати складним.

Є багато прийомів, використання яких спростить його форматування та редагування. Наприклад, для переходу до того чи іншого заголовка в документі досить встановити курсор миші на відповідний заголовок у його змісті та натиснути клавішу Ctrl на клавіатурі. Після цього клацнути лівою кнопкою миші заголовок – виконається перехід на відповідний заголовок у документі.

 *Якщо в текст документа внесено зміни (вставлення/вилучення сторінок, зміна заголовків (їх кількість і текст)), їх досить легко відобразити в змісті таким чином:*

1. Внести зміни до заголовків та (чи) зміни кількості сторінок, тексту, назв тощо.

2. Перейти на поле змісту й клацнути правою кнопкою миші – відкриється контекстне меню.

3. В контекстному меню клацнути **Оновити зміст і обрати Оновити цілком** – ОК.

У Microsoft Word є декілька засобів, що дають змогу швидко переглянути документ і внести до нього зміни. Один із них:

1. Клацнути меню **Вигляд**.

2. Установити позначку в полі **Область навігації** – відкривається вікно **Навігація**. У цьому вікні доступні три режими:

- Перегляд заголовків у документі.
- Перегляд сторінок документів.
- Перегляд результатів поточного пошуку.

Після встановлення позначки в полі **Область навігації** й обрання режиму “Перегляд заголовків у документі” вздовж лівого поля робочої частини вікна документа відкриється вертикальна область, у якій відображатиметься структура документа (заголовки, підзаголовки) (рис. 3.17).

Для перегляду тексту документа досить виконати перехід, клацаючи відповідні заголовки.

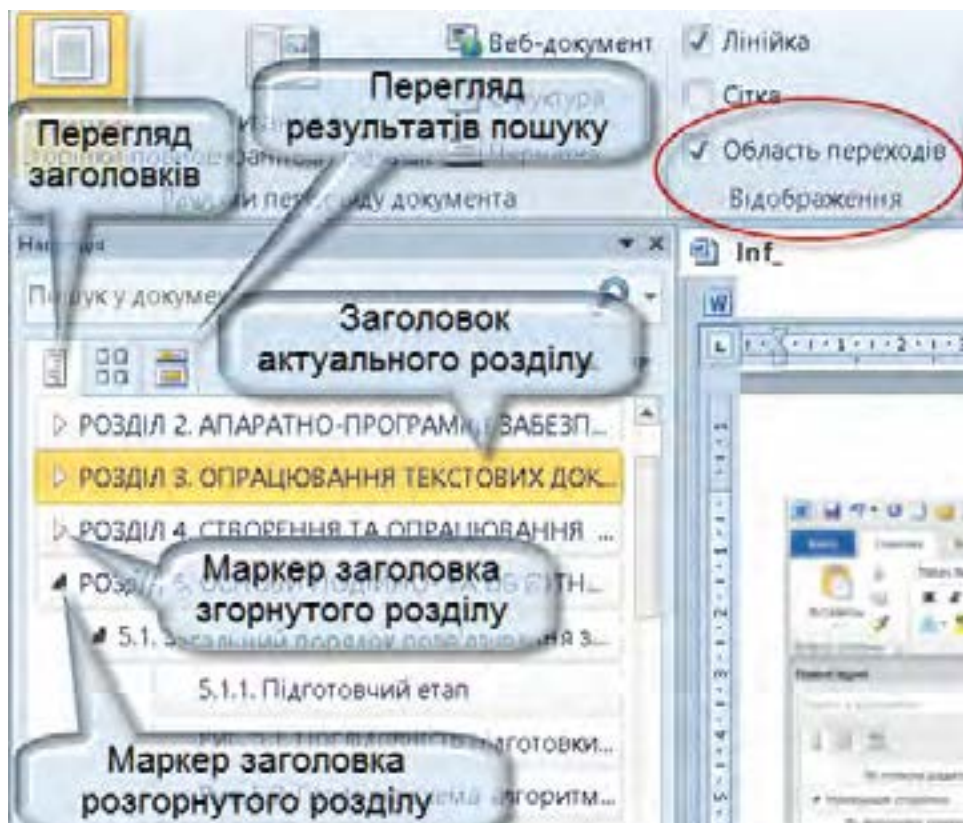


Рис. 3.17. Перегляд документа з використанням **Область навігації**

✓ У пакеті Microsoft Office є можливість почергового копіювання декількох фрагментів документа (в останніх версіях – 24) для їх подальшого використання в інших місцях документа або в інших документах.

Виконується це через одночасне розміщення в **Буфері обміну** кількох фрагментів тексту (і не тільки тексту). **Буфер обміну** Microsoft Office спільний для програмних продуктів Microsoft. Для роботи з ним у додатках пропонується спеціальна **Область завдань**, яка теж називається **Буфер обміну**.

Для обміну інформацією між кількома документами або переміщенням чи копіюванням фрагментів у межах одного документа в меню **Основне** клацнути **Буфер обміну** – зліва від текстового поля (робочого) відкриється область завдань **Буфер обміну** (рис. 3.18). **Буфер обміну** заповнюється скопійованими фрагментами, які показані умовними значками. Значки відповідають різним типам даних фрагментів.

У заголовку **Області завдань** є кнопки **Вставити все** та **Очистити все**, а також список скопійованих фрагментів. Усі вони можуть бути вставлені в будь-який відкритий на цей час документ.

Для вставлення всіх фрагментів у активний документ потрібно встановити курсор миші (клацнути) у відповідне місце документа й натиснути кнопку **Вставити все**.

Для вставлення окремого фрагмента потрібно встановити курсор миші (клацнути) у відповідне місце документа, помістити покажчик миші на потрібний фрагмент і клацнути його. Він буде вставлений у документ у поточну позицію курсора.

Для видалення фрагмента з **Буфера обміну** потрібно клацнути на стрілці поряд із фрагментом. Після цього розкриється список команд, які можна застосувати до фрагментів, з них обрати й натиснути **Видалити**.

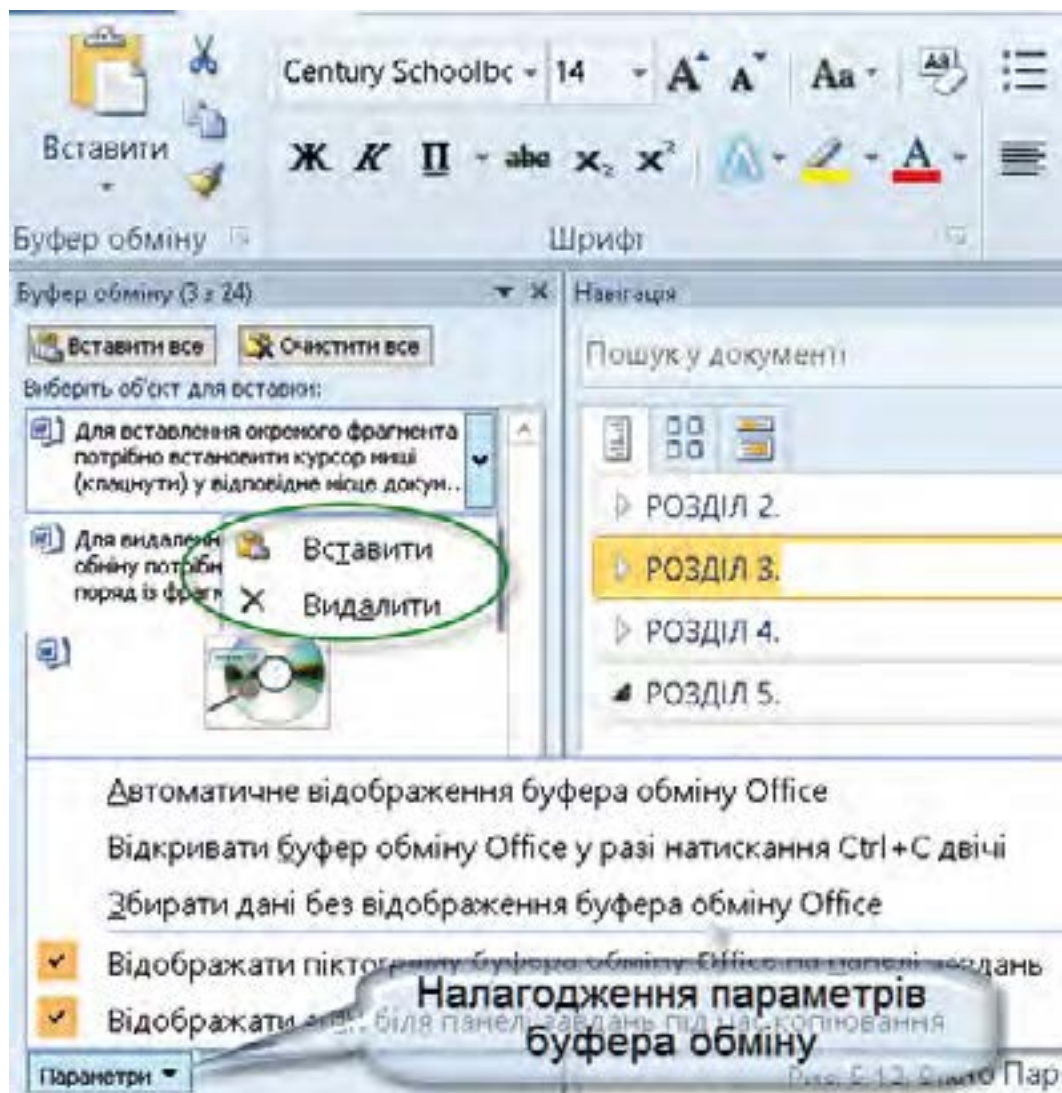


Рис. 3.18. Область завдань Буфер обміну

Перевіряємо себе

1. Який документ можна вважати складним? ✦
2. Які прийоми спростять форматування та редагування?
3. Якщо в текст документа внесено зміни (вставлення/вилучення сторінок, зміна заголовків (їх кількість і текст)), яким чином можна відобразити їх в змісті? ★
4. Які засоби можна використати для швидкого перегляду документа та внесення до нього змін? ▲
5. Який засіб використовується для обміну інформацією між кількома документами або переміщенням чи копіюванням фрагментів у межах одного документа? ▲
6. Як можна вставити одночасно декілька скопійованих фрагментів тексту? ✦
7. Як вставити окремий фрагмент тексту? ▲
8. Що слід зробити для вилучення фрагмента з Буфера обміну? ✦

Виконуємо

1. Відкрийте документ “Що я знаю про комп’ютер”. ▲
2. Додати у нього розділ “Як я користуюсь Інтернетом”. Для назви розділу виберіть стиль **Заголовок 1**. ✦
3. Виокремте кілька частин документа із назвами, яким надайте імена **Заголовок 2** та **Заголовок 3**. ★
4. Використати **Буфер** обміну для переміщення частин тексту: **Заголовок 3** на місце **Заголовок 2**. ✦
5. Створіть новий документ. ▲ Скопіюйте декілька фрагментів у документі “Що я знаю про комп’ютер” і вставте одночасно в створений документ. ✦
6. Очистіть Буфер обміну. ✦
7. Збережіть обидва документи. ▲

Практична робота № 4

Тема:	Створення текстового документа, що містить об’єкти різних типів
Мета:	Набути практичних навичок створення текстових документів з упровадженими та зв’язаними об’єктами

Завдання

1. Набрати текст відповідно до обраної теми.
2. Продумати та порадитись із учителем щодо необхідності та місця вставлення зображень у текст.

3. Продумати та порадитись із учителем щодо необхідності та місця вставлення таблиці в текст.

4. Виокремити в документі фрагмент тексту, який можна відформатувати в дві колонки.

5. Відформатувати зазначений фрагмент у колонки. Додати до цього фрагмента заголовок.

6. Зберегти створений документ.

Рекомендовані теми для практичної роботи

1. Предмети, які вивчають учні восьмого класу.

2. Що я знаю про Україну.

3. Найбільші річки України.

4. Що я знаю про Сполучені Штати Америки.

5. Мій особистий розклад на тиждень – мої улюблені справи.

6. Місто, де я народився (народилася).

7. Тварини, що живуть у лісах України.

Практична робота № 5

Тема:	Структура документа. Автоматизоване створення змісту і покажчиків
Мета:	Набути практичних навичок створення складних текстових документів

Завдання

1. У тексті документа, який створено в школі, виокремити заголовки першого та другого рівнів.

2. Застосувати до них відповідні стилі (використовуйте при цьому список **Стилів**).

3. До основного тексту документа застосувати стиль **Звичайний**.

4. Отримати від учителя вимоги до оформлення документа (передбачається застосування засобів **Абзац**, **Шрифт** тощо).

5. Переглянути структуру документа.

6. Для пояснення особливих для тексту документа термінів, власних імен, назв тощо створити **Предметний покажчик**.

7. На останній сторінці документа побудувати його зміст.

8. Зберегти документ.

Примітка. Додаткові вимоги до оформлення тексту надає вчитель.



СЛОВНИЧОК

Бланк – аркуш паперу з розміщеними на ньому постійними реквізитами.
Документ Word (*.doc) – формат текстового процесора Word; у цьому форматі документи зберігаються за замовчуванням.

Звичайний текст (*.txt) – найбільш універсальний формат, що зберігає текст без форматування; у текст вставляються тільки символи кінця абзацу;

- формат використовують для збереження документів, які можуть бути прочитані в застосунках, що працюють у різних операційних системах.
- Зміст документа** – впорядкований перелік заголовків і підзаголовків (зазвичай із номерами відповідних сторінок).
- Колонтитул** – текст та/або зображення, що розташовується внизу (нижній колонтитул) або вгорі сторінки документа (верхній колонтитул).
- Макрос** – команди та інструкції, згруповані в послідовність, яка запускається на виконання однією командою та призначена для автоматичного виконання певного завдання.
- Розділ у тексті документа** – частина документа, яка містить певний функціональний зміст.
- Стиль абзацу** – властивість абзацу (спосіб вирівнювання тексту, позиції табуляції, міжрядковий інтервал, інтервал перед і після, може містити формат символів).
- Стиль символів** – властивість символів (шрифт, розмір символів, накреслення, ефекти, колір тощо).
- Стиль списку** – властивість списку (спосіб вирівнювання, знаки нумерації або маркери, шрифти тощо).
- Стиль таблиці** – набір властивостей таблиці, який визначає вигляд меж, заливки, вирівнювання тексту, шрифти тощо.
- Стиль тексту** – набір значень властивостей тексту (у тому числі – рівень контуру, що використовується для автоматичної побудови документа), який має певну назву.
- Текст у форматі RTF (*.rtf)** – формат RTF (Rich Text Format – розширений текстовий формат), універсальний формат, що зберігає все форматування, перетворює коди керування на команди, які можуть бути прочитані та інтерпретовані в багатьох застосунках.
- Формат PDF (*.PDF)** – англ. Portable Document Format – формат файла, створений на основі формату Post Script (формату, призначеного для виведення документів на папір), який зберігає форматування документа.
- Формуляр** – сукупність реквізитів.
- Шаблон документа (*.dot)** – документ, на якому можуть базуватися інші документи.
- Шаблон електронного документа** – електронна форма, яка містить постійні реквізити та структуру документа.
- Штамп** – група реквізитів та їх постійних частин, відтворена на бланку (в шаблоні) як єдиний блок.



РОЗДІЛ 4. СТВОРЕННЯ ТА ОПРАЦЮВАННЯ ОБ'ЄКТІВ МУЛЬТИМЕДІА




Формати аудіо- та відеофайлів. Програмне забезпечення для опрацювання об'єктів мультимедіа. Засоби перетворення аудіо- та відеоформатів. Захоплення аудіо та відео, створення аудіо-, відеофрагментів. Побудова аудіо- та відеоряду. Додавання до відеокліпу відеоефектів та налаштування переходів між його фрагментами. Налаштування часових параметрів аудіо- та відеоряду. Сервіси розміщення аудіо- та відеофайлів у Інтернеті.


4.1. Формати аудіо- та відеофайлів. Конвертація аудіо- та відеофайлів


Мультимедіа – з одного боку, особливий тип документів, а з іншого – особливий клас програмного й апаратного забезпечення.


Мультимедійні документи відрізняються від звичайних тим, що крім традиційних текстових і графічних даних можуть містити звукові і музичні об'єкти, анімовану графіку (мультиплікацію), відеофрагменти.


Мультимедійне програмне забезпечення – програмні засоби, призначені для створення й відтворення мультимедійних об'єктів.

 **Мультимедіа** – програмний продукт, що містить зображення й текст, які супроводжуються звуком, відео, анімацією, та оснащений інтерактивним інтерфейсом з елементами керування.

 **Відео** – послідовність нерухомих зображень – кадрів, швидка зміна яких створює ілюзію руху об'єкта.


 *Для людського зору така ілюзія виникає при зміні 16 кадрів і більше за секунду. У кінотеатрах використовується зміна 24 кадрів за секунду, в телебаченні – від 25 кадрів за секунду. Що більша швидкість зміни кадрів, то якісніше зображення.*

 **Кодек** – програма, що перетворює потік даних або сигналів на цифрові коди, або навпаки. Звукові та візуальні дані потребують різних методів стиснення, а тому для них розроблені окремі кодеки.

 **Медіапрогравач** – пристрій або програмний засіб, який дає змогу відтворювати відео-, фото- і аудіоконтент.

Найпростішим у керуванні та найпоширенішим є медіапрогравач Windows Media Player.

Якщо для стиснення мультимедійних даних застосувати різні алгоритми, то ці дані будуть записані в файлах різних форматів. Є десятки таких алгоритмів і відповідно медіаформатів. Крім цього, існують формати, призначені для зберігання нестисненого звуку та відео.

 *Найпоширеніші формати.*

Аудіоформати:

– MP3 (Motion Picture Experts Group Layer 3) – використовує стиснення з втратами, підтримує стерео- і потокове передавання.


– WMA (Windows Media Audio) – ліцензований формат, розроблений корпорацією Майкрософт як альтернатива MP3.

Відеоформати:

– WMV (Windows Media Video) – дуже стиснутий формат, який потребує мінімального обсягу дискового простору на жорсткому диску комп'ютера;

– MPEG (Moving Pictures Experts Group) – забезпечує високу якість, підтримує додаткові можливості (захист від несанкціонованого копіювання, використання інтерактивних елементів) і потокове передавання відео;

– .SWF (Відео Flash) – використовується для передавання відео через Інтернет за допомогою програвача Adobe Flash.

 **Медіаконтейнер** – формат, у якому можна розміщувати в одному файлі мультимедійні дані різних типів і синхронізувати звук, відеозображення й текстову інформацію.

Медіаконтейнери:

– WAV (Waveform Audio Format) – звук у форматі WAV зберігається без втрати якості, але відсутність стиснення призводить до того, що обсяги wav-файлів дуже великі;

– AVI (Audio and Video Interleaved) – дає можливість об'єднувати нестиснені або закодовані різноманітними кодеками аудіо- та відеодані;

– MOV (QuickTime Movie) – як і формат AVI, дає змогу поєднувати аудіо- та відеопотоки, закодовані в різний спосіб, формат розроблений для програвача QuickTime.

Для відтворення (прослуховування чи перегляду) аудіо- та відеофайлів використовують спеціальні програми – програвачі – плеєри.


RealPlayer – програвач, який відтворює звук та відео більшості відомих форматів, зокрема поточкових, дає змогу записувати аудіофайли на компакт-диски в аналоговому вигляді.

QuickTime Player – надає змогу створювати й редагувати звук та відео, перекодувати й зберігати їх у різних форматах.

Win Amp – універсальний програвач мультимедійних файлів, який відтворює файли майже всіх поширених аудіо- та відеоформатів.

Media Player Classic – безкоштовний програвач із простим і зручним інтерфейсом, відтворює файли майже всіх поширених аудіо- та відеоформатів.

Windows Media Player – відтворює аудіо- та відеофайли більшості популярних форматів, входить до складу операційної системи Windows.


 Існує велика кількість форматів, призначених для зберігання аудіо- та відеофайлів. Найпоширеніші з них подано в табл. 4.1 і 4.2.

Таблиця 4.1

Популярні формати аудіофайлів (звукових файлів)

Формат звукового файла	Розширення	Характеристика
Audio Interchange File Format – AIFF	.aiff	Звуковий формат, який спочатку використовувався на комп'ютерах Apple і Silicon Graphics (SGI). Звукові файли цього формату зберігаються у 8-розрядному монофонічному (моно- або один канал) форматі, який не стискається, тому файли можуть мати великий розмір
UNIX Audio – AU	.au	Використовується для створення звукових файлів для комп'ютерів з операційною системою UNIX або для Інтернету
Musical Instrument Digital Interface – MIDI	.mid або .midi	Стандартний формат, який використовується для обміну музичною інформацією між музичними інструментами, синтезаторами та комп'ютерами
MPEG Audio Layer 3 – MP3	.mp3	Звуковий формат, стиснутий за допомогою кодека MPEG Audio Layer 3
Wave Form – звуковий файл Windows	.wav	Формат аудіофайлів, що зберігає звуки як форми хвиль; залежно від різних факторів, одна хвилина звуку може займати від 644 кілобайт до 27 мегабайт
Windows Media Audio	.wma	Формат, стиснутий за допомогою кодека Microsoft Windows Media Audio, схеми цифрового кодування звуку, розробленої корпорацією Майкрософт для розповсюдження записаної музики, зазвичай через Інтернет

Через велику кількість різноманітних форматів аудіо- та відеофайлів у користувачів часто виникає необхідність перекодувати ці файли з одного формату в інший.

 Процес перекодування файла з одного формату в інший називається **конвертацією файлів**.

Найпростішу конвертацію можна виконати через медіапрогра- вач Windows Media Player.

Популярні формати відео- та аудіофайлів

Формат файла	Розширення	Додаткові відомості
Відео Flash – Adobe Flash Media	.swf	Формат файла, який переважно використовується для передавання відео через Інтернет для відтворення програвачем Adobe Flash
Advanced Streaming Format – медіафайл Windows	.asf	Файл зберігає синхронізовані мультимедійні дані та може використовуватися для потокового відтворення аудіо- та відеовмісту, зображень і команд сценаріїв через мережу
Audio Video Interleave – відеофайл Windows	.avi	Мультимедійний формат файла, що використовується для збереження звуку та відеозображення у форматі Microsoft Resource Interchange File Format (RIFF). Один із найуживаніших форматів, оскільки звук і відеовміст, які стискаються за допомогою різноманітних кодеків, можуть зберігатись у файлі з розширенням .avi
Moving Picture Experts Group – файл фільму	.mpg або .mpeg	Це стандарт відео- й аудіокомпресії, який розвивається, створено Moving Picture Experts Group. Цей формат файла створено спеціально для використання на мультимедійних пристроях, наприклад, на відеокомпакт-дисках і дисках CD
Windows Media Video – відеофайл Windows Media	.wmv	У цьому форматі дані стиснені за допомогою кодека Windows Media Video. Потребує мінімального обсягу дискового простору на жорсткому диску комп'ютера

Існує також багато спеціалізованих програм для здійснення конвертації різноманітних мультимедійних даних. Однією з таких програм є Total Video Converter, який здійснює конвертацію аудіо- та відеофайлів більшості форматів.

Медіапрогравач Windows та інші програми використовують кодеки для створення й відтворення цифрових медіафайлів.

Кодек може складатися з двох компонентів: кодувальника й декодера.



Кодувальник виконує функцію стискання (кодування).

- ✓ *Декодер виконує функцію розпакування (декодування). Деякі кодеки мають обидва ці компоненти, а деякі – лише один із них.*

Кодеки взаємодіють з певними прикладними програмами, зокрема з медіапрогравачами, і допомагають їм відтворювати медіадані.

Залежно від вимог і ступеня підготовки користувача, можна обрати кодек як із попередньо встановленими режимами конвертації аудіо та відео, так і з широкими можливостями налаштування процесу цієї конвертації.

Таблиця 4.3

Програми, за допомогою яких здійснюється перетворення одного формату файлів на інший

Піктограма	Назва програми	Характеристика
	Format Factory	Безкоштовна універсальна програма для конвертування файлів відео, аудіо та зображень
	DVDFab	Програма для копіювання вмісту DVD і Blu-ray, а також для конвертації контенту з метою його подальшого перегляду на мобільному телефоні
	AVS Video Converter	Використовується для роботи з відео: редактор, конвертер, запис VCD, SVCD, DVD
	SUPER ©	Потужна програма для конвертації (перетворення) різноманітних форматів файлів аудіо та відео
	Switch Sound Converter	Зручний і потужний аудіоконвертер
	F2 Image Resizer	Безкоштовна програма для швидкої зміни розмірів, конвертації форматів і оптимізації фотографій
	Uni Converter	Універсальний перетворювач форматів векторної графіки
	Video charge	Різання, склеювання, конвертування відео та аудіо, створення скриншотів з відео, робота з файлами VOB і IFO, копіювання DVD
	Any Video Converter Free	Безкоштовна програма, що дає змогу конвертувати відеофайли в потрібний формат

Утиліти-конвертери мультимедія – це програми, що виконують перетворення файлів, які належать до одного типу даних, але в різних форматах. Універсальні конвертери дають можливість змінювати параметри формату або формат файлів різних типів. Прикладом такого конвертера відео- та аудіоформатів є Any Video Converter Freeware та ін. Програми для запису звуку і відео (програмами захоплення звуку, відео) називають **програмами-граберами**. Це такі програми, як Exact Audio Copy, Wonder share Streaming Video Recorder.

Існує велика кількість безкоштовних програм, що забезпечують перетворення одного формату файлів на інший.

Їх можна поділити на чотири умовні категорії. Вибір потрібної залежить від поставлених користувачем завдань, форматів, які підтримуються програмою, і можливостей з налаштування конвертації (табл. 4.3):

- програми, призначені для конвертації аудіо і відео для різних пристроїв (програвачів MP3, мобільних телефонів, плеєрів, ігрових приставок тощо);
- програми, орієнтовані на користувацький режим конвертації;
- гібридні програми, у яких поєдналися характеристики обох згаданих вище категорій;
- вузькоспеціалізовані програми, орієнтовані на виконання конвертації аудіофайлів.

Перевіряємо себе

1. Що називають мультимедія? ▲
2. Що таке мультимедійне програмне забезпечення? ◆
3. Що таке відео? ▲
4. Як називається програма, що перетворює потік даних або сигналів на цифрові коди, або навпаки? ◆
5. Який пристрій називають медіапрогравачем? ▲
6. Назвіть найпоширеніші аудіоформати. ◆
7. Назвіть найпоширеніші відеоформати. ◆
8. Поясніть призначення медіаконтейнера. ◆
9. Який процес називається конвертацією файлів? ★
10. Із чого складається кодек? ★
11. Як називаються програми, що виконують перетворення файлів? ▲

Виконуємо

1. Знайдіть мультимедійні документи на вашому комп'ютері. ▲
2. Поясніть, чим вони відрізняються від текстових і графічних документів. ▲
3. Скопіюйте в окрему папку файли, які належать до **Аудіоформатів**. ▲

4. Скопіюйте в окрему папку файли, які належать до **Відеоформатів**. ✦
5. Дослідіть особливості роботи з утилітами-конвертерами. ★

4.2. Програмне забезпечення для опрацювання об'єктів мультимедія

Нині відомо багато програм, які можна використати для створення відеокліпів із зображень або створених раніше відео- та аудіофайлів.

Розглянемо стандартну програму операційної системи Windows – **Windows Movie Maker**. Цю програму можна знайти в меню: **Запустити – Усі програми – Windows Movie Maker**.

Вікно програми Windows Movie Maker складається з чотирьох основних частин (рис. 4.1): **Панель операцій з відео**; **Панель вмісту (контенту)**; **Вікно попереднього перегляду** та **Область монтажу (Аркуш розкадрування/Шкала часу)**.

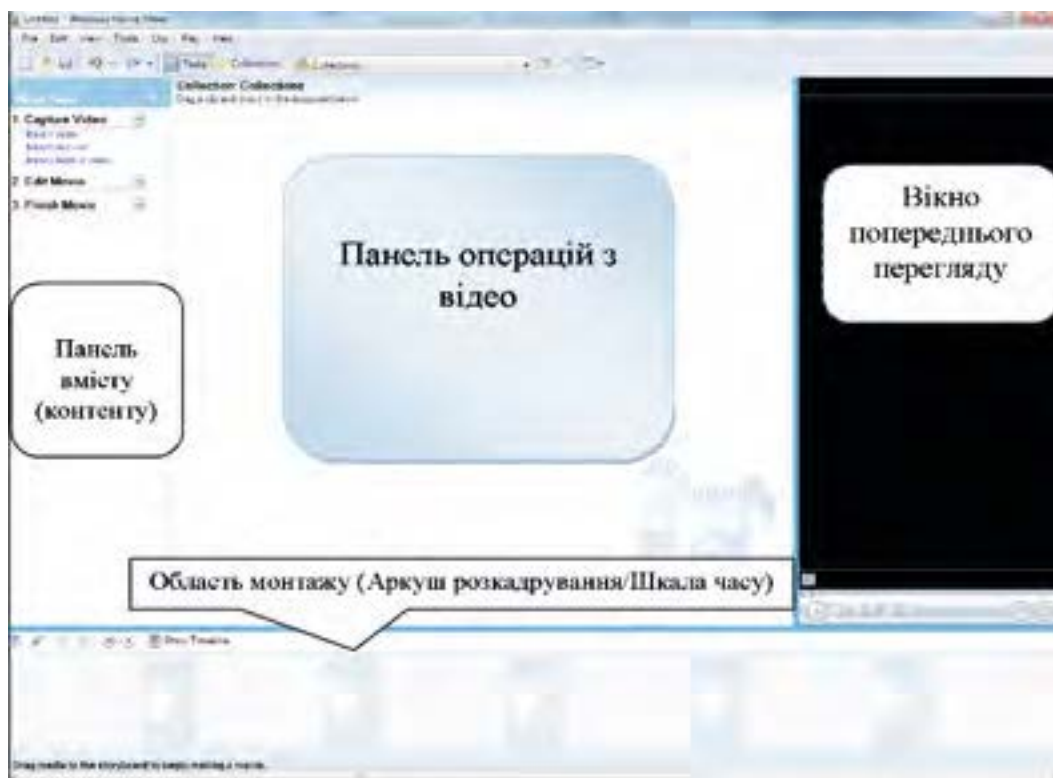


Рис. 4.1. Будова вікна програми Windows Movie Maker

Панель операцій з відео. Ця панель поділена на розділи, що відповідають покроковому процесу створення відео:

- **Запис відео (Capture Video)** – інструменти, які використовуються для початку створення відео: запис відео та імпорт існуючого відео, зображення чи звуку.

• **Монтаж фільму (Edit Video)** – інструменти перегляду існуючого відео, зображення, відеоефектів, прослуховування звуку або додавання у відео назв і титрів.

• **Завершення створення відео (Finish Movie)** – інструменти, призначені для збереження підготовленого відео на комп'ютері, компакт-диску чи надсилання засобами Інтернет.

• **Поради щодо створення фільмів (Movie Making Tips)** – довідковий матеріал про виконання операцій у програмі Windows Movie Maker.

Панель вмісту (контенту). На цій панелі відображаються кліпи, які містяться у вибраній на **Панелі операцій** рубриці. Тут відображаються всі файли відео, аудіо, зображення, відеопереходи і візуальні ефекти, які можна додати на шкалу розкадрування або часу.

Вікно попереднього перегляду. Це вікно можна використовувати для переглядання як окремих кліпів, так і всього проекту, перш ніж його зберегти.

Область монтажу. Це область, у якій створюються і монтуються проекти. Вона відображається у двох режимах: **Аркуш розкадрування** і **Шкала часу**. У процесі створення фільму можна перемикатися між цими двома режимами.

У режимі **Аркуш розкадрування** відображаються картинки всіх вибраних фрагментів і налаштовані переходи між ними.

У режимі **Шкала часу** відображаються відеофрагменти, довжини яких пропорційні часу їх відтворення, переходи між ними, звуковий супровід і титри.

Перевіряємо себе

1. Які вам відомі програми, котрі можна використати для створення відеокліпів із зображень або створених раніше відео- та аудіофайлів? ▲
2. Із яких основних частин складається вікно програми Windows Movie? ◆
3. Для чого призначені команди розділів **Панелі операцій з відео**? ◆
4. Для чого призначено **Панель вмісту (контенту)**? ◆
5. Для чого призначено **Вікно попереднього перегляду**? ★
6. Для чого призначено **Область монтажу**? ★

Виконуємо

1. Дослідіть **Панель вмісту (контенту)**. ◆
2. Дослідіть **Панель операцій з відео**. ◆
3. Дослідіть **Вікно попереднього перегляду**. ◆
4. Дослідіть **Область монтажу**. ◆

4.3. Створення й опублікування мультимедія



Захоплення аудіо та відео, створення аудіо-, відеофрагментів з використанням цифрових відео- або фотокамер. Побудова аудіо- та відеоряду. Додавання до відеокліпу відео ефектів та налаштування переходів між його фрагментами. Налаштування часових параметрів аудіо- та відеоряду. Збереження створених відеофільмів на носіях даних. Сервіси публікування відеофайлів. Подкастинг.

Додавання зображень і аудіо- та відеофайлів можна виконати за допомогою **Панелі операції**:

1. Для імпортування фотографій натисніть Import Pictures.
2. Щоб додати до проекту аудіофайл, потрібно натиснути Import audio or music.
3. Для додавання існуючого відео натиснути Import video.

Усі вибрані файли будуть додані на **Панель вмісту**, як показано на рис. 4.2.



Рис. 4.2. Вікно програми Windows Movie Maker, у якому на **Панель вмісту** додано фотографії та аудіофайл

✓ Для монтажу фільму потрібно виконати таке.

1. За допомогою миші перетягнути фотографії та аудіофайл із **Панелі операції** в **Область монтажу** на **Шкалі часу**.
2. Потім відсортувати фотографії, перетягуючи їх мишею по шкалі часу у такій послідовності, в якій вони мають відобразитися в створеному фільмі.

3. Стандартна тривалість відтворення фотографії 5 секунд. Тривалість показу конкретного зображення можна змінити, розтягуючи його – притиснувши ліву межу зображення лівою кнопкою миші.

4. У **Області монтажу** перейти зі **Шкали часу** на **Аркуш розкадрування**. Для переходу між цими двома режимами використовується кнопка перемикач між режимами Show Storyboard (рис. 4.3).



Рис. 4.3. Область монтажу, де виділено кнопку перемикач між режимами Show Storyboard

5. Для того щоб фото перегорталися з особливими ефектами, можна використати спеціальні переходи. Для їх перегляду потрібно клацнути на меню **Show video transitions**.

6. Для перегляду ефекту клацніть лівою кнопкою миші на іконці ефекту – він відобразиться у **Вікні перегляду**.

7. Після вибору ефекту його слід перетягнути мишкою на **Шкалу розкадрування** між двома фотографіями, до яких цей ефект буде застосований.

✓ Слід пам'ятати, що відеопереходи скорочують тривалість показу зображень на шкалі, оскільки відбувається накладання кадрів. Це означає, що після застосування відеопереходів на весь час програвання аудіо буде зайнятим. У такому випадку рекомендується додати ще декілька зображень.

✓ Для видалення ефекту слід вибрати кнопку на потрібному кадрі й натиснути клавішу **Delete**.

8. На **Панелі операцій** натиснути на посилання **Save to my computer**, дати ім'я створеному відео та обрати папку, де воно буде збережене.


9. Файл буде збережений у форматі *.WMV і надалі його можна використовувати як звичайний відеофайл: копіювати, відтворювати з використанням відеопрогравачів, конвертувати, вставляти в слайдові презентації тощо.

Після завершення процедури збереження рекомендується зачекати 5–10 хвилин, доки програма перетворить створений файл проекту на фільм. Відтак його можна опублікувати на Youtube або записати на будь-який носій.

✓ *Невеликі за розмірами відеофільми називаються кліпами.*

Текст, що виводиться наприкінці фільму, називають **титрами**, а будь-який інший текст у фільмі – **назвами**.

Мультимедійні файли, які розповсюджуються через Інтернет, називаються **подкастами** (англ. *podcast*). Вони відтворюються на портативних медіяпрогравачах або персональних комп'ютерах, можуть містити інтерв'ю, лекції чи будь-що інше, що належить до усного жанру.

 Термін *podcast* є поєднанням назви портативного програвача музики iPod та слова *broadcast* (радіопередача, трансляція).

У Windows Movie Maker можна захопити зображення окремого кадру з імпортованого відео- чи аудіофайла, а потім використати це зображення як статичне у створеному фільмі. Зображення, які захоплюються з відеокліпів у Windows Movie Maker, автоматично зберігаються як файли у форматі JPEG з розширенням .jpg.

1. У області **Вміст** клацніть відеокліп, з якого потрібно захопити зображення.

2. Знайдіть кадр у відеокліпі, який потрібно зберегти як окреме зображення, пересуваючи індикатор відтворення під **Вікном попереднього перегляду**. Для пошуку потрібного кадру можна скористатися кнопками **Наступний кадр** і **Попередній кадр**.

3. Виберіть у меню **Знаряддя** пункт **Зробити знімок із попереднього перегляду**.

4. Уведіть ім'я для файла зображення та натисніть кнопку **Зберегти**.

Можна також захопити окреме зображення з відеокліпу на **Аркуші розкадрування/шкалі часу**, але якщо захоплювати його з кліпу в області **Вміст**, на виході часто можна отримати зображення вищої якості. Можна провести захоплення (запис) аудіо(відео) за допомогою спеціальних програм (граберів).

Наприклад, програму Windows **Звукозапис** можна запустити, натискаючи меню **Пуск – Усі програми – Стандартні – Звукозаписувач**. Ця програма записує звуки в форматі WAV тривалістю 60 секунд (за допомогою мікрофона).

Перевіряємо себе

1. За допомогою якого інструменту можна виконати додавання зображень і аудіо- та відеофайлів? ✨
2. Які дії потрібно виконати для монтажу фільму? ✨
3. Що слід зробити для вилучення ефекту? ▲
4. Які відеофільми називаються кліпами? ✨
5. Які мультимедійні файли називаються подкастами? ▲

Виконуємо

1. Зробіть кілька фотографій робочого дня вашого класу. ▲
2. Відберіть фотографії, які найбільше відтворюють особливості навчання у вашій школі. ▲
3. Підберіть аудіофайли, які можуть супроводжувати фотографії. ★
4. Відберіть програму, за допомогою якої можна створити відеокліп. Пояснити такий вибір. ★
5. За допомогою обраної програми створіть відеокліп на 20 секунд. ★

Практична робота № 6

Тема:	Створення відеокліпу. Додавання відеоефектів, налаштування часових параметрів аудіо- та відеоряду
Мета:	Набути практичних навичок опрацювання об'єктів мультимедія

Користуючись Windows Movie Maker, за матеріалом, наданим учителем, створити мультимедійний файл.

Примітка. Файл має містити аудіо- та відеодані.

Практична робота № 7

Тема:	Розміщення аудіо- та відеоматеріалів у Інтернеті
Мета:	Набути практичних навичок опрацювання об'єктів мультимедія

Створити відеокліп “Мої друзі”.

У процесі створення відеокліпу дотримуйтесь етапів.

1. На першому етапі відеомонтажу кліпу створюєте новий проект та перетягуванням встановлюєте мультимедійні об'єкти зі збірника в **Область монтажу**.

2. На другому етапі створюєте переходи між зображеннями, додаєте відеоефекти, титри та голосовий супровід.

3. На третьому етапі вставляєте титри або текстовий супровід.

Підказки

1. Для оформлення відеокліпу використовуйте ваші власні фотогалереї.

2. Для супроводу показу підберіть та вставте улюблену музику.

3. Додайте текстові написи до відеофільму (рис. 4.4). Для цього виконайте такі дії:

- У меню **Tools** клацніть **Titles and Credits**.

- Відкриється вікно, в якому оберіть потрібне: вставлення титрів, назви перед обраним зображенням чи після нього тощо.

- Вставте потрібний текст.




Рис. 4.4. Меню Tools із виділеною командою вставлення назв і титрів

- Збережіть виконане.

Тривалість демонстрації вставлених титрів за замовчуванням – 3,5 секунди. Її можна змінити так само, як і тривалість демонстрації зображень.

Для титрів, що розміщуються всередині кліпу, можна перетягувати як ліву, так і праву межу.

Додайте дикторський звуковий супровід (рис. 4.5). Для цього:

- Виберіть точку на Шкалі часу.
- У меню Tools клацніть **Narrate Timeline** або виберіть кнопку  у лівій частині **Області монтажу**. Відкриється вікно запису.
- Для початку запису виберіть кнопку **Start**, а для призупинення чи завершення запису – кнопку **Done**.

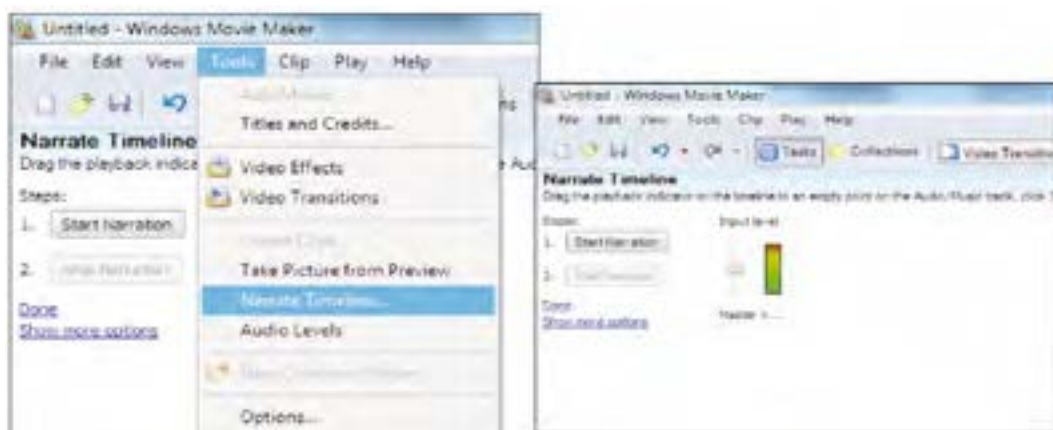


Рис. 4.5. Засоби запису дикторського тексту



СЛОВНИЧОК

Відео – послідовність зображень – кадрів, які створюють рухоме зображення.

Декодер – програма, що виконує функцію розпакування (декодування).

Кліп – невеликий за розмірами відеофільм.

Кодек – програма, що перетворює потік даних або сигналів на цифрові коди.

Кодувальник – програма, що виконує функцію стискання (кодування).

Конвертація файлів – процес перекодування файла з одного формату в інший.

Медіаконтейнер – формат, що дає змогу розміщувати в одному файлі мультимедійні дані різних типів і синхронізувати звук, відео й текстову інформацію.

Медіапрогравач – пристрій, за допомогою якого відтворюється відео-, фото- і аудіоконтент.

Мультимедія – програмний продукт, що містить зображення і текст, які супроводжуються звуком, відео, анімацією, та оснащений інтерактивним інтерфейсом з елементами керування.

Подкасти – мультимедійні файли, які розповсюджуються через Інтернет.

Титри – текст, що виводиться на початку і наприкінці фільму.



РОЗДІЛ 5. ОСНОВИ ПОДІЙНО- ТА ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ



Алгоритм можна описати, використовуючи звичайну мову. Для того щоб це виконав програмно-керований автомат (комп'ютер), потрібно алгоритм подати у формі, доступній для опрацювання і виконання комп'ютером.



Підготовка і розв'язування задач. Основні види мов програмування та їх складники. Призначення і структура візуального середовища програмування Lazarus. Використання форми для створення графічного інтерфейсу, визначення потрібних для цього об'єктів і їх властивостей. Створення і виконання найпростіших проєктів у середовищі Lazarus.

5.1. Загальний порядок розв'язування задач на комп'ютері



Алгоритм. Мова програмування. Математична модель. Компіляція програми.

Процес підготовки задач за допомогою мови програмування для їх реалізації на комп'ютері називають **програмуванням**. Мова програмування – це універсальний засіб, призначений для подання алгоритмів у вигляді, придатному для реалізації на комп'ютері. Кожний програміст має дотримуватися порядку і правил, які існують у певному колективі програмістів. Для початківців і учнів бажано роботу з розроблення і налагодження програм поділити на етап **підготовчий** і етап **виконання**.

5.1.1. Підготовчий етап

Сутність цього етапу полягає в докладному аналізі задачі, яку потрібно розв'язати, її формалізації і розробленні програмного коду. На цьому етапі бажано дотримуватися послідовності роботи, відображеної на рис. 5.1.



Рис. 5.1. Послідовність підготовки задачі для розв'язування на комп'ютері

1. *З'ясування сутності задачі.* Задача, яку потрібно запрограмувати, найчастіше формулюється словесно. Слід чітко з'ясувати її умову, визна-

чити вхідні та вихідні дані, форму подання результатів її розв'язання і таке інше.

Наприклад, задача може бути сформульована так: “Для автобуса Київ–Рівне, що прямує через Житомир, встановлена середня швидкість руху на всьому маршруті. Автобус з Києва до Житомира рухався з однією швидкістю, а з Житомира до Рівного – з іншою. Визначити, з якою швидкістю має рухатися автобус із Житомира до Рівного, щоб він прибув до кінцевої станції своєчасно”. З'ясовуючи сутність цієї задачі, потрібно: визначити (знайти в довіднику) відстані від Києва до Житомира і від Житомира до Рівного, встановлену швидкість автобуса на всьому маршруті і реальну швидкість автобуса на шляху від Києва до Житомира.

2. *Формалізація задачі.* Формалізувати задачу означає описати її термінами певної галузі знань. Зазначений етап не є обов'язковим для всіх задач. Наприклад, етап формалізації не обов'язковий для деяких обчислювальних задач із курсів шкільної математики, фізики, хімії, позаяк самі умови задач уже формалізовані. Для сформульованої вище задачі можна позначити: s_1 – відстань від Києва до Житомира; s_2 – відстань від Житомира до Рівного; v – швидкість руху для всього маршруту, визначена як середнє значення ($v = (s_1 + s_2)/t$, де t – час між прибуттям до кінцевого пункту і відбуттям із початкового пункту); v_1 – реальна швидкість руху автобуса від Києва до Житомира; v_2 – швидкість руху автобуса від Житомира до Рівного; яку потрібно визначити. Задачу можна сформулювати так: визначити швидкість v_2 , якщо відомі s_1 , s_2 , v і v_1 .

3. *Розроблення математичної моделі й вибір способу її раціональної реалізації.* Розробити математичну модель означає знайти математичну залежність між уведеними математичними величинами. Для наведеного прикладу цю модель можна записати так: $(s_1 + s_2)/v = s_1/v_1 + s_2/v_2$, тобто використати той факт, що потрібно вчасно дістатися кінцевого пункту. Вибрати раціональний спосіб реалізації моделі означає знайти варіант розв'язання отриманого рівняння. У цьому випадку можна запропонувати формулу $v_2 = s_2 / ((s_1 + s_2)/v - s_1/v_1)$.

4. *Розроблення алгоритму.* Розробити алгоритм – означає визначити послідовність інструкцій (вказівок), які можна реалізувати на комп'ютері і від виконання яких залежить правильне розв'язання задачі. Розроблення таких інструкцій називають **алгоритмізацією**. Ступінь деталізації вказівок може бути різним і залежить від мови програмування, за допомогою якої описуватиметься алгоритм. Для задачі, що розглядається, алгоритм, який може бути реалізований практично будь-якою мовою програмування, у словесній формі можна записати так.


1. Увести значення змінних s_1, s_2, v, v_1 .
2. Обчислити значення виразу $(s_1 + s_2)/v$ (загальний час руху).
3. Від значення, отриманого в п. 2, відняти значення s_1/v_1 .
4. Значення s_2 поділити на значення, отримане в п. 3.
5. Присвоїти змінній v_2 значення, отримане в п. 4, і вивести його на екран.

Графічна схема цього алгоритму зображена на рис. 5.2.


Розроблення програми. Розробити програму означає описати алгоритм відповідною мовою програмування. Зазначимо, що у комп'ютері виконуються лише програми, описані машинною мовою, тобто мовою двійкових символів 0 і 1.

Програміст нині користується більш доступною для нього мовою, так званою мовою високого рівня. Подання алгоритмів такою мовою програмування виконується точно й формалізовано, і подальший перехід від неї до машинної мови відбувається автоматично, за допомогою спеціальної програми – **транслятора**.

Отже, мова програмування високого рівня і система програмування відіграють роль посередника між людиною та комп'ютером.

 **Обов'язкові компоненти** будь-якої мови програмування високого рівня:

- алфавіт (аналогічно алфавіту звичайної людської мови);
- символи арифметичних і логічних операцій та деякі спеціальні знаки;
- набір так званих ключових слів, які не можна змінювати (наприклад, begin, end, var, for);
- ідентифікатори – комбінації символів для позначення змінних, функцій, файлів та інших об'єктів (приклади: alfa, n32, sun_2, a1).

 **Мова програмування високого рівня має строгі правила запису команд (операторів) і даних. Сукупність таких правил створює синтаксис мови, а зміст кожної команди та інших конструкцій мови – її семантику.**

Описаний порядок розроблення програм – це лише загальні рекомендації. Він залежить і від обраного середовища програмування. Наприклад, для візуальних середовищ програмування спочатку розробляється

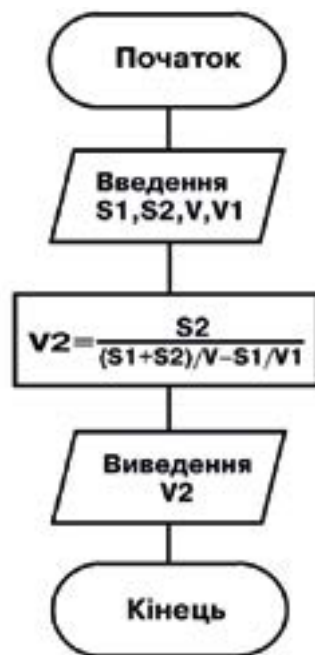


Рис. 5.2. Графічна схема алгоритму обчислення швидкості руху автомобіля

інтерфейс майбутньої програми, а потім – програмний код. Зазначмо, що досвідчені програмісти після аналізу завдання одразу починають розробляти програму і вводити її в комп’ютер. Однак у навчанні потрібно дотримуватися описаного порядку.

“Чужу” програму, особливо складну, без розроблення алгоритму і без коментарів, зрозуміти важко. Тому слід виробити культуру оформлення алгоритмів на прикладах простих задач.

5.1.2. Етап виконання

На цьому етапі зазвичай виконується послідовність дій, зображена на рис. 5.3.




Рис. 5.3. Орієнтовна послідовність дій на етапі виконання

1. *Уведення програми.* Розроблена програма, яку називають кодом (програмним кодом), уводиться в пам’ять комп’ютера найчастіше з клавіатури. У процесі введення програма відображається на екрані монітора, і помилки введення можна одразу виправляти. Для введення коду найчастіше використовується інтегроване середовище програмування. **Наголосимо, що у візуальному середовищі програмування спочатку на формі створюється її інтерфейс, а потім уводиться програмний код.**

2. *Компіляція програми.* Початковий текст програми мовою високого рівня, уведений у комп’ютер, автоматично перетворюється на машинний код за допомогою транслятора. Якщо текст програми не відповідає синтаксису мови програмування, трансляція не буде виконана. У таких випадках транслятор видає повідомлення про синтаксичні помилки. Є два види трансляторів: **компілятори та інтерпретатори.**

Компілятор одразу перетворює **весь текст** програми на двійковий код. Спочатку отримують **об’єктний код**, а потім – **виконуваний**, який розміщується в файлі з розширенням `exe`.

 **Виконуваний код** – це завершена програма, готова до виконання на комп’ютері з відповідним процесором, що функціонує під управлінням операційної системи, для якої відкомпільовано цю програму.

Інтерпретатор вибирає з тексту програми одну команду, аналізує її структуру, перетворює на машинні команди й одразу передає на виконання. Після успішного виконання цієї команди для наступної команди виконуються аналогічні дії.

Перевага компіляторів над інтерпретаторами полягає в тому, що створюється машинний код з високою швидкістю виконання. Цей код можна переносити на інший комп'ютер із аналогічним процесором і виконувати без трансляції. Застосовуючи інтерпретатор, програміст може ввести зміни до програми й продовжити її виконання. Така технологія дуже зручна для програмістів-початківців, а також для створення програм, які можуть виконуватися на дуже різних комп'ютерах.

Для створення програми використовується середовище програмування, обов'язковими компонентами якого є:

- текстовий редактор для введення та редагування програм;
- редактор зв'язків;
- транслятор (компілятор або інтерпретатор).

Середовища програмування містять також набір підпрограм обчислення значень стандартних функцій. Для об'єднання всіх модулів програми у єдиний файл (такий файл називають файлом, що виконується) у середовищах програмування застосовуються редактори зв'язків, засоби для налагодження програм та інші складники.

3. Тестування та налагодження програми. У фірмах-розробниках програм створюються спеціальні підрозділи, які займаються тестуванням програм. Успішна компіляція ще не означає, що в коді немає помилок. Програма після успішної компіляції може запрацювати, але вона може видавати неправильні результати, оскільки крім **синтаксичних** у ній можуть бути **логічні** помилки, які допустив програміст. Наприклад, якщо у виразі $y = a + b$ замість знака плюс насправді введено знак мінус, така помилка є логічною, і компілятор її не виявляє. Тому потрібно виконати тестування програм і усунути помилки. Зазвичай програма багаторазово виконується для різноманітних даних у допустимому діапазоні значень.

4. Аналіз отриманих результатів. На основі аналізу отриманих результатів робиться остаточний висновок про працездатність і надійність програми. Для навчальних задач, щоб бути впевненим, що програма розроблена правильно й у ній відсутні будь-які помилки, необхідно вручну розрахувати деякі проміжні результати і порівняти їх з результатами, які видає програма.

Перевіряємо себе

1. Які функції виконує транслятор? ▲
2. Які компоненти мови програмування є обов'язковими? ▲
3. Що називають синтаксисом мови програмування? ▲
4. Які існують типи трансляторів? ▲
5. Назвіть основні компоненти середовища програмування. ▲

6. Які дії виконуються під час підготовки задачі для розв'язування на комп'ютері? ★
7. Поясніть сутність формалізації задачі. ★
8. Наведіть приклад математичної моделі. ★
9. У чому полягають переваги та недоліки компіляторів і інтерпретаторів? ★
10. Чому потрібне тестування й налагодження програм? ★
11. Які основні дії виконуються у процесі виконання програм? ★
12. Який код програми називають таким, що виконується? ★
13. Поясніть сутність тестування й налагодження. ★

5.2. Мови програмування



Мови програмування високого та низького рівнів. Мови візуального програмування. Процедурні мови програмування. Об'єктно-орієнтовані мови програмування.

Початок розвитку мов програмування (МП) сягає ХІХ століття. Англійський учений Чарльз Беббідж створив проект механічної обчислювальної машини. Програми для неї розробила леді Ада Лавлейс (на її честь названа мова програмування Ada).

Однак мови програмування в сучасному розумінні стали розвиватися з появою ЕОМ.



Мова програмування – мова, призначена для запису алгоритму для його виконання комп'ютером.

Існують сотні мов програмування та їх модифікацій, проте лише деякі з них набули широкого визнання. На різних етапах розвитку ЕОМ були популярні такі мови програмування: Фортран, Кобол, Алгол-60, ПЛ-1, Алгол-68, Ada, С, Basic, Pascal, Prolog. Нині широкоживаними є мови С++ (вимовляємо “Сі плюс плюс”), С# (вимовляємо “Сі шарп”) і Java (вимовляємо “Джава”).

Існують сотні різноманітних класифікацій мов програмування. До основних класифікаційних ознак слід віднести такі: ступінь залежності мов від апаратних засобів, принципи програмування та орієнтація на клас задач. На рис. 5.4 подана схема класифікації мов програмування за цими ознаками.

Незалежно від того, якою мовою розроблена програма, у комп'ютері виконується лише програма машинною мовою, тобто мовою, що подається двійковими символами 0 і 1. Для перших ЕОМ самі програмісти розробляли програми машинними мовами. Але це була досить важка робота, яка вимагала також добрих знань структури і принципів взаємодії пристроїв комп'ютера. Пізніше з'явилися мови символічного коду-

вання (МСК), у яких команди програміст подавав не двійковим кодом, а символами, наприклад, ДОДАТИ А В. За цією командою до числа А додавалося число В.



Рис. 5.4. Класифікація мов програмування

Очевидно, що розроблення програм мовою МСК полегшувало роботу програміста. Перетворення символічного коду на машинні команди комп'ютер виконував автоматично.

Потім до МСК стали включати **макрокоманди, які реалізуються послідовністю з кількох машинних команд**. Використання макрокоманд у МСК ще більше полегшило розробку програм. Різновиди мов символічного кодування називають автокодом і мовою Асемблер (приклад фрагмента програми мовою Асемблер є на рис. 2.6).

Проте мови символічного кодування були все ще дуже складними. Програмування на них вимагало спеціальних знань, зокрема, знання структурних особливостей процесора і комп'ютера. Такі мови отримали назву **машинно-орієнтовані, або мови низького рівня**. Майже кожний тип комп'ютера мав власну мову програмування низького рівня, що обмежувало обмін програмами. Такі мови використовували системні програмісти для розроблення системного і прикладного програмного забезпечення ЕОМ третього покоління, сьогодні за їх допомогою розробляють драйвери пристроїв сучасних комп'ютерів.

Уже на перших етапах розвитку обчислювальної техніки розпочалося розроблення мов програмування, доступних для широкого кола користувачів і не зв'язаних із конкретною обчислювальною машиною. **Такі мови назвали мовами високого рівня**. Першою такою мовою, яка набула широкого визнання серед програмістів світу, стала мова Фортран, розроблена у 1954 році в США. Ця мова близька до звичайної мови алгебри й орієнтована на розв'язування обчислювальних задач. У 1960 році група вчених різних країн розробила мову програмування Алгол-60, яка також орієнтована на розв'язування обчислювальних задач, але її конструкції були більш наочними.

✓ *За ступенем залежності від апаратних засобів розрізняють мови низького і високого рівнів.*

Із розвитком і удосконаленням обчислювальних машин розширювалися й галузі їх застосування. Удосконалювалися й розвивалися також мови програмування, як шляхом спеціалізації, так і шляхом універсализації. Однією з перших спеціалізованих мов була мова Кобол, розроблена в 1961 році в США й орієнтована на розв'язування економічних задач.

Пізніше були розроблені десятки різних спеціалізованих мов, наприклад, Симула – мова моделювання, Лісп – мова для розв'язування інформаційно-логічних задач, Рапіра – мова для розв'язування навчальних задач.

Отже, з одного боку, потреби в розв'язуванні різноманітних задач, а з іншого – удосконалення і поява нових обчислювальних машин поставили нові вимоги до мов програмування. Сутність цих вимог полягала в тому, що вони мали забезпечити успішне розв'язування різних задач людської діяльності за максимального використання можливостей, що надають ЕОМ.

Мови програмування, які тією чи іншою мірою відповідають усім зазначеним вимогам, відносять до класу **універсальних мов**. Найбільш відомими мовами цього класу були мова ПЛ-1, розроблена в 1964 році, і мова Алгол-65. Утім, ці мови виявилися досить складними, крім цього, мова ПЛ-1 не задовольняла вимогу надійності написання програм. Потім набула поширення універсальна мова Pascal, розроблена спеціально для навчання програмування.

Особливою популярністю в усьому світі користувалася мова Бейсик. Вона незамінна для розв'язування простих задач, створення додатків до офісних програмних засобів.

✓ *За орієнтацією на види задач мови програмування поділяються на універсальні та спеціалізовані.*

Універсальні мови призначені для розв'язування широкого класу задач. До цих мов нині відносять Pascal, C, C++ та інші. Особливим класом універсальних мов є мови, реалізовані у **візуальних** середовищах програмування (Visual Basic, Delphi та інші).

Спеціалізовані мови програмування призначені для певного типу задач і певної предметної галузі. Існують десятки спеціалізованих мов програмування, наприклад, мови для роботи з базами даних, мови web-програмування.

За принципами програмування розрізняють процедурні, непроцедурні мови та мови об'єктно-орієнтованого програмування.

Процедурні мови побудовані на описі послідовної зміни стану комп'ютера, тобто значення комірок пам'яті, стану процесора та інших пристроїв. Вони маніпулюють даними в покроковому режимі, використовуючи послідовне виконання інструкцій. У процедурних мовах дотримуються чіткої структуризації програм, тому їх ще називають

мовами структурного програмування. Прикладами таких мов є Алгол, Pascal, перші версії мови Basic, частково – Fortran.

Непроцедурні мови ефективні для програмування пошуку даних у великих обсягах даних, а також для програмування задач, розв’язування яких неможливо точно й вичерпно описати (переклад, розпізнавання образів). У цих мовах алгоритм пошуку розв’язку вбудований у інтерпретатор мови. До непроцедурних відносять мови функціонального та логічного програмування.

✓ *Мови об’єктно-орієнтованого програмування (ООП) почали розвиватися наприкінці ХХ століття.*

Сьогодні вони є основними мовами високого рівня для професійного програмування. Ці мови містять конструкції, які дають змогу описувати об’єкти і їх класи і які мають засоби роботи з абстрактними типами даних. До таких мов відносять, зокрема, C++, Object Pascal, Java, C#.

Далі розглядатимемо середовища програмування **Free Pascal** і **Lazarus**.

Перевіряємо себе

1. Якою мовою описують програму, для того щоб вона могла бути виконана в комп’ютері? ▲
2. Поясніть сутність мови символічного кодування. ▲
3. Що називають макрокомандою? ▲
4. Які мови називають мовами низького рівня? ▲
5. Які мови називають мовами високого рівня? ▲
6. Як класифікуються мови програмування за ознакою залежності від апаратних засобів? ★
7. Які мови відносять до універсальних? ★
8. Наведіть приклади мов програмування високого рівня. ★
9. Як класифікуються мови програмування за орієнтацією на клас задач? ★
10. Як класифікуються мови програмування за принципами програмування? ★
11. Наведіть приклади мов об’єктно-орієнтованого програмування. ★

5.3. Основні відомості про мову Free Pascal і середовища програмування



Середовище програмування Free Pascal. Обов’язкові компоненти мов програмування.

Мову програмування Pascal розробив швейцарський учений Н. Вірт у 1971 році спеціально для навчання основ програмування. Вона названа на

честь видатного французького математика Б. Паскаля. Ця мова постійно розвивалася й удосконалювалася, у результаті чого зараз налічується десятки її версій. Нині популярною для навчання основ алгоритмізації і програмування у вищих і середніх закладах освіти є мова Free Pascal, яку розглянемо нижче. Зазначимо, що сучасна мова програмування майже завжди реалізується як система (середовище), що забезпечує весь комплекс робіт – від розроблення програм до отримання результатів її виконання. Обов'язковими складниками середовища програмування є текстовий редактор, компілятор і редактор зв'язків. Середовище Free Pascal забезпечує повний цикл розроблення й виконання програм у текстовому режимі. Мова Free Pascal підтримується також вільним інтегрованим середовищем із графічним інтерфейсом Lazarus.



Середовище програмування Lazarus забезпечує створення й виконання програм як із графічним (віконним) інтерфейсом, так і в текстовому (консольному) режимі.



Використання мови Pascal та середовищ Free Pascal і Lazarus здійснюється на основі відкритої для всіх ліцензійної угоди.

Середовище програмування Free Pascal сумісне з Borland Pascal і Object Pascal – Delphi. Компілятор Free Pascal реалізований для ОС Windows і Linux. Незважаючи на те що в професійному програмуванні найчастіше застосовуються мови C++, C#, Java, PHP та інші, мовою Pascal теж можна реалізувати сучасні складні інженерні й економічні обчислення.

Обов'язковими елементами мови Free Pascal є **символи, ключові слова та ідентифікатори**. У програмі застосовуються коментарі.

У програмах мовою Free Pascal дозволено використовувати такі символи:

– латинські літери A, B, C, ..., x, y, z (*компілятор не відрізняє великих літер від маленьких*);

– цифри 1, 2, 3, ..., 0;

– символи арифметичних і логічних операцій, синтаксичні та інші символи: +, -, *, /, <, >, =, \$, %, #, @, &, :, ; (,), [,], {, }, ..', ^, _.

Із символів мови складаються **ідентифікатори величин і ключові слова**.

Ключові слова мають строго визначене призначення і використовуються для позначення операторів мови, типів даних тощо. Змінювати їх не можна. Таких слів налічується декілька десятків, наприклад: **read, integer, writeln, const, else, case**.

Ідентифікатори призначені для іменування різних об'єктів (змінних, констант, міток тощо). Вони складаються з літер, цифр і символів підкреслення і починаються з літери або символу підкреслення. При-

клади ідентифікаторів: a_1, i, beta, z. Ідентифікаторами не можуть бути службові слова.

У тексті програми застосовуються коментарі для пояснення сутності дій у програмі, що виконуються. На виконання програми вони не впливають. Є три види коментарів:

- у фігурних дужках – {коментар};
- починається двома косими лініями – //коментар;
- між символами (* і *) – (*коментар*).

Середовище програмування Free Pascal можна запустити на виконання звичайними засобами ОС Windows. Наприклад, на робочому столі слід двічі клацнути ярлик системи. Відкриється вікно компілятора, у якому слід виконати команду **File**→**New**. У результаті відкриється головне вікно системи, зображене на рис. 5.5. У робочому полі вікна міститься код програми.

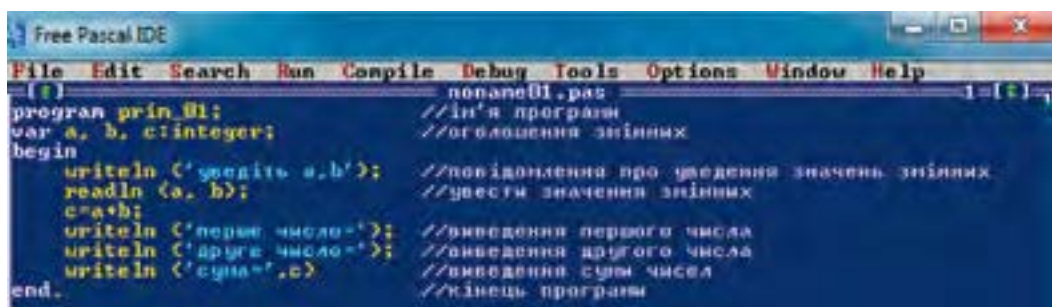


Рис. 5.5. Головне вікно системи Free Pascal з кодом програми

У верхньому рядку вікна містяться назва системи і три кнопки управління вікном.

Другий рядок – це головне меню системи, у якому десять пунктів. Більшість цих пунктів має призначення, аналогічне призначенню однойменних пунктів меню інших прикладних програм, наприклад, текстового редактора. Зокрема, команди пункту **File** забезпечують роботу з файлами програм (створення, зберігання, друкування файлів тощо), команди пункту **Edit** – редагування, копіювання, вставлення, видалення частин тексту програми. Пункт **Windows** слугує для керування вікнами, а **Help** – для отримання довідкової інформації.

✓ *Інші пункти меню мають таке призначення:*

- Search – пошук і заміна фрагментів програми;
- Run – виконання програми;
- Compile – компіляція програми;
- Debug – налагодження програми;
- Tools – використання допоміжних засобів;
- Options – налаштування середовища програмування.

Під головним меню розташоване поле з назвою імені поточного файлу `popame01.pas`, у яке вводиться програма. За замовчуванням файл записуватиметься до папки `C:\Programs` диска. У нижньому рядку перелічені функціональні клавіші та їх призначення (F1 – Help, F2 – Save та інші).

Правила роботи з інтерфейсом системи Free Pascal принципово не відрізняються від правил роботи в середовищі стандартних текстових редакторів і операційної системи Windows. Активація рядка меню може здійснюватися за допомогою миші. Для переходу в головне меню слід натиснути клавішу F10, а для повернення до робочої області – клавішу Esc.

Перевіряємо себе

1. Які складники середовища програмування є обов'язковими? ▲
2. Для чого призначене середовище програмування Lazarus? ▲
3. Якими операційними системами підтримується компілятор Free Pascal? ▲
4. Які основні дії виконуються у меню File середовища Free Pascal? ▲
5. Для чого призначене меню Compile середовища Free Pascal? ▲
6. Для чого призначені ключові слова мови програмування? ✦
7. Сформулюйте правило запису ідентифікатора. ✦
8. Які типи коментарів використовуються в мові Free Pascal? ✦

5.4. Загальні відомості про середовище візуального програмування Lazarus



Головне меню Lazarus. Форма. Компоненти на формі. Редактор тексту. Об'єкт – властивості й методи. Інспектор об'єктів. Події та їх опрацювання.

Інтегроване середовище програмування Lazarus забезпечує розроблення програм у **візуальному** та **консольному** режимах. Принципова різниця цих режимів полягає в такому. У візуальному режимі в спеціальному вікні, яке називається **формою**, спочатку створюється інтерфейс майбутньої програми, а потім розробляється програмний код.

Інтерфейс майбутньої програми будується зі спеціальних компонентів (кнопок, прапорців тощо) з різними властивостями. Компонент, перенесений на форму, стає об'єктом.

✓ *Для кожного компонента в середовищі існує програмний код, який автоматично переноситься в поле текстового редактора після розміщення на формі відповідного об'єкта.*

Програмісту не потрібно його розробляти. Отже, середовище Lazarus автоматично створює значну частину програмного коду, що сприяє прискоренню розроблення програми.

У консольному режимі інтерфейс не розробляється, одразу розробляється програмний код. Дані вводяться з клавіатури, а результати виконання коду виводяться на екран монітора.

Запуск середовища Lazarus здійснюється стандартними засобами ОС Windows, наприклад, можна двічі клацнути ярлик середовища на робочому столі або виконати команду **Пуск**→**Програми**→**Lazarus**. Відкриються вікна середовища, можливий вигляд яких зображений на рис. 5.6.

На вашому комп'ютері зовнішній вигляд середовища може відрізнятися від зображеного на рис. 5.6. Наприклад, може бути відсутнє вікно форми, а інші вікна можуть розташовуватися інакше й мати різні розміри. Кожне з вікон користувач може згорнути, змінити його розміри й місце розташування за допомогою стандартних засобів роботи з вікнами ОС Windows.

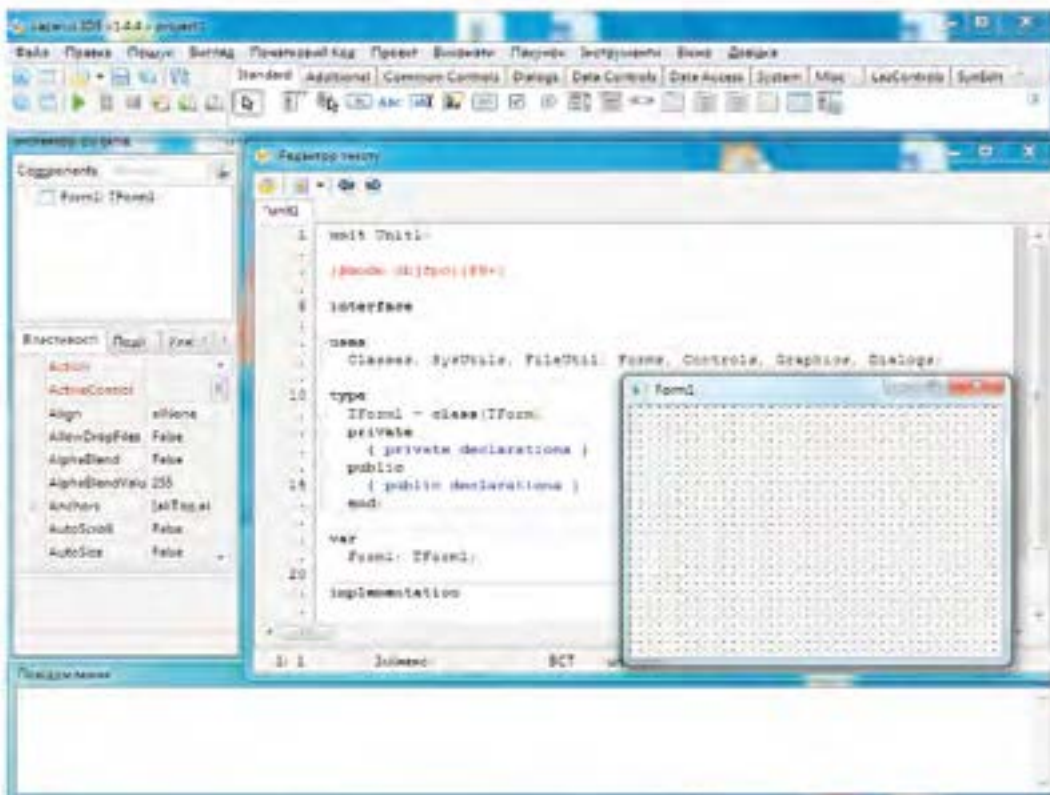


Рис. 5.6. Вікна середовища програмування Lazarus




Головне вікно Lazarus (рис. 5.7) складається з трьох частин: головного меню (**Файл**, **Правка** та інші), панелі інструментів (ліворуч під головним меню) і палітри компонентів на вкладках **Standard**, **Additional** та інших.

Головне вікно відкрите протягом усього часу роботи середовища. Якщо його закрити, закриються всі вікна й здійсниться вихід із середовища.






Рис. 5.7. Заголовок головного вікна середовища Lazarus

Призначення окремих пунктів головного меню аналогічне призначенню однойменних пунктів інших прикладних програм, зокрема, стандартного редактора тексту. Наприклад, команди меню **Файл** призначені для роботи з файлами, команди меню **Правка** – для редагування початкового коду програми, команди меню **Пошук** забезпечують пошук необхідної інформації, а команди меню **Вигляд** використовуються для налагодження зовнішнього вигляду елементів середовища.


На панелі інструментів розташовані кнопки команд, що часто використовуються. Вони дублюють деякі команди головного меню. Назви цих кнопок з'являються після встановлення на них курсора миші. Тут містяться, зокрема, кнопки **Нова форма** () , **Зберегти** () , **Виконати** () . Кнопки цієї панелі призначені для зручності й прискорення роботи користувача.

На палітрі компонентів розташована значна кількість компонентів, які користувач може розміщувати на формі, створюючи інтерфейс майбутньої програми. Ці компоненти згруповані за функціональним призначенням.

За замовчуванням відкриваються компоненти групи **Standard** (див. рис. 5.7). На початковому етапі вивчення середовища Lazarus компонентів групи **Standard** для оформлення вистачає нескладного інтерфейсу програми. Найчастіше застосовуються такі компоненти: **TButton** () , **TLabel** () , **TEdit** () .

Якщо компонентів групи **Standard** не вистачає, можна скористатися компонентами групи **Additional** та інших. Для розкриття компонентів інших груп потрібно клацнути назву відповідної групи.

Для перенесення компонента на форму потрібно клацнути кнопкою миші на цьому компоненті, потім клацнути нею в тому місці форми, де передбачається його розмістити. Якщо на палітрі компонентів двічі клацнути відповідний компонент, то він розташується в лівому верхньому куті вікна форми.

✓ *Якщо потрібно відмовитися від перенесення на форму вибраного компонента, слід клацнути кнопкою зі стрілкою, розташованою ліворуч на вкладці () .*

! **Компонент, перенесений на форму, стає об'єктом. Кожний об'єкт має властивості й методи.**

- ✓ *Властивості визначають зовнішній вигляд об'єкта, наприклад, його ім'я, розміри, колір.*
- ✓ *Методи визначають поведінку об'єкта, наприклад, як він реагуватиме на натиснення кнопки миші, зміну його розміру.*

Для стандартних об'єктів програміст не розробляє методи, він використовує їх під час розроблення програми. Для налагодження властивостей об'єктів застосовується **Інспектор об'єктів**. Значення деяких властивостей можуть змінюватися також програмно.

✓ У інспекторі об'єктів найчастіше застосовуються вкладки **Властивості** та **Події**. Кожна вкладка має таблицю з двома колонками. У лівій колонці розташовані назви властивості або події, а у правій – значення властивості або імена обробників подій. Для вибору властивості або події необхідно клацнути кнопкою миші по відповідному рядку, після чого можна вибрати або ввести її значення. На рис. 5.8 представлено вікно інспектора подій з інформацією про нову форму.

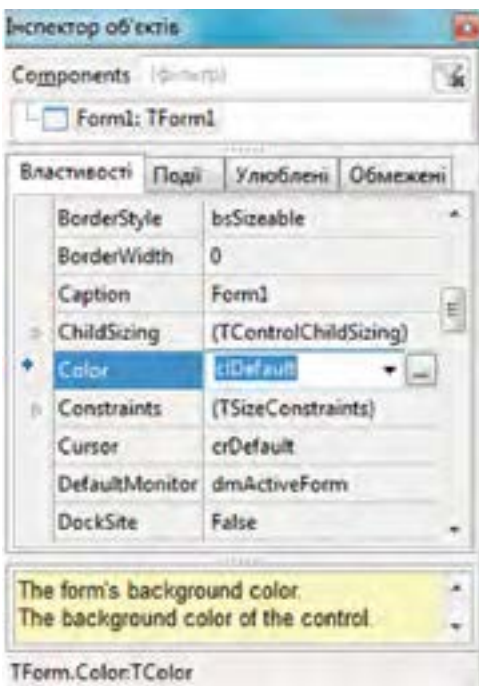


Рис. 5.8. Вікно Інспектора об'єктів із властивостями форми

Властивості поділяються на прості й складні. Прості визначаються лише одним значенням. Наприклад, властивість Caption (Заголовок) визначається рядком символів, а властивість Height (Висота) – числом.

Складні властивості визначаються сукупністю значень. Складною є, наприклад, властивість Font (Шрифт). Ліворуч від назви складних властивостей розташований трикутник. Якщо клацнути по світлому трикутнику (після цього він набуде темного кольору), відкриється список його значень, по темному – список згорнеться. Біля простих властивостей трикутника немає.

Активація будь-якої властивості здійснюється клацанням по ній кнопкою миші. Після активації властивості в кінці рядка можуть з'явитися кнопка з трьома точками і кнопка зі стрілкою вниз. Після клацання кнопки з трьома точками відкриється діалогове вікно для вибору відповідних значень. Наприклад, після активації такої кнопки властивості Color (див. рис. 5.8) відкриється вікно **Колір**. Після активації кнопки трикутника розкриється список можливих значень цієї властивості.

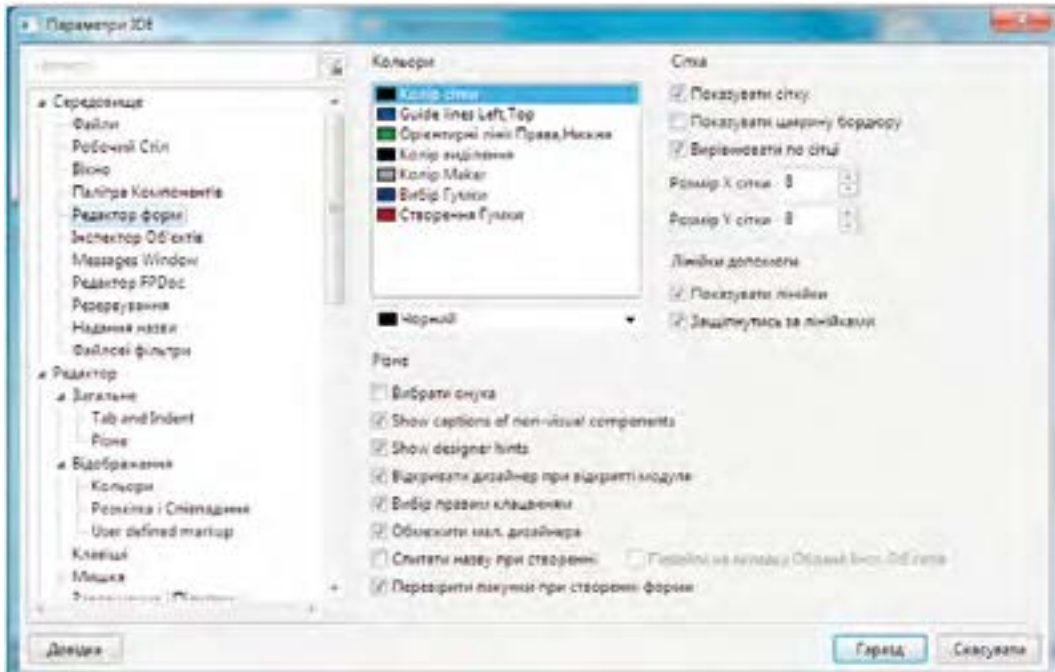


Рис. 5.9. Вікно для налагодження форми

У вікні **Редактор тексту** (див. рис. 5.6) розміщується програмний код. Рядки у вікні пронумеровані, що досить зручно для роботи користувача. Ключові слова виділені жирним, знаки пунктуації – червоним кольором, коментарі – синім, помилки – коричневим. У редакторі тексту наведена деяка структура майбутньої програми, процедури і функції. Її можна розглядати як деякий шаблон, що полегшує роботу програміста під час створення програми.

Робота в графічному режимі середовища Lazarus передбачає спочатку створення інтерфейсу програми, а потім – уведення програмного коду. Для створення графічного інтерфейсу використовуються компоненти палітри компонентів, які розміщуються на формі, а їх властивості встановлюються за допомогою інспектора об'єктів. Після розміщення компонентів на формі у вікні редактора тексту **автоматично генерується відповідний програмний код.** Користувач повинен доповнити його потрібними командами для розв'язування поставленого завдання. Усі файли, з яких складається програмний код, називаються **проектм**.

✓ **Форма** – вікно, в якому створюється інтерфейс програми користувача.

На початку роботи вікно форми порожнє (рис. 5.10) і містить лише заголовок та кнопки управління вікном. Користувач має заповнити форму необхідними компонентами з палітри компонентів для створення зовнішнього вигляду вікна програми, яку він розроблятиме.

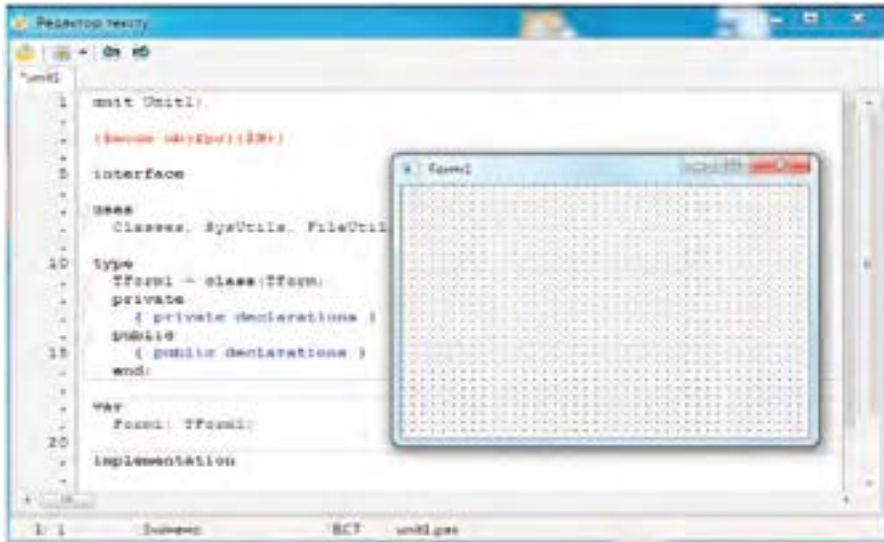


Рис. 5.10. Вікна редакторів тексту і форми

Налагодження вікна форми виконується за допомогою елементів керування і команд, розташованих у вікні **Параметри IDE** (англ.: Integrated Development Environment – інтегроване середовище розробки) (рис. 5.9).

Для відкриття цього вікна слід виконати команду **Інструменти**→**Параметри...** й у вікні, що відкриється, відкрити вкладку **Редактор форм**. У процесі створення нового проекту вікно редактора тексту відображається на екрані разом із формою (рис. 5.10).

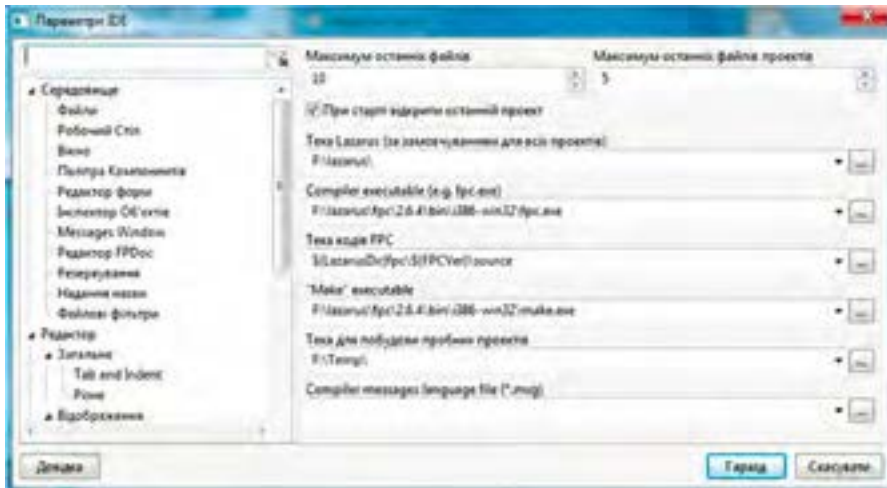


Рис. 5.11. Вікно Параметри IDE на вкладці Файли

За замовчуванням на передньому плані розташоване вікно редактора тексту, що закриває форму. Активувати вікно можна за допомогою клавіші F12 або команди **Вигляд**→**Перемкнути форму/модуль**.

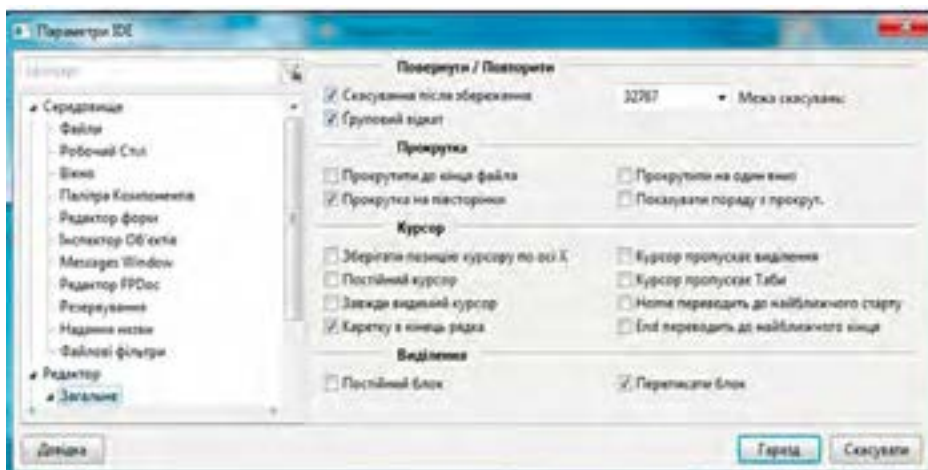


Рис. 5.12. Вікно Параметри IDE на вкладці Загальне

Для того щоб у вікні редактора тексту у процесі завантаження системи Lazarus автоматично з'являвся останній створений проект, необхідно за допомогою команди **Інструменти** → **Параметри...** відкрити вікно **Параметри IDE**, на вкладці **Файли** (рис. 5.11) увімкнути прапорець **При старті відкрити останній проект** і натиснути кнопку **Гаразд**.

Налагодження вікна редактора тексту виконується у вікні **Параметри IDE** на вкладках **Редактор** (див. рис. 5.11). Зміст вікна на вкладці **Загальне** зображено на рис. 5.12.

Зміна шрифту тексту програмного коду виконується на вкладці **Відображення**. Зміст вікна на цій вкладці зображено на рис. 5.13.

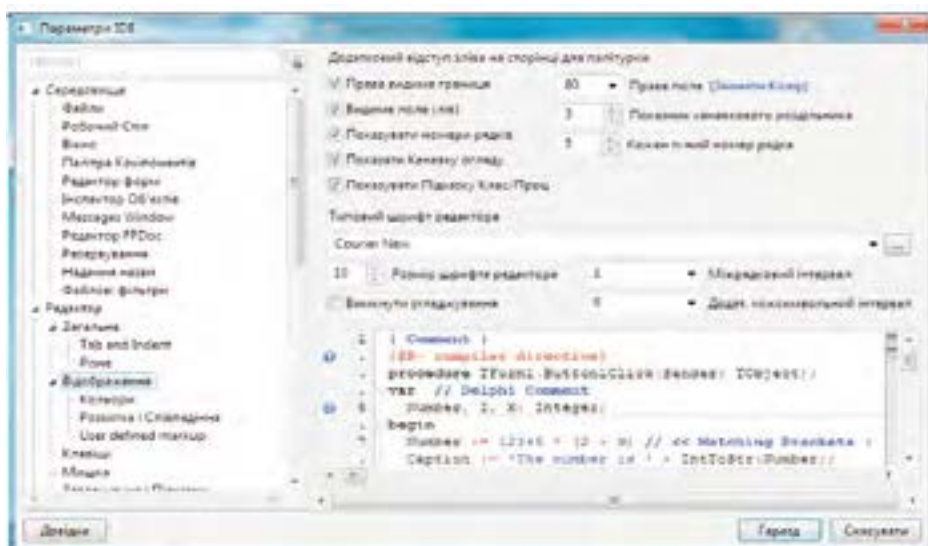


Рис. 5.13. Вікно для зміни шрифту програмного коду

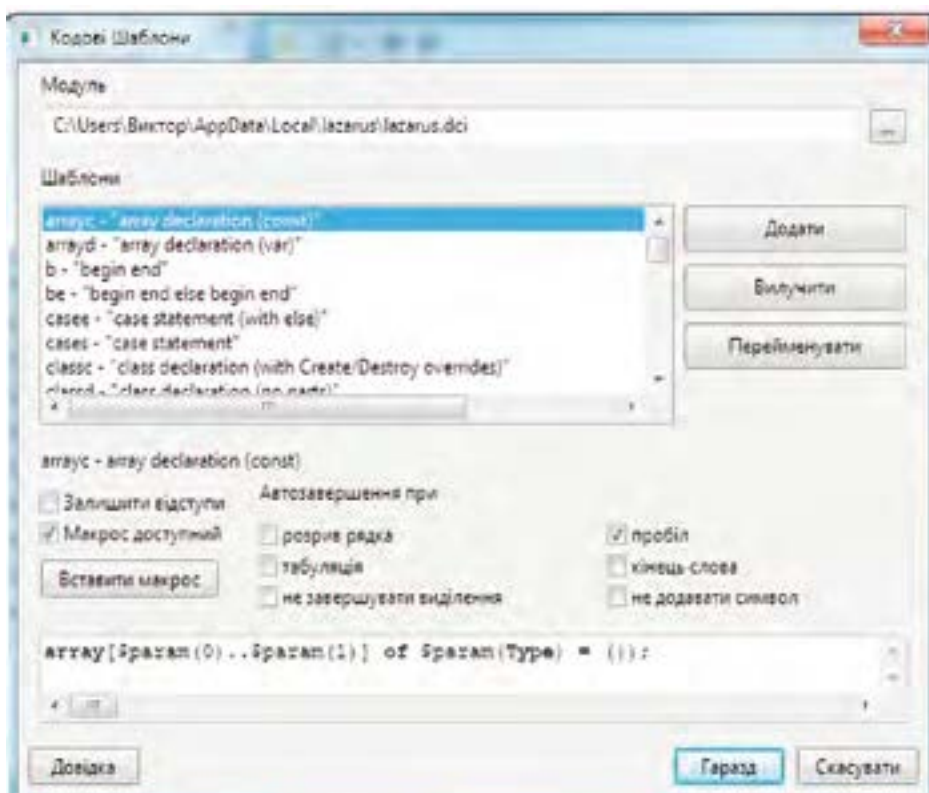


Рис. 5.14. Вікно для налагодження шаблонів

Редактор тексту забезпечує роботу з шаблонами. Як відомо, шаблон – це конструкція мови програмування, яка полегшує уведення програмного коду.

Налагодження шаблонів виконується у вікні **Кодові шаблони** (рис. 5.14), яке відкривається за допомогою команди **Інструменти – Кодові шаблони...** У цьому вікні наведено перелік шаблонів, які може застосовувати користувач. Для того щоб функціонував, наприклад, шаблон **begin end**; необхідно клацнути по ньому кнопкою миші, увімкнути прапорець **пробіл** і натиснути кнопку **Гаразд**. Тепер після введення символу **b** і натиснення клавіші **пробіл** автоматично з'являтиметься шаблон **begin end** ; .





Користувач може додавати нові шаблони, вилучати непотрібні, редагувати існуючі, використовуючи прапорці, кнопки та інші елементи вікна. Ці операції виконуються аналогічно відповідним операціям у середовищі текстового редактора.

Перевіряємо себе

1. Для чого призначене середовище Lazarus? ▲
2. Як можна запустити середовище Lazarus? ▲

3. Які операції можна виконувати над вікнами середовища Lazarus? ▲
4. Як переносяться компоненти палітри компонентів середовища Lazarus на форму? ▲
5. Для чого призначена форма? ▲
6. Як можна викликати вікно форми на передній план? ▲
7. Назвіть основні компоненти середовища Lazarus. ★
8. Які вікна має середовище Lazarus? ★
9. Із яких частин складається головне вікно середовища Lazarus? ★
10. Для чого призначена палітра компонентів середовища Lazarus? ★
11. Для чого призначена панель інструментів головного вікна середовища Lazarus? ★
12. Для чого призначений редактор тексту середовища Lazarus? ★
13. Для чого призначений інспектор об'єктів? ★
14. Поясніть сутність простих і складних властивостей. ★
15. Як налагоджується вікно форми? ★
16. Як можна змінити шрифт у вікні редактора тексту? ★
17. Для чого призначені шаблони? ★

Виконуємо

1. Запустіть середовище Lazarus. Відкрийте вікно форми. Змініть розміри та місце розташування вікна форми. ▲
2. Розмістіть вікна середовища Lazarus так, як вам сподобається. ▲
3.  Перенесіть на форму компоненти TButton і TEdit. Змініть їх розміри та місце розташування. Встановіть їх колір і назви. ★
4.  Надайте формі назву Початок і потрібний колір. Використовуючи вікно для налагодження форми, встановіть інші властивості форми. ★
5.  Послідовно встановіть курсор миші на всіх кнопках панелі інструментів. Спробуйте з'ясувати призначення кожної з них. ★
6.  У групі Standard послідовно встановіть курсор миші на всіх компонентах палітри компонентів. З'ясуйте призначення основних з них. ★

5.5. Основи візуального програмування в середовищі Lazarus



Візуальне програмування. Об'єкти та їх властивості. Класи і методи. Об'єктно-орієнтоване програмування. Подійне програмування.

Концепція середовища Lazarus ґрунтується на проектуванні графічного інтерфейсу користувача на основі форми, яка є аналогом вікон Windows. Основною структурною одиницею програмування в середовищі Lazarus є

компоненти – об’єкти (кнопка, поля-списки, текстові поля та інші), які вибираються з палітри компонентів середовища і вставляються в форму.

✓ *Середовище Lazarus містить компоненти, які є об’єктами й можуть використовуватися для оформлення графічного інтерфейсу.*

Вони згруповані за функціональним призначенням. Найчастіше застосовуються компоненти групи Standard, яка містить стандартні елементи інтерфейсу. На початковому етапі вивчення основ візуального програмування використовуються компоненти саме цієї групи.

Іноколи використовуються також компоненти групи Additional, яка містить додаткові компоненти для роботи з діалоговими вікнами. Компоненти інших груп на цьому етапі майже не застосовуються.

✓ *Найуживанішими є такі компоненти групи Standard:*

TMainMenu – меню програми;

TPopupMenu – локальне меню, що викликається за допомогою правої кнопки миші;

TButton – командна кнопка;

TLabel – поле для розміщення однорядкових написів;

TEdit – однорядковий текстовий редактор;

TMemo – багаторядковий текстовий редактор;

TScrollBar – лінійка прокручування;

TGroupBox – група елементів;


TPanel – панель для об’єднання декількох компонентів;

TActionList – список дій.

✓ *Компоненти, вставлені в форму, і сама форма є об’єктами.*

Об’єкти мають власні атрибути (властивості). Їх значення можна змінювати з фіксованого набору, що містить саме середовище, а також їх може розробляти користувач. Повний перелік властивостей форми і компонентів міститься в інспекторі об’єктів на вкладці **Властивості**. У табл. 5.1 наведені основні властивості форми, більшість із яких можуть бути також властивостями інших компонентів.

Форма, а також інші об’єкти можуть **реагувати на події**. Перелік можливих подій міститься в **Інспекторі об’єктів** на вкладці **Події**. Подія виникає, наприклад, у разі клацання по об’єкту кнопкою миші, переміщення по об’єкту вказівника миші тощо. Завдання програміста полягає у розробленні програмного коду, який описує реакцію об’єкта на ту чи іншу подію.

 Середовище візуального програмування **Lazarus** реалізує **об’єктно-орієнтоване** програмування (ООП), а також дає змогу створювати процедури опрацювання подій.


✓ **ООП** – це методика розроблення програм, яка ґрунтується на поняттях **об’єкт**, **метод** і **клас**.

Головна ідея ООП полягає в об’єднанні даних, з якими працює програмний код, і процедур їх опрацювання у єдине ціле. Як уже зазначалося, кожний **об’єкт** має ім’я і властивості. Наприклад, об’єкт “циліндр” може мати такі властивості: радіус основи, висоту, об’єм, колір.

✓ **Опрацювання даних і подій** (“поведінка” об’єкта) та його взаємодія з іншими об’єктами описуються за допомогою **методів**, які є аналогами підпрограм процедурних мов програмування.

✓ **Клас** – сукупність способів подання даних і методів їх опрацювання.

Умовно клас можна розглядати як певний шаблон, із якого можна виготовити скільки завгодно об’єктів одного типу. Тому часто кажуть, що клас – це шаблон об’єкта, а об’єкт – екземпляр класу.

 **Подійне програмування** – програмування, в якому передбачається зміна виконання програми залежно від певних дій користувача (наприклад, клацання кнопкою миші), повідомлень від інших програм, повідомлень, що надійшли з мережі, тощо.

Для цього у кодї програми на кожну подію передбачається відповідний **обробник подій**.

Таблиця 5.1

Основні властивості форми

Властивість	Опис властивості
1	2
Caption	Заголовок форми
Color	Колір заливки форми
Cursor	Тип курсора, що з’являється на формі
Enabled	Доступність. Якщо значення цієї властивості True, то форма реагує на подію, якщо False, то не реагує. За замовчуванням форма завжди має значення True
Font	Тип, колір і розмір шрифту в об’єктах (не в заголовку форми)
Height	Висота вікна в пікселях
Left	Горизонтальна координата (X) лівого краю компоненти відносно екрана (у пікселях)
Name	Містить ім’я компонента, за яким програма здійснює звернення до нього. Кожному компоненту автоматично присвоюються імена, наприклад, Form1, Button1. Змінювати їх не рекомендується

1	2
Position	Положення вікна на екрані; <code>poDesigned</code> – після запуску застосунку форма в фіксованому положенні; <code>poDefault</code> – форма може перебувати в довільному місці; <code>poScreenCenter</code> міститься в центрі екрана
Top	Вертикальна координата (Y) лівого верхнього кута форми відносно екрана (у пікселях)
Visible	Визначає, чи буде компонент відображатися на екрані. Якщо ця властивість має значення <code>True</code> , то компонент відображається, якщо <code>False</code> , то компонента не видно
Width	Ширина об'єкта (у пікселях)

✓ *Програмування, в якому використовуються поняття “подія” і програмний код, яким описується її опрацювання (“обробник події”), називають подійним програмуванням.*

Програма в середовищі Lazarus є **проектом**. Проект об'єднує кілька файлів, з яких створюється єдиний файл, що виконується.

✓ *Мінімальний склад файлів проекту такий:*

файл опису проекту (.lpi); файл проекту (.lpr); файл ресурсів (.lrs); модуль форми (.lfm); програмний модуль (.pas).

✓ *У результаті компіляції з файлів проекту створюється єдиний файл, що виконується, з розширенням `exe`.*

Ім'я файла збігається з іменем проекту. Користувач розробляє програмний модуль, усі інші додаються до проекту автоматично.

✓ *Програмний модуль має таку загальну структуру:*

```

unit ім'я модуля;           //заголовок модуля
interface                  //розділ опису (список модулів, типів, кон-
стант, змінних, процедур і функцій)
implementation //розділ коду програми
end.                        //кінець модуля

```


✓ *Розділ опису має таку загальну структуру:*


```

interface
uses список_модулів;
type список_типів;
const список_констант;
var список_змінних;
procedure ім'я_процедури;
...
function ім'я_функції;

```


Заголовок модуля починається ключовим словом **unit**, за ним – ім'я модуля і крапка з комою. Розділ опису починається ключовим словом **interface**. Тут описуються компоненти програмного коду: типи, класи, процедури і функції. Розділ **implementation** містить програмний код опрацювання даних, який розробляє користувач.

 Користувач розробляє **лише власний код програми**, що реалізує поставлене завдання, інший необхідний код **додається середовищем автоматично** і його **не потрібно змінювати**.

 *Середовище Lazarus забезпечує розроблення також звичайних консольних програм.*

Консольна програма має таку структуру:

```
program ім'я програми;  
uses modul1, modul2, ..., modulN;  
const опис констант;  
type опис типів;  
var опис змінних;  
begin //далі розташовані оператори мови (тіло програми)  
оператор_1;  
оператор_2;  
...  
оператор_N;  
end. //кінець програми
```

Тіло програми починається ключовим словом **begin**. Оператори в мові Pascal відокремлюються один від одного крапкою з комою. Тіло програми закінчується словом **end**.

Перевіряємо себе

1. У чому полягає сутність концепції візуального програмування в середовищі Lazarus? ▲
2. Що називають об'єктом у середовищі Lazarus? ▲
3. Наведіть приклад об'єкта середовища Lazarus та його властивостей. ▲
4. Чи може користувач визначити нестандартні властивості об'єкта? ▲
5. Що називають проектом у середовищі Lazarus? ▲
6. Яким ключовим словом починається розділ опису? ▲
7. Яким ключовим словом починається програмний код опрацювання даних? ▲
8. Назвіть основні поняття об'єктно-орієнтованого програмування. ✦
9. У чому полягає основна ідея об'єктно-орієнтованого програмування? ✦
10. Поясніть сутність методу об'єктно-орієнтованого програмування. ✦

11. Поясніть сутність класу в об'єктно-орієнтованому програмуванні. ✦
12. У чому полягає сутність подійного програмування? ✦
13. Яке розширення мають файли, які розробляє користувач? ✦
14. Які основні файли входять до складу проекту? ★
15. Яку загальну структуру має програмний модуль? ★
16. Яку загальну структуру має розділ опису? ★
17. Назвіть основні компоненти розділу опису. ★
18. Яку загальну структуру має консольна програма? ★

Виконуємо





1. Розмістіть на формі з назвою Нова форма об'єкти Label1, Button1 і Edit1 (як на рис. 5.15). Зверніть увагу, що розміри об'єктів Button1 і Edit1 змінені. ▲
2. Встановіть заголовок форми Початок. Розмістіть на формі об'єкти Button1, Button2 і Button3. Змініть місце їх розташування та розміри. Вилучіть із форми всі об'єкти. ▲
3.  На формі з іменем Домашнє завдання розмістіть об'єкти Label1, Label2 і Button1. Встановіть раціональні, на ваш погляд, їх розміри. Самостійно визначте для них текст і назви. ▲
4.  Надайте формі ім'я Мій проект. Розмістіть на формі об'єкти Memo1 і Button1. Змініть розміри об'єктів так, як зображено на рис. 5.16, а для об'єкта Button1 встановіть властивість **натисни**. ✦
5.  На формі з іменем Спільна розмістіть об'єкти PopupMenu, CheckBox і Memo. Експериментально спробуйте визначити їх функціональне призначення. ★
6.  Виберіть із палітри компонентів необхідні для створення форми, зображеної на рис. 5.17. ★



Рис. 5.15. Приклад розміщення об'єктів на формі

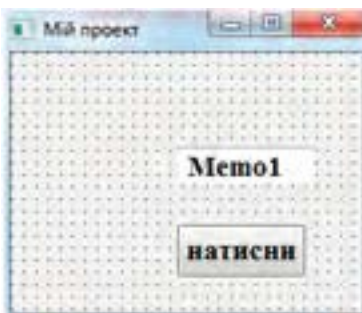


Рис. 5.16. Приклад форми й об'єктів з властивостями

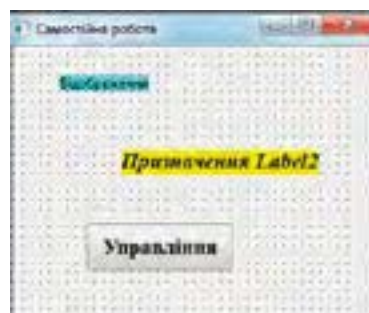


Рис. 5.17. Приклад визначення назв об'єктів

5.6. Створення проекту в режимі візуального програмування середовища Lazarus



Графічний інтерфейс. Компоненти графічного інтерфейсу. Створення проекту.

Існують різні підходи до створення проектів у візуальному режимі середовища Lazarus.

Але загальний порядок їх розроблення однаковий.

1. Створюється графічний інтерфейс програми.
2. Розробляється програмний код мовою Free Pascal, за допомогою якого будуть функціонувати елементи інтерфейсу.
3. Компілюється, зберігається і виконується проект.

Для створення інтерфейсу використовується вікно форми, а для введення коду – вікно редактора тексту. За будь-якої зміни змісту форми автоматично змінюється програмний код у вікні редактора тексту.

Оскільки ми ще не знайомі зі синтаксисом мови програмування Free Pascal, методику створення проекту розглянемо на найпростішому прикладі.

Нехай на формі розташована кнопка із написом “Натисніть”. Після натискання цієї кнопки з’являється повідомлення “Молодець”.

Дотримуватимемося такої послідовності дій:

1. Після завантаження середовища Lazarus створюємо новий проект. Для цього виконуємо команду **Проект** → **Новий проект...** . Відкриється діалогове вікно **Створити новий проект** (рис. 5.18).

2. Вибираємо верхній рядок із назвою **Програма** і натискаємо кнопку **Гаразд**. На екрані будуть розташовані вікна форми і редактора тексту.

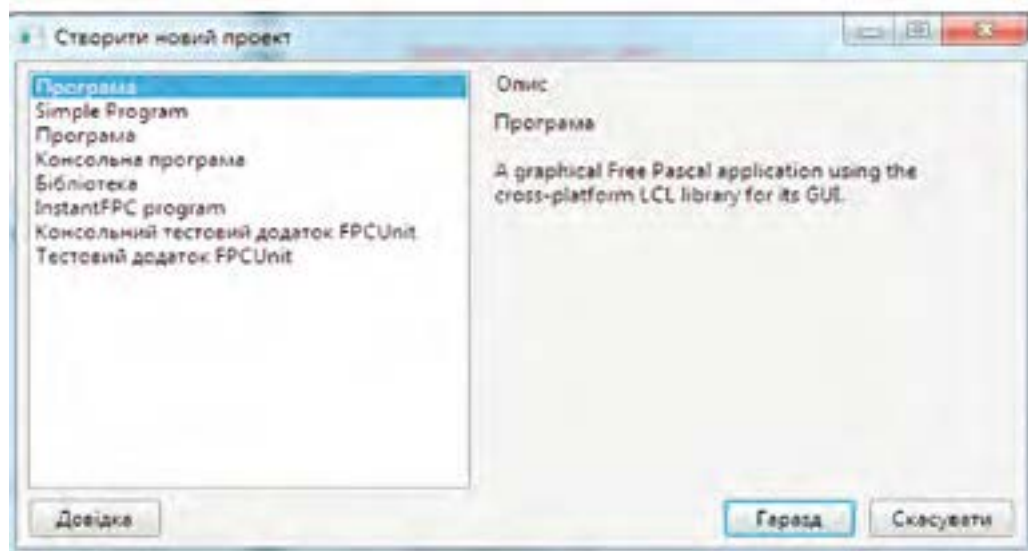


Рис. 5.18. Вікно створення нового проекту

3. Зберігаємо створений проект. Для цього виконуємо команду **Проект** → **Зберегти проект як...** Відкриється вікно **Зберегти проект project1 (*.lpr)**, зображене на рис. 5.19.

4. Створюємо папку, в якій зберігатимуться файли проекту. Для цього в адресне поле цього вікна можна встановити бажане місце збереження папки (у нашому прикладі для цього пропонується папка **lazarus**, що зберігається у кореневому каталозі диска **F:**). Можна вибрати інше місце, але ми не будемо змінювати запропоновану папку і стандартними засобами ОС Windows, наприклад, за допомогою кнопки **Створити папку**, створимо тут папку з іменем **Pro_01**. У цій папці зберігатимуться файли нашого проекту (нагадаємо, що проект міститиме кілька файлів).

5. Після створення папки відкриваємо її (кляцанням на її імені) і зберігаємо, для чого натискаємо кнопку **Зберегти**: зберігається файл **Project1**, який містить відомості про проект. Одразу відкриється вікно для збереження програмного коду з назвою **Зберегти unit1 (*.pas)**, зображене на рис. 5.20. У цьому вікні також натискаємо кнопку **Зберегти**. Отже, проект збережено. Якщо тепер відкриємо папку **Pro_01**, побачимо в ній перелік файлів, потрібних для подальшої роботи.

Зокрема, тут є файл із текстом програми **unit1.pas**, файл **unit1.lfm**, що містить відомості про форму **Form1**, файл про параметри налагодження системи програмування – **project1.lpr**, файл **project1.lpi** з параметрами конфігурації проекту. Далі після будь-яких змін у проекті необхідно зберегти їх за допомогою команди **Проект**→**Зберегти**.

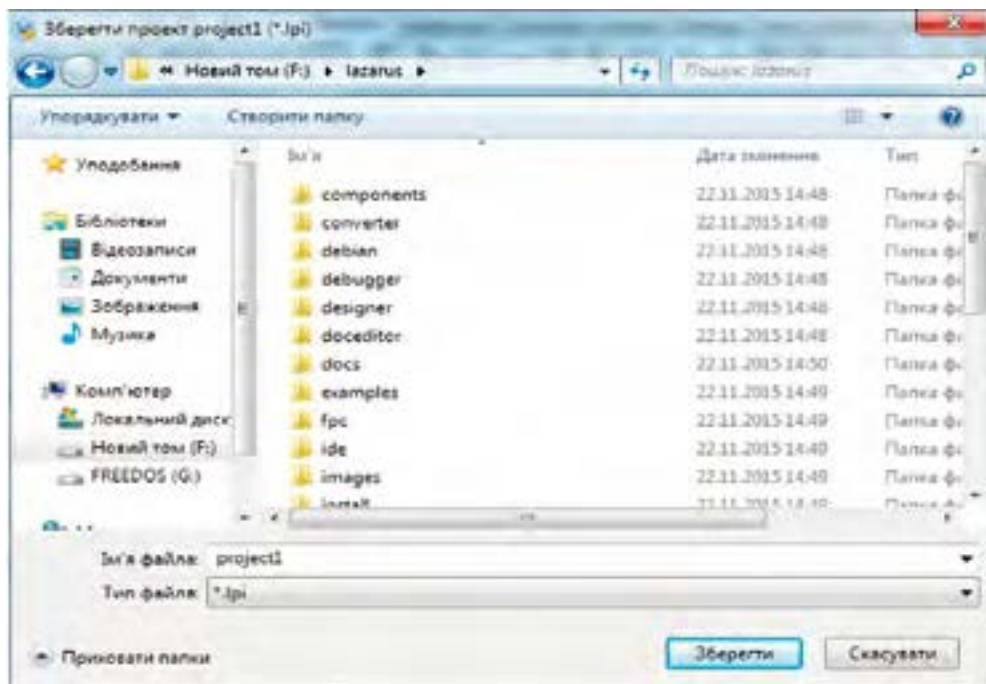


Рис. 5.19. Вікно збереження проекту

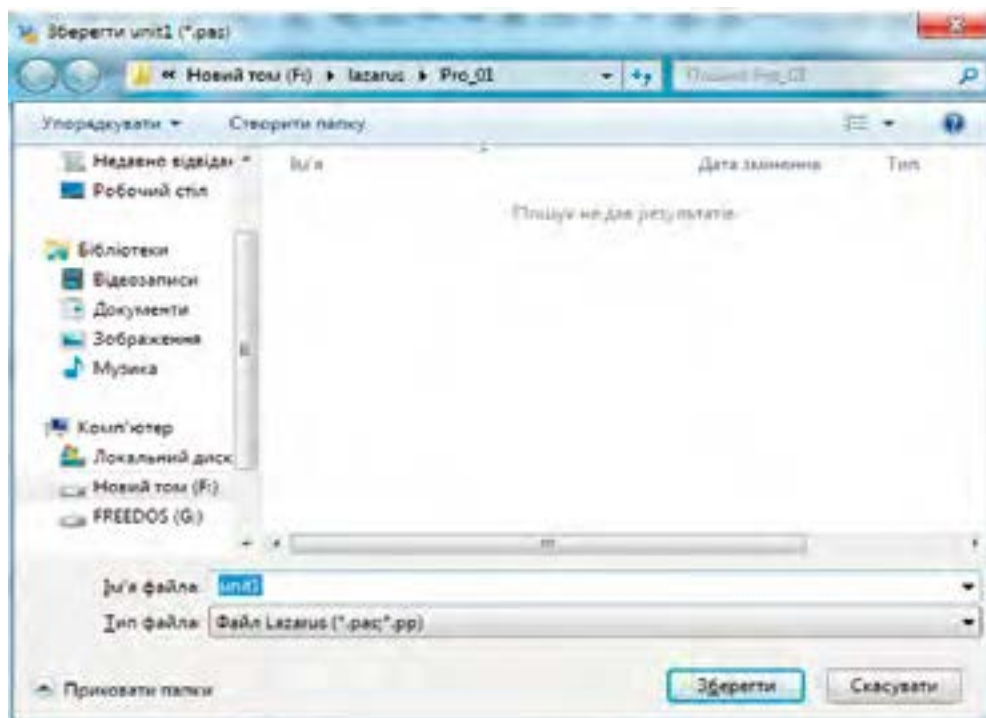


Рис. 5.20. Вікно збереження програмного коду

6. Перейдемо до візуального програмування. На цей момент маємо лише один об'єкт – Form1. Надамо формі іншу властивість, а саме – назву Приклад_01. Для цього у вікні інспектора об'єктів знаходимо властивість Caption (Заголовок). За замовчуванням вона має значення Form1. Виділяємо рядок Caption, вводимо назву Приклад_01 і натискаємо клавішу Enter. Ця назва одразу з'явиться в заголовку форми. Аналогічно можна присвоїти нові значення властивостям Height (висота) і Width (ширина). Нагадаємо, що змінити розміри вікна форми можна і стандартними засобами роботи з вікнами ОС Windows. Змінити місце розташування форми на екрані можна за допомогою властивості Position (місце форми на екрані). Після вибору цієї властивості у правому стовпці властивості з'явиться кнопка трикутника. Натиснемо на неї, в результаті чого відкриється список можливих значень цієї властивості (рис. 5.21).

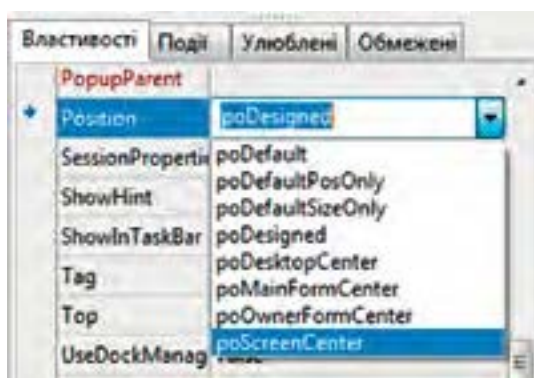


Рис. 5.21. Перелік значень властивості Position


Якщо вибрати значення властивості `poScreenCenter`, то форма буде розташована в центрі екрана.

7. Розмістимо на формі кнопку. Для цього на панелі компонентів двічі клацнемо компонент `TButton`. У результаті об'єкт `Button1` з'явиться на формі (рис. 5.22).



Рис. 5.22. Вікно форми з кнопкою

Кнопку можна перетягнути у будь-яке місце форми. Тепер у нас є два об'єкти: форма і кнопка.

 У вікні інспектора об'єктів можна змінювати значення властивостей лише вибраного (активованого) об'єкта.

Змінити ширину і висоту об'єкта `Button1` можна як стандартними засобами Windows, так і редагуванням значень властивостей вікна в **Інспекторі об'єктів**.

8. Надамо об'єкту `Button1` властивість “Клацніть мишею”. Виділяємо об'єкт, у вікні інспектора вибираємо `Caption`, уводимо нове значення властивості і натискаємо клавішу `Enter`. Цей текст одразу з'явиться всередині кнопки (рис. 5.23).

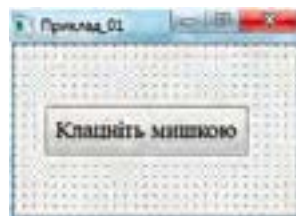



Рис. 5.23. Вікно форми з назвою кнопки

9. Отже, ми бажаємо, щоб після клацання цієї кнопки замість фрази “Клацніть мишею” з'явилося слово “Гаразд”. Для цього у вікні інспектора об'єктів обираємо кнопку `Button1` і вибираємо подію `OnClick` (опрацювання клацання миші) і двічі клацаємо поле праворуч від назви `OnClick`.

 У результаті у вікні редактора тексту з'явиться текст:

```
procedure TForm1.Button1 Click (Sender: TObject);
begin
end;
```

Це фрагмент програмного коду (процедури) опрацювання події `Click` (клацання миші). Команди, що будуть розташовані між словами `begin` і `end`, виконуватимуться як результат події.

10. Установимо курсор у рядок між словами `begin` і `end` й уведемо `Button1.Caption := 'Гаразд';`.

Ця команда означає: присвоїти властивості `Caption` об'єкта `Button1` значення `Гаразд`. Збережемо створену програму за допомогою команди **Проект** → **Зберегти проект**.

11. Виконуємо компіляцію і запускаємо програму на виконання.

Це можна зробити різними способами: за допомогою команди **Виконати** → **Компілювати**, **Виконати** → **Виконати**, кнопки **Виконати** на панелі інструментів або натиснувши клавішу **F9**.



Рис. 5.24. Результат виконання програми

У результаті виконаних дій у папці нашого проекту буде створено ехе-файл, а у вікні **Повідомлення** буде виведено протокол створення проекту.





Якщо в програмі буде виявлено помилки, то про них висвітляться повідомлення також у цьому вікні. Після виконання програми форма набуде вигляду, як на рис. 5.24.

На цьому створення проекту завершується. Пам'ятаймо, що після будь-яких змін у проекті потрібно виконати команду **Проект** → **Зберегти проект**.

Перевіряємо себе

1. За допомогою якої команди створюється проект? ▲
2. Для чого створюється папка у процесі розроблення проекту? ▲
3. Поясніть, як надати значення властивостям форми. ▲
4. Як можна змінити місце розташування кнопки на формі? ▲
5. Якими способами можна виконати компіляцію і запуск програми? ▲
6. Яку роль виконує форма в розробленні проекту? ◆
7. Для чого призначене вікно редактора тексту? ◆
8. Яка інформація виводиться у вікні повідомлення? ◆
9. Назвіть основні файли, які зберігаються в папці проекту. ★
10. Як можна надати значення новим властивостям кнопки? ★
11. Як призначаються події для об'єктів, розташованих на формі? ★
12. Поясніть сутність команди `Button2.Caption:='Вітаю!';`. ★

Виконуємо

1. Розмістіть на формі об'єкти `Button1`, `Edit1`, `Label1` із властивостями, зображеними на рис. 5.25. ▲
2. Для об'єктів, зображених на рис. 5.25, розробити код, за допомогою якого після натиснення кнопки `Button1` у полі `Edit1` з'явиться текст, уведений у поле `Label1`. ★
3.  Розмістіть на формі об'єкти `Button1`, `Button2` і `Edit1`. Розробити проект, який реалізує таке завдання: після клацання кнопки `Button1` у текстовому полі з'являється слово `Вітаю`, а після клацання кнопки `Button2` поле стає порожнім. ◆
4.  Виберіть необхідні компоненти й розробіть проект, у результаті виконання якого об'єкти матимуть властивості, зображені на рис. 5.26. ★
5.  Розмістіть на формі об'єкти `Button1` і `Edit1`. Надайте цим об'єктам властивості, зображені на рис. 5.27. Якщо натиснути на кнопку `Змінити текст`, у текстовому полі з'явиться текст, уведений користувачем у текстове поле. Розробіть проект, що реалізує це завдання. ★
6.  Доповніть форму, зображену на рис. 5.27, об'єктами `Button2` і `Edit2`. Самостійно сформулюйте нове завдання і розробіть проект, що його реалізує. ★

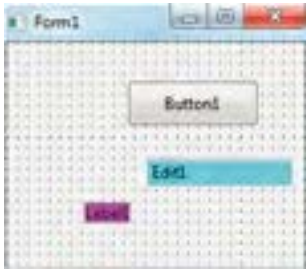


Рис. 5.25. Форма з об'єктами для встановлення їх властивостей



Рис. 5.26. Форма з об'єктами для розроблення програмного коду



Рис. 5.27. Форма з двома об'єктами для розроблення програмного коду

5.7. Розроблення проекту з простим кодом



Проект із програмним кодом. Змінні та команди програмного коду. Взаємодія об'єктів

Взаємодія об'єктів. У попередньому розділі ми розглянули загальний порядок створення проекту в середовищі Lazarus. Оскільки мову Free Pascal ми ще не вивчали, то в наведеному прикладі жодне опрацювання даних не виконували. Фактично розроблення проекту було обмежене розробленням інтерфейсу з опрацюванням події клацання кнопки.

Розробимо проект для реалізації такого завдання. Друзі з Києва і Житомира домовилися про зустріч. Вони одночасно виїхали з міст на велосипедах і рухалися по шосе назустріч із різною швидкістю. Відстань між Києвом і Житомиром – 140 км. Визначити, через скільки годин вони зустрінуться.

Дотримуватимемося такої послідовності в роботі.

1. Створюємо проект. Нагадаємо, що для цього слід виконати команду **Проект**→**Новий проект**. Відкриється вікно **Створити новий проект**, у якому виділяємо верхній рядок Програми і натискаємо кнопку **Гаразд**. Зберігаємо проект за допомогою команди **Проект**→**Зберегти проект як...**

2. Визначаємо місце збереження проекту. Обираємо, наприклад, кореневу папку диска **F:**. За допомогою кнопки **Створити папку** створюємо тут папку з іменем, наприклад, **Proba_01**. Потім відкриваємо і зберігаємо цю папку. Так будуть збережені відомості про проект у файлі **project1** і відкриється вікно **Зберегти unit1(*.pas)**.

3. Натискаємо кнопку **Зберегти**. Нагадаємо, що після цього в папці **Proba_01** будуть збережені основні файли проекту. Пам'ятаємо, що після будь-яких змін у проекті необхідно виконати команду **Проект** → **Зберегти**.

4. Надаємо властивостям форми значень, поданих у табл. 5.2.

Таблиця 5.2

Властивості форми

Назва властивості	Позначення властивості		Значення властивості
Заголовок	Caption		Зустріч друзів
Висота	Height		250
Ширина	Width		420
Шрифт	Font	Назва шрифту	Times New Roman
		Стиль	Звичайний
		Розмір	10

5. Для створення інтерфейсу обираємо чотири компоненти TLabel, два компоненти TEdit, один компонент TButton і розміщуємо їх на формі орієнтовно так, як показано на рис. 5.28.

6. Для об'єктів типу Label встановлюємо властивості, зображені на рис. 5.29, а колір об'єкта Label1 – clAqua. Для об'єктів типу Edit для властивості Text вводимо пробіл, а їх колір – clYellow.



Рис. 5.28. Варіант розміщення об'єктів на формі

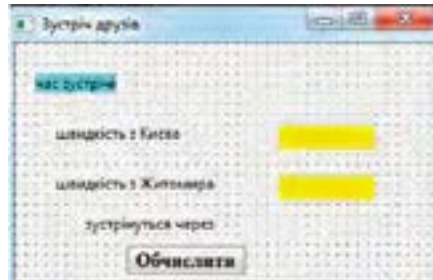


Рис. 5.29. Вигляд форми після надання властивостей об'єктам

Для об'єкта Button1 встановлюємо властивості, які містяться в табл. 5.3.

Таблиця 5.3

Властивості об'єкта Button

Назва властивості	Позначення		Значення
Текст на кнопці	Caption		Обчислити
Шрифт	Font	Назва	Times New Roman
		Стиль	Жирний
		Розмір	12

На цьому розроблення інтерфейсу програми завершується. Розпочинаємо розроблення програмного коду.

Уведемо в поля Edit1 і Edit2 значення швидкості руху велосипедистів. Для цього виділяємо на формі об'єкт Edit1, в інспекторі об'єктів вибираємо

рядок Text, праворуч від нього уводимо, наприклад, число 16 і натискаємо клавішу Enter. Аналогічно в поле об'єкта Edit2 вводимо число 17.

7. Для об'єкта Button1 створимо обробник подій OnClick. Для цього виділяємо цей об'єкт, у вікні інспектора об'єктів вибираємо рядок OnClick і двічі клацаємо лівою кнопкою миші праворуч від цієї назви.

8. На передньому плані має бути вікно редактора тексту, а курсор встановлено між словами begin і end.

Уводимо такий програмний код:

```
procedure TForm.Button1Click (Sender:TObject);
var
v1, v2: integer;
t: real;
begin
v1:=StrToInt (Edit1.Text);
v2:=StrToInt (Edit2.Text);
t=140/(v1+v2);
Label4.Caption:='зустрінуться через - '+FloatToStr(t);
end;
```

✓ Що означає кожний рядок уведеного програмного коду.

Рядок 1 – заголовок програми.

Рядок 2 – початок розділу оголошення змінних, які будуть використовуватися в програмі.

Рядок 3 – оголошуються змінні v_1 і v_2 цілого типу (integer). У змінній v_1 зберігатиметься швидкість руху велосипедистів з Києва, а у змінній v_2 – з Житомира.

Рядок 4 – оголошується змінна t – час, через який зустрінуться велосипедисти.

Рядок 5 – словом begin починається блок обчислення і виведення часу зустрічі.

Рядок 6 – командою в цьому рядку виконуються складні дії, з якими ознайомимося пізніше. Спрощено будемо вважати, що команда цього рядка присвоює число в полі Edit1 змінній v_1 . Фактично за допомогою цієї команди властивість Text об'єкта Edit1 присвоюється змінній v_1 .

Рядок 7 – властивість Text об'єкта Edit2 присвоюється змінній v_2 (інакше кажучи, число з поля Edit2 присвоюється змінній v_2).

Рядок 8 – обчислюється значення часу зустрічі й присвоюється змінній t .

Рядок 9 – обчислене значення виводиться в поле Label4.

Рядок 10 – кінець блоку програми.

9. Виконаємо компіляцію. Для цього виконуємо команду **Виконати** → **Компілювати**. У наведеному вище коді у восьмому рядку ми навмисне допустили синтаксичну помилку (восьмий рядок – у присво-

юванні значення змінній відсутній символ двокрапка “:”), щоб звернути увагу на те, що за наявності синтаксичних помилок компіляція не може бути виконана. Тоді генерується повідомлення про помилку, подане на рис. 5.30.

У програмному коді рядок, у якому допущено помилку, виділено іншим кольором. Виправляємо помилку і ще раз компілюємо програму. На цей раз компіляція завершується успішно, про що висвічується відповідне повідомлення.

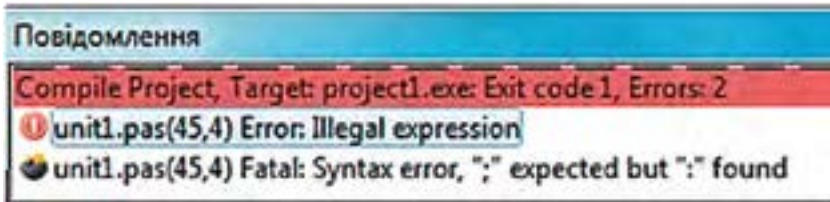


Рис. 5.30. Повідомлення про синтаксичну помилку

10. Запускаємо програму на виконання за допомогою команди **Виконати**→**Виконати**. Потім натискаємо на кнопку **обчислити**. Отримуємо результат, зображений на рис. 5.31.

Після останніх змін зберігаємо проект за допомогою команд **Проект** → **Зберегти проект**. Закрити проект можна за допомогою команди **Проект** → **Закрити проект**, а відкрити – за допомогою команди **Проект** → **Відкрити проект...**

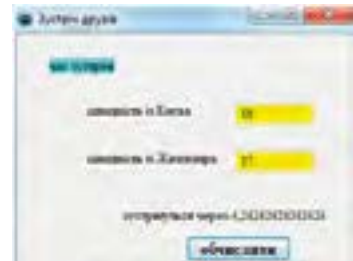










Рис. 5.31. Результат виконання програми

Перевіряємо себе

1. За допомогою якої властивості формі надається назва? ▲
2. Для чого слугує властивість Height? ▲
3. Для чого слугує компонент TEdit? ▲
4. Для чого слугує компонент TLabel? ▲
5. Як обирається назва шрифту об'єктів у формі? ★
6. Як реагує середовище Lazarus на синтаксичні помилки? ★
7. Поясніть сутність команди `v1:=StrToint(Edit1.Text);`. ★
8. Поясніть сутність команди `t:=140/(v1+v2)`. ★

Виконуємо

1. Визначте необхідні об'єкти для створення інтерфейсу проекту обчислення площі прямокутника. ▲
2. Створіть проект для обчислення значення виразу: $y = a + b$. ★

3.  Відомі катети прямокутного трикутника. Розробіть проект для обчислення його площі. 
4.  Сторінка тексту містить 28 рядків, у кожному – 70 символів. Розробіть проект обчислення обсягу пам'яті для збереження 50 сторінок тексту. 
5.  Розробіть проект для перетворення значення маси в кілограмах на пуди і фунти (1 кг дорівнює $\frac{1}{16}$ пуда; 1 кг дорівнює 2,5 фунта). 
6.  Проаналізуйте об'єкти, зображені на рис. 5.32. Сформулюйте умову задачі, яку можна реалізувати за допомогою цього інтерфейсу. Розробіть проект, що реалізує завдання. 

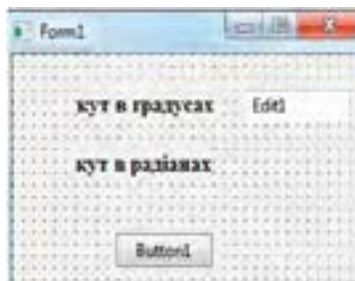


Рис. 5.32. Інтерфейс для розроблення програмного коду

**Практична
робота № 8**

Тема:	Створення об'єктно-орієнтованої програми, що відображає вікно повідомлення
Мета:	Набути практичних навичок створення програм з віконним інтерфейсом

Завдання. На формі розмістити компоненти Button1 і Edit1. Після натиснення на кнопку Button1 у текстовому полі з'являється повідомлення: Вас вітає Lazarus.

1. Запустити на виконання середовище Lazarus. Створити у ньому проект. Створити й зберегти папку з іменем Robot_5.
2. Надати формі ім'я Форма_1.
3. Надати значення таким властивостям форми: розміри форми: Height – 200, Width – 300; колір форми (властивість Color) – виберіть самостійно.
4. Розмістити на формі об'єкт Button1. Змінити властивість Caption. Увести, наприклад, Клацніть. Інші властивості об'єкта вибрати самостійно.
5. Розмістити на формі компонент Edit1. Самостійно визначити необхідні значення його властивостей і надати їх.
6. Перейти у вікно інспектора об'єктів і виділити рядок Button1. На вкладці Подія вибрати рядок OnClick і двічі клацнути кнопкою миші праворуч від цієї назви. У редакторі тексту з'явиться заголовок для введення команди, що реагує на натиснення кнопки.
7. У редакторі тексту ввести команду Edit1.Text:= 'Вас вітає Lazarus';. Зберегти і виконати проект.

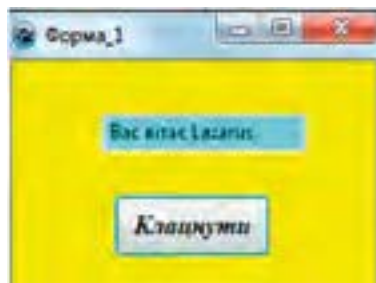


Рис. 5.33. Контрольний результат виконання програми

8. Переконатися, що отриманий результат аналогічний представленому на рис. 5.33.

9. Закрити проект. Відкрити й виконати проект ще раз. Переконатися, що програма виконується правильно. Закрити проект.

**Практична
робота № 9**

Тема:	Створення програми з кнопками та написами
Мета:	Набути практичних навичок створення програм з віконним інтерфейсом

Завдання. Відомі курси долара та євро, встановлені Національним банком України. Розробити програму для визначення суми грошей у гривнях після обміну 120 доларів і 80 євро.

1. Визначити назви і кількість компонентів, потрібних для створення інтерфейсу програми.

2. Створити проект і зберегти його. Надати ім'я формі й значення іншим її властивостям.

3. Розмістити на формі визначені компоненти і надати значення їх властивостям.

4. У **Інспекторі об'єктів** для об'єкта Button1 вибрати подію OnClick.

5. Увести необхідні команди в редактор тексту.

6. Виконати компіляцію програми.

7. Виконати програму для різних значень курсу валют. Переконатися, що програма функціонує правильно.

8. Зберегти і закрити проект.



СЛОВНИЧОК

Lazarus – візуальне середовище програмування, призначене для розроблення й виконання програм у текстовому та графічному режимах.

Алгоритмізація – процес розроблення алгоритму.

Виконуваний код – програма, готова до виконання на комп'ютері з відповідним процесором і операційною системою.

Клас – сукупність змінних і методів, що описує множину об'єктів.

Компіляція програми – процес перетворення програми на машинний код.

Метод – програма, що описує поведінку об'єкта.

Мова програмування – алгоритмічна мова, призначена для запису алгоритму і даних для виконання на комп'ютері.

Об'єктно-орієнтоване програмування – методологія програмування, що ґрунтується на поданні програми у вигляді сукупності об'єктів, кожний із яких є екземпляром певного класу.

Подійне програмування – Програмування, у якому порядок виконання програми залежить від певних дій користувача.

Форма – вікно середовища Lazarus створення інтерфейсу програми.

РОЗДІЛ 6. АЛГОРИТМИ РОБОТИ З ОБ'ЄКТАМИ ТА ВЕЛИЧИНАМИ



Основи алгоритмізації та програмування. Консольний режим роботи середовища Lazarus, структура консольної програми, типи даних мови Free Pascal, їх оголошення, засоби їх уведення в текстовому й візуальному режимах. Арифметичні та логічні вирази, операції над рядковими даними. Базові структури алгоритмів, алгоритми з розгалуженнями, програми, оператори циклів із лічильником, передумовою та післяумовою. Програми для малювання геометричних та інших фігур, використання в проектах рисунків із зовнішніх файлів.

6.1. Консольний режим роботи середовища Lazarus

Консольний режим зручно використовувати для вивчення операторів мови програмування та її синтаксичних конструкцій.

Для створення нової консольної програми в середовищі Lazarus потрібно виконати команду **Проект** → **Новий проект**. Потім у вікні, що відкриється, слід виконати команду **Консольна програма** → **Гаразд**. Відкриється вікно **Нова консольна програма**, у якому не бажано змінювати параметри генерування коду, а слід лише натиснути кнопку **Гаразд**. На екрані з'явиться вікно редактора тексту, в якому наведена загальна структура програми мовою Free Pascal, тобто певний шаблон, який допомагає і прискорює роботу програміста. Це вікно зображене на рис. 6.1.

У цьому вікні наведено лише фрагмент загальної структури консольної програми мовою Free Pascal, якого досить для пояснення сутності методики розроблення програми. Насправді у вікні редактора тексту на комп'ютері буде значно складніша структура. Розглянемо основні компоненти зображеного фрагмента.

Перший рядок – це заголовок програми, який складається з ключового слова **program** та імені програми `Project1`. Це ім'я присвоюється програмі автоматично. За бажанням його можна змінити, наприклад, заголовок може мати такий вигляд: `program prog_01`.

Ключовим словом **uses** починається підключення до програми необхідних модулів. **Модуль** – окрема спеціальна програма, яка розширює можливості мови програмування.

У вікні, зображеному на рис. 6.1, бачимо, що підключено, зокрема, модуль `Classes`, який забезпечує роботу з компонентами, і модуль `SysUtils`, призначений для роботи з файлами і каталогами. Не потрібно підключати жодних інших модулів, вони встановлюватимуться за замовчуванням.

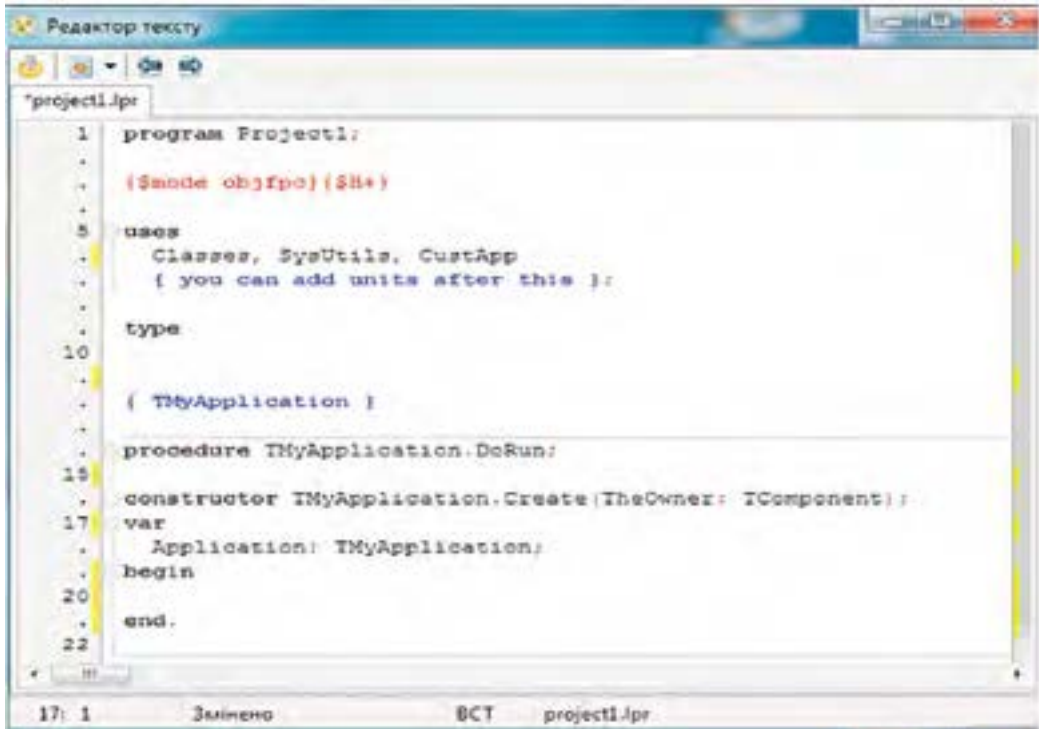


Рис. 6.1. Вікно редактора тексту в консольному режимі

Далі починається розділ опису. Тут зазвичай описуються константи, типи і змінні.

- ✓ Між словами **begin** і **end**, розміщується тіло програми, яку розробляє програміст.

У тексті програми можуть міститися коментарі, які виконують інформативну роль і не впливають на виконання програми. Коментарі обмежуються фігурними дужками або починаються двома косими рисками.

- ✓ Після уведення консольної програми вона зберігається, відкривається, компілюється і запускається на виконання так само, як і у візуальному проекті.

Отже, консольне застосування має таку загальну структуру:

заголовок програми;
uses modul1, modul2,..., modulN;
розділ опису;
тіло програми;

- ✓ Більш детально структура консольної програми має такий вигляд:

program ім'я програми;
uses modul1, modul2, ..., modulN;

```

const опис констант;
type опис типів;
var опис змінних;
begin    //далі розташовані оператори (тіло програми)
оператор_1;
оператор_2;
...
оператор_N;
end.    //кінець програми

```

Тіло програми починається ключовим словом **begin**. Оператори у мові Pascal відокремлюються один від одного крапкою з комою. Тіло програми закінчується словом **end**.

Наведемо найпростіший приклад консольної програми:

```

program pro_01;
const a=5;           //оголошено константу a
var b, c: integer; //змінні b і c оголошені типу integer
begin
    b:=a+8; c:=b+a;   //операції додавання і присвоєння значень
змінним
    writeln ('c=', c); //виведення результату на екран
readln           //затримання зображення на екрані з повідомленням про результат
end.

```

Після уведення й виконання програми на екрані з'явиться вікно з повідомленням: c=18.

Для завершення виконання програми і зняття з екрана цього вікна слід натиснути клавішу Enter.

Для реалізації цієї програми у віконному режимі до неї необхідно внести відповідні корективи. Програма має набути такого вигляду:

```

program pro_01a;
const a=5;
var b, c: integer;
begin
    b:=a+8; c:=b+a;
    Label1.Caption:='c='+FloatToStr (c);
end;

```

На формі розмістимо компоненти Label1 і Button1 і надамо їм властивості, показані на рис. 6.2.

Активуємо кнопку Button1, встановлюємо для неї у вікні інспектора об'єктів подію OnClick, двічі клацаємо праворуч від неї кнопкою миші й у вікні редактора тексту вводимо програму. Далі компілюємо її й виконуємо. Після натиснення на кнопку "Обчислити" з'явиться результат, представлений на рис. 6.3.

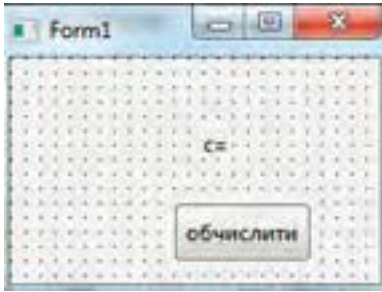


Рис. 6.2. Компоненти на формі для створення проекту

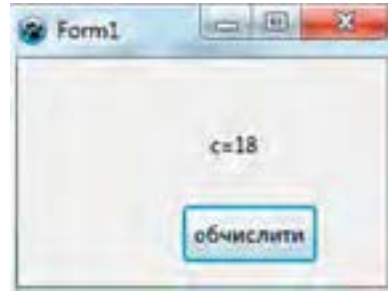


Рис. 6.3. Результат виконання програми

✓ На основі описаного можна дійти такого висновку: принципова різниця між методикою розроблення візуального проекту та консольної програми полягає в тому, що для візуального проекту додатково створюються об'єкти, дії яких програмуються, а введення/виведення даних здійснюється через зміну відповідних властивостей об'єктів.


Перевіряємо себе

1. Що вказується в заголовку консольної програми? ▲
2. Що оголошується в розділі опису консольної програми? ▲
3. Що розміщується між ключовими словами **begin end**? ▲
4. Як позначаються коментарі в консольній програмі? ▲
5. Наведіть загальну структуру консольної програми. ★
6. Що називають модулем у консольній програмі? ★
7. Яку функцію виконує модуль SysUtils? ★
8. Поясніть загальний порядок створення консольної програми. ★
9. Яка різниця в створенні візуального проекту від консольної програми? ★
10. Наведіть детальну структуру консольної програми. ★


Виконуємо

1. Визначте помилки в коді програми: ★

```
programm n2;  
const b=21;  
begin  
  a:=a+b;  
end.
```
2. 🏠 Створіть консольну програму для додавання двох цілих чисел. ★
3. 🏠 Створіть візуальний проект для додавання двох цілих чисел. ★

4.  Уведіть і виконайте консольну програму в середовищі Lazarus: ★

```
program rek;  
const b=3;  
var a: integer;  
begin  
  a:=5*b;  
  writeln ('a=', a);  
  readln;  
end.
```

5.  Доопрацюйте програму п. 4 для її реалізації у віконному режимі. ★

6.2. Дані, змінні, константи

Дані в комп'ютерних системах – це факти, відомості, повідомлення, результати вимірювань, спостережень тощо, подані в формалізованому вигляді, придатному для їх збереження, опрацювання та передавання. Дані зберігаються в пам'яті комп'ютера й можуть бути різних видів: числові, символічні, логічні та інші.


✓ *Кожний вид даних має певну множину значень, над якими можуть виконуватися заздалегідь визначені операції.*

Наприклад, над числовими даними можуть виконуватися арифметичні операції, а над логічними – логічні, з якими ми познайомимося пізніше. Множину значень даних разом із множиною операцій, що може виконуватися над ними, називають **типом даних**.

✓ *Тип даних визначає розмір комірки (ділянки) пам'яті, у якій зберігатиметься те чи інше значення.*

Під час виконання програми дані зберігаються **в комірках оперативної пам'яті**. Якщо значення комірки пам'яті можна змінювати у процесі виконання програми, її вміст називається **змінною**, інакше – **константою**. Отже, константа – це значення, що не змінюється протягом усього часу виконання програми.

Мінімальна величина комірки – 1 байт. Для кожного типу даних визначена фіксована кількість байтів. Змінна має ім'я (ідентифікатор), тип і значення. Ідентифікатор слугує для звернення до комірки пам'яті, в яку потрібно записати або з якої прочитати значення.

 **Змінна в програмуванні** – це ділянка пам'яті з присвоєним їй ім'ям (ідентифікатором), у якій зберігаються дані певного типу.

Імена змінних складаються з букв, цифр і символу підкреслення, наприклад, `snik`, `a2`, `b_1`, `i`. Першим символом ідентифікатора не може бути цифра. Мова Free Pascal вимагає обов'язкового оголошення змінних.

✓ *Оголосити змінну – означає вказати її ім'я і тип.*

Розділ оголошення змінних розміщується між заголовком програми і словом **begin**. Він починається словом **var** і має таку структуру:

```
var <ім'я змінної>:<тип змінної >;
```

Ім'я змінної відокремлюється від типу двокрапкою. Оголошення змінної завершується крапкою з комою, наприклад, **var i:integer;**. Декілька однотипних змінних можна оголосити разом, указавши їх імена через кому, наприклад: **var a_1, b_2: real;**. Змінні різних типів можна оголосити за допомогою одного слова **var**, наприклад:

```
var  
alpha, n1: integer; //змінні alpha і n1 типу integer  
rh, a_1:real; //змінні rh і a_1 типу real  
Константи описуються за такою структурою:
```

```
const ім'я константи = значення;
```

Наприклад:

```
const  
r=5.25; //константа r дійсного типу  
c=21; //константа c цілого типу
```

У мовах програмування розрізняють статичні та динамічні дані. Якщо протягом усього періоду виконання програми комірки пам'яті задіяні постійно, не звільняються і не виділяються для інших даних, то дані, що в них зберігаються, називаються **статичними**, інакше – **динамічними**. Далі розглядаються лише статичні типи даних.



Статичні типи даних поділяються на **прості** (скалярні) та **структуровані**.

✓ *Прості типи даних є цілісними (атомарними, неподільними) елементами.*

✓ *Структуровані типи даних складаються з множини елементів.*

Прикладами структурованих типів даних є масиви, множини, файли, рядки символів та інші. Будь-яка універсальна мова програмування забезпечує роботу з цілою родиною простих типів даних. Прості типи поділяються на **стандартні (базові)**, які мають зарезервовані ідентифікатори (наприклад, **integer**, **real**, **boolean**) і змінювати їх не можна, й **типи, які програміст визначає сам**. У цьому підручнику розглядаються лише прості типи даних.

✓ *Класифікація простих типів даних мови Free Pascal зображена на рис. 6.4.*

Цілочислові, символічні та логічні дані називають **порядковим типом даних**, позаяк їх легко впорядкувати за певним критерієм. Інші типи даних, наприклад дійсні, впорядкувати складніше.

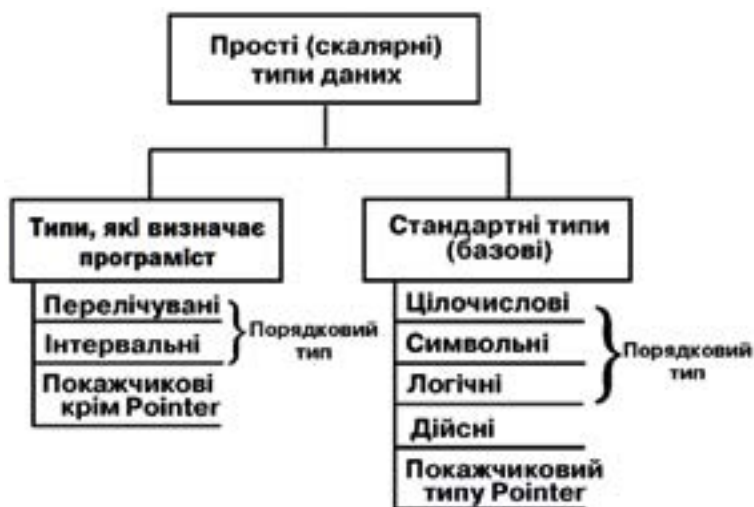


Рис. 6.4. Класифікація простих типів даних мови Free Pascal

Цілочислові типи даних. Цей тип даних може займати в пам'яті комп'ютера від 1 до 8 байтів. Усього в мові є дев'ять цілочислових типів даних. Цілі числа можуть записуватись у десятковій і шістнадцятковій системах числення. Перед шістнадцятковими числами ставиться символ \$, наприклад, \$8A5F. У табл. 6.1 наведено п'ять найбільш уживаних цілочислових типів даних.

Таблиця 6.1

Цілочислові типи даних

Ідентифікатор типу	Кількість байтів	Діапазон
byte	1	0 ± 255
word	2	0 ± 65535
longword	4	0 ± 4294967295
shortInt	1	-128 ± 127
integer	4	-2147483648 ± 2147483647

Приклад оголошення цілочислових змінних:

var i, j: integer; k1: byte; a_1: word;

Дійсний тип даних. Дійсні числа подаються у спеціальному форматі: $M \cdot 10^P$, де M – мантиса; P – порядок. Мантиса в поданні числа має задовольняти умови $0 < M \leq 1$. Для запису порядку числа використовується два розряди, а для мантиса – від 8 до 20. На записі мантиса відокремлюється від порядку літерою E . Наприклад, десяткове число 374,5 можна записати так: .3745E03. У мові Free Pascal існує шість типів дійсних чисел. У табл. 6.2 наведені найбільш уживані з них.

Таблиця 6.2

Типи дійсних чисел

Ідентифікатор типу	Кількість байтів	Кількість цифр у мантисі
Single	4	до 8
Real	8	до 16
Double	8	до 16
Comp	8	до 20

Приклад оголошення змінних дійсного типу:

```
var t_1, t_2: double; rk: real; a1: single;
```

Дані логічного типу. Дані логічного типу можуть набувати лише двох значень: true (істинне) і false (хибне). Типи логічних даних наведені в табл. 6.3.

Таблиця 6.3

Типи логічних даних


Ідентифікатор типу	Кількість байтів
Boolean	1
Bytebool	1
Wordbool	2
Longbool	4

Приклад оголошення змінних логічного типу:


```
var dek, bm: boolean; a_1: wordbool;
```

Символьні дані. Символьний тип даних позначається словом char. Дані цього типу займають у пам'яті 1 байт. Дані символьного типу – це окремі символи, що записуються в одинарних лапках, наприклад: 'к', 'а', 'п'. Приклад оголошення змінних символьного типу: `var sam, v1, b_2: char;`

Дані типу дата – час. Цей тип позначається TDateTime. Він призначений для одночасного оголошення дати і часу.

 Прості типи даних, які визначає програміст.

Крім простих стандартних типів даних мова Free Pascal надає можливість програмісту самому визначати нові прості типи даних, зокрема, перелічуваний та інтервальний типи.

 Для цього застосовується ключове слово **type** із такою структурою:

```
type новий_тип_даних = визначення типу;
```

Після створення нового типу даних можна оголошувати змінну цього типу:

var список_змінних_нового_типу : новий_тип_даних;

Перелічуваний тип даних. У цьому типі перелічуються значення, яких може набувати змінна. Змінна типу оголошується за такою структурою:

var ім'я_змінної: (значення_1, значення_2,..., значення_N);

Приклад оголошення змінних перелічуваного типу:

var color: (red, blue, green);

Із прикладу випливає, що змінна color може набути значень red, blue або green.

Приклад:

grivna = (_1, _2, _5, _10, _20, _50, _100, _200, _500);

var gr1, gr2: grivna;

У прикладі створено новий тип даних grivna й оголошені змінні gr1, gr2 цього типу.

Інтервальний тип даних. Подається границями своїх значень за такою структурою:

var ім'я_змінної: мінімальне_значення .. максимальне_значення;

Наприклад: **type** date=1..30; symb='a'..'d';


Із прикладу випливає, що змінна date може набувати будь-якого значення в діапазоні від 1 до 30, а змінна symb – будь-яку літеру від a до d.

Приклад

type days = 1..7;

var a_1: days;

У прикладі створено новий тип даних days і оголошена змінна a_1 цього типу, яка може набувати будь-якого значення в діапазоні від 1 до 7.

 **Значення присвоюються змінним за допомогою оператора присвоєння, який позначається двокрапкою і знаком дорівнює (:=) і має таку структуру:**

<ім'я_змінної>:=<значення>;

Значення – це вираз, змінна, константа або функція. Спочатку обчислюється значення, що потім присвоюється змінній, ім'я якої записане ліворуч від оператора. Значення має бути узгоджене з типом змінної, якій воно присвоюється. Наприклад, якщо змінна a_1 оголошена типу integer, то оператор a_1:=21; буде виконаний правильно, а в операторі a_1:='s' тип змінної не узгоджений із типом виразу, а тому він виконаний не буде.

Приклад. У програмі pr_2 всі типи змінних і виразів узгоджені, тому програма виконується правильно.

program pr_02;

var a, b, c: real; {оголошення змінних a, b, c дійсного типу}

begin

```
a:=21.4; b:=33.25; {присвоєння змінним a, b дійсних значень}  
c:=a*b-a/2; {обчислення значення виразу і присвоєння його змінній c}  
writeln ('сума=', c); {виведення на екран результату}  
end.
```

Дані рядкового типу – рядок символів. На рис. 6.4 його не віднесено до простих типів даних, тому що принципово він є структурованим типом. Але його часто зараховують не до структурованих, а до скалярних типів даних. Особливої помилки тут немає. Це пояснюється тим, що з точки зору структур даних рядки є структурованим типом. Однак завдяки їх широкому застосуванню для них виділено стандартний ідентифікатор `string`. Тому з точки зору назви типу рядки можна віднести й до стандартного (базового) типу. Рядкові змінні оголошуються за такою структурою:

```
var ім'я_змінної: string [довжина_рядка];
```

Якщо довжина рядка не вказана, то кількість символів у рядку не може перевищувати 255.

Приклад.

```
var str1: string; str2: string [20];
```

У цьому прикладі змінна `str1` може мати до 255 символів, а змінна `str2` — до 20 символів. У виразах рядкові дані беруться в лапки, наприклад: 'файл'. Доступ до окремих символів рядка здійснюється за допомогою операції індексування, яка позначається символами []. Наприклад, у результаті виконання операторів `str1:='клавіатура'; write (str1[4]);` буде виведено символ `v`.



Константа – це величина, значення якої не змінюється у процесі виконання програми.



Константи у мові Free Pascal бувають прості й типізовані.

Тип простої константи не потрібно оголошувати, його компілятор розпізнає за її значенням. Наприклад, якщо у програмі трапляється число 27, то компілятор однозначно його сприймає як ціле. У мові Free Pascal використовуються такі типи простих констант: десяткові цілі числа, шістнадцяткові цілі числа, десяткові дійсні числа, символи, рядкові та логічні.

Цілочислові константи записуються в звичайному форматі у допустимому діапазоні, наприклад, 248113, -305.

Дійсні десяткові константи записуються в такому вигляді: <ціла частина>.<дробова частина>E<порядок>. Наприклад, 291E02, -21.25E-01, що відповідає таким числам: 29100, -2,125.

Шістнадцяткові цілі константи записуються шістнадцятковими символами, яким передуює символ \$, наприклад, \$E5C, \$0A.

Символьна константа – це будь-який символ у лапках, наприклад: 'a', '+'. Для перетворення цілого числа в діапазоні 0ч255 на ASCII-символ використовується функція `chr`. Наприклад, після виконання операто-

рів `a_1:=chr(90)`; `a_2:=chr(97)` змінна `a_1` набуде значення Z, а змінна `a_2` – значення a.

Рядкова константа – це будь-яка послідовність символів у лапках, наприклад: `'Київ'`, `'a_1+a_2='`. Максимальна довжина рядкової константи становить 255 символів.

Логічна константа має значення `true` (істинне) або `false` (хибне).

У мові Free Pascal використовуються також **прості іменовані константи**, тобто константи, яким присвоюються імена (ідентифікатори). Потім можна звертатися до констант за їх іменами. Тип константи в цьому випадку не вказується, він автоматично визначається за її значенням. Оголошення іменованих констант має таку структуру:

`const <ім'я константи>=<константа>;`

Приклади: `const i=5; a_1='н'; b_1='файл'; b_2=51.25E-01;`

✓ *Типізована константа оголошується з типом і значенням.*

Оголошення типізованих констант має таку структуру:

`const <ім'я константи>:<тип>=<значення>;`

На відміну від простих іменованих констант значення **типізованих констант** можна змінювати у процесі виконання програми так само, як і значення змінних. Але типізовані константи відрізняються від змінних тим, що вони отримують значення на початку роботи програми, а змінні набувають значень у процесі виконання програми.

Приклади оголошення типізованих констант: `const i:integer=1; a_1:string='байт'; b_1:boolean=true; x:char='М';`.

Приклад.

```
program pr_06;
const n:integer=21; m:integer=5; {оголошення типізованих констант}
var y:integer; {оголошення змінної}
begin
  y:=n*m; writeln ('y=',y); {обчислення значення виразу і його виведення }
  readln {призупинення виконання програми}
end.
```


За допомогою програми `pr_06` обчислюється добуток типізованих констант `n` і `m`, який присвоюється змінній `y`. Це значення виводиться на екран.

Перевіряємо себе

1. Яку мінімальну довжину має комірка пам'яті? ▲
2. Які дані називають статичними? ▲
3. На які типи поділяються статичні дані? ▲

4. Які типи даних є порядковими? ▲
5. Яким словом позначається символічний тип даних? ▲
6. Які значення мають дані логічного типу? ▲
7. Як позначаються дані логічного типу? ▲
8. Що називають даними в комп'ютерних системах? ✦
9. Що називають типом даних? ✦
10. Які дані називають динамічними? ✦
11. Поясніть сутність простих типів даних. ✦
12. Назвіть основні типи цілочислових даних. ✦
13. Назвіть основні типи дійсних чисел. ✦
14. Наведіть структуру оператора присвоювання. ✦
15. Де зберігаються дані під час виконання програми? ★
16. На які типи поділяються прості дані? ★
17. Чому рядкові дані відносять і до скалярних, і до структурованих типів? ★
18. Поясніть, як записуються дійсні числа. ★
19. Наведіть визначення змінної в програмуванні. ★
20. За якою структурою оголошуються змінні в мові Pascal? ★
21. Назвіть типи простих констант мови Pascal. ✦
22. Наведіть приклади рядкових констант. ✦
23. За якою структурою оголошуються прості іменовані константи? ✦
24. Наведіть приклади логічних констант. ✦
25. Наведіть приклади дійсних десяткових констант. ✦
26. Поясніть сутність простої іменованої константи. ★
27. Наведіть приклади оголошень простих іменованих констант. ★
28. За якою структурою оголошуються типізовані константи? ★
29. Наведіть приклади оголошень типізованих констант. ★




Виконуємо

1. Запишіть оголошення змінних a і b логічного типу, а змінної c – дійсного. ▲
2.  Знайдіть помилки в оголошеннях: a) var x, y; real; b) var x; y: boolean; c) var x: y: integer; d) var x: real; x1=.345E03: real; e) const n: integer=25; m: integer=.51E10. ✦
3. Доведіть, що в програмі sam_05 відсутні помилки. ✦



```

program sam_05;
  var a, b, c: real;
  begin
    a:=5; b:=6.6;
      
```

```
c:=a+b;
writeln ('c=', c)
end.
```

-  Запишіть розділ оголошення таких типів змінних: a, b – цілі числа; c – дійсні числа; x, y – символічні. ▲
-  Знайдіть помилки в оголошенні змінних: var a_1; a_2:rial; a_3 f_4:byte; x:boolean;. ✦
-  Програма sam_06 має забезпечити введення та множення двох цілих чисел. Але в ній є помилки. Проаналізуйте програму, знайдіть у ній помилки, виправте їх. Який результат буде отриманий після виправлення помилок? ★


```
program sam_06
var a_1; x:integer;
var a_2:char;
begin
  writeln ('увести числа a_1, a_2')
  readln (a_1, a_2);
  x:a_1*a_2
  writeln ('x=', x)
  readln
end.
```

-  Доопрацюйте програму п. 6 для її реалізації у віконному режимі. ★

6.3. Уведення та виведення даних

Уведення й виведення даних у середовищі Lazarus для текстового та віконного режимів відрізняються. У текстовому режимі дані вводяться безпосередньо з клавіатури й виводяться на екран. У віконному режимі дані вводяться й виводяться за допомогою зміни властивостей об'єктів інтерфейсу користувача, а також деяких функцій.


6.3.1. Уведення та виведення даних у консольному режимі

-  Для введення даних з клавіатури в мові Free Pascal використовується два оператори:

```
read (список імен змінних);
readln (список імен змінних);
```

Імена змінних списку відокремлюються комою, наприклад, read (beta, tr1, a2);. Оператори read і readln відрізняються лише тим, що після виконання оператора readln курсор буде переведений на початок наступного рядка, а після виконання оператора read він залишається на поточному

рядку. Кожний із цих операторів перериває виконання програми й “очікує” введення такої кількості даних, скільки він містить змінних.

 *Якщо вводяться числові дані, то вони відокремлюються пробілом. Символьні дані пробілом не відокремлюються.*

Після введення останнього значення слід натиснути клавішу Enter і виконання програми буде продовжене. Перше введене значення буде присвоєне першій змінній в операторі, друге значення – другій змінній і т. д.

Припустімо, що фрагмент програми має такий вигляд:


```
var a1, a2: integer; a3: real;
begin
read (a1, a2, a3);
.....
```

Якщо з клавіатури будуть введені числа 21 100 35.25, то змінна *a1* набуде значення 21, змінна *a2* – значення 100, а змінна *a3* – значення 35,25. Звернімо увагу, що змінні *a1* і *a2* оголошені цілого типу (integer), а змінна *a3* – дійсного типу (real). Тому обов’язково потрібно вводити перші два числа цілими, а третє може бути як ціле, так і дійсне.

Приклад. Програма обчислення площі прямокутного трикутника за значеннями його катетів може бути такою:

```
program pr_03;
var a, b, s: real; {оголошення змінних}
begin
  writeln ('увести значення катетів'); {повідомлення про введення
значень катетів}
  readln (a, b); {уведення значень катетів}
  s:=0.5*a*b; {обчислення площі трикутника}
  writeln ('площа=', s); {виведення значення площі прямокутного
трикутника}
end.
```

Значення катетів вводяться з клавіатури й присвоюються змінним *a* і *b*. Вони мають дійсний тип (real). Обчислена площа трикутника зберігається в змінній *s*.

 **Для виведення даних на екран монітора в мові Free Pascal використовуються такі оператори:**

```
write (список значень);
writeln (список значень).
```

До списку значень можуть бути включені змінні, константи, функції, вирази, які відокремлюються комами. Якщо елементом списку значень є текст, він береться в лапки. Різниця між наведеними операторами полягає лише у тому, що після виконання першого з них курсор залиша-

ється на поточному рядку, а після виконання другого – переводиться на наступний. Наприклад, якщо змінні a , b , c набувають відповідно таких значень: 20, -37, 125, то в результаті виконання фрагмента програми:

```
var a, b, c: integer;  
begin
```

```
...
```

```
write (a);  
write ( ' ', b, ' ', c);
```

ці числа будуть надруковані в одному рядку: 20 -37 125. Звернімо увагу, що змінні в операторі `write` відокремлені двома пробілами, тому й виведені числа відокремлені також двома пробілами.

Ці ж самі числа в результаті виконання такого фрагмента:

```
var a, b, c: integer;  
begin
```

```
...
```

```
writeln (a, ' ', b);  
write (c);
```

будуть виведені двома рядками:

```
20 -37  
125
```

Корисно разом із даними виводити повідомлення, яке пояснює їх. Наприклад, у результаті виконання фрагмента програми:

```
var a, b, c: integer;  
begin
```

```
...
```

```
writeln ('a= ', a, ' b= ', b, ' c= ', c);
```

числа будуть виведені в одному рядку в такому вигляді: a=20 b=-37 c=125.

Приклад. Фрагмент програми, що забезпечує обчислення та виведення на екран площ двох квадратів.

```
program pr_04;  
var a, b: integer;      {оголошення змінних}  
begin
```

```
writeln ('уведіть значення a, b'); {повідомлення про введення значень змінних}
```

```
readln (a, b);          {уведення значень змінних}
```

```
writeln ('площа першого квадрата =', a*a); {виведення площі 1-го квадрата}
```

```
writeln ('площа другого квадрата =', b*b) {виведення площі 2-го квадрата}
```

```
end.
```

Якщо у процесі виконання наведеної програми будуть уведені числа 7 і 12, то на екран буде виведено:

площа першого квадрата =49
площа другого квадрата =144.

✓ Для того щоб у процесі виконання програми в консольному режимі результат виведення на екран можна було одразу прочитати (не натискаючи клавіші *Alt+F5*), доцільно після операторів виведення поставити порожній оператор *readln* або *read*.

Він призупиняє виконання програми, і результат буде показуватися доти, доки не буде натиснута клавіша *Enter*. З урахуванням цього фрагмент програми *pr_04* можна записати так:

```
program pr_05;  
var a, b: integer;      {оголошення змінних}  
begin  
writeln ('увести a, b'); {повідомлення про введення значень змінних}  
readln (a, b);          {введення значень змінних}  
writeln ('площа 1-го квадрата=', a*a); {виведення площі 1-го квадрата}  
readln;                {призупинення виконання програми}  
writeln ('площа 2-го квадрата=', b*b); {виведення площі 2-го квадрата}  
readln                 {призупинення виконання програми}  
end.
```

У процесі виконання програми *pr_05* вона зупиняється двічі: перший раз після виведення значення площі 1-го квадрата і другий – після виведення значення площі 2-го квадрата.

6.3.2. Уведення і виведення даних у віконному режимі

Для організації введення і виведення даних у віконному режимі існує два способи:

✓ За допомогою вбудованих діалогових вікон *InputBox* і *ShowMessage*.

Ці вікна не встановлюються у форму. Для їх виведення на екран використовуються відповідно функція *InputBox()* і функція *ShowMessage*. У перше вікно вводяться необхідні дані, а в друге – виводяться дані. Ці функції використовуються безпосередньо в коді програми.

✓ За допомогою об'єктів, розташованих на формі.

Для введення та виведення даних використовуються різні об'єкти, наприклад, однорядкові *TEdit*, *TLabel* і багаторядкові *ListBox*, *TMemo*. Об'єкти *TEdit* і *TMemo* можуть застосовуватися як для введення, так і для виведення даних. Розглянемо спочатку приклади використання об'єкта *TEdit* для введення і об'єкта *TLabel* для виведення. Інші об'єкти опишемо пізніше, під час їх використання.

Сутність введення і виведення даних цим способом полягає в зміні значень властивостей *Caption* і *Text* указаних компонентів. Раніше вже

наводилися приклади введення і виведення даних з використанням цих компонентів, але без розкриття їх сутності.

✓ *Необхідно завжди пам'ятати, що система Lazarus сприймає значення властивостей Caption і Text компонентів Edit і Label як рядковий тип.*

✓ *Перетворення даних рядкового типу на інші типи здійснюється за допомогою таких функцій:*

StrToFloat (s) – перетворює символи рядка s на дійсне число.

StrToInt (s) – перетворює символи рядка s на ціле число.

StrToDateTime (s) – перетворює символи рядка s на дату і час.

✓ *Зворотне перетворення здійснюється за допомогою функцій:*

FloatToStr (v) – перетворює дійсне число v на рядок.

IntToStr (v) – перетворює ціле число v на рядок.

DateTimeToStr (v) – перетворює дату і час v на рядок.

Таблиця 6.4

Властивості об'єктів

Об'єкт	Caption	Text	Font			Color
			Шрифт	Стиль	Розмір	
Label1	Перше число		Times New Roman	Курсив	12	
Label2	Друге число		Times New Roman	Курсив	12	
Label3	Сума		Times New Roman	Жирний курсив	12	clYellow
Label4	Добуток		Times New Roman	Жирний курсив	12	clYellow
Edit1		Пробіл	Times New Roman	Жирний	12	clSkyBlue
Edit2		Пробіл	Times New Roman	Жирний	12	clSkyBlue
Button1	Обчислити		Times New Roman	Жирний	12	

У Інспекторі об'єктів для властивості Text об'єкта Edit1 уведемо число 12, а для об'єкта Edit2 – число 8. У результаті маємо отримати орієнтовно такий зміст форми, який зображено на рис. 6.5.

Для об'єкта Button1 вибираємо подію OnClick і вводимо такий програмний код:

```
procedure TForm1.Button1Click(Sender: TObject);
var a, b, s, p: real;
```

```

begin
a:=StrToFloat (Edit1.Text);
b:=StrToFloat (Edit2.Text);
s:=a+b;
p:=a*b;
Label3.Caption:='Сума='+FloatToStr (s);
Label4.Caption:='Добуток='+FloatToStr (p);
end.

```

Після успішної компіляції запускаємо програму на виконання і потім натискаємо кнопку “Обчислити”. Маємо отримати результат, зображений на рис. 6.6.

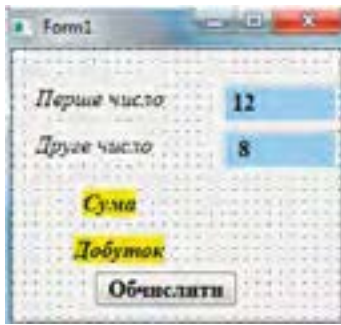


Рис. 6.5. Графічний інтерфейс для виконання завдання

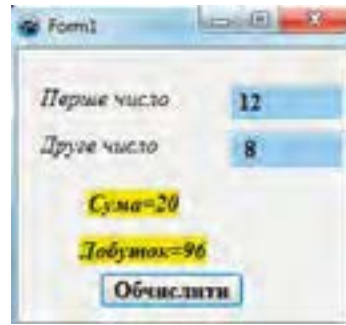


Рис. 6.6. Результат виконання програми

Перевіряємо себе

1. Які оператори використовуються для введення даних із клавіатури? ▲
2. Запишіть структуру оператора read. ▲
3. Запишіть структуру оператора readln. ▲
4. Які оператори використовуються для виведення даних на екран? ▲
5. Назвіть способи введення/виведення даних для графічного інтерфейсу. ▲
6. Яка різниця між операторами read і readln? ◆
7. Запишіть структуру оператора write. ◆
8. Запишіть структуру оператора writeln. ◆
9. Яка різниця між операторами write і writeln? ◆
10. Поясніть сутність введення даних за допомогою компонента Edit. ◆
11. Поясніть сутність виведення даних за допомогою компонента Label. ◆
12. Наведіть приклади запису оператора readln. ★
13. Для чого в програмі використовується порожній оператор readln? ★
14. Наведіть приклади запису оператора writeln. ★

15. Чи можна в операторі `writeln` використовувати логічний вираз? ★
16. За допомогою якої функції здійснюється перетворення рядкових даних на цілі числа? ★
17. За допомогою якої функції здійснюється перетворення чисел дійсного типу на рядковий тип? ★

Виконуємо



1. Змінні a , b і c оголошені символьного типу. Після виконання оператора `readln (a, c, b)` були введені такі символи: `h, r, w`. Визначте значення змінних a , b , c . ▲
2. У процесі введення даних змінні a і b набули значень відповідно 135 і 27. Знайдіть значення, що буде виведено на екран після виконання оператора `writeln ('сума=', a+b)`. ◆
3. Проаналізуйте і виконайте програму `sam_07`.

```

program sam_07;
var a, b, c, x: real;
begin
  writeln ('увести числа a, b, c');
  readln (a, b, c);
  x:=a*b-c;
  writeln ('x=', x);
  readln
end.

```

Доведіть, що в програмі помилок немає і що вона виконується правильно. ★


4.  Розробіть консольну програму для обчислення значення виразу: $a/b+(a/b-3b)$, в якому змінні a і b набувають цілочислових значень. Доведіть, що програма виконується правильно. ◆
5.  Визначте синтаксичні помилки в програмі `sam_08`.

```

program sam_08;
var a, b, c;real;
begin
  writln (увести a, b');
  readln (a, b(
  c:=a+2b-a/b;
  writeln ('c', c)
end.

```

Після виправлення помилок уведіть програму, виконайте компіляцію і добийтеся, щоб програма функціонувала правильно. ★


6.  Консольна програма `sam_09` розроблена для обчислення значення виразу $4*(a_1+a_2)$. Але в програмі є помилки.



```

program sam_09;
var a_1, a_2: integer;
begin;
  writeln (‘’)
  a_3:=(a_1+a_2)*4;
  writeln (”a_3=”, a_3)
end.

```

Виправить помилки в програмі й доведіть, що вона функціонує правильно. ★

7.  Розробіть проект із графічним інтерфейсом для додавання трьох цілих чисел і визначення їх добутку. ★


8.  Визначте помилки в програмному коді програми з елементами графічного інтерфейсу:

```

var a1, a2, a3: iteger;
begin
  a1:=StrToFloat (Edit1.Text);
  y:=a1*a2-a1;
  Label1.Caotion:=’Результат=’+FloatToStr (y);
end.

```

6.4. Вирази, операнди та операції

 Вираз складається із операндів (констант, змінних і функцій), знаків операцій і круглих дужок і визначає порядок виконання операцій над даними.


Константи, змінні та функції називають **операндами** виразу. Найпростіший вираз складається з одного операнда: константи, змінної або функції. Приклади виразів: $a+2*c-(a*b)$; $(-21.5/c+4)*b/a$.

Залежно від типу операндів і операцій, що використовуються у виразі, а також типу отриманого результату, розрізняють **арифметичні вирази** (результат арифметичного типу), **логічні вирази** (результат логічного типу) і **рядкові вирази** (результат рядкового типу).

6.4.1. Арифметичні вирази

У арифметичних виразах мови Free Pascal використовуються операції додавання (+), віднімання (-), множення (*), ділення (/), цілочислове ділення (div) й остача від ділення (mod).

Операнди можуть бути цілого і дійсного типів. У процесі запису арифметичного виразу слід дотримуватися таких правил і рекомендацій.

 1. Забороняється застосовувати підрядкові та надрядкові символи. Наприклад, вираз $3*(a^2+b)$ неправильний, правильним є вираз $3*(a*a+b)$.


2. Операції необхідно вказувати явно. Не можна писати $21(a+b)$, правильним є запис: $21*(a+b)$.

3. Не можна записувати дві операції безпосередньо одна за одною. Наприклад, вираз $a*-b$ є некоректним, його слід записати так: $a*(-b)$.

4. Кожній дужці, що відкривається, у виразі має відповідати дужка, що закривається.

5. Типи операндів виразів мають бути узгоджені. Не можна, наприклад, додавати символ до числа. Але в деяких випадках узгодження типів виконується автоматично. Наприклад, якщо виконується арифметична операція над числами, одне із яких ціле, а друге дійсне, то ціле число буде перетворене на дійсне, після чого операція виконуватиметься над дійсними числами. Результатами операцій додавання, віднімання і множення є цілі числа, якщо обидва операнди також є цілими числами, в інших випадках будуть отримані дійсні числа. Результатом операції ділення завжди є дійсне число.

Операції в арифметичному виразі виконуються з урахуванням їх пріоритету, а операції, які мають однаковий пріоритет, – у порядку їх запису, тобто зліва направо.

 *Правила пріоритету операцій такі:*

спочатку обчислюються значення функцій, потім виконуються операції множення і ділення, а після цього – операції додавання та віднімання. Якщо у виразі є круглі дужки, то насамперед виконуються ті операції, які записані в них. Наприклад, значення виразу $(a+b) + a*(c+d)$ обчислюється так: спочатку визначається сума $(a+b)$, потім – $(c+d)$, остання сума множиться на a , і до цього результату додається перша сума.

Приклад. Програма `pr_07` обчислює значення арифметичного виразу $3,5(a^2+b):(2b-a)$, в якому a і b – дійсні числа, що вводяться з клавіатури.

```
program pr_07;
var a, b, y:real;           {оголошення змінних}
begin
writeln ('введіть числа a і b'); {повідомлення про введення}
readln (a, b);              {уведення значень змінних}
y:=((a*a+b)*3.5)/(2*b-a); {обчислення значення арифметичного виразу}
writeln ('y=', y);        {виведення результату}
readln                     {призупинення виконання програми}
end.
```

6.4.2. Логічні вирази

Результатом обчислення значення логічного виразу може бути лише одне зі значень: `true` (істинно) або `false` (хибно). Логічні вирази складаються з логічних операндів, логічних операцій та круглих дужок.

Операндами можуть бути результати операцій порівняння, яких у мові Free Pascal шість: > (більше), < (менше), = (дорівнює), <> (не дорівнює), >= (більше або дорівнює), <= (менше або дорівнює). Наприклад, якщо $x=2$, а $y=7$, то результатом операції порівняння $x>y$ є false, тому що x не більше y , а результатом порівняння $4*x>y$ є true.


Основними логічними операціями є такі: not (ні), and (і), or (або), xor (виключне або). Ці операції ще називають відповідно операціями заперечення, кон'юнкції, диз'юнкції та додавання за модулем 2. Результати виконання наведених логічних операцій над логічними виразами А і В, які набувають логічних значень, наведені в табл. 6.5.

Таблиця 6.5

Результати виконання логічних операцій

A	B	not A	A and B	A or B	A xor B
Істинно	Істинно	Хибно	Істинно	Істинно	Хибно
Істинно	Хибно	Хибно	Хибно	Істинно	Істинно
Хибно	Істинно	Істинно	Хибно	Істинно	Істинно
Хибно	Хибно	Істинно	Хибно	Хибно	Хибно

Як бачимо з табл. 6.5, результат операції not є істинним, якщо логічний вираз є хибним, і, навпаки, хибним, якщо вираз є істинним. Результат операції and є істинним за умови, що обидва вирази є істинними. Результат операції or є істинним, якщо істинним є хоча б один із виразів А чи В. Результат операції xor є хибним, якщо значення змінних збігаються.

 Логічні операції мають такий пріоритет: not, and, or, xor.

Якщо у виразі є дві операції одного пріоритету, то першою виконується та, яка у виразі розташована лівіше. Найпершими виконуються операції в круглих дужках.

Нехай змінні мають такі значення: $x=1$, $y=2$, $z=3$. Тоді значення виразу: $((x>y) \text{ or } (x<z)) \text{ and } (\text{not}((x>0) \text{ or } (y>z)))$ обчислюється в такій послідовності: спочатку обчислюється значення виразу $((x>y) \text{ or } (x<z))$, який набуває значення *істинне*, потім обчислюється значення виразу $(x>0) \text{ or } (y>z)$, який також набуває значення *істинне*. Над останнім виразом виконується операція заперечення, тому він набуває значення *хибне*. Останньою виконується операція and і вираз набуває значення *хибне*.

Нижче наведена програма, що реалізує розглянутий приклад.

```

program pr_08;
var x, y, z:integer; s:boolean;           {оголошення змінних}
begin
  writeln ('увести значення змінних'); {повідомлення про введення}
  readln (x, y, z);                     {уведення значень змінних}

```

$s:=((x>y) \text{ or } (x<z)) \text{ and } (\text{not}((x>0) \text{ or } (y>z)))$; {обчислення значення логічного виразу}

```
writeln ('результат=', s);           {виведення результату}
readln                               {призупинення виконання програми}
end.
```

Якщо виконати програму для значень змінних $x=5, y=4, z=6$, то отримаємо результат false.

Логічні операції можуть виконуватися і над двійковими даними. У табл. 6.6 наведені результати виконання цих операцій над двійковими числами 0 і 1.

Таблиця 6.6

Результати виконання логічних операцій над двійковими даними

A	B	not A	A and B	A or B	A xor B
0	0	1	0	0	0
0	1	1	0	1	1
1	0	0	0	1	1
1	1	0	1	1	0

Наприклад, якщо $A=01110110$ і $B=10111100$, то в результаті виконання операції A and B буде отримано такий результат:

A	0	1	1	1	0	1	1	0
B	1	0	1	1	1	1	0	0
A and B	0	0	1	1	0	1	0	0

6.4.3. Операції над рядковими даними

✓ *Кілька рядків можна об'єднати в один рядок за допомогою операції об'єднання, яка позначається символом додати (+).*

Наприклад, у результаті виконання операції:

'середо'+'вище'+ ' '+Lazarus' отримаємо: середовище Lazarus.

✓ *Рядки можна порівнювати між собою.*

Наприклад, у результаті виконання операції 'abc'>'ab' отримаємо true, а у результаті виконання операції 'abc'='ab' – false.

✓ *Можна звертатися до окремих символів рядка.*

Наприклад, у результаті виконання фрагмента програми:

```
var s:string;
begin
s:='процесор';
writeln (s[5]);
```

на екрані отримаємо символ e.

- ✓ Якщо до програми підключений модуль *System*, то доступними для опрацювання рядків є такі функції і процедури.

Функції:

`length(s)` – повертає довжину рядка `s` ;

`concat(s1, s2)` – об’єднує рядки `s1` і `s2` ;

`copy(s, n, m)` – із рядка `s` копіюється `m` символів, починаючи з позиції `n`.

`pos(s1, s2)` – повертає номер позиції, з якої рядок `s1` входить у рядок `s2`.

Процедури:

`delete(s, n, m)` – із рядка `s` вилучається `m` символів, починаючи з позиції `n`.

`insert(s1, s, m)` – у рядок `s` вставляється рядок `s1`, починаючи з позиції `m`.

Нижче подано код програми, у якій демонструється використання деяких функцій над рядковими даними.

```
program prog_10;
var s1, s2, s3: string;
begin
  s1:='процесор';
  s2:='цес';
  s3:='новий';
  writeln ('довжина=', length (s1));
  writeln ('перших 6 символів: ', copy (s1, 1, 6));
  writeln ('цес починається з № ', pos (s2, s1));
  writeln ('після об’єднання: ', concat (s3, ' ', s1));
  writeln ('на позиції №5: ', s1[5]);
  readln;
end.
```

У результаті виконання цієї програми на екран буде виведено:

довжина=8

перших 6 символів: процес

цес починається з № 4

після об’єднання: новий процесор


на позиції № 5 : e

Перевіряємо себе

1. Які типи виразів є у мові Pascal? ▲
2. Яким може бути найпростіший вираз? ▲
3. Які існують константи в мові Pascal? ▲
4. Як записуються константи цілих чисел? ▲
5. Як записуються символні константи? ▲
6. Які значення може мати логічна константа? ▲

7. Із яких елементів складається арифметичний вираз? ✨
8. Виконайте операцію `or` над двійковими числами 10100011 і 01010001. ✨
9. Наведіть правила запису арифметичних виразів. ★
10. Яких результатів набувають логічні операції `and` і `or`? ★
11. Поясніть сутність операції `pos` над рядками `s1` і `s2`. ★
12. Наведіть приклад використання функції `length` над рядком. ★

Виконуємо

1.  Проаналізуйте й уведіть програму `sam_10`.

```

program sam_10;
const a1:integer=21; a2:integer=4;
var a3:real;
begin
    a3:=(a1*a1+a2)/2-a1/a2;
    writeln ('a3=', a3)
end.

```

Ця програма розроблена для обчислення значення виразу $((a1)^2+a2)/2-a1/a2$. Виконайте програму і доведіть, що вона функціонує правильно. 🟢






2. Визначте, за яких допустимих значень змінної `x` вираз `not((x=3) or (x>5))` набуде значення *істинне*, а за яких – *хибне*. ✨
3. Проаналізуйте й уведіть програму `sam_11`, яка обчислює значення виразу `((a<10) and (b>7)) or (a>b)`.

```

program sam_11;
const a:integer=4; b:integer=12;
var c:boolean;
begin
    c:=((a<10) and (b>7)) or (a>b);
    writeln ('c=', c)
end.

```

Виконайте програму і покажіть, що вона функціонує правильно. ✨


4.   Розробіть фрагмент програми, за допомогою якого у рядок `монр` вставляється новий рядок, у результаті чого отримуємо слово `монітор`. ★
5.  Розробіть програму для обчислення значення арифметичного виразу $a(b+c)/(a-c)$. ✨
6.  Розробіть програму для обчислення значення логічного виразу `((x>3)and (y=5)) or (x=y)`. Значення змінних вводяться за допомогою клавіатури. ✨
7.  Проаналізуйте програму `sam_12`, яка обчислює значення логічного виразу `((x=y) or (x>y)) and (x=5)`. виправіть помилки та введіть програму. ★

```

program sam_12;
const x:integer=20;
var y:=integer; c:boolean;
begin
    writeln ('увести y')
    readln (y);
    c:=((x=y) or (x>y)) and (x=5)
    writeln ('c=', c);
    readln
end.

```

Після успішної компіляції виконайте програму. За яких значень змінної у програма видає значення *істинне* і за яких – *хибне*?

8.  Розробіть фрагмент програми, у якій використовуються функції `length`, `insert`, `copy` над рядками: фрагмент, мен і пам'яті. ★

6.5. Розроблення програм для реалізації лінійних алгоритмів

Розглянемо декілька прикладів розроблення програм для лінійних алгоритмів.

1. Комп'ютер видає на екран запит учневі: “Ваше ім'я”. Після його введення висвітлюється новий запит – “у якому ви класі?”. Після введення відповіді на друге запитання на екрані висвітлюється: “Ви навчаєтеся ...у класі”.

Позначимо ім'я учня змінною x , а клас – змінною y . Уважатимемо, що максимальна довжина імені учня і назви класу не перевищує 10 символів. Нижче подано програму, що реалізує описаний алгоритм.

```

program n11_10;
var x, y: string [10];           {оголошення змінних рядкового типу}
begin
    writeln ('як Вас звати?'); {прохання ввести ім'я}
    readln (x);                {уведення імені}
    writeln ('у якому ви класі?'); {прохання ввести клас}
    readln (y);                {уведення класу}
    writeln (x, ', Ви навчаєтеся у ', y, 'класі'); {виведення імені і
                                                    класу}
    readln                    {призупинення виконання програми}
end.

```

2. Василь, Галина і Петро збирали гриби. Потім вони зважили зібрані гриби й поділили між собою порівну. Розробимо програму, за допомогою якої визначається, скільки кілограмів грибів дісталось кожному.

Позначимо масу грибів, зібраних Василем, змінною a , масу грибів Галини – змінною b , а масу грибів Петра – змінною c . Змінна s – загальна

маса грибів. Програма, що реалізує це завдання, має ім'я n11_11. Виконайте програму для різних значень змінних і доведіть, що вона функціонує правильно.

```
program n11_11;
var a, b, c, s: real;           {оголошення змінних дійсного типу}
begin
  writeln ('увести a, b, c'); {повідомлення про введення значень
змінних}
  readln (a, b, c);           {уведення значень змінних}
  s:=(a+b+c)/3;              {обчислення середнього значення}
  writeln ('середнє=', s);   {виведення середнього значення}
  readln                     {призупинення виконання програми}
end.
```

3. У квадрат зі стороною a вписано круг. Розробимо програму визначення різниці площ квадрата і круга.

Уважатимемо, що значення змінної a вводиться з клавіатури. Позначимо площу квадрата змінною s_1 , площу кола – змінною s_2 , а різницю площ квадрата і круга – змінною r . Один із варіантів програми, що реалізує завдання, подано нижче.

```
program n11_12;
var a, s1, s2, r: real;        {оголошення змінних дійсного типу}
begin
  writeln ('увести значення змінної a'); {повідомлення про
введення значення змінної a}
  readln (a);                 {уведення значення змінної a}
  s1:=a*a;                    {обчислення значення площі квадрата}
  s2:=3.14*s1/4;              {обчислення значення площі круга}
  r:=s1-s2;                   {обчислення різниці площ}
  writeln ('різниця=', r);    {виведення різниці площ}
  readln                      {призупинення виконання програми}
end.
```

Виконайте програму і переконайтеся, що для значення змінної $a=4$ результат різниці $r=3.4400000000E+00$.

4. Відстань між пунктами А і В дорівнює l км. Одночасно назустріч один одному з пункту А починає бігти учень Микола, а з пункту В – їхати на велосипеді учениця Оля. Швидкість руху Миколи дорівнює v_1 км/год, а Олі – v_2 км/год. Потрібно розробити програму, що визначає, через скільки годин вони зустрінуться.

Уважатимемо, що значення всіх змінних уводяться за допомогою клавіатури. Позначимо час до зустрічі змінною t . Програму виконання поставленого завдання можна подати в такому вигляді.


```

program n11_13;
var t, l, v1, v2: real;           {оголошення змінних дійсного типу}
begin
    writeln ('увести значення l, v1, v2'); {повідомлення про введення
                                           значень змінних}
    readln (l, v1, v2);             {уведення значень змінних}
    t:=1/(v1+v2);                  {обчислення часу зустрічі}
    writeln ('зустрінуться через ', t, ' годин'); {виведення обчисленого
                                                    результату}
    readln                          {призупинення виконання програми}
end.

```

Виконайте програму і доведіть, що вона функціонує правильно.

5. У басейні ємністю 10000000 л залишилося p літрів води. До нього підведені дві помпи, одна з яких щохвилини подає в басейн a літрів води, а друга за кожну хвилину викачує b літрів води ($a > b$). Потрібно розробити програму, що визначає, через скільки хвилин басейн буде повний, за умови, що помпи вмикаються одночасно.


Використаємо такі змінні: t – кількість хвилин, за яку басейн наповниться повністю; m – кількість літрів води, яку потрібно накачати, щоб басейн був повний. Програма, що реалізує наведене завдання, може мати такий зміст.

```


program n11_14;
var p, a, b, m, t: real;         {оголошення змінних дійсного типу}
begin
    writeln ('увести значення змінних p, a, b'); {повідомлення про
                                                    введення}
    readln (p, a, b);              {уведення значень змінних p, a, b}
    m:=10000000-p;                {кількість літрів, які потрібно долити}
    t:=m/(a-b);                   {обчислення часу наповнення басейну}
    writeln ('наповниться через ', t, ' хвилин'); {виведення результату}
    readln                          {призупинення виконання програми}
end.

```

Виконуємо

1. Розробіть алгоритм і програму обчислення значення функції $y=(5x-1)(x+1)+x^2$ для будь якого значення x . ★
2. Розробіть алгоритм і програму обчислення значення функції $y=(a^2+b+c)/(a^2+b+c-4ac)$ для додатних значень a, b, c за умови, що знаменник не дорівнює нулю. ★
3.  Автомобіль і пішохід одночасно розпочали рух з пункту А в пункт В, які розташовані на відстані l км один від одного. Автомобіль, досягнувши пункту В, не затримуючись, повертає і рухається в пункт А.

Через 4 години після початку руху він зустрічає пішохода на відстані $l/4$ від пункту А. Розробіть алгоритм і програму для визначення, з якою швидкістю рухалися автомобіль і пішохід. ★

4.  Відстань від Харкова до Полтави автошляхом дорівнює 144 км. О 8 годині ранку з Харкова у напрямку до Полтави починає рух велосипедист зі швидкістю p км/год. О 10 годині ранку з Полтави у напрямку до Харкова починає рух мотоцикліст. Розробіть алгоритм і програму у візуальному середовищі Lazarus для визначення, з якою середньою швидкістю повинен рухатися мотоцикліст, щоб їхня зустріч відбулася на середині шляху. ★

**Практична
робота № 10**

Тема:	Описання та виконання лінійних алгоритмів опрацювання величин у середовищі програмування
Мета:	Набути практичних навичок описання та виконання лінійних алгоритмів

Завдання. Є кімната, довжина якої m м, ширина n м і висота h м. У одній стіні є двері висотою h_1 м і шириною n_1 м, а в другій – вікно шириною n_2 м і висотою h_2 м. Скільки повних рулонів шпалер шириною 1 м і довжиною 10 м потрібно, щоб обклеїти стіни кімнати?

Підказка: Значення параметрів рулонів шпалер можна ввести як типізовані константи, а для обчислення кількості повних рулонів використати “заокруглення вгору”.

1. Розробити програму обчислення потрібної кількості рулонів шпалер.
2. Увести програму. Виконати її компіляцію. виправити усі синтаксичні помилки й запустити програму на виконання.
3. Довести, що програма виконується правильно.
4. Зберегти програму.

**Практична
робота № 11**

Тема:	Налагодження програми
Мета:	Набути практичних навичок налагодження програм із віконним інтерфейсом

Завдання.

1. Розробити віконний інтерфейс програми обчислення потрібної кількості рулонів (Практична робота № 10).
2. Виконати компіляцію. виправити всі синтаксичні помилки й запустити програму на виконання.
3. Відкоригувати розміри, колір об’єктів на формі, виконати компілювання програми в остаточному вигляді.
4. Довести, що програма виконується правильно.
5. Зберегти програму.



Дані логічного типу – дані, що набувають лише двох значень (true і false).

Операнди виразу – константи, змінні та функції, значення яких (повернуті якими) використовуються для обчислення значення, яке має повернути вираз.

Рядковий тип даних – послідовність символів довжиною до 255 знакомісць.

Символьний тип даних – окремі символи.

Стандартні функції – найчастіше вживані функції, для обчислення значень яких розроблені спеціальні програми.

Структуровані типи даних – дані, що об'єднуються у структури (масиви, записи, множини тощо).

6.6. Розроблення програм для реалізації алгоритмів із розгалуженнями

Раніше вже розглядалась алгоритмічна конструкція розгалуження, яка дає змогу виконавцеві алгоритму обрати один із двох шляхів подальших дій залежно від істинності певної умови. У багатьох мовах програмування високого рівня, зокрема в мові Pascal, для реалізації алгоритмів із розгалуженою структурою використовуються такі оператори умовного переходу:

```
if <умова> then <оператор>;
```

```
if <умова> then <оператор 1> else <оператор 2>;
```

Тут **if** (якщо), **then** (то), **else** (інакше) – зарезервовані слова, <умова> – довільний логічний вираз, <оператор 1> і <оператор 2> – довільні оператори.

Перший оператор реалізує одноальтернативне розгалуження, другий – двохальтернативне.



Зазначмо, що перший тип оператора умовного переходу називають його скороченою формою.


Виконання скороченої форми оператора умовного переходу починається з обчислення значення логічного виразу <умова>. Якщо умова істинна, то виконується <оператор>, якщо ж хибна, то виконання умовного оператора на цьому завершується.


Приклад 1. Розглянемо такий фрагмент програми тестування учня, в якій змінна `vidpovid` призначена для введення учнем відповіді на чергове запитання, змінна `prav` містить правильну відповідь, а змінна `osinka` містить накопичену оцінку до відповіді на запитання:

```
if vidpovid = prav then osinka := osinka + 1;
```

При виконанні вказаної команди спочатку буде перевірена умова `vidpovid = prav`, яка буде істинною, якщо учень дасть правильну відповідь, і в цьому випадку виконається оператор `osinka := osinka + 1`, тобто оцінка учня зросте на 1 бал. Якщо ж учень дасть неправильну відповідь,

то значення умова `vidpovid = prav` буде хибним, команда розгалуження завершить роботу й оцінка залишиться без змін.

 Оператор умовного переходу, записаний у повній формі, виконується так. Спочатку обчислюється значення булевого виразу <умова>. Якщо умова істинна, то виконується <оператор 1> і керування передається наступному за умовним оператору (<оператор 2> пропускається). Якщо ж умова хибна, то <оператор 1> пропускається, а виконується лише <оператор 2> і на цьому дія умовного оператора завершується.

 *Звернімо увагу на таку синтаксичну деталь.* Символ “ ; ” є символом, що відокремлює оператори. Оскільки умовний оператор завершується діями, які записані після слова `else`, то запис крапки з комою перед `else` є синтаксичною помилкою.

Зазначимо, що після ключових слів `then` та `else` можуть бути розташовані інші оператори умовного переходу. Така конструкція використовується, якщо є більше двох можливих варіантів дій. Розглянемо програму, в якій один оператор умовного переходу вкладено в інший.

Приклад 2. Обчислимо значення такої функції:

$$y = \begin{cases} x + 3, & \text{якщо } x < 4 \\ 1, & \text{якщо } 4 \leq x \leq 5 \\ 2x + 4, & \text{якщо } x > 5 \end{cases}$$

Блок-схема розв’язування задачі представлена на рис. 6.7. За цією блок-схемою складемо програму. Гілка “ні” першого розгалуження, якій відповідає слово `else` оператора `if`, містить друге розгалуження. Тому після слова `else` першого оператора `if` слід записати другий оператор умовного переходу.

```
program n_2;
var x,y:real;
begin
  writeln ('уведіть x');
  readln (x);    {уведення значення змінної x}
  {обчислення значення функції}
  if x<4 then
    y:=x+3
  else
    if x>5 then
      y:=2*x+4
    else
      y:=1;
  writeln ('y=',y);    {виведення значення функції}
end.
```

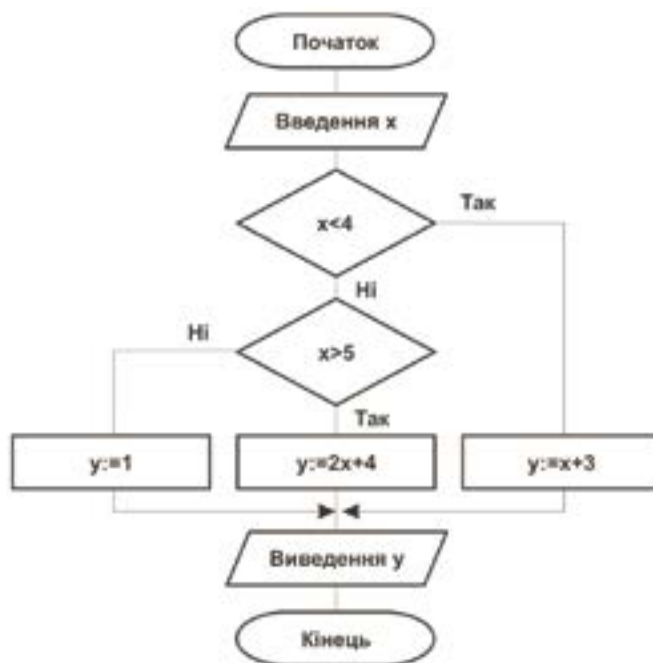


Рис. 6.7. Блок-схема алгоритму обчислення значення функції

Зазначимо, що в логічних виразах, які використовуються в операторах умовного переходу, інколи доцільно застосовувати логічні операції (not, or, and). Використання цих операцій дає можливість скоротити кількість операторів умовного переходу.

Приклад 3. Напишемо програму для обчислення значення функції:

$$y = \begin{cases} k + x, & \text{якщо } x < 0,5 \text{ і } k \geq 1 \\ 2k - x, & \text{інакше} \end{cases}$$

```

program p_3;
var x,k,y:real;
begin
  writeln ('уведіть x,k');
  read (x,k); {уведення значень змінних x,k}
  {обчислення значення функції}
  if (x<0.5) and (k>=1) then y:=k+x else y:=2*k-x;
  writeln ('y=',y); {виведення значення функції}
end.
  
```

✓ У мові Pascal дія ключових слів *then* та *else* поширюється лише на один, наступний, оператор.

Якщо в разі істинності або хибності певного логічного твердження необхідно виконати кілька операторів, то їх слід оточити *операторними дужками*, роль яких відіграють ключові слова **begin** і **end**.

```

if <умова> then
begin
    <оператор 1>;
    ...
    <оператор N>
end
else
begin
    <оператор K>;
    ...
    <оператор M>
end;

```

Якщо логічний вираз <умова> істинний, виконуватимуться оператори <оператор 1>, ..., <оператор N>, якщо хибний – оператори <оператор K>, ..., <оператор M>. Групу операторів, що розпочинається зі слова **begin**, а завершується словом **end**, можна розглядати як один складений оператор. Розглянемо приклад використання складеного оператора.

Приклад 4. Напишемо програму для обчислення значення функції:

$$\begin{aligned}
 p &= \sqrt{x} - y, && \text{якщо } x > 1 \text{ і } y < 4 \\
 q &= |x > y|, && \text{якщо } x > 1 \text{ і } y < 4 \\
 &\text{інакше } t = x/y
 \end{aligned}$$

```

program n_4;
var
    x, y, p, q, t: real;
begin
    writeln ('уведіть x,y');
    readln (x,y); {уведення значень змінних x,y}
    if (x>1) and (y<4) then
        {перевірка умови}
        begin
            p:=sqrt(x)-y; {обчислення значення функції p}
            q:=abs(x+y); {обчислення значення функції q}
            writeln (p, ' ', q) {виведення значень функцій p та q}
        end
    else
        begin
            t:=x/y; {обчислення значення функції t}
            writeln (t) {виведення значення функції t}
        end;
end.

```

6.7. Розроблення програм для реалізації алгоритмів із повтореннями для опрацювання величин

Оператори циклу призначені для реалізації циклічних алгоритмічних структур. Можливі дві принципово різні ситуації:

- кількість повторень відома наперед;
- кількість повторень заздалегідь визначити неможливо.

Зрозуміло, що управління циклом у цих випадках здійснюється по-різному. У першому випадку управління здійснюється за допомогою параметра циклу – змінної, яка послідовно змінює своє значення, а в другому випадку використовується умова – вираз логічного типу, від значення якого залежить виконання чи завершення циклу.

У мові Pascal реалізовано три різновиди операторів циклу:

- цикл з лічильником,
- цикл з передумовою,
- цикл із післяумовою.



Будь-який оператор циклу складається з двох частин: **заголовка циклу** й **тіла циклу**.

У заголовку циклу записуються умови, за яких виконання циклу триватиме або завершиться, а в тілі циклу містяться оператори, виконання яких потрібно повторювати.



Оператор циклу з лічильником

Оператор циклу з параметром (або, як його ще називають, циклу з лічильником) має такий синтаксис:

```
for <змінна> := <початкове значення> to <кінцеве значення> do  
<оператор>
```

або

```
for <змінна> := <початкове значення> downto <кінцеве значення>  
do <оператор>
```

Тут

for, to, downto, do – службові слова;

<змінна> – ідентифікатор деякої змінної порядкового типу даних, яка називається лічильником;

<початкове значення> – вираз, тип якого збігається з типом лічильника і його значення стає початковим значенням лічильника;

<кінцеве значення> – вираз, тип якого збігається з типом лічильника і задає його кінцеве значення;

<оператор> – оператор тіла циклу.

Робота цього оператора здійснюється за таким алгоритмом.

1. Обчислюється початкове значення.
2. Це значення присвоюється параметру.
3. Обчислюється кінцеве значення.
4. Порівнюється значення параметра з кінцевим значенням.

5. Якщо воно перевищує (у випадку з використанням слова **to**) кінцеве значення або менше (у випадку з використанням слова **downto**), ніж кінцеве значення, то оператор циклу завершує роботу. У іншому випадку виконується тіло циклу і значення параметра змінюється на наступне (попереднє); відбувається перехід до пункту 4.

Особливо слід підкреслити, що:

- і початкове, і кінцеве значення обчислюються до виконання оператора циклу і більше не перераховуються;
- ідентифікатор змінної є параметром циклу і згідно зі стандартом не повинен змінюватися всередині оператора циклу, нехтування цим правилом може призвести до непередбачуваних наслідків;
- після завершення циклу значення параметра циклу вважається не визначеним.

Приклад 1. Вивести таблицю квадратів натуральних чисел другого десятка.

```
for n := 11 to 20 do writeln(n, n * n: 5);
```

Приклад 2. Вивести великі латинські літери на екран у зворотному порядку.

```
for ch := 'Z' downto 'A' do write(ch: 2);
```

Якщо тіло циклу містить більш ніж один оператор, то ці оператори записуються в операторних дужках **begin...end**;

Приклад 3. Використання складеного оператора в тілі циклу:

```
for i:=1 to n do
begin
  s:=s+a;
  y:=2*x
end;
```

✓ Оператор **while...do**

Цикл із передумовою реалізується оператором **while...do**, синтаксис якого наведено нижче. У цьому операторі умова продовження циклу перевіряється перед початком кожного виконання операторів тіла циклу.

```
while <логічний вираз> do      {заголовок циклу}
<оператор>;                       {тіло циклу}
```

Оператор тіла циклу виконуватиметься доти, доки істинним буде логічний вираз, який вказаний у заголовку циклу. Якщо в тілі циклу міститься декілька операторів, то вони беруться в операторні дужки **begin...end**;. Блок-схема оператора **while...do** зображена на рис. 6.8.

Приклад 4. Із клавіатури послідовно вводяться і додаються цілі числа, які не перевищують 10. Процес потрібно завершити при введенні числа більше 10.



Рис. 6.8. Блок-схема оператора циклу з передумовою

Програма, що розв’язує цю задачу, наведена нижче.

```
program n14_1;
var x, s: integer;           {оголошення змінних типу integer}
begin
  s:=0; x:=0;               {початкові значення змінних}
  while x<10 do             {початок циклу}
  begin
    s:=s+x;                 {обчислення суми введених чисел}
    writeln ('увести x');   {повідомлення про введення}
    readln (x)              {уведення числа}
  end;
  writeln ('s=', s);        {виведення значення суми}
  readln                    {призупинення виконання програми}
end.
```

✓ Оператор repeat...until

Цикл із післяумовою реалізується оператором repeat...until, у якому умова завершення циклу перевіряється після кожного виконання операторів тіла циклу. Синтаксис цього оператора такий:

```
repeat
<оператори>                {тіло циклу}
until <логічний вираз>    {заголовок циклу}
```

Оператори тіла циклу repeat...until виконуються доти, доки записаний після слова until логічний вираз є хибним.

Після останнього оператора в тілі циклу repeat...until символ “;” не записується. Оператори тіла циклу з післяумовою можна не брати в операторні дужки, навіть у тому випадку, коли їх декілька. Блок-схему оператора repeat...until зображено на рис. 6.9.

Приклад 5. Наведемо програму знаходження найбільшого спільного дільника цілих чисел, які вводяться з клавіатури. У програмі реалізовано алгоритм Евкліда.

```
program n14_2;
var a, b: integer;          {оголошення змінних цілочислового типу}
begin
  writeln ('увести значення a, b'); {повідомлення про введення}
  readln (a, b);            {уведення значень a, b}
  repeat {початок циклу}
    if a>b then a:=a-b;      {оператор тіла циклу}
    if b>a then b:=b-a;      {оператор тіла циклу}
  until a=b;                {перевірка умови завершення циклу}
```

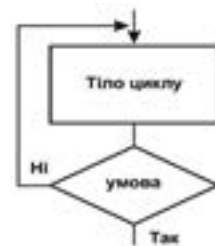


Рис. 6.9. Блок-схема оператора циклу з післяумовою

```
writeln ('НСД=', a);           {виведення результату}
readln                        {призупинення виконання програми}
end.
```

- ✓ У деяких задачах всередині операторів циклу доцільно використовувати оператори *break* і *continue*.

Перший із них перериває виконання циклу, а другий перериває виконання операторів тіла циклу і передає керування його заголовку.

Приклад 6. Розглянемо гру. Двоє гравців домовилися кидати гральні кості по черзі не більше 1000 разів. Виграє той, хто набере більшу суму балів. Однак гравці можуть кидати кості лише доти, доки суддя не підніме червоний прапорець. Нижче наведено фрагмент програми, яка моделює гру:

```
for i:=1 to 1000 do           {заголовок циклу}
begin
  <оператори тіла циклу>;
  if prapor='червоний' then
    break                     {якщо підняти червоний прапорець,
                               цикл завершити}
  end;                         {кінець циклу}
```


Перевіряємо себе

1. Для чого використовуються оператори циклу? ▲
2. Яку структуру має оператор циклу *repeat...until*? ▲
3. Які ключові слова використовуються в операторі циклу з параметром? ◆
4. Як виконується оператор циклу *while...do*? ★

Виконуємо

1. Визначте, що буде виведено в результаті виконання фрагмента програми:

```
var i, y, b: integer;
begin
  y:=1; b:=4;
  for i:=2 to b do y:=y+1;
  y=y*y;
  writeln (y);
end;
```

2.  Розробіть програму для розв'язування таких задач:

- а) вивести значення \sqrt{x} для парних чисел x у діапазоні від 2 до 10;
- б) визначити суму цілих додатних непарних чисел, які менші 30.

Практична робота № 12

Тема:	Описання та виконання алгоритмів з повтореннями та розгалуженнями для опрацювання величин
Мета:	Набути практичних навичок створення програм із повтореннями та розгалуженнями для опрацювання величин

Завдання. Створити алгоритм, описати його мовою Паскаль, увести й налагодити програму.

1. Задано два парних числа a_1 і a_2 . Знайти суму всіх непарних чисел, більших a_1 , але менших a_2 . $a_1 = 20$; $a_2 = 60$.

2. Для випробування нового автомобіля вирішено першого дня проїхати z км, а кожного наступного дня збільшувати пробіг на y відсотків порівняно з попереднім днем. Через скільки днів дистанція пробігу досягне s км? $z = 00$; $y = 3$; $s = 500$.

3. Прямокутні паралелепіпеди мають в основі квадрат зі стороною r , висота першого дорівнює h , а висота кожного наступного збільшується на z . Висота останнього дорівнює H . Знайти об'єм кожного паралелепіпеда. $r = 5$; $h = 3$; $z = 0.2$; $H = 5$.

Підказка: Звернути увагу на вибір ідентифікаторів для висот паралелепіпедів.

4. З яблуні було знято m яблук і покладено порівну в n кошиків ($m > n$), але декілька яблук залишилося (менше ніж n). Скільки яблук покладено в кожний кошик і скільки залишилося? $m = 37$; $n = 7$.

6.8. Розроблення програм для реалізації алгоритмів із графічним відображенням даних

Створення геометричних та інших фігур, використання у проектах рисунків із зовнішніх файлів.

Середовище Lazarus дає змогу розробляти проекти, в процесі виконання яких створюються схеми, ілюстрації та інші графічні зображення. Засоби і принципи їх створення до деякої міри аналогічні тим, якими користується художник під час створення малюнка.

Як відомо, основними засобами праці художника є полотно, олівці, пензлі, акварельні фарби тощо. Фактично такі ж віртуальні засоби має і середовище Lazarus. Художник, малюючи, наносить на полотно точки, лінії, кола, квадрати та інші графічні примітиви різного кольору, стилю й товщини. До деякої міри аналогом таких дій художника в системі Lazarus є спеціальні підпрограми, які створюють найпростіші графічні зображення.

6.8.1. Створення графічних зображень у середовищі Lazarus

✓ Підпрограми, які виконують такі функції, називають *методами*.

Для створення графічних зображень середовище Lazarus містить **спеціалізовані класи**, основними з яких є TCanvas (канва, полотно), TFont (шрифт), TPen (перо) і TBrush (пензель).

✓ *Об'єкти, пов'язані з цими класами, автоматично створюються для всіх видимих елементів і доступні через їх властивості Canvas, Font, Pen і Brush.*

Кожний об'єкт має властивості й методи, за допомогою яких і створюються графічні зображення.

Графічне зображення виводиться на поверхню об'єкта (найчастіше форми Form). Поверхні об'єкта Form для рисування відповідає властивість Canvas. Отже, Canvas – це властивість об'єкта Form, це полотно, що розміщується на формі.

Водночас полотно є об'єктом типу TCanvas, який має свої **методи** для рисування на полотні прямокутників, ліній, кіл та інших зображень. Цей об'єкт також має властивості, які дають змогу визначати стиль графічних зображень (товщину лінії, колір лінії, колір тла заливки тощо).

Полотно складається з окремих точок (**пікселів**). Положення пікселів на полотні визначається їх координатами, тобто номером пікселя по горизонталі (координата x) і номером пікселя по вертикалі (координата y).

Координати (0, 0) – це лівий верхній кут канви. Координати зростають зверху вниз і зліва направо. Значення координат правої нижньої точки полотна залежить від його розміру. Розмір можна дізнатися, звернувшись до властивостей Height і Width об'єкта, на якому виконується малюнок. Кольори пікселів задаються властивістю Pixels. Змінити колір будь-якого пікселя на об'єкті Form1 можна за допомогою команди:

Form1.Canvas.Pixels [X, Y]:=Color;, де Color – змінна або константа типу TColor.

Найчастіше вживані значення властивості Color містяться в табл. 6.7.

Таблиця 6.7

Основні значення властивості Color

Значення	Назва	Значення	Назва
clGreen	Зелений	clAqua	Блакитний
clGray	Сірий	clBlak	Чорний
clRed	Червоний	clGray	Сірий
clBlue	Синій	clYellow	Жовтий
clWhite	Білий	clTeal	Бірюзовий

Колір будь-якого пікселя об'єкта Form1 можна отримати за допомогою команди: Color:=Form1.Canvas.Pixels [X, Y];, де Color – змінна типу TColor.

Розглянемо найпростіший приклад створення графічного зображення. Створимо проект, за допомогою якого на формі відображається прямокутник. Для цього створюємо проект звичайним способом.

✓ Нагадаємо, що після запуску Lazarus необхідно виконати команду **Проект** → **Новий проект...** й у вікні, що відкриється, виконати команду **Програма** → **Гаразд**.

З'явиться вікно форми, у яке розмістимо поле для рисування `TPaintBox` (цей компонент розташований на вкладці `Additional`) і об'єкт `TButton`. У інспекторі об'єктів можна встановити деякі властивості цих об'єктів. Наприклад, для об'єкта `PaintBox1` встановимо значення `Width` 150 (ширина по осі X) і значення `Height` 120 (висота по осі Y).

Активуємо об'єкт `Button1`, вибираємо в інспекторі об'єктів на вкладці **Події** подію `OnClick` і двічі клацаємо кнопкою миші праворуч від назви цієї події. У вікні редактора тексту автоматично буде створена процедура:

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
end;
```

Між словами **begin** **end** потрібно написати код, за допомогою якого створюватиметься графічне зображення. Для рисування прямокутника використовується метод `Rectangle`. Для рисування його на об'єкті `PaintBox1` можна скористатися командою:

```
PaintBox1.Canvas.Rectangle(x1, y1, x2, y2);
```

де x_1 та y_1 – координати лівого верхнього кута прямокутника, а x_2 та y_2 – координати його правого нижнього кута. З урахуванням зазначеного код рисування прямокутника набуде змісту, зображеного на рис. 6.10.

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    PaintBox1.Canvas.Rectangle(10, 20, 100, 100);  
end;  
end.
```

Рис. 6.10. Код для зображення на формі прямокутника



Рис. 6.11. Прямокутник на формі

Після успішної компіляції програми запускаємо програму і натискаємо кнопку `Button1`. З'явиться прямокутник, зображений на рис. 6.11.

Якщо потрібно, щоб методи малювання виконувалися безпосередньо після запуску програми (без використання кнопки `Button1`), слід для форми вибрати в інспекторі об'єктів подію `OnPaint` і створити код, наприклад такий, який представлений на рис. 6.12.

```

procedure TForm1.FormPaint(Sender: TObject);
begin
    Form1.Canvas.Rectangle(100, 50, 300, 100);
end;
end.

```

Рис. 6.12. Код рисування прямокутника для події OnPaint

Після запуску цього проекту на формі одразу з'явиться прямокутник.

Перевіряємо себе

1. Назвіть основні класи для створення графічних зображень. ▲
2. Які основні об'єкти використовуються для рисування? ▲
3. Назвіть основні значення властивості Color. ▲
4. З чого складається полотно? ✦
5. Яку систему координат має полотно? ✦
6. За допомогою якої команди можна на формі нарисувати прямокутник? ✦
7. Як можна отримати колір пікселя об'єкта Form1? ★
8. Як для об'єкта Button1 можна встановити подію OnClick? ★

6.8.2. Інструменти й методи для створення графічних зображень

Для створення графічних зображень використовуються методи класу TCanvas, які дають змогу будувати еліпси, лінії, багатокутники тощо й виводити текст у графічному режимі, і три класи, що визначають інструменти виведення фігур і текстів:

- TFont – (шрифти);
- TPen – (олівець, перо);
- TBrush – (пензель).

6.8.3. Графічні інструменти

Клас TPen слугує для створення об'єкта перо, призначеного для креслення ліній і контурів. Він має такі основні властивості:

- * Color – визначає колір ліній, що кресляться пером;
- * Style – визначає стиль (вид) ліній;
- * Width – товщина ліній у пікселях.

Тип Style визначено як перелічувальний тип, можливі значення якого наведені в табл. 6.8.

Стилі ліній

Стиль	Значення
psSolid	Суцільна лінія (значення за замовчуванням)
psDash	Штрихова лінія
psDot	Лінія з крапок (пунктирна)
psDashDot	Штрих-пунктирна лінія
psDashDotDot	Штрих-пунктир-пунктирна лінія
psClear	Відсутність ліній

Властивості класу `TPen` за замовчуванням мають такі значення: `Color=clBlack`, `Mode=pmCopy`, `Style=psSolid`, `Width=1`.

Клас `TBrush`. Цей клас визначає колір і зразок для заливання замкнених фігур. Він має такі властивості:

`Bitmap` – слугує для запису растрового зображення, що використовується при заповненні пензлем;

`Color` – колір пензля;

`Style` – стиль пензля.

За замовчуванням властивість `Color` має значення `clWhite` (білий), а властивість `Style` – значення `bsSolid` (суцільна лінія). Якщо властивості `Style` присвоєне значення `bsClear`, значення властивості `Color` ігнорується, але в неї записується значення `clWhite`. Цей клас також має свої методи, які тут не розглядаються.

Клас `TFont`. Шрифт має такі характеристики, як ім'я, стиль, розмір. Його основні властивості:

`Charset` – номер набору символів. За замовчуванням властивості присвоюється значення `DEFAULT_CHARSET`;

`Color` – колір шрифту;

`Height` – висота шрифту в екранних пікселях;

`Style` – стиль шрифту. За замовчуванням використовуються звичайні символи.

Тип `Style` може мати такі значення:

– `fsBold` – жирний шрифт;

– `fsItalic` – курсив;

– `fsUnderline` – шрифт, підкреслений однією прямою лінією;

– `fsStrikeOut` – шрифт, перекреслений однією прямою лінією.

Стиль шрифту може визначатися не одним значенням, а їх комбінацією, наприклад, жирний курсив.

6.8.4. Методи класу TCanvas

Клас TCanvas створює канву, на якій можна рисувати пером, пензлем і вводити текст шрифтом. У класі TCanvas є достатня кількість методів роботи з графічними інструментами Pen, Brush і Font. Нижче наведені ті з них, які застосовуються найчастіше і які ми використовуватимемо в процесі розроблення проектів.

1. Методи для роботи з пером:

MoveTo (X, Y: Integer) – переміщує перо в зазначену точку без креслення лінії;

LineTo (X, Y: Integer) – креслить відрізок прямої лінії від поточного положення пера до зазначеної точки, не захоплюючи останню;

Arc (X1, Y1, X2, Y2, X3, Y3, X4, Y4: Integer) – креслить дугу еліпса, вписаного у прямокутник зі сторонами, паралельними межах канви, і двома кутами в точках (X1, Y1) і (X2, Y2). Початок дуги міститься в точці перетину променя, що проходить через центр еліпса і точку (X3, Y3), а кінець дуги – на перетині променя, що проходить через центр еліпса і точку (X4, Y4).

Розробимо програму побудови на формі двох ліній і дуги. Перша лінія червоного кольору, товщина – за замовчуванням; починається в точці з координатами 10, 10 і закінчується в точці з координатами 100, 10. Друга лінія синього кольору, стиль – пунктир, колір – синій. Лінія починається в точці з координатами 10, 40 і закінчується в точці з координатами 100, 40. Дуга зеленого кольору, стиль і товщина – за замовчуванням. Програма реалізації цього завдання зображена на рис. 6.13.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  Form1.Canvas.Pen.Color:=clGreen; //олівець зеленого кольору
  Form1.Canvas.Arc (10,60,110,160,30,60,90,160); //будова дуги
  Form1.Canvas.MoveTo (10,10); //переміщення олівця
  Form1.Canvas.Pen.Color:=clRed; //олівець червоного кольору
  Form1.Canvas.LineTo (100,10); //будова першої лінії
  Form1.Canvas.Pen.Style:=psDash; //стиль олівця
  Form1.Canvas.Pen.Width:=5; //товщина олівця
  Form1.Canvas.Pen.Color:=clBlue; //олівець синього кольору
  Form1.Canvas.MoveTo (10,40); //переміщення олівця
  Form1.Canvas.LineTo (100,40); //будова другої лінії
end;
end.
```

Рис. 6.13. Код побудови ліній і дуги

У результаті виконання програми отримаємо зображення, представлені на рис. 6.14.



Рис. 6.14. Зображення на формі двох ліній і дуги

2. Методи для роботи з пером і пензлем:

Rectangle (X1, Y1, X2, Y2: Integer) – креслить прямокутник зі сторонами, що паралельні межах канви, й діагоналлю, заданою вершинами в точках (X1, Y1) і (X2, Y2), із заповненням його актуальним пензлем;

RoundRect (X1, Y1, X2, Y2, X3, Y3: Integer) – креслить прямокутник зі згладженими кутами. Перші чотири параметри аналогічні параметрам методу **Rectangle**, а параметри X3 і Y3 задають відповідно ширину й висоту прямокутника, що охоплює еліпс, дуги якого використовуються для згладжування кутів;

Ellipse (X1, Y1, X2, Y2: Integer) – креслить еліпс в охоплюючому прямокутнику і заповнює його актуальним пензлем відповідно до властивостей останнього;

Chord (X1, Y1, X2, Y2, X3, Y3, X4, Y4: Integer) – креслить дугу еліпса, сполучаючи кінці дуги хордою і заповнюючи отриману фігуру пензлем;

Pie (X1, Y1, X2, Y2, X3, Y3, X4, Y4: Integer) – виконує такі ж дії, як і метод **Chord**, але кінці дуги сполучаються відрізками прямої з центром еліпса (створюється сектор еліпса).

Для скорочення тексту програми в ній на початку коду доцільно використовувати оператор **with**. Він забезпечує доступ до властивостей і методів об'єкта **Canvas**. Він має таку структуру: **with Canvas do begin**. На рис. 6.15 зображено код будови еліпса червоного кольору й сектора зеленого кольору з використанням оператора **with**.

```

procedure TForm1.Button1Click(Sender: TObject);
begin
  with Canvas do begin
    Brush.Color:=clRed;      //пензель червоного кольору
    Pen.Width:=3;           //перо товщиною 3 пікселя
    Pen.Style:=psDot;       //стиль пера (пунктир)
    Ellipse (10, 10, 100, 100); //будова еліпса
    Brush.Color:=clGreen;   //пензель зеленого кольору
    Pie (10, 110, 100,200,50,110,100,160); //будова сектора
  end;
end;
end.

```

Рис. 6.15. Код побудови еліпса й сектора



Рис. 6.16. Зображення на формі кола й сектора

Звернімо увагу на те, що розміри еліпса обмежені квадратом, тому в цьому випадку буде побудоване коло. У результаті виконання програми отримуємо геометричні фігури, зображені на рис. 6.16.

3. Методи для роботи з пензлем:

`FillRect (x1, y1, x2, y2)` – прямокутник зафарбовується пензлем актуального кольору.

`FrameRect (x1, y1, x2, y2)` – прямокутник обводиться пензлем актуального кольору.

```

procedure TForm1.Button1Click(Sender: TObject);
begin
  with Canvas do begin
    Brush.Color:=clBlue;           //пензель синього кольору
    FillRect (100, 50, 200, 110); //зафарбований прямокутник
    Brush.Color:=clRed;           //пензель червоного кольору
    FrameRect (100, 130, 170, 170); //прямокутник не зафарбований
  end;
end;
end.

```

Рис. 6.17. Код побудови двох прямокутників

Розробимо код побудови двох прямокутників різних розмірів, розташованих один під одним. Область верхнього прямокутника має синій колір, а нижній обводиться пензлем червоного кольору і не зафарбовується. Код програми, що будувє на формі таке зображення, поданий на рис. 6.17, а результат його виконання – на рис. 6.18.



Рис. 6.18. Зображення на формі двох прямокутників

4. Методи роботи зі шрифтом:

Розглянемо лише метод `TextOut (X, Y, Text)`, який виводить рядок `Text` так, щоб лівий верхній кут прямокутника, що його охоплює, роз-

міщувався в точці (X, Y). Перо наразі переміщується у правий верхній кут виведеного прямокутника.

На рис. 6.19 показано програмний код побудови трьох геометричних фігур, усередині кожної з них наведені їх назви.

На рис. 6.20 зображено результат виконання програмного коду.

Розробимо ще один код для створення на формі трьох геометричних фігур, варіант яких зображено на рис. 6.21.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  with Canvas do begin
    Brush.Color:=clYellow; //пензель жовтого кольору
    Rectangle (10,10,110,50); //будова прямокутника
    TextOut (20,20, 'Прямокутник'); //текст у прямокутнику
    Pen.Color:=clRed; //перо червоного кольору
    MoveTo (10,170); //перо у початкову точку
    LineTo (30,70); //креслення лінії
    LineTo (160,170); //креслення лінії
    LineTo (10,170); //креслення лінії
    Brush.Color:=clGray; //пензель сірого кольору
    TextOut (30, 140, 'Трикутник'); //текст у трикутнику
    Brush.Color:=clRed; //пензель червоного кольору
    Pen.Width:=3; //перо товщиною 3 пікселя
    Ellipse (150,70,230,120); //будова еліпса
    Pen.Width:=1; //перо товщиною 1 піксель
    TextOut (170,90, 'Еліпс'); //текст в еліпсі
  end;
end;
end.
```

Рис. 6.19. Код побудови об'єктів із текстом



Рис. 6.20. Об'єкти форми з текстом

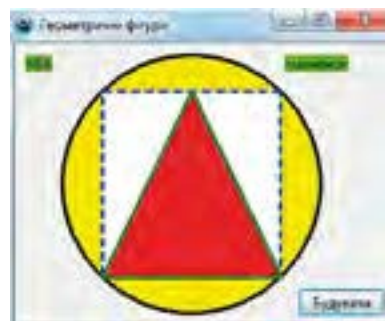


Рис. 6.21. Геометричні фігури

Код, що відображає ці геометричні фігури з текстом, поданий на рис. 6.22.

```

1 procedure TForm1.Button1Click(Sender: TObject);
2 begin
3   with Canvas do begin
4     Pen.Width:=3;
5     Pen.Style:=psSolid;
6     Brush.Color:=clYellow;
7     Ellipse (40,10,260,230);
8     Pen.Style:=psDash;
9     Pen.Color:=clBlue;
10    Brush.Color:=clWhite;
11    Rectangle (74,41,226,199);
12    Pen.Style:=psSolid;
13    Pen.Color:=clGreen;
14    Brush.Color:=clRed;
15    Polygon ([Point (74,199),Point (150,41),Point (226,199)]);
16    Brush.Color:=clLime;
17    TextOut (10,10, 'MiЙ');
18    TextOut (230,10, 'малюнок');
19    end;
20 end;
21 end.


```

Рис. 6.22. Код побудови трьох геометричних фігур

Перевіряємо себе

1. Які інструменти використовуються для виведення фігур і тексту? ▲
2. Назвіть основні властивості об'єкта перо. ▲
3. Назвіть основні властивості об'єкта пензель. ▲
4. Які значення властивостей має об'єкт пензель? ▲
5. Назвіть основні властивості шрифту. ▲
6. Які існують стилі ліній? ◆
7. Які значення властивостей класу TPen застосовуються за замовчуванням? ◆
8. Які значення властивостей має шрифт? ◆
9. Назвіть основні методи для роботи з пером. ◆
10. Назвіть основний метод для роботи зі шрифтом. ◆
11. Назвіть основні методи для роботи з пензлем. ★
12. Наведіть приклад команди для креслення хорди. ★
13. Наведіть приклад команди для креслення сектора. ★
14. Наведіть приклад команди для креслення еліпса. ★
15. Наведіть приклад команди для креслення багатокутника. ★

Виконуємо

1.  Розробіть код для креслення піраміди, зразок якої зображено на рис. 6.23.

Порівняйте розроблений код із кодом, зображеним на рис. 6.24. Укажіть переваги та недоліки кожного з них. ★

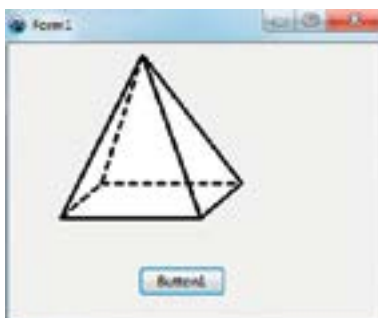





Рис. 6.23. Піраміда у вікні форми

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  with Canvas do begin
    Pen.Width:=3;
    Pen.Color:=clBlack;
    Polygon ([Point(115,10), Point(45,150), Point(165,150),
    Point(200,120), Point(115,10), Point(165,150)]);
    MoveTo (45,150);
    Pen.Style:=psDash;
    LineTo(80,120); LineTo(200,120);
    MoveTo (115,10);
    LineTo (80,120);
  end;
end;
end.
```

Рис. 6.24. Код побудови піраміди

2.  Розробіть код, за допомогою якого зображується квадрат, у який вписане коло. Колір і стиль ліній оберіть самостійно. ★
3.  Розробіть код для креслення фасаду дачного будиночка. ▲
4.  Розробіть програмний код, за допомогою якого у рівнобедрений трикутник вписане коло. Зразок рисунка зображено на рис. 6.25. Порів-

найте розроблений код із кодом, поданим на рис. 6.26. Визначте переваги та недоліки кожного з них. ★

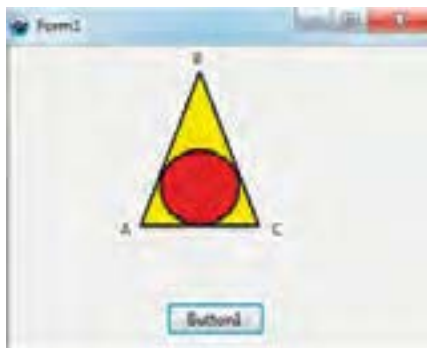



Рис. 6.25. Рівнобедрений трикутник із вписаним колом

```

procedure TForm1.Button1Click(Sender: TObject);
begin
  with Canvas do begin
    Pen.Width:=2;
    Brush.Color:=clYellow;
    Polygon ((Point (100,140), Point (145,20), Point (190,140)));
    Brush.Color:=clRed;
    Ellipse (115, 81, 176,140);
    Brush.Color:=clWhite;
    TextOut (85,135,'A');
    TextOut (140,2,'B');
    TextOut (200, 135,'C');
  end;
end;
end.

```

Рис. 6.26. Код побудови трикутника зі вписаним колом

5.  Розробіть код для креслення конуса. ✦


6.  Розробіть код для креслення зображення, поданого на рис. 6.27. Використайте оператор циклу. Розроблену програму порівняйте з програмою, наведеною на рис. 6.28. Укажіть переваги й недоліки кожної з них. ★



Рис. 6.27. Завдання для розроблення коду з оператором циклу

```

procedure TForm1.Button1Click(Sender: TObject);
var i, k, x, y:integer;
begin
  x:=30; y:=100; k:=1;
  with Canvas do begin
    Brush.Color:=clGreen;
    Rectangle (20,40, 315,170);
    Brush.Color:=clYellow;
    Pen.Width:=2;
    Pen.Color:=clRed;
    for i:=1 to 7 do begin
      Ellipse (x,y, x+20, y+(20*k));
      x:=x+20; k:=k*(-1);
    end;
    TextOut (140,175,'Вільярд');
  end; end;end.

```

Рис. 6.28. Код з оператором циклу для побудови зображення

6.8.5. Відображення рисунків із зовнішніх файлів

Середовище Lazarus забезпечує виведення на форму готових графічних зображень із зовнішніх файлів. Базовим класом для цього є описаний у модулі Graphics абстрактний клас TGraphic та його підкласи TIcon, TBitmap, TMetafile, TImage. Виведення графічних зображень здійснюється зазвичай на компонент TImage, який міститься на вкладці Additional. Компонент TImage підтримує виведення ілюстрацій у багатьох форматах, у тому числі в форматах bmp, ico, jpeg. Основні властивості цього компонента подані в табл. 6.9.

Таблиця 6.9

Основні властивості компонента TImage

Властивість	Опис
1	2
Picture	Ілюстрація, що відображується в полі компонента
Width, Height	Розмір компонента. Якщо розмір компонента менший ніж розмір ілюстрації і значення властивостей AutoSize, Stretch і Proportional дорівнює false, то зображується лише частина ілюстрації
Proportional	Виконується автоматичне масштабування ілюстрації без викривлення. Щоб масштабування було виконане, значення властивості має дорівнювати true

1	2
Stretch	Автоматичне масштабування ілюстрації відповідно до реальних розмірів компонента. Якщо розмір компонента не пропорціональний розміру ілюстрації, то ілюстрація буде викривлена. Ця властивість не впливає на рисунки типу iso
AutoSize	Автоматична зміна розміру компонента відповідно до реальних розмірів ілюстрації
Center	Визначає розміщення ілюстрації в полі компонента по горизонталі, якщо ширина ілюстрації менша за ширину поля компонента. Якщо значення властивості дорівнює false, то ілюстрація примикає до правої межі компонента, а якщо true, то розміщується по центру
Visible	Компонент й ілюстрація розташовуються на поверхні форми
Canvas	Поверхня, на яку можна вивести ілюстрацію

Графічні зображення із зовнішніх файлів можна присвоїти компоненту Image вручну, під час розроблення інтерфейсу, а також програмно, в процесі виконання проекту.

А. Відображення зображень у процесі розроблення інтерфейсу

Спосіб полягає у присвоєнні властивості Picture компонента TImage імені файла, в якому зберігається зображення, яке потрібно розмістити на компоненті. Відобразимо, наприклад, файл P6130628, у якому зберігається фото Михайлівського собору. Дотримуватимемося такої послідовності дій:

1. Щоб не вказувати повний шлях до файла P6130628, скопіюємо його в папку, в якій створюватимемо проект. Обираємо, наприклад, папку LAZAR диска F:.

2. Уже відомим способом створимо проект.

3. Розміщуємо на формі компонент TImage (міститься на вкладці Additional).

4. Активуємо компонент TImage і в інспекторі об'єктів надаємо значення true властивості Proportional. Клацаємо рядок Picture і натискаємо в цьому рядку кнопку з трьома крапками.

5. Відкриється вікно **Діалог завантаження зображення – Image1. Picture**. У цьому вікні натискаємо кнопку **Завантажити**. У вікні з'явиться файл P6130628, який ми скопіювали в папку LAZAR (рис. 6.29).

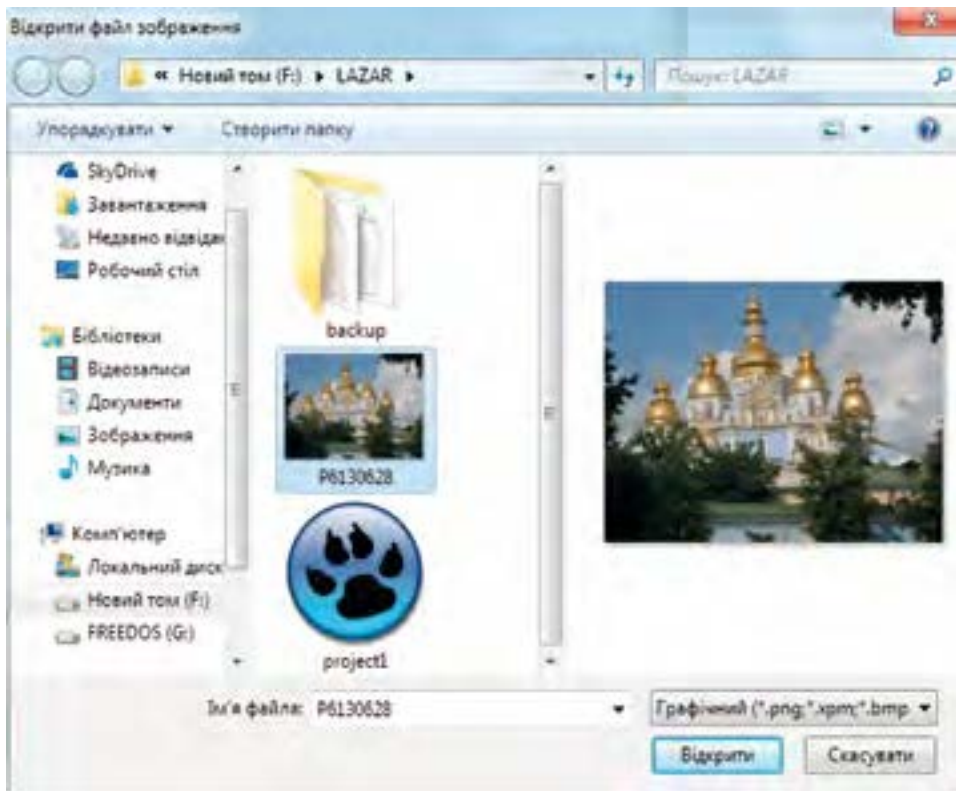


Рис. 6.29. Вікно для відкриття файла зображення

Ім'я цього файла переносимо в рядок **Ім'я файла**.

6. Натискаємо кнопку **Відкрити** й у вікні, що відкриється, натискаємо кнопку **Гаразд**. У результаті зміст файла відобразиться на об'єкті Image (рис. 6.30).

Б. Відображення зображень у процесі виконання коду

Для відображення на компоненті TImage графічного зображення за допомогою цього способу використовується метод LoadFromFile (), що належить об'єкту Picture.

Розглянемо приклад відображення на компоненті TImage1 змісту файла 12.23.tif.

Дотримуватимемося такої послідовності дій:

1. Скопіюємо файл 12.23.tif у папку LAZAR, у якій створюватимемо проект.

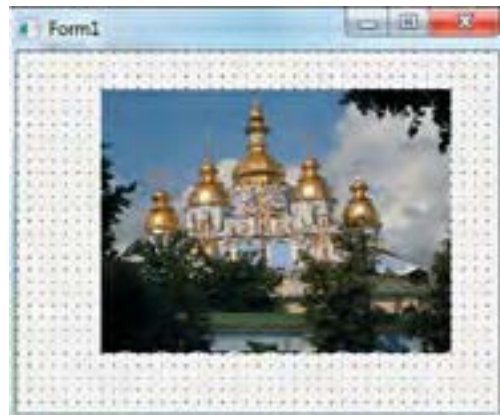


Рис. 6.30. Фото на об'єкті Image

2. Створимо проект.
3. Розмістимо на формі компоненти TImage1 і TButton1. Властивості Caption об'єкта Button1 надамо значення Показати. Активуємо об'єкт Image1 і в Інспекторі об'єктів встановлюємо значення true властивості Proportional.
4. Двічі клацаємо кнопкою миші на об'єкті Button1.
5. У текстовому редакторі набираємо код, поданий на рис. 6.31.

```

procedure TForm1.Button1Click(Sender: TObject);
begin
    Image1.Picture.LoadFromFile('12_23.tif');
end;

```

```

end.

```

Рис. 6.31. Код для відображення файлу 12.23.tif

6. Виконаємо компіляцію і запустимо програму на виконання. Відкриється вікно форми, в якому натискаємо кнопку Показати. У результаті відкриється зміст файлу 12.23.tif, зображений на рис. 6.32.



Рис. 6.32. Рисунок з файлу 12.23.tif, розташований на об'єкті Image





Графічні можливості середовища Lazarus забезпечують також побудову графіків функцій і графічне відображення даних, що сприяє наочності їх подання.

Для цього застосовуються різноманітні об'єкти й методи середовища, зокрема, сектори, прямокутники, лінії. Приклад графічного відображення даних розглядається у практичній роботі № 13.

Перевіряємо себе

1. На який компонент форми здійснюється виведення графічних зображень із зовнішніх файлів? ▲
2. Яке призначення має властивість AutoSize? ▲
3. Поясніть призначення властивості Proportional. ▲
4. Поясніть сутність відображення графічних зображень у процесі розроблення інтерфейсу. ✦
5. Для чого файл із графічним зображенням копіюється в папку проекту? ✦
6. Який метод застосовується для відображення графічного зображення? ✦
7. Поясніть порядок розміщення графічного зображення в процесі створення графічного інтерфейсу. ★
8. Поясніть порядок розміщення графічного зображення у процесі виконання програмного коду. ★

Виконуємо

1.  Створіть у графічному редакторі Paint який-небудь малюнок і збережіть його у файлі. Відобразіть зміст цього файлу на компоненті Image в процесі створення інтерфейсу. ▲
2.  Створіть у графічному редакторі Paint рисунок і збережіть його у файлі. Створіть код для розміщення цього рисунка на компоненті Image. ✦
3.  Знайдіть у Інтернеті фото Андріївського узвозу в Києві й збережіть його у файлі. Відобразіть знайдене фото на компоненті Image у процесі створення інтерфейсу. ▲
4.  Знайдіть у Інтернеті фото Києво-Печерської лаври й збережіть його у файлі. Створіть код відображення змісту цього файлу на компоненті Image. Додайте на форму відомості про фото Києво-Печерської лаври у вигляді окремого текстового об'єкта. ✦

Практична робота № 13

Тема:	Описання та виконання алгоритмів з графічним відображенням даних
Мета:	Набути практичних навичок налагодження програм із віконним інтерфейсом і графічним відображенням даних

Завдання

1. Створити програму, яка виводить на полотно зображення з чотирьох файлів у вигляді невеликих копій; під кожною з них розташована кнопка, у разі натиснення на яку з'являється зображення в збільшеному розмірі.



СЛОВНИЧОК

Дані логічного типу – дані, що приймають лише два значення (true і false).
Змінна в програмуванні – ділянка пам'яті з присвоєним їй ім'ям (ідентифікатором), у якій зберігаються дані певного типу.

Константа – це величина, значення якої не змінюється у процесі виконання програми.

Модуль – окрема спеціальна програма, яка розширює можливості мови програмування.

Операнди виразу – константи, змінні й функції.

Рядковий тип даних – послідовність символів довжиною до 255.

Символьний тип даних – окремі символи, що записуються в одинарних лапках.

Стандартні функції – найчастіше вживані функції, для яких розроблені спеціальні програми.

Структуровані типи даних – дані, що об'єднуються в структури (масиви, записи, множини та інші).

РОЗДІЛ 7. ТЕХНОЛОГІЇ ОПРАЦЮВАННЯ ЧИСЛОВИХ ДАНИХ У СЕРЕДОВИЩІ ТАБЛИЧНОГО ПРОЦЕСОРА



Електронні таблиці (ЕТ) створюються у спеціальному програмному середовищі, яке називають електронною таблицею або табличним процесором. У вільно поширюваному офісному пакеті Libre Office – це електронні таблиці Libre Office Calc (рис. 7.1), а в офісному пакеті Microsoft Office – Microsoft Office Excel (рис. 7.2). Електронний документ, який створюється табличним процесором, називають Книгою. Книга складається з аркушів, вміст одного з яких виводиться на екран. Аркуш відповідно складається з комірок (клітинок).



Обчислювальні алгоритми в середовищі табличного процесора; абсолютні, відносні й мішані посилання на комірки та діапазони комірок; формули з використанням посилань на комірки та діапазони; умовне форматування; автофільтр і розширений фільтр; проміжні підсумки; типи діаграм відповідно до мети їх застосування; математичні, статистичні, логічні функції табличного процесора; упорядкування даних за значеннями одного чи кількох полів; шаблони електронних таблиць; підготовка таблиці до друкування.

Увага! У цьому розділі деякі команди та назви функцій подані англійською мовою, для того щоб при використанні нелокалізованих (або частково локалізованих) версій табличних процесорів і їх програмних надбудов не виникало труднощів.

7.1. Обчислення в середовищі табличного процесора



Властивості об'єктів відображаються у вигляді даних, зв'язки між властивостями об'єктів описуються у вигляді математичних моделей. У електронних таблицях ці моделі подаються через запис у комірки формул.

Дії (арифметичні, порівняння або впорядкування – для числових даних, тільки порівняння або впорядкування – для текстових) виконуються над вмістом комірки або вмістом діапазону комірок, а звернення до даних виконуються за адресою комірки або діапазону комірок, іменем, тобто за посиланнями на дані.

Для позначення арифметичних операцій застосовуються такі символи:
+ – додавання; – – віднімання; * – множення; / – ділення.

Формула може містити посилання на комірки, які розташовані на іншому робочому аркуші чи навіть у таблиці іншого файла.

Один раз уведена формула може бути в будь-який час модифікована. Вбудований Менеджер формул допомагає користувачеві знайти помилку чи неправильне посилання у великій таблиці.

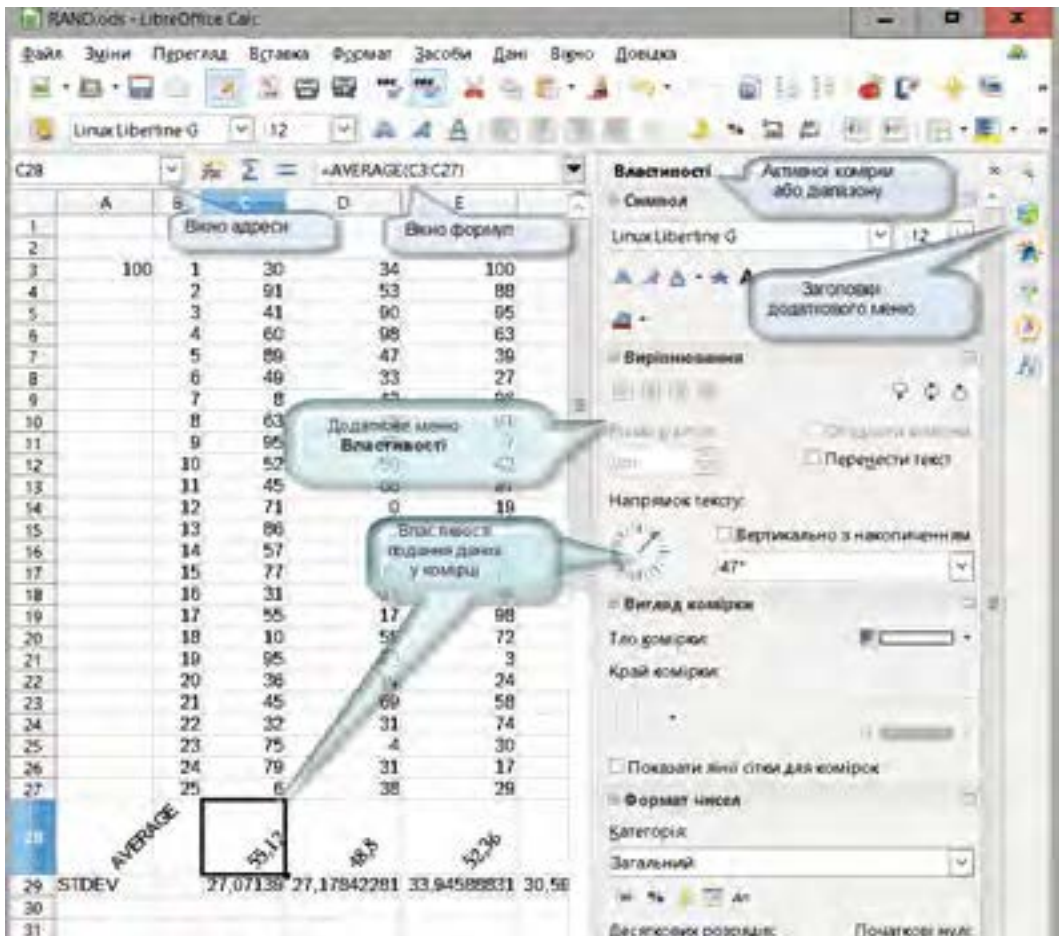


Рис. 7.1. Будова вікна Libre Office Calc

Табличні процесори забезпечують роботу зі складними формулами, які містять кілька операцій.

Для зручності можна включити режим, у якому ЕТ відобразатимуть не результат обчислення формули, а власне формулу.

Нехай у комірці **A1** таблиці міститься число 100, а в комірці **B1** – число 20. Щоб розділити перше число на друге та результат помістити в комірку **B1**, у комірку **B1** слід увести формулу $=A1/B1$ і натиснути **Enter**.

Уведення формули можна виконати інакше, не вводячи посилань на комірки, а вказуючи на них мишею (при роботі зі сенсорним екраном – виконуючи одинарний дотик):

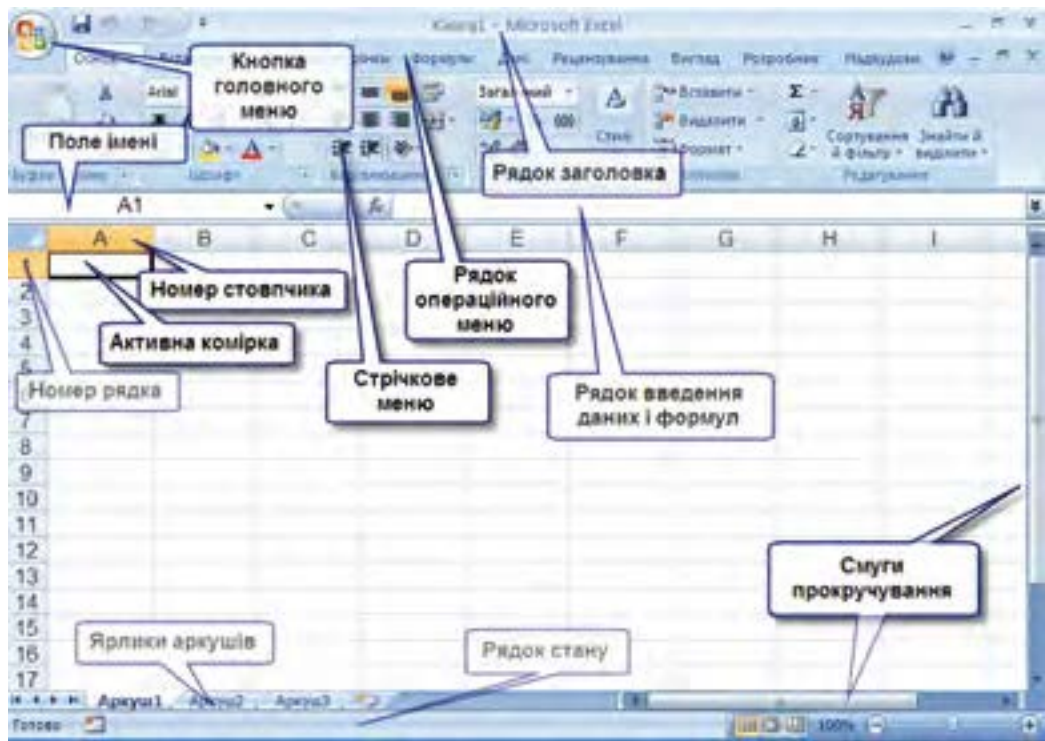


Рис. 7.2. Будова вікна Microsoft Excel 2007

до комірки, в якій має бути розміщено результат обчислень за формулою, ввести знак рівності (=) ⇒ клацнути лівою кнопкою миші комірку A1 ⇒ у комірці з'явиться адреса першої комірки формули ⇒ увести знак операції ділення "/" ⇒ клацнути на комірці B1 ⇒ у комірці з'явиться адреса другої комірки ⇒ натиснути **Enter**.

У всіх табличних процесорах можливе подання адрес комірок у форматі з нумеруванням як рядків (англ. row), так і стовпців (англ. column) (формули матимуть вигляд, наприклад, =RC[-2]-R[-1]C[-1]). Перехід до такого подання показано на рис. 7.3.

Застосування складних формул продемонструємо на прикладі (рис. 7.4). Нехай потрібно обчислити вартість виконання певних робіт.

У стовпці D таблиці зазначено час (у годинах), витрачений на виконання роботи, у стовпці E – вартість однієї години роботи, а в стовпці F – проміжну суму, яку треба сплатити за роботу. У комірці F6 потрібно показати загальну вартість усіх робіт. Для цього в неї слід записати таку формулу: = F3+F4+F5.

Для обчислення податку на додану вартість отриману суму слід помножити на 0,15 і результат помістити в комірку: F7: =F6*0,15.

Для обчислення кінцевої суми, яка підлягає оплаті (наприклад, у комірці F8), треба спочатку отримати проміжні суми, а потім результат помножити на 1,15. Формула матиме такий вигляд: =(F3+F4+F5)*1,15.

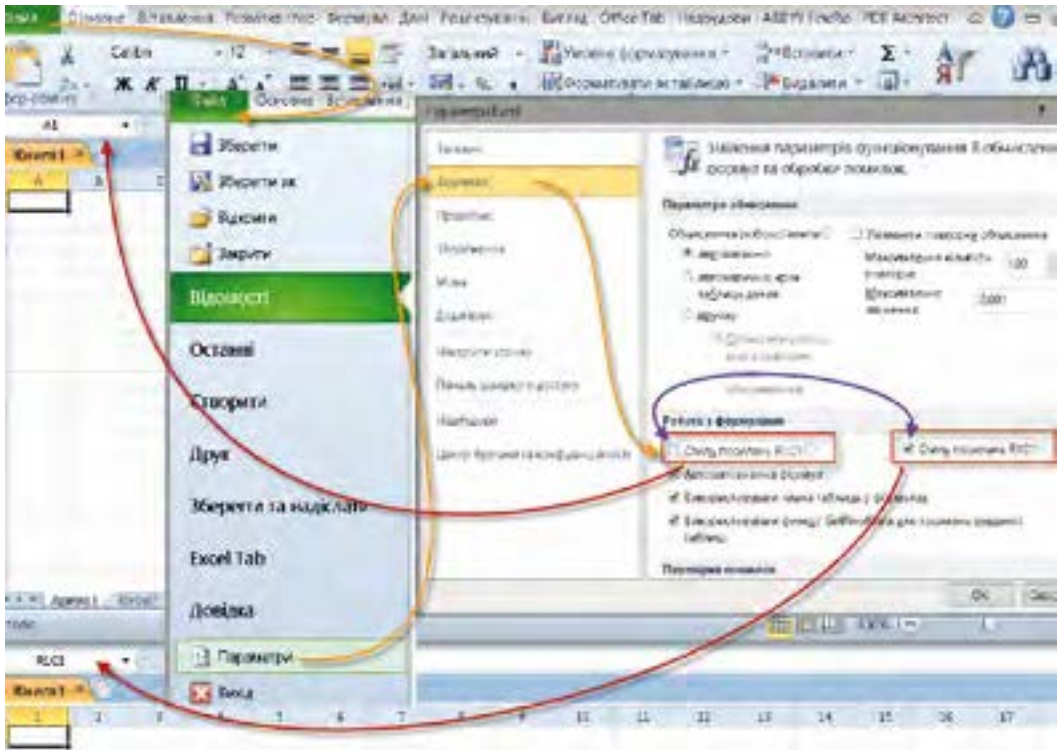


Рис. 7.3. Перемикання форми подання адрес комірок у Microsoft Excel 2010

Звичайно, можна було б додати вміст комірок **F6** і **F7**. Для додавання кількох чисел можна також використати функцію суми **SUM()**, тоді формула матиме такий вигляд: **= SUM(F3:F5)*1,15**.

Для редагування вмісту комірки (комірок) їх потрібно спочатку активізувати. Далі слід включити режим редагування, натиснувши на клавіатурі клавішу **F2**, або подвійним клацанням лівої кнопки миші. Редагування можна виконувати й у полі **Рядок введення формул і даних** (розташованому під панелями інструментів), у якому подається тільки формула для редагування, а не результат її обчислення.

Для звернення до значення, що міститься в комірці, розташованій на іншому робочому аркуші, потрібно вказати ім'я цього аркуша разом з адресою відповідної комірки. Наприклад, для звернення до комірки **D6** на робочому аркуші **Sheet3** потрібно ввести формулу **=Sheet3!D6**.

✓ *Якщо в назві аркуша є пробіли, то назва подається в лапках. Адреси комірок мають бути зазначені латинськими літерами.*

Інформаційне зв'язування двох комірок можна спростити, якщо скопіювати значення вихідної комірки в буфер (за допомогою комбінації клавіш **Ctrl+C**) й активізувати комірку, в якій має з'явитися результат.

	C	D	E	F	G	H
1						
2		Назва роботи	час, год	вартість	сума	
3		Встановлення і налагодження ОС	4	35.00 грн	140.00 грн	
4		Підключення до локальної мережі	0.5	20.00 грн	10.00 грн	
5		Встановлення офісного програмного забезпечення	3	25.00 грн	75.00 грн	
6		СУМА			225.00 грн	
7		Податок на додану вартість (ПДВ)			45.00 грн	
8		Сума з ПДВ			270.00 грн	

Рис. 7.4. Застосування складних формул для обчислення загальної вартості виконаних робіт

Потім потрібно викликати з меню **Правка (Edit)** директиву **Спеціальна вставка (Paste Special)**, далі в діалоговому вікні натиснути на кнопку **Вставити зв'язок (Paste Link)** (рис. 7.5).

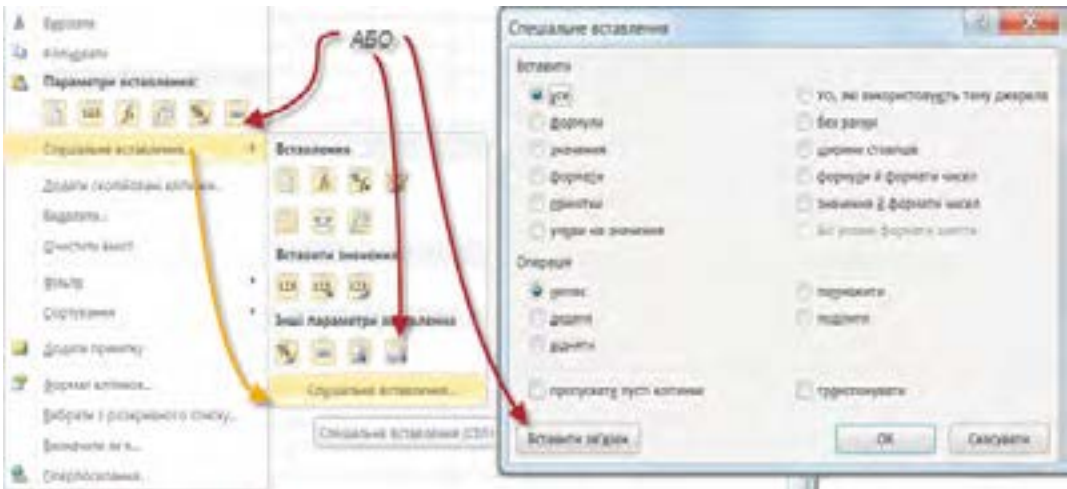


Рис. 7.5. Спеціальне вставлення (MS Office 2010)

✓ У програмах Excel і Libre Office Calc можна ввести посилання і на комірку, розташовану в іншій таблиці (іншому файлі).

Після введення посилання значення, які містяться в комірках, автоматично поновлюватимуться при кожному завантаженні.

✓ Для обчислення суми вмісту групи комірок не потрібно називати (перераховувати) в формулі окремі комірки або вводити діапазон у форматі F3:F5. Досить виокремити всю групу та надати їй ім'я.

Надалі це ім'я можна буде використовувати в формулах. Для надання імені групі комірок виконуємо таке (для MS Office 2003):

меню **Вставка (Insert)** ⇒ відкрити підменю **Ім'я (Name)** ⇒ викликати директиву **Надати** ⇒ у полі введення діалогового вікна **Надання імені (Define Name)** вказати ім'я цієї групи (у це вікно виведено список уже наданих групових імен, які розташовані на аркуші) ⇒ натиснути **Додати (Add)** ⇒ **ОК**.

Процедура надання імен групам комірок у MS Office 2007–2010 й у LibreOfficeCalc простіша – досить виділити діапазон і у вікні адреси комірки розкрити список імен, увести в вікно адреси ім'я групи (рис. 7.6).

✓ У Libre Office Calc ім'я може бути набране тільки латиницею. Ім'я групи має починатися з літери й містити не більше 255 символів.

Не допускається використовувати пробіли. Ім'я групи не має збігатися з адресами комірок (A1, B6 тощо).

Якщо таблиця містить заголовки рядків і стовпців, то їх також можна використовувати як імена цих діапазонів. Для цього потрібно:

виокремити сусідні рядки (стовпці), включаючи перші комірки, де розташовані імена меню ⇒ меню **Вставка (Insert)** ⇒ відкрити підменю **Ім'я (Name)** ⇒ викликати директиву **Створити (Create)** ⇒ у діалоговому вікні, що відкрилося, потрібно вказати місцезоташування імен (у першій чи останній комірці рядка чи стовпця) ⇒ **ОК**.

Для редагування функцій:

двічі клацнути лівою кнопкою миші комірку, в якій міститься функція ⇒ в комірці, яка містить результат обчислення, або в **Рядку формул** ⇒ відредагувати ⇒ **Enter**

Звернення до вмісту комірок (посилання) можуть бути **відносними**, **абсолютними** та **мішаними**.

Якщо адресу комірки записати в формулу, наприклад, так: A4, або діапазону комірок так: A8:B20, то при копіюванні або перенесенні формули в іншу комірку цю адресу буде змінено – номери стовпців збільшено або зменшено на стільки, на скільки стовпців перенесено формулу, і так само буде змінено й номери рядків.

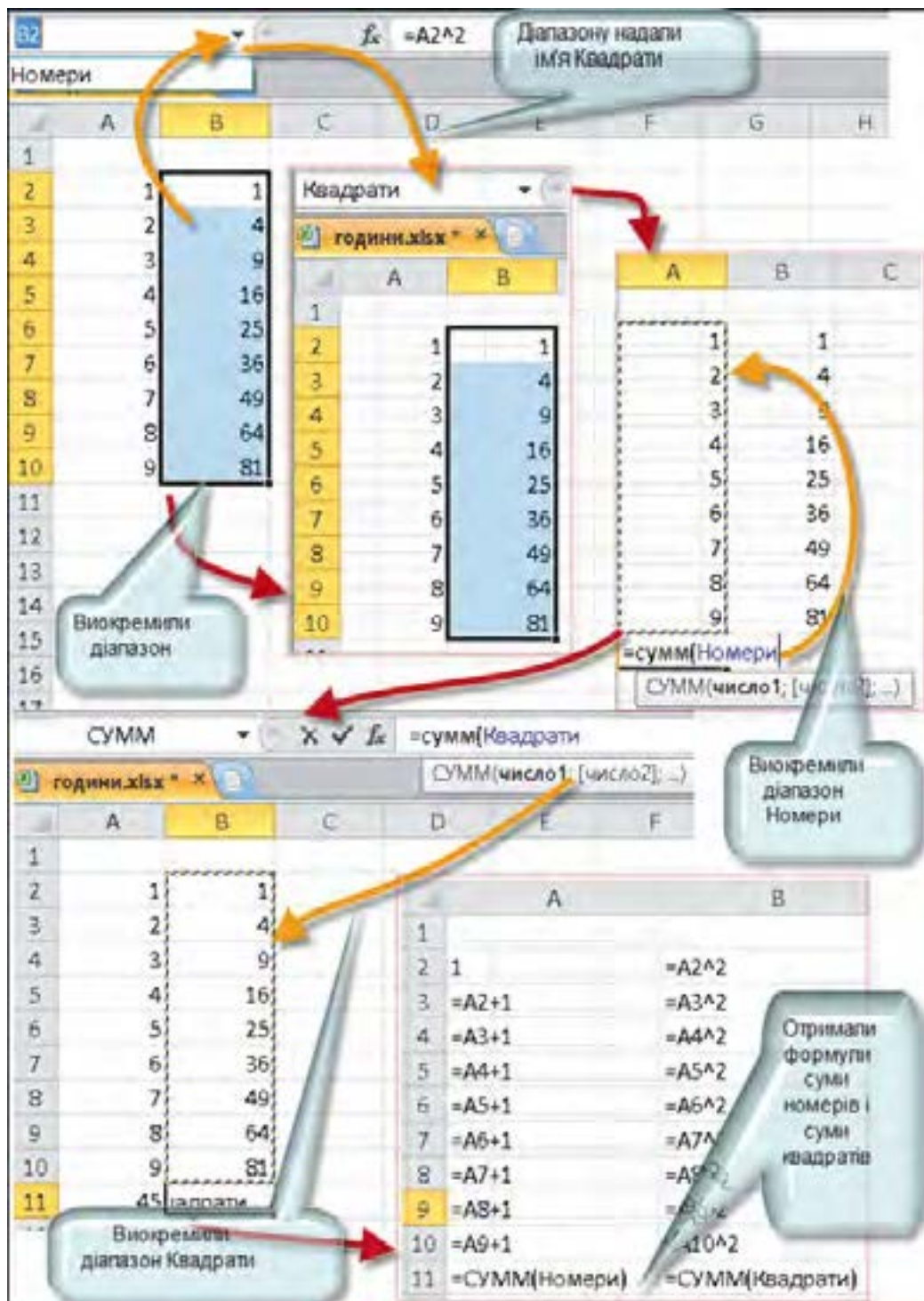


Рис. 7.6. Процедури надання імені групі комірок і використання імен у формулах (MS Office 2010)



Адреси комірок (і діапазонів комірок), які змінюються при копіюванні (перенесенні) формул, називаються **відносними**.

Якщо потрібно, щоб при копіюванні формул посилання не змінювалися, їх записують так: \$A\$4, \$A\$8:\$B\$20.



Адреси комірок (і діапазонів комірок), які не змінюються при копіюванні формул, називаються **абсолютними**.



Абсолютним є й адресування з використанням імен.

Інколи потрібно, щоб при копіюванні формули змінювалися тільки значення стовпця або рядка. У такому випадку символ “\$” розташовують лише перед номером рядка або літерою стовпця, наприклад: A\$29 або \$A29. Такі записи адрес є мішаними.






Адреси комірок (і діапазонів комірок), які лише частково змінюються при копіюванні формул, називаються **змішаними**.

Інколи потрібно в комірці певного діапазону (наприклад: A5:A25) ввести значення, кожне наступне з яких більше від попереднього на певне значення (“крок”, або різниця).









Слід у комірку A5 ввести перше значення, а в комірку A6 формулу =A5+a, де a – значення різниці. Після цього досить скопіювати вміст комірки A6 у комірки A7:A25, і вони заповняться необхідною послідовністю чисел.

Виконуємо


-  Проаналізуйте рис. 7.4 і його опис у тексті. Створіть ЕТ. Які посилання доцільно використовувати у формулах, записаних у комірках F3–F5? Чому? ▲
-  Модифікуйте створену за попереднім завданням ЕТ таким чином, щоб значення податку на додану вартість можна було оперативно змінювати, вводячи з клавіатури. ★
-  Модифікуйте створену за попередніми завданнями ЕТ таким чином, щоб значення податку на додану вартість можна було оперативно змінювати й розраховувати для кожного виду роботи окремо.

Підказка: додати стовпчик (комірки G2–G5) із назвою “ПДВ робіт”. Для зручності використайте абсолютне посилання на комірку зі значенням коефіцієнта для ПДВ. ★

-   Проаналізуйте рис. 7.5 і його опис у тексті. Як називається група комірок, що відповідає діапазону A2–A10? Яким чином ці комірки заповнено значеннями? Як можна назвати числа, розміщені в комірках A11 та B11? ★

5.  Створіть ЕТ, подану на рис. 7.6. Модифікуйте таблицю таким чином: збільшіть кількість рядків до двадцяти; змініть формули у стовпчику А так, щоб можна було отримувати послідовності чисел, які відрізняються не на 1, а на довільну величину (крок). ★
6.  Знайдіть у Інтернеті пояснення щодо особливостей форматів файлів *.ods і *.csv. Занотуйте основні відмінності між форматами файлів. Збережіть створену електронну таблицю в форматах, що відрізняються від того, в якому її подано. ✦
7.   Виконайте, якщо можливо, пересилання телефонної книги з мобільного телефону на комп'ютер (у файл *.csv) і відкрийте цей файл текстовим редактором Блокнот, текстовим процесором і табличним процесором. Зробіть висновки. ★

Перевіряємо себе

1. Назвіть основні формати збереження ЕТ табличного процесора Microsoft Excel 2007. ▲
2. Для чого призначений формат *.xml? Знайдіть у Довідці Excel 2010 необхідні відомості. ▲
3. Для чого призначені формати *.xlt та *.xls? ✦
4. До яких програмних засобів можливий експорт електронних документів, створених у Excel 2010? ✦
5. Які відмінності між **Шаблонами** текстового процесора, презентаційної системи і табличного процесора? Чим вони зумовлені? ✦
6.  Знайдіть у Довідці Excel 2010 відомості щодо форматування та властивостей ЕТ, створених у застосунку Excel 2010, які не зберігаються у файлах інших форматів. ✦
7. У яких випадках доцільно використовувати різні види посилань у формулах? Наведіть приклади й перевірте їх у таблиці. ✦
8. Яким чином можна надати ім'я несуміжним коміркам? Знайдіть у Довідці Excel 2010 необхідні відомості. ✦
9. Проаналізуйте рис. 7.5 і його опис у тексті. Коли доцільно використовувати кожен із видів вставлення?

7.2. Призначення та використання основних математичних функцій табличного процесора



Одиницею зберігання даних в ЕТ є вміст комірки. Для того щоб ЕТ виконала опрацювання формули, запис формули, який вводиться в комірку таблиці, має починатися зі знака рівності (=).

Оскільки деякі формули та їх комбінації трапляються дуже часто, то табличні процесори містять понад 200 заздалегідь запрограмованих формул, які називаються **функціями** (табл. 7.1).

✓ **Функції** – наперед визначені формули, за якими ЕТ виконують обчислення в певній послідовності для величин, які називаються **операндами**, або **аргументами**.

Аргумент функції може займати одну комірку чи розміщуватися в групі комірок.

Є **аргументи** різних типів: число, текст, логічне значення (TRUE та FALSE), масиви, значення помилки (наприклад, #N/A) або посилання на комірку. У кожному окремому випадку потрібно використовувати відповідний тип аргументу. Константи, формули або функції також використовуються як аргументи.

Із функцій можна формувати вирази, де операндом однієї з функцій може бути значення, яке повертає інша (такий запис у програмуванні називають **вкладенням**).

Для введення функцій використовується команда **Вставка функції**, яка розташована в **Рядку формул (рядку введення даних)**. Натисненням кнопки **Вставка функції** відкривається вікно **Вставка функції**, яке містить упорядкований за алфавітом повний список функцій. У цьому списку можна легко знайти потрібну функцію. При наведенні курсора на ім'я функції внизу списку програма надає її короткий опис.

Якщо функцію не знайдено, її пошук необхідно виконувати за категоріями, клацнувши в рядку **Категорії** або використавши команду **Знайти**. Багато функцій мають досить незначні відмінності, тому при пошуку слід увести короткий опис дії, яку потрібно виконати, та натиснути на кнопку **Знайти** (рис. 7.7).

При введенні функції в формулу діалогове вікно **Вставка функції** відображає ім'я функції, всі її аргументи, опис функції та кожного аргументу, поточний результат функції та всієї формули.

Використання вікна **Вставка функції** полегшує введення функцій під час створення формул, які містять функції, та допомагає вставити правильну формулу й потрібні аргументи (рис. 7.8).

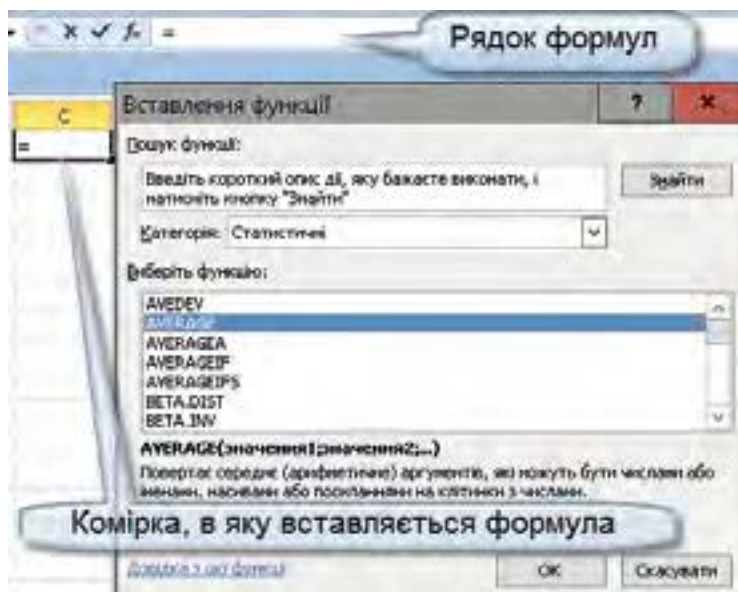


Рис. 7.7. Вікно введення функції

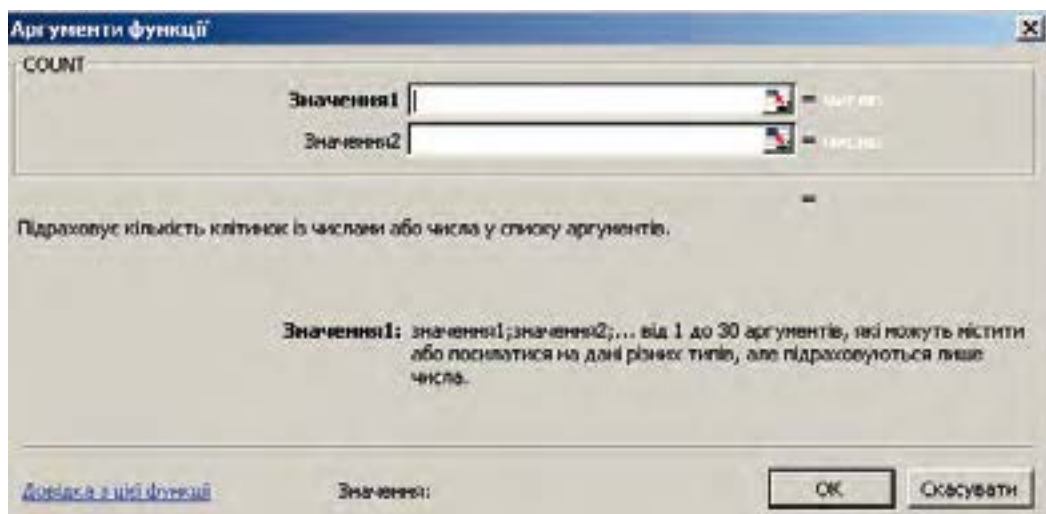


Рис. 7.8. Вікно Майстра Аргументи функції (Office 2010)

✓ Для вставлення функції слід:

клацнути ім'я функції (під полем **Виберіть функцію** буде показано синтаксис цієї функції та її короткий опис) ⇒ двічі клацнути ім'я функції ⇒ функцію та її аргументи буде відображено в вікні **Майстра Аргументи функції** ⇒ вказати правильні аргументи, користуючись коротким описом та поясненнями, які наводяться під полями аргументів ⇒ **ОК**.

Майстер розрізняє аргументи, які обов'язково мають бути враховані, та необов'язкові (опційні) аргументи. Щоб задати аргумент функції, потрібно або ввести його адресу з клавіатури, або в таблиці виокремити діапазон, у якому він розташований. Тоді адреса аргументу функції з'явиться у відповідному полі діалогового вікна **Майстра**.

Після цього в зазначеній комірці таблиці з'явиться результат обчислень, але комірка залишиться виділеною. У **Рядку формул** можна побачити обрану функцію разом з відповідними аргументами.

✓ Для редагування функцій:

двічі клацнути лівою кнопкою миші на комірці, в якій записана функція ⇒ виконати редагування в комірці, яка містить результат обчислення, або в **Рядку формул** ⇒ **Enter**

Таблиця 7.1

Деякі математичні функції електронних таблиць *Microsoft Excel*

№ з/п	Назва функції	Опис
1	2	3
1	ABS()	Повертає абсолютне значення числа
2	CEILING() ОКРУГЛІТ()	Заокруглює число до найближчого цілого і кратного вказаному значенню
3	COS()	Повертає косинус числа
4	DEGREES() ГРАДУСИ()	Перетворює радіани на градуси
5	EXP()	Повертає число e, піднесене до вказаного степеня
6	FACT()	Повертає факторіал числа
7	FLOOR() ОКРУГЛІВНИЗ()	Заокруглює число до меншого за модулем, у напрямку нуля
8	GCD() НОД()	Повертає найбільший спільний дільник
9	INT()	Заокруглює число до найближчого меншого цілого
10	LCM() НОК ()	Повертає найменше спільне кратне
11	МОД() ОСТАТ()	Повертає остачу від ділення
12	PI()	Повертає число π
13	POWER() СТЕПЕНЬ()	Повертає число, піднесене до степеня
14	RADIANS() РАДИАНИ()	Перетворює градуси на радіани

Закінчення табл. 7.1

1	2	3
15	RAND() СЛЧИС()	Повертає випадкове число в інтервалі від 0 до 1
16	RANDBETWEEN() СЛУЧМЕЖДУ()	Повертає випадкове число в зазначеному інтервалі
17	ROUND() ОКРУГЛИ()	Заокруглює число до вказаної кількості знаків
18	ROUNDDOWN() ОКРУГЛВНИЗ()	Заокруглює число до меншого, у напрямку до нуля
19	CEILING() ОКРУГЛВВЕРХ()	Заокруглює число вгору, у напрямку від нуля
20	SIGN() ЗНАК()	Повертає знак числа (1, якщо число більше нуля, 0, якщо число 0, -1, якщо число менше нуля)
21	SIN()	Повертає синус кута (значення кута подається в радіанах)
22	SQRT() КОРЕНЬ()	Повертає невід'ємне значення квадратного кореня
23	SUM() СУММ()	Підсумовує аргументи
24	SUMIF() СУММЕСЛИ()	Підсумовує вміст комірок, визначених за вказаною умовою
25	SUMIFS()	Підсумовує вміст комірок у діапазоні, який відповідає кільком умовам
26	SUMPRODUCT() СУММПРОИЗВ()	Повертає суму добутків відповідних елементів масиву
27	SUMSQ() СУММКВ()	Повертає суму квадратів аргументів
28	TAN()	Повертає тангенс числа
29	TRUNC() ОТБР()	Видаляє дробову частину числа

Більш докладні відомості про застосування математичних функцій та їх синтаксис можна отримати з Довідки Microsoft Excel.

Наприклад, правила заокруглення, які зазвичай використовують у процесі обчислень, в ET Microsoft Excel 2007 доповнені умовами, які можуть накладатися на результат заокруглення, і створено відповідні формули.

Досить часто доводиться розв'язувати задачі, подібні до такої.

Приклад. 105 л пального потрібно розмістити в каністрах, ємність кожної з яких 20 л. Скільки потрібно каністр?

Відповідь. Для цього знадобиться 6 каністр, одна з яких міститиме тільки 5 л пального.

Тому, якщо деякий вантаж можна розмістити порціями, не більшими за певну (одне вантажомісце, у задачі (прикладі) це каністра для пального), то навіть для порції вантажу, маса якої значно менша від визначеної для вантажомісця, треба надати окреме вантажомісце, тобто для визначення кількості вантажомісць потрібно виконати “заокруглення вгору” (ROUNDUP).

Застосування деяких формул до чисел показано на рис. 7.9. До чисел першого рядка застосовано формули, назви яких подані в стовпці F. Формули застосовані в рядках 2...7.

	A	B	C	D	E	F
1	394,588298	398,753033	399,006699	401,195038	402,905759	
2	394	398	400	402	402	MROUND
3	394,59	398,76	399,01	401,2	402,91	ROUNDUP
4	396	400	400	402	404	EVEN
5	395	399	401	403	403	ODD
6	394,58	398,75	399	401,19	402,9	ROUNDDOWN
7	394,59	398,75	399,01	401,2	402,91	ROUND

Рис. 7.9. Приклад застосування формул, призначених для заокруглення чисел

Аргументи функцій, так само як і простих формул, можуть розташовуватися на різних аркушах книги.

Наприклад, кілька постачальників постачають однаковий товар, але для кожного з них облік постачання ведеться на окремому аркуші, на якому також зберігається ціна товару від постачальника. Потрібно знайти загальну кількість товару і його вартість, якщо облік постачання товару в натуральному вимірі ведеться на різних аркушах: Пост_1, комірки B2:E9; Пост_2, комірки B2:E9; Пост_3, комірки B4:E12, а ціни зберігаються в комірках Пост_1!A10, Пост_2!A10, Пост_3!A10.

Вираз: =SUM(SUM(Пост_1!B2:E9)*Пост_1!A10;SUM(Пост_2!B2:E9)*Пост_2!A10;SUM(Пост_3!B4:E12)*Пост_3!A10) поверне значення загальної вартості товару від трьох постачальників.

✓ Особливим випадком аргументу є масив даних.


⚠ **Масив** – сукупність однотипних величин, кожен елемент якої нумерований.

✓ Масив є структурою даних. Звернення до окремого значення (елемента масиву) здійснюється за його номером, в ET його значення розміщують у блоці суміжних комірок.

Формули, в яких аргументами є масиви, можуть описувати дії впорядкування елементів, арифметичні дії над елементами кількох масивів, пошуку елементів за певними ознаками.


Можна також використовувати формули, результатом обчислення яких є не одне значення, а масив (сукупність значень).

Можна написати й власні формули, що застосовуються до діапазонів комірок, результатом обчислення яких буде діапазон комірок. Наприклад: **=F4:F9–G4:G9**.













 Для введення подібних формул (формул масиву):

виділіть діапазон комірок, що мають містити результати обчислення формули масиву (розмір виділеного діапазону має відповідати кількості значень, що створюються формулою) \Rightarrow натисніть клавішу **F2** \Rightarrow уведіть потрібну формулу, вказуючи посилання на діапазони комірок, що мають використовуватися в обчисленнях \Rightarrow завершіть введення формули натисканням сполучення клавіш **Ctrl+Shift+Enter** (а не натисканням кнопки **OK** у підменю вибору даних!).











Програма Excel помістить формулу у фігурні дужки, що є ознакою формули масиву. У комірках виділеного діапазону будуть представлені результати обчислення формули.

 Табличний процесор завжди інтерпретує масив як єдине ціле, тому змінити значення в окремих комірках масиву неможливо. Можна лише вказати для окремих комірок різні параметри форматування (у тому числі використати умовне форматування). Комірки не можуть бути переміщені з масиву, а нові комірки – додані в масив.


Перевіряємо себе

-  Які типи значень можуть бути операндами математичних функцій? 
- Які математичні функції можна застосувати до числа -1 ? 
- Чим відрізняються *Грошовий* і *Фінансовий* формати подання чисел? 
-  Чи є масивом іменована група комірок, у яких містяться як числові значення, так і текст? 
-  У яких випадках доцільно використовувати автоматичне заповнення комірок? 
-  У першому році нашої ери 1 г золота покладено в банк під 3 % річних. Скільки золота за такою угодою банк мав би видати сьогодні? 
-  Дехто оформив у банку кредит на 120000 грн на 10 років під 30 % річних із помісячною виплатою (тіло кредиту виплачується однаковими внесками, відсоток сплачується з суми, яка була у використанні на місяць, що закінчився). Який прибуток має банк з такої угоди? 

Виконуємо

-   Знайдіть (використовуючи Довідку Microsoft Excel або довідкову систему іншого табличного процесора, з яким ви працюєте) в табл. 7.1 усі функції, які не потребують операндів. ▲
-  Знайдіть (використовуючи Довідку) в табл. 7.1 усі функції, які потребують тільки одного операнда. ▲
-  Знайдіть (використовуючи Довідку) в табл. 7.1 усі функції, які потребують тільки двох операндів. ▲
-  Знайдіть (використовуючи Довідку) в табл. 7.1 усі функції, які потребують не менше двох операндів. ◆
-  Запишіть із використанням математичних функцій ABS і COS вираз $y = \frac{\cos x}{x}$. Значення x уводимо в окрему комірку. ◆
-   Створіть ЕТ, за допомогою якої можна обчислити значення квадратного тричлена за його коефіцієнтами. ◆
-   Створіть таблицю, яка обчислює арифметичні значення коренів квадратних чисел натурального ряду від n до m , подаючи їх із точністю до трьох десяткових знаків. ◆
- Створіть ЕТ для обчислень значень функції $y = 2x^2 - 5x + 10$ в інтервалі значень x від -10 до 10 з кроком 1 . Для створення масиву значень x використайте автоматичне заповнення комірок. Як зробити так, щоб крок послідовності (різницю) можна було змінювати без редагування формули? ★

7.3. Призначення та використання основних логічних функцій табличного процесора

 Для опрацювання даних із вибором способу опрацювання використовують логічні вирази, тобто формули, які описують перевірку умови (або кількох умов). Значень, яких можуть набувати такі вирази, лише два: істинне або хибне (англ. true або false), так або ні, 1 або 0. Наприклад, вираз $C1 > 10$ матиме значення true, якщо в комірці $C1$ міститиметься число більше 10, і false – в усіх інших випадках.

Логічні вирази використовують для визначення дій, які виконуються після перевірки істинності певної умови, наприклад, формула: $=IF(G7 <> 0; F7/G7; «на нуль ділити не можна»)$, яку записано в комірку $H7$, означає, що вміст комірки $F7$ буде поділено на вміст комірки $G7$, якщо та містить ненульове значення або не є порожньою, результат обчислення буде записаний у комірку $H7$, інакше в комірку $H7$ буде записано текст “на нуль ділити не можна”.

Подібні вирази ви вже використовували, записуючи алгоритмічні структури розгалуження в навчальному середовищі програмування Scratch і описуючи їх мовою програмування Паскаль.

Очевидно, що якщо в функції IF(умова; результат1; результат2) заповнені крім умови обидва місця для результатів, отримуємо структуру розгалуження першого типу – “якщо умова – то оператори1 – інакше – оператори2” (рис. 7.10), тобто повне розгалуження.

Якщо заповнено тільки місце для умови і першого результату, то отримуємо структуру “розгалуження” другого типу – неповне розгалуження (рис. 7.11).

✓ *Формули, які повертають логічні значення, також допускають використання вкладення.*

Приклад 1. Якщо необхідно створити таблицю для відображення накопичення значень певного параметра для кількох об’єктів, а потім визначити відповідність накопичених сум певним критеріям, можна скористатись виразом: =IF(SUM(D15:M15)<10;\$E\$17;IF(SUM(D15:M15)<50;\$E\$18;IF(SUM(D15:M15)<100;\$E\$20))).



Рис. 7.10. Структура “розгалуження” першого типу (повне розгалуження)



Рис. 7.11. Структура “розгалуження” другого типу

У комірках D15:M15 містяться значення параметра (наприклад, оцінки, отримані протягом семестру), а в комірках E17, E18, E20 – назви груп, до яких належать накопичені значення.

Створення таблиць, призначених для опрацювання масивів даних, отриманих для груп іменованих об’єктів (наприклад, розташованих у різних містах філій банку), може спростити застосування функції SUMIFS, яка виконує додавання значень з використанням певної умови.

Приклад 2. Нехай у стовпці С розташовані значення прибутку філій установи, а в стовпці В – назви міст, у яких вони розташовані. Формула

=SUMIFS (B4:B30;«Черкаси»;C4:C30) поверне суму прибутків тільки тих філій, які розташовані в м. Черкаси.

Приклад 3. Нехай n учнів склали 10 тестів, відповідь на кожен із яких можна оцінити від 0 до 10 балів. Треба створити ЕТ для визначення результату тестування кожного учня з використанням оцінок “недостатньо”, “задовільно”, “добре”, “відмінно”, якщо відомо, що ці оцінки виставляють при сумах набраних балів: до 30, від 31 до 50, від 51 до 70, від 71 до 100 відповідно.

З цією метою доцільно застосувати математичні та логічні функції, як це показано на рис. 7.12.

Формулу (вираз), яку використано для віднесення результатів тестування певної особи до однієї з чотирьох груп, подано в формі, яка передбачає можливість окремого введення значень критеріїв (меж інтервалів) і назв груп (з цією метою використано посилання на комірки N4:N7).

Вираз для встановлення відповідності значення й назви інтервалу

Прізвище і.і.	1	2	3	4	5	6	7	8	9	10	сума	оцінка	
Баніков І.	9	2	9	2	10	10	6	9	8	8	73	відмінно	30 недостатньо
Візь М.	4	6	4	8	9	6	9	9	1	8	64	добре	50 задовільно
Буряк Є.	7	4	6	8	7	7	7	7	5	5	63	добре	70 добре
Дерюгін Ф.	3	4	7	3	4	1	3	2	1	1	29	недостатньо	100 відмінно
Бремєва П.	7	7	0	4	1	9	2	3	4	3	40	задовільно	
Лавченко А.	4	0	2	7	8	6	1	5	9	1	43	задовільно	
Матвієнко В.	1	2	0	7	5	3	6	4	7	2	37	задовільно	
Паркуян Л.	5	3	2	4	9	8	1	9	9	2	52	добре	
Пухан В.	7	9	6	7	0	6	0	9	6	3	55	добре	
Сабо В.	2	8	9	5	2	0	5	5	0	8	44	задовільно	
Серебрянников В.	8	4	2	6	8	10	4	7	2	3	54	добре	
Яшин А.	8	10	6	10	9	9	6	6	4	9	77	відмінно	

Масив даних

Значення верхніх меж інтервалів (критерії) і назви інтервалів

Рис. 7.12. Електронна таблиця для підведення підсумків тестування

Формулу (рис. 7.12)

=IF(SUM(B4:K4)<30;\$O\$4;IF(SUM(B4:K4)<50;\$O\$5;IF(SUM(B4:K4)<70;\$O\$6;\$O\$7))) можна подати у вигляді опису алгоритму :

якщо SUM(B4:K4)<30

то оцінка = \$O\$4

інакше якщо SUM(B4:K4)<50






то оцінка = \$O\$5

інакше якщо SUM(B4:K4)<70







то оцінка = \$O\$6

інакше назва інтервалу = \$O\$7

Перевіряємо себе

-  З якою метою замість явного подання (приклад 3) назв інтервалів у формулі використано посилання? Навіщо ці посилання мають форму абсолютних посилань? ▲
-  Яка частина (частини) формули описує критерії віднесення значення до певного інтервалу? ★
-  Як можна розширити список учнів (рис. 7.12)? ★
-  Які зміни потрібно внести до формули, для того щоб можна було оперативно змінювати критерії віднесення значень до певного інтервалу? ★
-  Чому в формулі жодного разу не використано значення нижньої межі інтервалів? ★
- Яку оцінку “поставить” алгоритм, якщо значення у стовпці L перевищать 100? Що треба зробити для того, щоб уникнути такої помилки? ★

Виконуємо

-  Спробуйте описати словесно й побудувати графічне подання алгоритму, який описано формулою
$$=IF(SUM(D15:M15)<10; \$E\$17; IF(SUM(D15:M15)<50; \$E\$18; IF(SUM(D15:M15)<100; \$E\$20)))$$
. ★
-  Перепишіть алгоритм, передбачивши надання назві інтервалу значення без використання посилання на комірку. Яка частина таблиці після цього перестане бути необхідною? ▲
-   Створіть таблицю для обчислення найменшого спільного кратного двох чисел. ★
-  Створіть таблицю для обчислення найбільшого спільного дільника двох чисел. ★
-  Створіть таблицю для перевірки того, чи можна побудувати трикутник, заданий значеннями довжин трьох сторін. Використайте функцію SUM і логічний вираз. ★

7.4. Призначення й використання основних статистичних функцій табличного процесора



Опрацювання великих наборів даних і отримання на основі їх аналізу відомостей щодо об'єкта або сукупності об'єктів є однією з основних задач науки статистики.

✓ *Потреба в статистичному аналізі даних виникає в тому випадку, якщо потрібно визначити, чи пов'язані між собою дві або декілька величин, подій (як впливає паління тютюну на виникнення певних захворювань у людини, яким буде врожай зернових, якщо протягом зими випала певна кількість снігу, тощо), спрогнозувати певні явища в природі (передбачити погоду, знайти родовище корисних копалин тощо), спрогнозувати розвиток виробництва.*

З цією метою необхідно майже завжди досліджувати великі обсяги даних. Найпростішим прикладом є завдання: визначити межі змін розміру деталі, яка виробляється автоматичною лінією.

Для цього відбирають досить велику (300...1000 штук) кількість готових деталей з партії, вимірюють їх розміри. Для розміру (або кількох розмірів), що є важливим для наступних етапів виготовлення та подальшої експлуатації виробу, частиною якого буде деталь, складають таблицю.

✓ *Статистичне опрацювання отриманих даних має відповідати на такі запитання.*

“Яке середнє значення розміру деталі?”

“В яких межах змінюється цей розмір?”

“Яким є стандартне відхилення розміру від середнього для партії деталей (вибірки) і чи можна це значення застосувати до всіх деталей, які виробляються автоматичною лінією?”

“З якою надійністю можна гарантувати, що розмір деталі, взятої довільно з партії, міститься в певних межах?”

“Скільки деталей певного розміру є в партії?”

Приблизно такі самі запитання можна поставити, досліджуючи дані соціологічних опитувань (скільки людей проголосують на виборах за певну партію), дані, отримані в процесі спостережень за популяціями тварин, навіть опрацьовуючи дані, отримані в процесі виконання лабораторних робіт з фізики, хімії тощо.


Тому для подібних досліджень використовують методи, які в ЕТ описані у вигляді статистичних функцій. Статистичні функції ЕТ забезпечують можливість опрацювання результатів фізичних, хімічних експериментів, соціальних досліджень тощо. У результаті такого опрацювання великих масивів однотипних даних можна визначити взаємозв'язок між величинами, прогнозувати значення даних у залежностях, тобто визначити тенденцію розвитку певного процесу.

Статистичні функції ЕТ забезпечують можливість опрацьовувати результати фізичних, хімічних експериментів, коли потрібно опрацьовувати великий масив однотипних даних, визначити взаємозв'язок між величинами, виконати статистичний аналіз даних, спрогнозувати наступні

значення даних у деякому діапазоні величин, тобто визначити тенденцію розвитку певного процесу тощо.

Для обчислення середнього арифметичного значення є статистична функція **СРЗНАЧ** (англ.: **AVERAGE**)

Функція має такий вигляд: **AVERAGE(число1; число2;...)**, де **число1**, **число2**, ... – від 1 до 255 аргументів, для яких обчислюється середнє.


 Аргументами можуть бути тільки числа або імена, масиви або посилання, *які містять числа*, при цьому порожні комірки функція не враховує, але комірки з нульовими значеннями беруть участь у обчисленнях.

Якщо аргумент масиву чи посилання містить текст, логічні значення або порожні комірки, то ці значення беруться до уваги. Аргументи, які є значеннями помилки або текстом, який не можна перетворити на числа, призводять до помилки.

Для обчислення найбільшого (максимального) значення даних, що містяться в діапазоні комірок, слугує функція **МАКС**. (англ.: **MAX**)

Функція має такий вигляд: **MAX(число1;число2;...)**, де **число1**, **число2**, ... – від 1 до 30 чисел, серед яких визначається максимальне значення.

Причому функція може опрацьовувати не тільки дані, подані у вигляді чисел, можна задавати аргументи також у вигляді логічних значень або текстовими поданнями чисел.

 Під час роботи з масивами функція опрацьовує тільки числа.

Приклад. У таблиці в діапазоні комірок **A1:A6** міститься послідовність чисел: {4; 5; 2; 3; 4; 5; 2; 3}. Визначити максимальне значення числа в цій послідовності. Для цього створюємо формулу = **МАКС(A1:A6)**, яка повертає число 5.

Для обчислення мінімального значення з певного діапазону значень використовують функцію **МИН** (англ. **MIN**). Синтаксис функції аналогічний синтаксису функції **МАКС**.

Для визначення номера певного найбільшого значення в заданому діапазоні комірок зручно використовувати статистичну функцію **НАИБОЛЬШИЙ** (англ. **LARGE**). Функція визначає *i*-те найбільше значення з множини даних, наприклад, функцію можна застосувати для визначення першого, другого й третього місць серед усіх учнів класу, які брали участь у змаганнях.

Функція має такий вигляд: **НАИБОЛЬШИЙ(масив; i)**, де **масив** – діапазон даних, серед яких потрібно визначити *i*-те найбільше значення; *i* – це позиція в діапазоні даних.

- ✓ Різниця між значеннями, що повертають функції **МАКС** і **МИН** для вибірки, називається “**розмахом вибірки**”.
- ✓ У вибірці (наборі значень) **мода** – значення, що трапляється найчастіше.
- ✓ **Медіана** – значення, рівновіддалене від максимального і мінімального значень.
- ✓ **Середнє** – середнє арифметичне значення.

Жодне з цих чисел саме по собі не характеризує повною мірою те, як розташовані дані на числовій осі. Уявімо, що дані згруповані в трьох областях, одна половина даних близька до деякого малого значення, а друга – до двох інших великих значень. **Медіана** і **середнє** значення при цьому будуть близькі до порожньої середини, а мода, найімовірніше, дорівнюватиме найменшому значенню, що часто трапляється.

Для обчислення медіани використовують функцію **МЕДИАНА** (англ. **MEDIAN**).

Функція має вигляд: **MEDIAN**(число1;число2;...), де **число1**; **число2**; ... – від 1 до 255 аргументів, для яких потрібно знайти медіану.

- ✓ Якщо кількість чисел у масиві є парною, функція **MEDIAN** обчислює середнє з двох чисел, які розташовані посередині.

Приклади: **МЕДИАНА**(1; 2; 3; 4; 5) дорівнює 3, **МЕДИАНА**(1; 2; 3; 4; 5; 6) дорівнює 3,5, середнє арифметичне 3 і 4.

Мода обчислюється за допомогою функції **МОДА** (англ. **MODE**)

Якщо набір даних не містить однакових даних, то функція **МОДА** повертає значення помилки #Н/Д (немає даних).

Приклад **MODE** ({5,6; 4; 4; 3; 2; 4}) дорівнює 4.

Подібно працює і функція **РАНГ** (англ. **RANK**), але за її допомогою визначають місце (ранг) значення у масиві значень. Повертає числове значення, яке є місцем аргументу в списку чисел. (Якщо впорядкувати список, то ранг числа дорівнюватиме його позиції.)

Функція має вигляд: **RANK**(число; посилання; порядок), де **число** – число, ранг якого потрібно визначити; **посилання** – посилання на список чисел. Нечислові значення в посиланні ігноруються; **порядок** – параметр, що визначає, як розподіляються порядкові номери.

Якщо порядок дорівнює 0 (нулю) або не вказаний, функція визначає ранг числа, спираючись на припущення, що посилання є списком, відсортованим за спаданням. Якщо порядок має будь-яке ненульове значення, функція визначає ранг числа на основі припущення, що посилання є списком, відсортованим за зростанням.

- ✓ Функція **RANK** призначає числам, що повторюються, однаковий ранг. Це впливає на ранги наступних чисел. Наприклад, якщо в

списку цілих чисел, відсортованих за зростанням, число 10 трапляється двічі та має ранг 5, то число 11 матиме ранг 7 і жодному числу не буде призначено ранг 6.

Наприклад, нехай після проведення змагань отримано список, який містить прізвища спортсменів і суми балів, отримані ними з усіх вправ. Треба визначити місце, яке посів кожний спортсмен (рис. 7.13). Зверніть увагу на те, що друге й сьоме місця присвоєні двом спортсменам, а третє й восьме місця – нікому.

	A	B	C
1	Прізвище І.	Сума	Місце
2	Василенко М.	25	6
3	Сумський В.	56	1
4	Іванов І.	43	2
5	Кравченко М.	43	2
6	Біба К.	29	7
7	Сабо І.	34	4
8	Банніков М.	23	7
9	Пузач Т.	32	5
10	Мунтян А.	10	9

Рис. 7.13. Приклад застосування функції РАНГ

Уявімо, що деяка фізична величина змінюється так, що в кожний момент часу набуває певного значення. Дослідникові невідомо, чи є в появі певних значень величини якась закономірність. Отже, потрібно насамперед з'ясувати, які значення величини з'являються частіше за інші, від чого залежить їх поява. Така задача виникала, коли намагалися дослідити залежність швидкості молекул від температури, у процесі розробки пристроїв, які є основою сучасного телебачення, тощо.

Подібні задачі виникають і тоді, коли потрібно оцінити якість партії приладів, призначених для вимірювання певної величини. Найпростіше це зробити, якщо виміряти одну й ту саму (відому наперед з високою точністю) величину кілька разів різними екземплярами цього приладу.

Для опису таких задач та їх дослідження засобами електронних таблиць використовують функцію **ЧАСТОТА** (англ. **FREQUENCY**).

Функція має вигляд: **FREQUENCY(масив_даних; масив_інтервалів)**, де *масив_даних* – масив або посилання на набір значень, для яких потрібно обчислити частоти. Якщо *масив_даних* не містить жодних значень, формула **FREQUENCY** повертає масив із нулів; *масив_інтервалів* – масив або посилання на сукупність інтервалів, за якими потрібно

згрупувати значення масиву даних. Якщо *масив_інтервалів* не містить жодних даних, функція **FREQUENCY** повертає кількість елементів масиву даних.

Функція повертає масив, елементи якого дорівнюють кількості елементів даних, значення яких потрапляють у визначені інтервали – частоти. Кількість елементів у масиві частот на один більша за кількість елементів у масиві інтервалів. Додатковий елемент масиву містить кількість значень, які перевищують верхню межу інтервалів. Наприклад, під час обчислення трьох діапазонів (інтервалів) значень, уведених у три комірки, для відображення результатів потрібно ввести функцію **FREQUENCY** у чотири комірки. У додаткову комірку функція **FREQUENCY** повертає кількість значень у масиві даних, які перевищують значення верхньої межі третього інтервалу, якщо такі існують.



Формули, які повертають значення у вигляді масивів, слід вводити як формули масивів.



Відношення частоти певної події до всієї кількості подій називається “**відносною частотою події**”. Якщо подію визначити як “знаходження значення з масиву даних у певному діапазоні”, то сума всіх відносних частот подій завжди дорівнюватиме 1, або 100 %.

На рис. 7.14 подано приклад аналізу даних, розміщених у комірках **A2:W2**. Треба було визначити, як ці значення розподіляються в інтервалах 0...20; 21...40; 41...60; 61...80; 81...100, значення верхніх меж яких занесене в комірки **A4:A8**. Оскільки наперед було відомо, що значення, менші за 0, і значення, більші за 100, неможливі, то для масиву результатів виділено п'ять комірок – **B4:B8**. Для комірки **B4** засобами ЕТ позначено комірки, що впливають на її значення: це діапазон даних **A2:W2** і комірка **A4**, в якій міститься значення верхньої межі першого інтервалу. Для контролю та наступного обчислення відносних частот у комірці **B9** уведено формулу **=SUM(B4:B8)**. Значення, яке повертатиме ця формула, має дорівнювати кількості елементів масиву даних, розміщеного у комірках **A2:W2**.

Значення в комірці **B4**, як показано стрілками, впливає на значення в комірках **C4=ROUND(B4/\$B\$9*100;1)** і **B9=SUM(B4:B8)**.

Застосування функції **ЧАСТОТА** дає можливість побудувати особливий вид діаграми – гістограму, на якій показано, як часто трапляються події, описані у вигляді масиву даних і масиву інтервалів.

Значення в комірці **B4**, як показано стрілками, впливає на значення в комірках **C4=ROUND(B4/\$B\$9*100;1)** і **B9=SUM(B4:B8)**.



*Застосування функції **ЧАСТОТА** дає можливість побудувати особливий вид діаграми – гістограму, на якій показано, як часто трапляються події, описані у вигляді масиву даних і масиву інтервалів.*

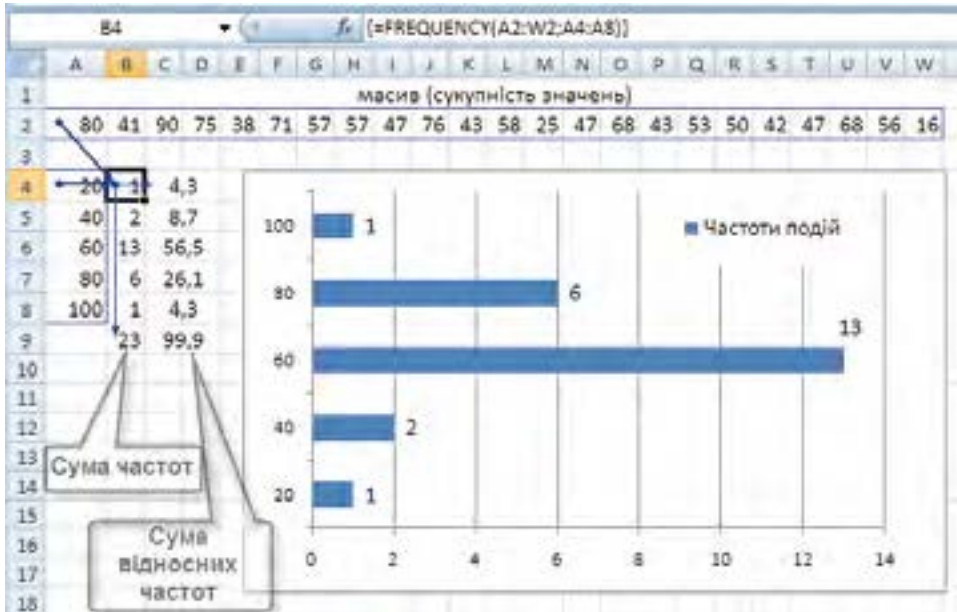




Рис. 7.14. Приклад аналізу даних з використанням функції **ЧАСТОТА** і побудови гістограми




Смужки, що відображають елементи ряду, підписані значеннями частот подій (цифрове значення біля маркера даних певного інтервалу), тому подібна діаграма може бути використана не тільки для якісного, а й для кількісного аналізу. Зокрема, її (якщо на ній відобразити значення відносних частот) можна використати для попереднього оцінювання **ймовірності** певної події, яка полягатиме в тому, що випадковим чином вибране з масиву даних значення потрапить у наперед визначений інтервал значень.

Перевіряємо себе




1. Що є об'єктом дослідження статистики?
2. У процесі розв'язування яких задач виникає потреба в застосуванні статистичних формул?
3. Що таке розмах вибірки? Чому не можна однозначно судити про вибірку, знаючи лише середнє арифметичне значення і розмах?
4. Який параметр масиву чисел називається частотою? Відносною частотою?
5. Чому дорівнює сума частот значень для вибірки? Сума відносних частот?
6. Які параметри значень, отриманих у результаті вимірювань, є суттєвими для визначення якості процесу вимірювання?

7.   Як можна використати ЕТ для опрацювання результатів соціологічного опитування, для якого використано анкету, що містила п'ять запитань, на кожне з яких можна було відповісти, увівши число 0...10? Запропонуйте варіант, який дасть змогу отримати дані не тільки щодо опитування загалом, а й за кожним запитанням окремо. ★

Виконуємо

1.    Відомо, що деякі деталі (наприклад, поршневі пальці та поршні автомобільних двигунів) добирають за групами, які позначають певним кольором. Створіть ЕТ, вхідними даними для якої будуть масиви значень діаметрів поршневих пальців, значення граничних розмірів для груп, кольори груп. Кількість груп – не більше чотирьох. У результаті маємо отримати для кожної деталі (поршневого пальця) номер групи та колір, яким деталь слід позначити. ★

Порада: дані (начебо результати вимірювання з використанням мікрометра) можна згенерувати, застосувавши формулу $=20+\text{ROUND}(\text{RAND}()*0,1;3)$, а умови записати як $=\text{IF}(\text{A3}\leq 20,025;1;\text{IF}(\text{A3}\leq 20,05;2;\text{IF}(\text{A3}\leq 20,075;3;\text{IF}(\text{A3}\leq 20,1;4))))$. Можна використати умовне форматування.


2.   Визначте, як розташовані дані в ЕТ, поданій на рис. 7.14. Відтворіть цю таблицю. ✨
3.  Визначте, як розташовано дані в ЕТ, поданій на рис. 7.13. Відтворіть цю таблицю. ▲

7.5. Умове форматування



Форматування комірок і написів у комірках полягає у наданні певного кольору та накреслення літерам, заливці тла, кольору і товщини – межам комірок.

Особливим варіантом форматування, який не використовується в текстових процесорах, є **умове форматування**.



 **Умове форматування** – надання певного формату поданню вмісту комірок залежно від значень числових величин, які містяться в цих комірках, значень логічних виразів, для обчислення яких використовуються дані інших комірок.

Умове форматування використовують, коли необхідно певним чином виділити комірку залежно від значення даних, які зберігаються в ній або в іншій комірці. Наприклад, якщо ЕТ призначено для обчислення стану грошового рахунку певної особи, можна позначати червоним кольором значення залишку, якщо воно менше, ніж наперед визначене (наприклад, якщо з карткового рахунку знято більше коштів, ніж на


Для того щоб позначити червоним кольором тексту значення температур, що зберігаються в комітках В2...В10, залежно від того, чи перебуває за цієї температури і нормального тиску деяка речовина в рідкому стані, необхідно у певну комірку (наприклад, А2) ввести значення температури плавлення (наприклад, для води – 0°C, бензолу – 5,45°C, камфори – 178,5°C). Потім виконати дії в певній послідовності (див. рис. 7.15).


Можливі й інші варіанти умов і форматів, які реалізуються при набутті логічним виразом значення “істина” (true).

Перевіряємо себе



1. Що таке умовне форматування? ▲
2. Як накласти дві-три умови в процесі умовного форматування? ✦
3. Що таке стиль? ▲
4. Як створити новий стиль? ▲
5. Як відбувається імпортування стилів? ✦
6. Що таке автоформат? ▲
7. Які параметри автоформату можна виставляти додатково? ✦
8.  Які типи значень можна використовувати при описанні правил (логічних виразів) умовного форматування? ▲
9. Які з перших шести правил (умов) можна застосувати до значень текстового типу (див. рис. 7.15)? ✦
10. Яке правило можна записати як $a = x$, якщо a – константа, x – значення в комірни? До яких типів даних може бути застосоване це правило? ✦
11. Яке правило можна записати як $a < x$, якщо a – константа, x – значення в комірни? До яких типів даних може бути застосоване це правило? ✦
12. Яке правило можна записати як $x < b$, якщо b – константа, x – значення в комірни? До яких типів даних може бути застосоване це правило? ✦
13. Яке правило можна записати як $a < x < b$, якщо a і b – константи, а x – значення в комірни? До яких типів даних може бути застосоване це правило?
14.  Чому не завжди доцільно використовувати правило “Дорівнює...” до значень числового типу (дійсних чисел)? Що потрібно зробити для того, щоб порівняти, наприклад, правильність розв’язку задачі, отриманого у вигляді числа, з еталонним? ★

Виконуємо

1.  Наведіть приклади доцільності застосування умовного форматування. ▲


2.  Для діапазону комірок A3:G8 встановіть правило умовного форматування: “Якщо значення, введене в комірку, менше за значення, яке зберігається в комірці A2, залити комірку блідо-червоним кольором”. ★


3. Нехай у комірках стовпця з одинадцятої по двадцяту розташовані числа від 1 до 10. Створіть умови форматування, за якими уведення певного числа в п'яту комірку цього самого стовпця викликало б забарвлення певним кольором потрібної кількості комірок. ★

4.   Таблиця містить такі дані про учнів школи: прізвище, зріст і вік учня. До баскетбольної секції приймають дітей зростом не менше 160 см. Вік не може перевищувати 13 років. Вихідні дані для заповнення таблиці підібрати самостійно, не менше десяти рядків. У таблиці має бути позначено зеленим кольором прізвища дітей, які можуть бути прийняті до секції. ★

Порада: у окремому стовпці сформулюйте логічний вираз, значення якого використайте для умовного форматування.

7.6. Створення і налагодження діаграм

 Діаграми – це засоби наочного подання даних, які полегшують порівняння, виявлення закономірностей і тенденцій змін даних. Усі сучасні табличні процесори надають користувачеві можливість побудови діаграм. Діаграми можуть будуватися для одного або кількох рядів даних. Діаграми типу **Графік** можуть унаочнювати зв'язок між двома величинами, відображати графік функції.

 Спеціальні види діаграм, тривимірні діаграми. Налagodження діаграм.

При розгляді застосування формул масиву використано новий тип діаграм – **гістограму**, який відрізнявся тим, що для стовпчикової діаграми вісь категорій розташована вертикально (див. рис. 7.6). Таке подання розподілу частот подій використовується досить часто для ілюстрування результатів економічних і соціометричних досліджень, фізичних експериментів.

Для підприємства, яке має кілька філій (видів діяльності), унаочненям внеску кожної філії (виду діяльності) в загальний прибуток (загальні витрати) може бути стовпчикова діаграма, в якій висота стовпчика, що відповідає часовому інтервалу (рік, місяць, тиждень), пропорційна сумі коштів, отриманих (витрачених) підприємством, а кожна філія (вид діяльності) позначена іншим кольором.

Для подання даних щодо змін температури повітря (день–ніч), мінімального та максимального значення досягнень групи спортсменів, вартості цінних паперів на біржі використовують так звану **біржову**

діаграму, на якій відображаються одночасно мінімальне, максимальне й середнє значення деякої величини.

Для того щоб порівняти й унаочнити внески кількох напрямів діяльності в загальну діяльність, використовують “пелюсткову” діаграму.

Діаграму можна створити на окремому аркуші або розташувати як упроваджений об’єкт на аркуші даних. Крім цього, діаграму можна опублікувати на веб-сторінці. Щоб створити діаграму, потрібно спочатку ввести для неї дані на аркуші. Виділивши ці дані, слід запустити **Майстер діаграм** для покрокового створення діаграми, під час якого вибираються її тип і різні параметри, або за допомогою панелі інструментів **Діаграма** створити базову діаграму, яку згодом можна змінити.

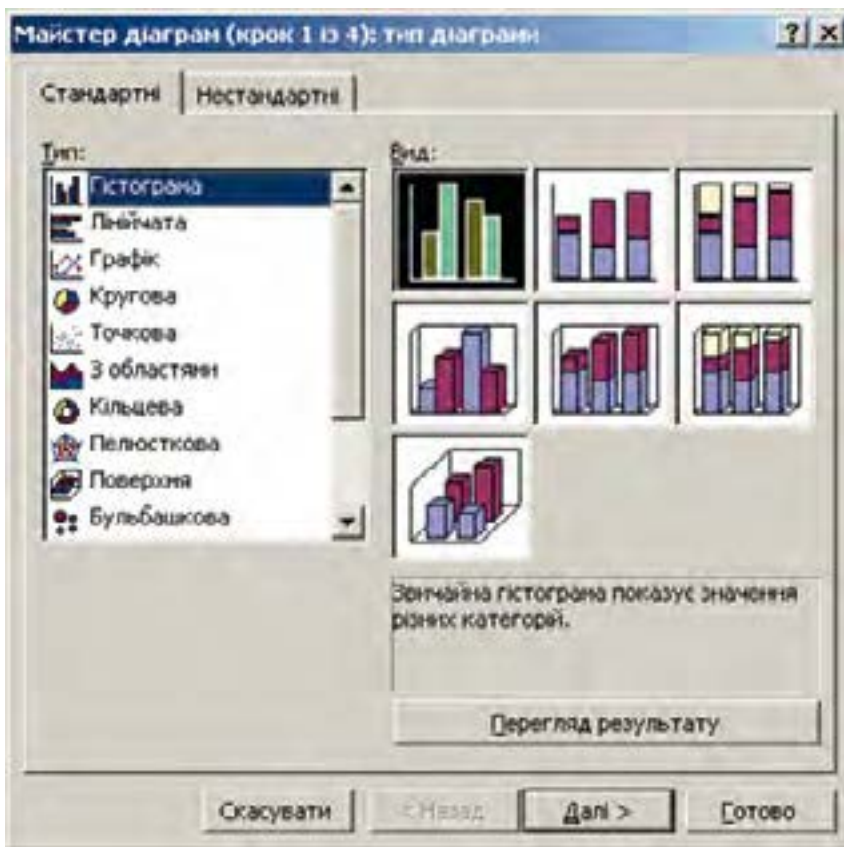


Рис. 7.16. Вибір типу діаграми

✓ Діаграма зв’язана з даними аркуша, на основі яких вона створена, і в разі зміни даних автоматично оновлюється.

Нижче наведено елементи діаграми:

Маркер даних. Смуга, область, точка, сектор або інший об’єкт на діаграмі, який представляє одну точку даних або значення клітинки

аркуша. Пов'язані один з одним маркери даних на діаграмі утворюють ряд даних. Послідовність маркерів даних представляє один ряд даних.

Ряд даних. Пов'язані одна з одною точки даних, нанесені на діаграму. Кожний ряд даних на діаграмі має власний колір або інший спосіб позначення та представлений на легенді діаграми. Діаграми всіх типів, за винятком кругової, можуть містити кілька рядів даних.

Основні лінії сітки. Лінії, які можна додати до діаграми для поліпшення сприйняття й оцінки відображуваних даних. Лінії сітки починаються від поділок на осі та перетинають область побудови й візуалізують основні інтервали на осі. На діаграмі можна також вивести на екран проміжні лінії, які позначають інтервали в межах основних інтервалів.

Імена категорій. Microsoft Excel використовує заголовки стовпців або рядків як імена інтервалів осі категорій. При налагодженні діаграми їх можна замінити іншими.

Імена рядів даних діаграми. Microsoft Excel також використовує заголовки стовпців або рядків як імена рядів даних. Імена рядів даних показані в легенді.

Легенда. Область, у якій подано кольори або інші способи позначення, що відповідають рядам даних або категоріям на діаграмі.

Наприклад, якщо навести вказівник на легенду, з'явиться підказка, яка містить слово "Легенда".

Аркуш діаграми. Аркуш книги, який містить лише діаграму.

Упроваджена діаграма. Діаграма, розташована на аркуші даних, а не на **Аркуші діаграми**. Впроваджені діаграми зручні, коли потрібно переглянути або надрукувати діаграму або звіт зведеної діаграми разом із вихідними даними й іншими відомостями, які містяться на аркуші. У обох випадках діаграма зв'язується з вихідними даними на аркуші, тобто в разі оновлення даних аркуша оновлюється створена на їх основі діаграма.

✓ *Якщо затримати курсор миші на елементі діаграми, з'явиться підказка з назвою цього елемента.*

✓ *Створення діаграми розпочинається з вибору її типу. Але слід пам'ятати, що до початку створення діаграми необхідно повністю закінчити формування структури таблиці, наповнити її даними.*

Процедура створення діаграми для Microsoft Excel 2003 і попередніх могла бути виконана з використанням **Майстра діаграм (Chart Wizard)** покроково, за п'ять кроків.

На рис. 7.17 показано послідовність побудови діаграми для Microsoft Excel 2007–2010.

Після кроку 4, на якому на аркуші з'являється порожнє полотно діаграми, у вікні Excel 2007–2010 з'являється контекстний інструмент

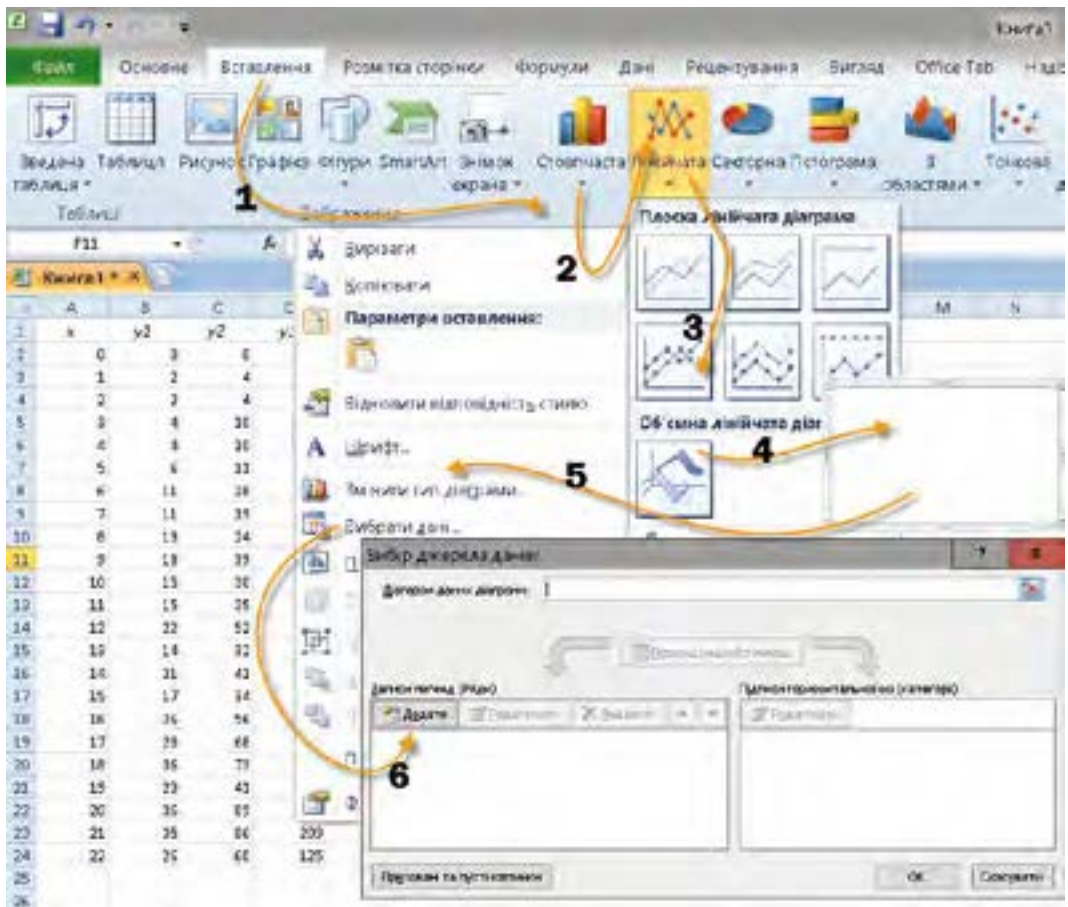


Рис. 7.17. Процедура створення діаграми для Microsoft Excel 2007–2010

“Робота з діаграмами”, що містить три стрічки: “Конструктор”, “Макет”, “Формат”. Інструменти роботи з діаграмами в Excel 2007–2010 прості й зрозумілі (рис. 7.18).

Відмінність від попередніх версій полягає в тому, що не обов’язково дотримуватися послідовності дій.

Можливий і інший спосіб створення діаграми, як показано на рис. 7.17 (кроки 5, 6 і наступні). Після того як на аркуші з’явилося полотно діаграми, можна, не користуючись стрічковими меню, натисканням правої кнопки миші на полотні діаграми через команду **Вибрати дані...** контекстного меню викликати вікно **Вибір джерела даних** (кроки 5 і 6 на рис. 7.17 і рис. 7.18).

✓ У вікні **Вибір джерела даних** є такі об’єкти:

Список **Елементи легенди (ряди)**. Відображає список імен існуючих рядів даних.

Кнопка Додати. Цю кнопку треба натиснути, щоб додати до діаграми новий порожній ряд. Щоб додати ім'я та значення для нового ряду, слід клацнути поле **Ім'я** або **Значення** й виділити діапазон на аркуші або ввести ім'я та значення в поля. Дані, введені в полях **Ім'я** та **Значення**, не додаються до аркуша.

Кнопка Видалити. Вилучає вибраний ряд даних із діаграми.

- ✓ *Додавання та вилучення рядів даних на діаграмі не впливає на дані на аркуші.*
- ✓ *Якщо в ту частину таблиці, за якою будувалася діаграма, буде внесено зміни, то програма Excel автоматично модифікує діаграму.*



*Рис. 7.18. Процедура створення діаграми для Microsoft Excel 2007–2010 (використання стрічкових меню групи **Робота з діаграмами**)*

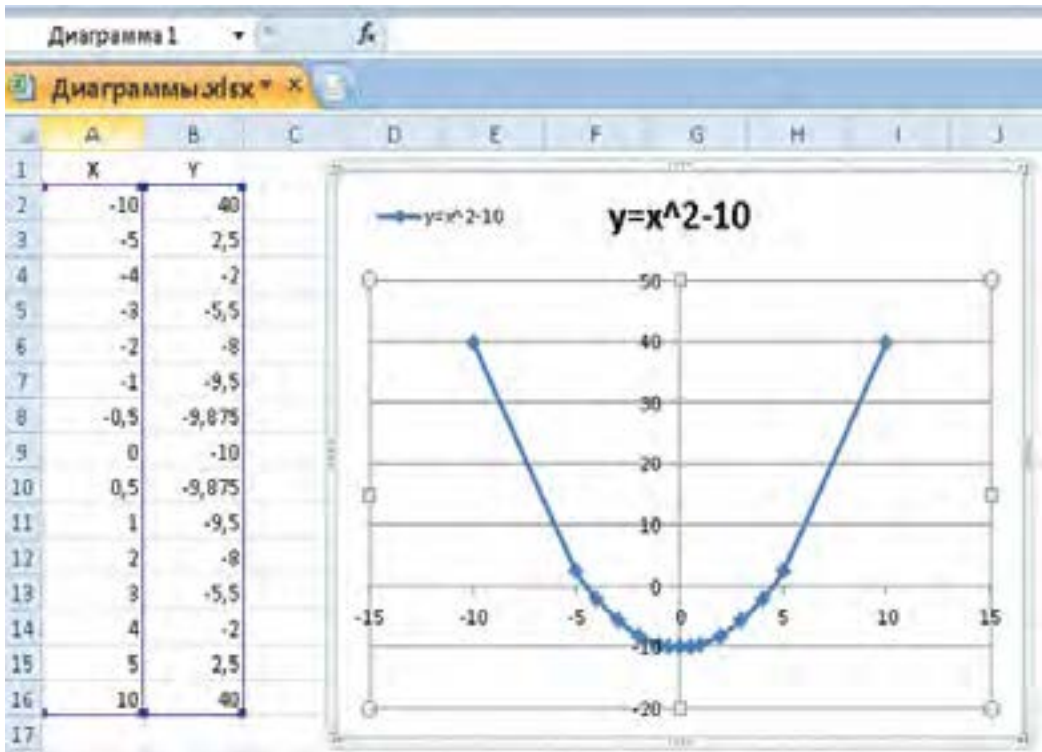


Рис. 7.19. Процедура створення діаграми для Microsoft Excel 2007

Виділіть нові дані в таблиці й перетягніть їх за допомогою миші на діаграму. Для цього поставте курсор миші на межу маркірованої області і, не відпускаючи ліву кнопку миші, перемістіть її на діаграму. Щойно відпустите кнопку миші, діаграма буде змінена (актуалізована).

Якщо діаграма розташована на окремому робочому аркуші, то для її актуалізації можна використовувати команду **Вибрати дані (New Data)** з меню **Вставлення (Insert)**. У діалоговому вікні команди потрібно вказати область таблиці, що була додана. Для цього або виділіть область, або вкажіть її адресу, попередньо створивши новий ряд (кроки 5 і 6 на рис. 7.17).

Щоб додати текст у будь-який елемент діаграми, клацніть на ньому правою кнопкою миші, виберіть команду **Змінити текст**, а потім уведіть текст.

Розглянемо особливості деяких діаграм.

✓ **Точкова діаграма** – єдина діаграма, для якої осі X і Y можна вважати осями декартової системи координат і будувати графіки функцій так само, як на аркуші паперу.

Для правильної побудови графіка функції не потрібно мати набір значень аргументу з постійним кроком, оскільки вісь X у даному випадку є віссю значень, а не віссю категорій.

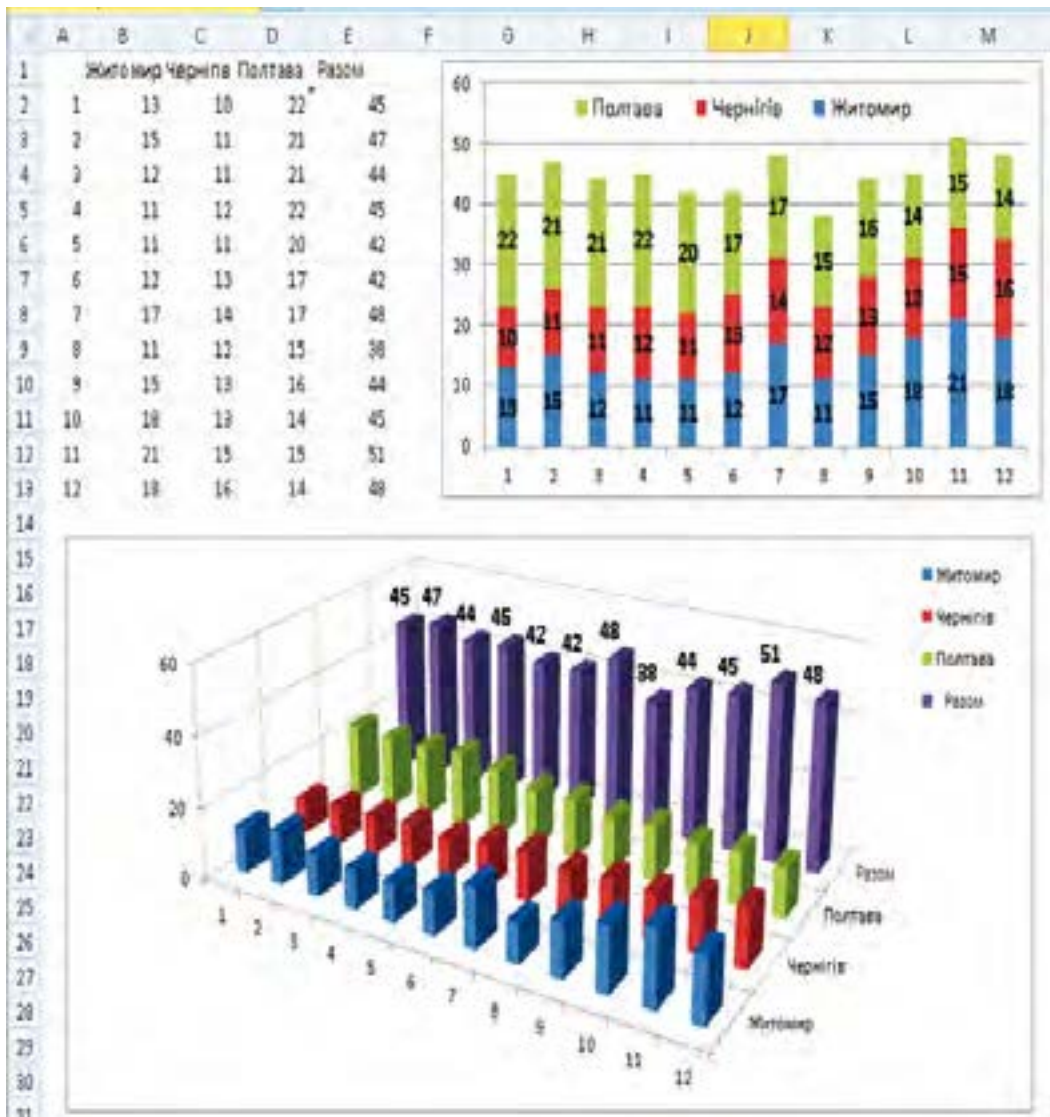


Рис. 7.20. Два різних способи подання даних стовпчиковими діаграмами

На рис. 7.19. показано графік функції $y = 0.5x^2 - 10$, побудований як точкова діаграма.

Нехай протягом 12 місяців три філії одного підприємства давали прибутки, розміщені в стовпчиках B, C, D. На рис. 7.20 подано зміни прибутків кожної філії та всього підприємства у вигляді двох стовпчикових діаграм – із накопиченням та у тривимірному вигляді.

✓ На циклічних і цільових діаграмах текст можна вводити лише в призначені для цього рамки, які відображаються під час додавання діаграми або елемента діаграми.



Рис. 7.21. Редагування елементів діаграми

✓ Для змін вигляду будь-якого елемента діаграми його слід актуалізувати натисненням на ньому лівої кнопки миші з наступним викликом меню натисненням правої кнопки.

Послідовність дій подано на рис. 7.21.





✓ Щоб додати елемент, який не передбачено стандартним макетом діаграми, слід натиснути кнопку **Додати фігуру** на панелі інструментів **Макет**.

Перевіряємо себе

1. Які основні елементи містить стандартний макет діаграми? ▲
2. Як викликати певний тип діаграми? ▲
3. Як викликати й застосувати стандартний макет діаграми? ▲
4. Чи можна змінювати тип діаграми після зв'язування її з даними? Коли це можливо і доцільно? ◆
5. Діаграма якого типу вимагає двох наборів даних? ◆
6. Яким чином можна додати дані на діаграму? ◆

7. У яких випадках потрібно користуватися контекстним меню редагування елементів діаграми? ✦
8. Як додати підписи (значення) до зображень елементів ряду і коли це доцільно робити? ✦
9. Як редагувати текстові елементи діаграми? ✦

Виконуємо

-  Побудуйте графік функції $y=2x^2-20$ для значень $-10 \leq x \leq 10$. ▲
-   Визначте з графіка приблизні значення коренів рівняння $2x^2 - 3x - 20 = 0$. Перевірте обчисленням. ✦
-  Створіть ЕТ і діаграми, подані на рис. 7.20. ✦

7.7. Упорядкування даних у таблицях. Автоматичні та розширені фільтри



Пошук значень здійснюється поелементним порівнянням значень і зразка. Так само, як у текстовому редакторі та файльовій системі, пошук може виконуватися з використанням шаблонів.



Упорядкування здійснюється з урахуванням розміщення літер у алфавіті мови, призначеної для тексту. Структури подання даних.

Табличні процесори надають можливості для пошуку будь-якого вмісту на робочому аркуші й, при потребі, заміни його новим вмістом.

✓ Пошук провадиться у виокремленому діапазоні комірок, а за відсутності виокремлення – в усьому робочому аркуші.

Можна виокремити декілька робочих аркушів і тим самим заданий пошук потрібного вмісту відразу в декількох аркушах. При створенні зразка шуканого вмісту можна використовувати будь-які літери, цифри та спеціальні символи. Крім цього, є такі символи підстановки.

✓ Знак питання (?) застосовується для позначення будь-якого символу (але одного), зірочка (*) – для позначення будь-якої кількості будь-яких символів. Для завдання пошуку самого символу підстановки (? або *) слід увести перед ним хвилясту риску (~).

При використанні як пошуку, так і заміни можна зазначити, чи має враховуватись точне написання шуканого вмісту (малі або великі літери). Можна зазначити, чи має провадитися пошук зразка лише як окремого вмісту комірки чи як довільної частини вмісту комірки.

Можна також вказати напрямок пошуку: в рядках зліва направо чи в стовпцях зверху вниз.

✓ Для виконання пошуку:

У меню **Правка (Редагування)** ⇒ виконати команду **Знайти** (або натиснути сполучення клавіш **Ctrl+F**) ⇒ у вікні діалогу **Знайти** в полі **Знайти** зазначити послідовність символів, які потрібно відшукати (зразок) (рис. 7.22. А) ⇒ у разі потреби можна зазначити інші параметри пошуку – натиснути **Параметри** (рис. 7.22. В) ⇒ для завершення пошуку натиснути кнопку **Закрити**.

Під час пошуку Excel виділяє першу послідовність символів (з урахуванням напрямку пошуку), яка відповідає заданому зразку, встановлює на неї курсор. При потребі можна, не закриваючи, змістити відкрите вікно діалогу. Для пошуку наступного входження натиснути кнопку **Зайти далі**. Для пошуку в зворотному напрямку при натисканні кнопки **Зайти далі** утримуйте натиснутою клавішу **Shift**.

✓ Сполучення клавіш, які дозволяють продовжувати пошук і без виведення на екран вікна діалогу **Знайти**: **Shift+F4** – пошук наступного входження; **Ctrl+Shift+F4** – пошук попереднього входження.

Клацання миші на вкладці **Замінити** дозволяє перейти з вікна **Знайти** у вікно **Замінити**.

У меню **Правка** виконати команду **Замінити** або скористатися сполученням клавіш **Ctrl+H** ⇒ у полі **Знайти** ввести послідовність символів, які потрібно відшукати (зразок) (рис. 7.22. А) ⇒ у полі **Замінити** на вказати послідовність символів для заміни.

У разі потреби можна зазначити інші параметри пошуку – натиснути **Параметри** (рис. 7.22. В) ⇒ для продовження пошуку натиснути кнопку **Знайти далі** ⇒ для виконання заміни натиснути кнопку **Замінити**.

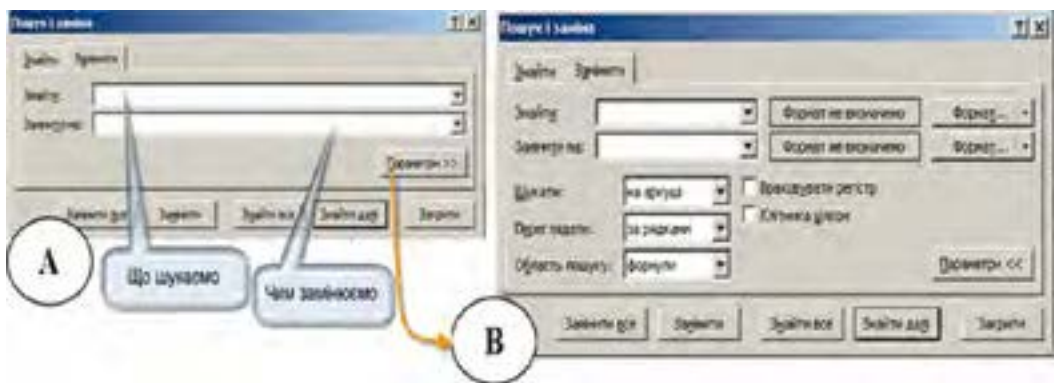



Рис. 7.22. Пошук вмісту для заміни: А – пошук, В – розширений пошук

Кнопка **Замінити все** дозволяє виконати заміну всіх знайдених послідовностей символів, які трапляються. При цьому потрібно бути впевненим, що така заміна за замовчуванням усіх знайдених послідовностей дасть бажаний результат.


Усі табличні процесори мають зручний інструментарій для виконання сортування та фільтрації даних.

 Під **сортуванням** розуміємо **впорядковування** даних за одним або кількома заданими критеріями.

Програма виконує декілька варіантів сортування. Найпростішим варіантом сортування є впорядковування за зростанням. Цей варіант сортування встановлено за замовчуванням.

Упорядковування даних таблиці в алфавітно-цифровому порядку здійснюється за зростанням або спаданням значень. Числа сортуються від найменшого від'ємного до найбільшого додатного, а текст – за алфавітом (за зростанням або спаданням).

Як ключ застосовують виокремлений стовпчик або стовпчик, в якому знаходиться курсор.

 *Сортування даних за кількома полями.*


Засоби Excel дозволяють одночасно сортувати записи за трьома полями. Послідовність сортування полів вибирається в діалоговому вікні **Сортування діапазону** в списках, що розкриваються: **Сортувати за**, **Потім за**, **В останню чергу за**.

Розташовані поряд із кожним списком перемикачі за зростанням, за зменшенням дозволяють вказати напрям сортування.

Перемикач **Ідентифікувати діапазон даних за** дає можливість ідентифікувати дані за підписами або позначенням стовпців аркуша.

При потребі сортування за чотирма і більше полями варто виконати кілька послідовних сортувань. Щоб не втрачати результати попереднього сортування, потрібно спочатку виконати сортування за останніми трьома ключами, а потім – за найпершим.

Фільтрування використовується в роботі з великими таблицями і дає можливість бачити не всю таблицю, а лише її частину, яка висвітлюється за певними ознаками (критеріями).

 *Щоб вибрати критерії, за якими здійснюється фільтрування, потрібно виконати такі дії:*

виокремити діапазон (або всю таблицю), скористатись командою **Дані** ⇒ ⇒ **Фільтр** ⇒ **Автофільтр**. Відразу в кожній комірці верхнього рядка з'явиться кнопка списку.

Клацаючи кнопки списку, вибираємо відповідні критерії фільтрування, тобто відбору. Одержимо таблицю з відібраними даними. В цій

таблиці можна отримувати потрібні суми, добутки, виконувати інші дії, як і в будь-якій таблиці.

Упорядкування за зростанням (за спаданням)

Цей варіант сортування можна застосовувати до даних, наведених у числових і текстових типах. Застосовуючи сортування за зростанням (у разі сортування за спаданням порядок у списках буде протилежним), слід враховувати особливості роботи програми, а саме:

- порожні комірки завжди вміщуються у кінець відсортованого списку (як у варіанті сортування за зростанням, так і у варіанті за спаданням);
- числові типи даних сортуються від найменшого від'ємного до найбільшого додатного (у варіанті сортування за спаданням – навпаки);
- текстові типи даних сортуються позначково зліва направо);
- текстові дані сортуються за таким порядком: спочатку цифри, потім пробіл та символи цифрових клавіш верхнього регістра, і тільки після цього літери в алфавітному порядку;
- під час сортування логічних значень значення **ХИБНІСТЬ (False, 0)** ставиться перед значенням **ІСТИНА (True, 1)**.

Сортування за кількома критеріями відбору

Програма дає змогу виконувати сортування за кількома критеріями відбору.


Розглянемо такий варіант сортування:

- виділити діапазон комірок, де слід виконати сортування;
- виконати команду меню **Дані ⇒ Сортування**;
- у діалоговому вікні **Сортування даних** у полях **Сортувати за і Потім за** вказати стовпці в тому порядку, який потрібен (спочатку за стовпцем “Вік”, а потім за стовпцем “Особа”). Натиснути **ОК**.

Таких умов відбору можна вказати три, причому деякі стовпці можна сортувати за зростанням, а деякі за спаданням. Для проведення сортування за чотирма і більше критеріями відбору слід виконувати сортування в декілька етапів (починаючи з найменш важливого на першому етапі та з найбільш важливого на другому).

Для сортування даних у стовпцях за днями тижня, місяцями слід скористатися ключем (який викликається кнопкою **Параметри**) у діалоговому вікні **Сортування даних**.

Фільтр – швидкий і легкий спосіб пошуку та відображення даних, які задовольняють певні критерії. Після застосування фільтра в таблиці відображаються тільки ті дані, які задовольняють критерії відбору.

 Програма Microsoft Excel має два різновиди фільтрування даних – **автофільтр** (для простих умов відбору) та **розширений фільтр** (для складених умов відбору).

✓ *Зверніть увагу, що фільтрування відрізняється від сортування тим, що після фільтрування дані в списку не впорядковуються, з них лише тимчасово приховуються записи, які не задовольняють критерії відбору.*

Але приховані записи в будь-який момент можна відобразити на екрані.

Найзручнішим інструментом фільтрування є **Автофільтр**. Цей інструмент містить такі фільтри:

- **Перші 10** – для відображення перших 10 записів даних;
- **Сортування за зростанням (сортування за спаданням)** – відображає всі записи, впорядковані за зростанням (за спаданням);
- **Умова ...** – інструмент для відображення на екрані записів, які задовольняють певні критерії, що можна задати у діалоговому вікні **Користувацький автофільтр**. Тут можна задавати як прості, так і складені умови фільтрування.

Розглянемо застосування автофільтра на прикладі. Нехай у списку треба показати всіх осіб віком до 12 років.

✓ *Слід зробити так:*

- виокремити стовпці з даними;
- виконати команду меню **Дані** ⇒ **Фільтр**;
- після виконання команди праворуч у назвах виокремлених стовпців з'являться кнопки зі стрілками;
- натиснути на кнопку стовпця з назвою “Вік” та обрати з переліку можливих варіантів фільтрування **Фільтри чисел між ...** ;
- у діалоговому вікні **Користувацький автофільтр** ввести необхідні умови;

Розширений фільтр. Якщо потрібно виконати відбір даних за складених умов відбору, зручніше застосувати інструмент фільтрування “розширений фільтр”, який викликається з позиції меню **Додатково**. У діалоговому меню **Розширений фільтр**, яке з'явиться після натиснення, можна ввести потрібні умови фільтрування списку або копіювання результатів фільтрування до іншого діапазону комірок.

Умови можуть бути такими: кілька умов для одного стовпця даних; одна умова для кількох стовпців даних; одна умова для одного стовпця та інша для іншого; набір умов для кількох стовпців; пошук за умовою у вигляді формули.

✓ *Рекомендації зі створення списку, до якого застосовуватимуться функції фільтрування та сортування.*




Для використання цих функцій таблиця має мати певну структуру, тому дані слід уводити, керуючись рекомендаціями, поданими в табл. 7.2.

Організація структур даних


№ з,п	Особливості стовпця	Форма подання даних
1	2	3
1	Подібні елементи розташовувати в одному стовпці	Упорядковувати дані слід таким чином, щоб в усіх рядках подібні елементи містилися в одному й тому самому стовпці
2	Діапазон розташовується окремо	Слід залишити принаймні один порожній стовпець і один порожній рядок між діапазоном споріднених даних і рештою даних на аркуші. У цьому разі Microsoft Excel ефективніше виявляє та виділяє діапазон під час сортування, фільтрування або вставлення автоматичних проміжних підсумків
3	Важливі дані мають міститися над діапазоном або під ним	Слід уникати розташування важливих даних ліворуч або праворуч від діапазону, тому що при фільтруванні діапазону вони можуть стати прихованими
4	Рядки та стовпці мають бути відображені	Перш ніж вносити зміни до діапазону, слід переконатися, що приховані рядки та стовпці відображено. Якщо в діапазоні не відображаються деякі рядки або стовпці, можливе помилкове видалення даних
Формат даних		
5	Використовувати форматovanі підписи стовпців	Рекомендується створювати в першому рядку діапазону даних заголовки стовпців. Ці підписи використовуються в Microsoft Excel для створення звітів, пошуку й упорядкування даних. Для заголовків стовпців слід використовувати шрифт, вирівнювання, тло, межу або регістр, відмінні від параметрів форматування, які мають дані в діапазоні. Перш ніж увести підписи стовпців, слід установити для комірок текстовий формат
6	Використовувати межі комірок	Якщо потрібно відокремити підписи від даних, краще це виконати за допомогою меж, а не вставляти порожні рядки або риски між підписами та рядками даних
7	Уникати використання пустих рядків і стовпців	Для того щоб ЕТ краще знаходила й виокремлювала діапазони споріднених даних, не слід вставляти в діапазон порожні рядки або стовпці
8	Не застосовувати пробіли перед даними або після них	Зайві пробіли на початку або в кінці комірки заважають правильному сортуванню або пошуку. Для зсування тексту в комірці слід використовувати відступи

1	2	3
9	Розширення форматів даних і формул	Якщо в кінець діапазону даних додаються нові рядки, на них автоматично поширюються наявне форматування та формули. Для того щоб відбувалося розширення формату, три з п'яти попередніх комірок мають мати однаковий формат. Для того щоб відбувалося розширення формул, усі попередні формули мають бути однакові

Перевіряємо себе

1. Чому при застосуванні фільтрів і сортуванні дані мають бути введені зі суворим дотриманням певних правил? ✦
2. Чим відрізняється сортування від фільтрування? ✦
3.   У латинському алфавіті й українському алфавіті є літери, однакові на вигляд. Що буде зі списком адрес шкіл України після сортування, якщо у назві міста Львів літера і набиратиметься інколи як латинська, а інколи – як українська? Перевірте. ✦
4. До яких наслідків може призвести порушення структури, рекомендованої в табл. 7.2, п. 1? ✦
5. До яких наслідків може призвести порушення структури, рекомендованої в табл. 7.2, п. 8? ✦
6. Які інструменти для фільтрування даних вам відомі? ▲
7. З яких етапів складається створення розширеного фільтра? ▲
8. Які можливості надає розширений фільтр? ▲
9. Наведіть приклади доцільності використання автофільтра, фільтра користувача, розширеного фільтра. ✦
10.  Що спільного між розділами var і type програми мовою Паскаль і табл. 7.2? ★

Виконуємо

1.  З веб-порталу фірми КРІ-Сервіс завантажте прайс-лист http://web-portal.kpIService.com.ua/kpi_new/price/price_opt.zip.
2. Відкрийте аркуш “Вспомогательное оборудование” і встановіть автофільтр (Дані ⇒ Фільтр ⇒ Автофільтр).
3. За допомогою встановленого фільтра виберіть з прайс-листа джерела безперебійного живлення back pro (кнопка списку поля Тип).
4. Упорядкуйте результат фільтрування за дрібнооптовою ціною, а потім – роздрібною ціною в порядку зростання (Дані ⇒ Сортування).

7.8. Підсумки і проміжні підсумки



Сортування і фільтрування даних. Типізація даних у мові програмування і в табличному процесорі. Об'єкти та їх властивості. Формули в табличному процесорі, аргументами яких є масиви.



Опрацювання даних як інформаційний процес. База даних. Фільтри і формули, аргументом яких є масив, як спосіб отримання нової інформації.

Якщо вже створено електронну таблицю з дотриманням правил, викладених у табл. 7.2, то це означає, що ми маємо новий об'єкт, який можна назвати **базою даних (БД)**.



База даних – певним чином структурована сукупність даних, отриманих для певного класу об'єктів.

Найпростішим способом подання даних для описання одного об'єкта певного класу є **запис**, у якому для кожної властивості виокремлено **поле** з наперед визначеним форматом подання даних. У електронній таблиці таким **записом** може слугувати **рядок**, а **полем** – **комірка**.

Застосовуючи фільтр до даних, розміщених у комірках ЕТ, які заповнено за правилами, викладеними в табл. 7.2, тобто до бази даних, ми здійснюємо **запит** до бази даних.

Запит до БД – інформаційний процес, результатом якого є отримання нової інформації.

Це не означає, що інформація “виникає з нічого” – інформація виникає як результат спільного опрацювання запиту і даних. Частина інформації, яку ми отримуємо в результаті запиту, вже міститься у БД, частину подаємо тоді, коли формулюємо умови запиту (табл. 7.3).

Наприклад, у БД містяться дані щодо поживності основних продуктів, які ми можемо споживати, їх вартість. Якщо ми робимо запит: *“Показати, які продукти й скільки мені потрібно взяти з собою в похід на 10 днів, за умови, що передбачаються витрати енергії 3500 ккал щодня, загальна маса продуктів не має перевищувати 3 кг, мінімальна порція кожного продукту має бути не менше 50 г, продукти мають бути досить різноманітними, а загальна вартість їх не має перевищувати 2000 грн”*, то до інформації з БД застосовуємо досить складне правило, сформульоване з урахуванням декількох вимог. Отже, запит теж містить певну інформацію.

Можна створити список із проміжними підсумками, використовуючи команду **Підсумки** в меню **Дані**. Але якщо список із проміжними підсумками вже створений, його можна модифікувати, редагуючи формулу з функцією **ПРОМІЖНІ.ПІДСУМКИ**.

У табличному процесорі серед інших є функція **Проміжні підсумки**, яка повертає проміжний підсумок у список або базу даних.

Синтаксис:

ПРОМІЖНІ.ПІДСУМКИ (номер_функції; посилання1; посилання2;...)

Номер_функції – число від 1 до 11, яке вказує, яку функцію використовувати для обчислення підсумків у списку.

Посилання1; Посилання 2; – від 1 до 29 інтервалів або посилань, для яких підводяться підсумки.

✓ Якщо в таблиці вже є формули підведення підсумків для аргументів посилання 1; посилання 2; ... (так звані вкладені підсумки), то ці вкладені підсумки ігноруються, щоб уникнути подвійного підсумовування.

✓ Функція ПРОМІЖНІ.ПІДСУМКИ ігнорує всі приховані рядки, які стають такими в результаті фільтрування списку. Це важливо, оскільки стає можливим підвести підсумки тільки для тих даних, що містяться в рядках, які відтворюються після фільтрування списку.

Таблиця 7.3

Деякі функції, які застосовуються для опрацювання даних

№ з/п	Функція	Опис функції
1	СРЗНАЧ (AVERAGE)	Повертає середнє арифметичне своїх аргументів
2	СЧЁТ (COUNT)	Підраховує кількість чисел у списку аргументів. Використовується для отримання кількості числових значень у інтервалах або масивах значень
3	СЧЁТЗ	Підраховує кількість непорожніх значень у списку аргументів. Використовується для підрахунку кількості комірок з даними в інтервалі або масиві
4	МАКС	Повертає найбільше значення з набору значень
5	МИН	Повертає найменше значення з набору значень
6	ПРОИЗВЕД	Перемножує числа, вказані як аргументи, і повертає їх добуток
7	СУММ	Додає всі числа в інтервалі комірок

Те, що ПРОМІЖНІ.ПІДСУМКИ, призначені для діапазону комірок, за наявності сформованого для цього ж діапазону комірок фільтра, обчислюються тільки для комірок, значення яких “пройшли фільтр”, дуже важливо. Це дає змогу сформулювати запит до БД, який можна описати так: “Покажи результат опрацювання значень у комірках стовця, розташованих у рядках, дані в яких відповідають певним об’єктам”.

Приклад. Нехай дані щодо об'єктів класу “зернові культури” розташовані в комірках (полях) рядків, які відповідають стовпцям із назвами “Назва культури”, “Врожайність, центнерів з гектара”, “Країна”, “Рік”. Тоді за запитом “Покажи середню врожайність для “Назва культури” = “Кукурудза” і “Країна” = “США”” отримаємо певне значення. Для такого ж запиту, але для “Назва культури” = “ячмінь” отримаємо інше значення. Змінюючи фільтр за країною, можемо отримати інформацію щодо врожайності зернових у різних країнах протягом кількох років. Порівнюючи отримані значення між собою, можна дійти висновку щодо доцільності вирощування певної культури в конкретній країні.

Подібним чином можна виконати й інші дослідження, застосовуючи спільно засоби фільтрування даних і опрацювання результатів з використанням функції ПРОМІЖНІ.ПІДСУМКИ.

Якщо заздалегідь запланувати важливі для аналізу даних запити, то доцільно створити **Зведену таблицю**. У цій таблиці вже мають бути сформовані (копіюванням значень або зв'язуванням комірок) рядки, комірки (поля) яких містять дані, що описують певну властивість об'єкта. Дані мають бути занесені з дотриманням вимог, поданих у табл. 7.2.

✓ Для створення зведеної таблиці виокремте будь-яку комірку в діапазоні ⇒ у меню **Дані** оберіть команду **Зведена таблиця** ⇒ натисніть кнопку **Далі**, щоб погодитися з налагодженнями, запропонованими на першому кроці ⇒ перевірте, чи правильно зазначено діапазон даних ⇒ натисніть кнопку **Далі**; ⇒ натисніть кнопку **Макет**. (У MS Excel 2007 перехід до макета відбувається автоматично) ⇒ у діалоговому вікні **Макет** перетягніть кнопку **Account** (Рахунок) зі списку праворуч до області рядків ⇒ перетягніть кнопку **Amount** (Сума) зі списку праворуч до області **Дані**; ⇒ натисніть кнопку **ОК** ⇒ вкажіть, де потрібно розмістити результат – на новому аркуші або в певному розділі на створеному аркуші, і натисніть кнопку **Готово**.

За даними **Зведеної таблиці** можна будувати діаграми, які відображатимуть отримані дані. Ці діаграми називатимуться **Зведеними діаграмами**.

Перевіряємо себе

1. Як відображаються властивості об'єкта в електронній таблиці? ✦
2. У яких випадках доцільно застосовувати попереднє сортування (фільтр) до даних, які є аргументами проміжних підсумків? ✦

3. Яку інформацію дасть застосування до даних обчислення проміжних підсумків функція з номером 3? Чи можна такий запит використати задля контролю правильності формування зведеної таблиці? ★
4. Що потрібно зробити насамперед, щоб створити зведену таблицю? ▲
5. Чому відповідають і що містять поля зведеної таблиці? ★
6. Чому потрібно строго дотримуватися однаковості форматів даних, які є аргументами проміжних підсумків? ▲

Виконуємо

Створіть таблицю, яку можна вважати базою даних щодо успішності у навчанні учнів класу.

Таблиця має забезпечувати виконання запитів:

“Який середній бал учнів класу з інформатики, математики, фізики тощо?”

“Скільки учнів мають середній бал з предмета, нижчий за середній із усіх предметів?” та подібних.

7.9. Надання значень параметрам сторінки.

Друківання електронної таблиці

У Microsoft Excel можна створити **Стандартний шаблон книги** і **Стандартний шаблон аркуша**.

Стандартний шаблон книги – Книга.xlt створюється для зміни використовуваного за замовчуванням формату новостворюваних порожніх книг. Надалі цей шаблон буде використовуватись у Microsoft Excel для створення порожньої книги при запуску або для створення книги без зазначення шаблону.

Стандартний шаблон аркуша – Аркуш.xlt створюється для зміни використовуваного за замовчуванням формату новостворюваних порожніх аркушів. Цей шаблон буде використовуватися в Microsoft Excel для додавання до книги нового аркуша.

Шаблон книги: книга, яка містить аркуші, стандартний текст (наприклад, заголовки сторінок, підписи стовпців і рядків), формули, макроси та інші елементи форматування, що мають бути у створюваних на основі цього шаблону книгах.

Шаблон аркуша: книга з одним аркушем, на якому застосовано елементи форматування, стилі, текст та інші дані, що мають бути на всіх нових аркушах такого самого типу.

Створення шаблону:

у меню **Файл** вибрати команду **Зберегти як** ⇒ у списку **Тип файла** вибрати пункт **Шаблон** ⇒ у рядку **Ім'я файла** набрати ім'я (при створенні шаблону книги для використання за замовчуванням, ввести текст **Книга**, при створенні спеціального шаблону книги, ввести довільне припустиме ім'я файла; при створенні шаблону аркуша для

використання за замовчуванням, ввести текст **Аркуш**, при створенні спеціального шаблону аркуша ввести довільне припустиме ім'я файла) ⇒ натиснути **Зберегти**.

У полі Папка можна вказати папку, в якій має бути збережений шаблон. Якщо створюється шаблон книги або аркуша для використання за замовчуванням, то йому відповідно надається ім'я **Книга.xlt** або **Аркуш.xlt** і він зберігається в папці **XLStart**, яка завантажується за замовчуванням та відображається при запуску Microsoft Excel. Microsoft Excel використовуватиме цей шаблон для створення книги або для вставки нових аркушів. Звичайне розташування папки **XLStart**:

C:\Program Files\Microsoft Office\Office11\XLStart

Для створення спеціального шаблону книги або аркуша необхідно вибрати папку **Шаблони**, звичайне розташування якої:

C:\Documents and Settings\ім'я користувача\Application Data\Microsoft\Templates

Щоб зображення першої сторінки шаблону було показано в полі **Перегляд** діалогового вікна **Шаблони (Загальні шаблони, область завдань Створити книгу)**, необхідно:

✓ *вибрати у меню **Файл** команду **Властивості** ⇒ відкрити вкладку **Документ** ⇒ установити прапорець **Створити малюнок для попереднього перегляду** (рис. 7.23).*



Рис. 7.23. Вікно Властивості робочої книги MS Excel

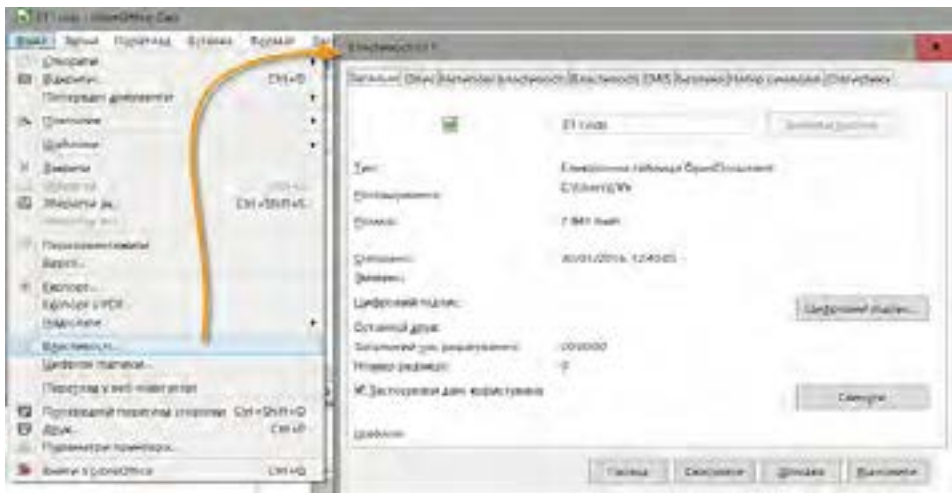


Рис. 7.24. Вікно Властивості робочої книги Libre Office Calc

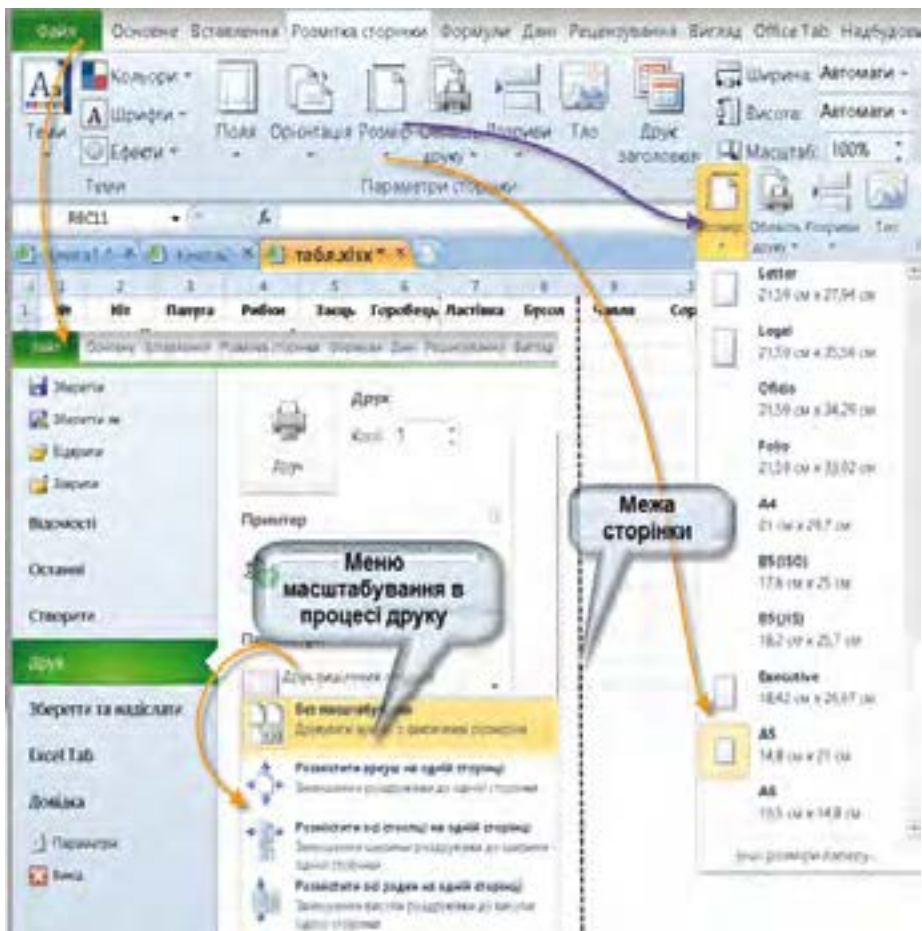


Рис. 7.25. Установлення параметрів сторінки і друкування ET Microsoft Excel 2010

Як бачимо з рис. 7.23, у цьому вікні можна також вказати ще низку додаткових відомостей про документ: тему роботи, керівника роботи, назву установи, в якій виконується робота, і вказати зв'язок з іншими документами (гіперпосилання).





Подібним чином можна змінювати властивості ЕТ й у Microsoft Excel 2010 і 2013 (рис. 7.24).

Якщо потрібно підготувати електронну таблицю для виведення на друк, слід виконати дії з розмітки сторінки, подібні до тих, які з цією метою виконувалися в текстовому процесорі.

Слід зауважити, що ЕТ Excel, на відміну від текстового процесора Word, для увімкнення відображення у вигляді макета сторінки, вимагає обов'язкового встановлення в ОС хоча б одного принтера.

Особливість друкування таблиць полягає в тому, що вони не завжди можуть бути розміщені на стандартному аркуші. Тому при друкуванні таблиць передбачено можливість додаткового автоматичного масштабування (рис. 7.25). Воно застосовується в тому випадку, якщо на аркуш не поміщається невелика частина таблиці (кілька стовпців або рядків).

Перевіряємо себе

-  Як має бути оформлена таблиця – джерело даних, щоб за нею можна було створити зведену таблицю? ▲
-  Для чого використовується Макет зведеної таблиці? ✦
-  Як потрібно змінити зведену таблицю, якщо: ✦
 - до початкового діапазону були додані нові записи?
 - поточні початкові дані були змінені?
 - не підходить структура зведеної таблиці?
 - потрібні інші функції для обчислень?
 - деякі дані полів мають бути приховані?
- Як побудувати діаграму на основі зведеної таблиці? ✦
-  Чим відрізняється створення шаблону книги і шаблону аркуша? ▲
- У яких випадках доцільно застосувати автоматичне масштабування таблиці? ✦

Виконуємо

- Створіть таблицю – список друзів, куди запишіть їхні номери телефонів та адреси електронних скриньок (поля: Номер; Прізвище І.П.; Телефон; Ел. пошта).

2. Відформатуйте заголовок таблиці: виділіть блок комірок A1:D1 і введіть команду контекстного меню **Формат комірок**; на вкладці **Вирівнювання** виберіть опції: по горизонталі – по центру виділення, по вертикалі – по верхньому краю; виділіть текст жирним шрифтом.

3. Викличте контекстне меню та виберіть команду **Формат клітинок**; на вкладці **Вирівнювання** задайте параметри: по горизонталі – за значенням, по вертикалі – по нижньому краю, переносити за словами – встановити прапорець, орієнтація – горизонтальний текст (за замовчуванням). Скопіюйте форматування на всі комірки, які передбачається заповнити.

4. Надайте кожному прізвищу порядковий номер, використовуючи автозаповнення. Введіть дані.

5. Відобразіть рамку і внутрішні лінії таблиці: виділіть блок комірок із даними в правому нижньому куті таблиці.

6. Відформатуйте таблицю таким чином, щоб вона повністю розмістилася на аркуші формату А5 (альбомної орієнтації). Надрукуйте аркуш на мережному принтері (або на віртуальному принтері, який створює документ формату *.pdf).

7. Закрийте ET.

**Практична
робота № 14**

Тема: Розв’язування задач на обчислення

Мета: Набути практичних навичок створення електронних таблиць

Завдання

1. У таблиці подано дані про площу і населення деяких країн Європи. Ввести у відповідні клітинки формули для обчислення:

1) загальної площі та загальної кількості населення; ▲

2) густоти населення в кожній країні (осіб/км²); ▲

3) частки, яку становить населення кожної з цих країн щодо загальної кількості населення в усіх країнах. ★

Країна	Площа, км ²	Населення, млн осіб	Густота населення	Відсоток
Україна	603700	46,3		
Франція	547030	67,3		
Іспанія	504782	40,5		
Швеція	449964	9,1		

2. Створити таблицю для опрацювання результатів вимірювання густини речовини методом обмірювання і зважування. ★

3. Створити таблицю для опрацювання результатів вимірювання швидкості звуку в повітрі (на відстані S від спостерігача стріляють із гармати, спостерігач фіксує час між спалахом і звуком). ★

Практична робота № 15

Тема:	Використання математичних, логічних і статистичних функцій табличного процесора. Умове форматування
Мета:	Набути практичних навичок створення електронних таблиць із використанням математичних, логічних і статистичних функцій табличного процесора

Завдання

З використанням електронної таблиці здійснити опрацювання даних за допомогою статистичних функцій.

1. Дано відомості про учнів класу, що включають усі оцінки, отримані кожним учнем протягом одного семестру. Підрахувати кількість кожної з оцінок (від 1 до 12 балів), знайти середній бал кожного учня і середній бал усієї групи. Створити діаграму, яка ілюструє відсоткове співвідношення оцінок у групі. ✦

Підказка: використайте функцію ЧАСТОТА (FREQUENCY) або функцію СЧЕТЕСЛИ (COUNTIF)

2. Визначити моду та медіану набору даних.

Практична робота № 16

Тема:	Упорядкування даних у таблицях. Автоматичні та розширені фільтри
Мета:	Набути практичних навичок створення електронних таблиць із використанням упорядкування даних, автоматичних і розширених фільтрів, проміжних висновків

Створити таблицю “Річки Європи”, використовуючи такі дані: довжина (км) і площа басейну (тис. кв. км): Волга – 3688 і 1350; Дніпро – 2285 і 504000; Дністер – 1 362 і 72 100; Дунай – 2850 і 817; Рейн – 1330 і 224; Ельба – 1150 і 148; Вісла – 1090 і 198; Луара – 1020 і 120; Урал – 2530 і 220; Дон – 1870 і 422; Сена – 780 і 79; Темза – 340 і 15. ✦

Упорядкувати таблицю за назвами річок. Знайти переклад назви кожної річки англійською, ввести в рядок із назвою річки додаткові комірки (додати стовпець “Англ. назва”), повторити впорядкування за алфавітом по цьому стовпцю. ★

Додати до рядка кожної річки країну (країни), через які вона протікає, додавши в таблицю додаткові стовпці. ✦

Визначити найдовшу і найкоротшу річки. ▲

Підрахувати сумарну та середню площу басейнів річок, сумарну й середню протяжність великих річок Європи (виконати з використанням проміжних підсумків для сумарних значень). ✦

З використанням умовного форматування виокремити назви річок, довжина яких більша за середню. Якщо для країни в таблиці вказано більш ніж одну річку, підрахувати їх загальну довжину з використанням фільтра і проміжних висновків. ★



СЛОВНИЧОК

База даних – певним чином структурована сукупність даних, отриманих для певного класу об'єктів.

Вісь значень – вісь, мітки якої розташовані пропорційно до їх значень, координатна вісь.

Вісь категорій – вісь, мітки якої розташовані рівномірно, незалежно від значень, які їм приписуються.

Запит – відомості щодо відбору об'єкта (об'єктів), містять значення властивостей, за якими *об'єкт* може бути зарахований до певної групи.

Зведена таблиця – таблиця, у якій зібрано підсумкові значення, одержані за спеціальними формулами з великих масивів даних.

Легенда – елемент діаграми, в якому подано позначення рядів даних.

Маркер даних – смуга, область, точка, сектор або інший об'єкт на діаграмі, який представляє одне значення комірки аркуша.

Область діаграми – прямокутна область, на якій відображаються елементи діаграми.

Область побудови діаграми – частина області діаграми, обмежена осями.

Основні лінії сітки – лінії, які можна додати до діаграми для поліпшення сприйняття й оцінювання відображуваних даних.

Ряд даних – пов'язані одна з одною точки даних, нанесені на діаграму.

Фільтр – засіб для відбору даних за певною ознакою (ознаками).

Шаблон аркуша – книга з одним аркушем, на якому застосовано елементи форматування, стилі, розташовано текст та інші дані, які мають бути у створюваних на основі цього шаблону аркушах.

Шаблон книги – книга, яка містить аркуші, стандартний текст (наприклад, заголовки сторінок, підписи стовпців і рядків), формули, макроси та інші елементи форматування, які мають бути у створюваних на основі цього шаблону книгах.

Шкала – мітки на осі значень.



РОЗДІЛ 8. РОЗВ'ЯЗУВАННЯ КОМПЕТЕНТНІСНИХ ЗАДАЧ

1. Створити таблицю “Озера Європи”, використовуючи такі дані щодо площі (кв. км) і найбільшої глибини (м): Ладозьке – 17700 і 225; Онезьке – 9510 і 110; Каспійське море – 371000 і 995; Венерн – 5550 і 100; Чудське з Псковським – 3560 і 14; Балатон – 591 і 11; Женевське – 581 і 310; Веттерн – 1900 і 119; Боденське – 538 і 252; Мелоре – 1140 і 64. Визначити найбільше і найменше за площею озеро, найглибше і наймілкіше озеро. Додати країну (країни), в яких розташовані озеро. ★

2. Розробіть код для відображення на формі екрана монітора, зображеного на рис. 8.1. Порівняйте розроблений код з кодом, зображеним на рис. 8.2. Вкажіть переваги та недоліки кожного з них.



Рис. 8.1. Монитор

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  with Canvas do begin
    Brush.Color:=clGray;
    Rectangle (10,10,305,205);
    Brush.Color:=clBlue;
    Rectangle (25,25,290,195);
    Brush.Color:=clGray;
    Pie (100,180,200,225, 100,205,200,205);
    TextOut (30,11, 'FLATRON');
    TextOut (150, 100, 'Winodws 10');
    TextOut (30,175, 'ПУСК');
  end;
end;
end.
```

Рис. 8.2. Код для зображення екрана монітора



Рис. 8.3. Блок-схема алгоритму для формулювання умови задачі

3. На формі зображене коло радіусом r із центром у точці з координатами $X[0], Y[0]$. Розробіть проект визначення, чи знаходиться точка з координатами $X[1], Y[1]$ всередині цього кола.

4. У банк покладено s гривень під k відсотків річних. Розробити у візуальному режимі середовища Lazarus проект визначення, через скільки років накопичена сума становитиме більше p гривень, якщо протягом цього часу не знімати ні відсотків, ні тіла вкладу.

5. Проаналізувати алгоритм, блок-схема якого зображена на рис. 8.3. Сформулювати можливу умову задачі, яку він реалізує. Розробити у візуальному режимі Lazarus проект для реалізації цього алгоритму.

6. Для випробування нового автомобіля вирішено першого дня проїхати s км, а кожного наступного дня збільшувати пробіг на p відсотків порівняно з попереднім днем. Розробити проект визначення, через скільки днів пробіг досягне z км.

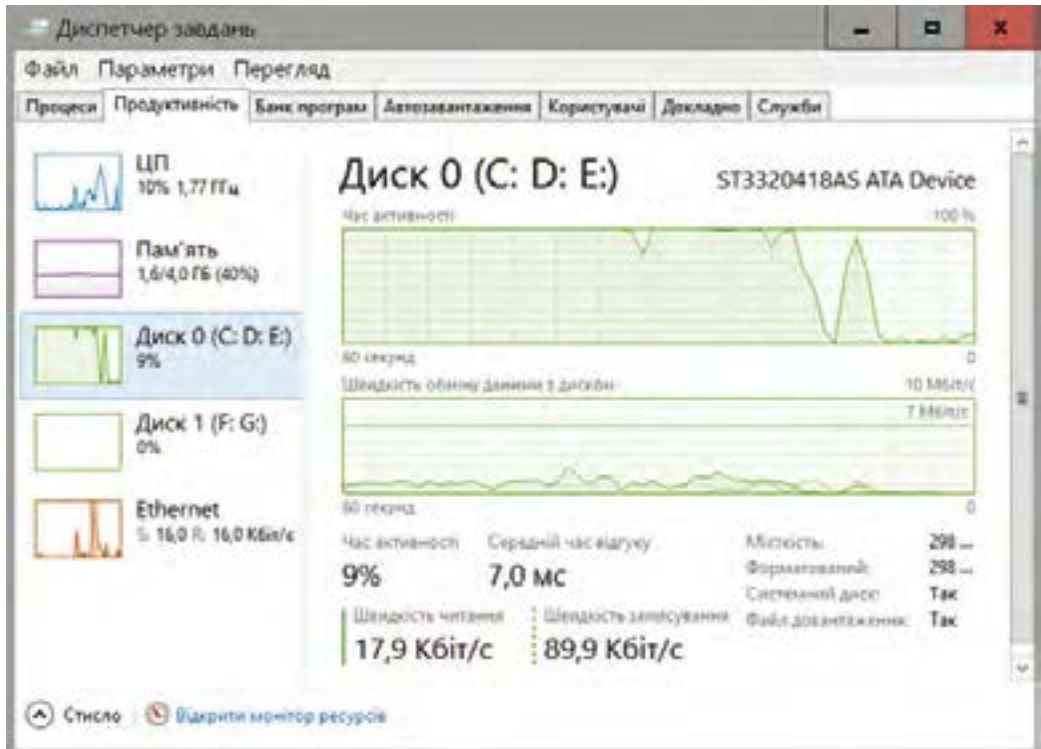


РОЗДІЛ 9. ВИКОНАННЯ ІНДИВІДУАЛЬНИХ І ГРУПОВИХ НАВЧАЛЬНИХ ПРОЕКТІВ

Орієнтовні задачі

1. Проаналізувавши рисунок і знаючи, що копія екрана знята на початку сеансу роботи невдовзі після завантаження ОС, спробуйте

- 1) описати комп'ютер, на якому воно отримане;
- 2) визначити, що виконувалося протягом останніх 60 секунд;
- 3) обґрунтувати зроблені припущення. ★



Екранна копія показів Диспетчера завдань.

2. П'ятеро друзів подорожують трьома видами транспорту: поїздом, літаком і теплоходом. Василь проплив 250 км теплоходом, проїхав 180 км поїздом і пролетів 1200 км літаком. Микола проплив теплоходом 260 км, проїхав поїздом 250 км і пролетів літаком 1160 км. Сергій пролетів літаком 1200 км, проїхав поїздом 110 км і проплив теплоходом 145 км. Марія проїхала поїздом 150 км, пролетіла літаком 1800 км і пропливла теплоходом 110 км. Світлана пролетіла літаком 10200 км, проїхала поїздом 80 км, і пропливла теплоходом 100 км. ★

Побудувати на основі наведених даних електронну таблицю. Додати до таблиці стовпець, у якому відобразатиметься загальна кількість кілометрів, яку проїхав кожен із друзів. Обчислити загальну кількість

кілометрів, яку хлопці проїхали поїздом, пролетіли літаком і пропливли теплоходом (для кожного виду транспорту окремо). Обчислити загальну кількість кілометрів, яку дівчата проїхали поїздом, пролетіли літаком і пропливли теплоходом (для кожного виду транспорту окремо). Обчислити сумарну кількість кілометрів, яку проїхали всі друзі. Визначити найбільшу і найменшу кількість кілометрів, пройдених друзями (для кожного виду транспорту). Визначити середню відстань, подолану кожним видом транспорту.

3. Опишіть словами послідовність дій, які потрібно виконати для розв'язання такої задачі: “Для випікання однієї житньої хлібини потрібно 300 г житнього борошна, 200 г пшеничного борошна, 10 г солі, 30 г цукру, 5 г дріжджів; для випікання однієї білої хлібини потрібно 600 г пшеничного борошна, 10 г солі, 60 г цукру, 5 г дріжджів. Скільки потрібно кожної зі складових для випікання 10 житніх і 20 білих хлібин?” Перевірте обчислення, придумайте декілька подібних задач і запишіть у зошит їх умови. Створіть ЕТ для виконання обчислень. ★

Орієнтовні теми навчальних проектів із предметної галузі навчального курсу “Інформатика”

Промінь, Дніпро, МІР, СМ1420, СМ1425, Пошук – хто і де їх створював.
Історія успіху Епл.

Катерина Логвинівна Ющенко – життєвий шлях і роль у історії кібернетики.

Проект “Союз” – “Аполлон”. Що в ньому було “made in Ukraine”?

Перші бортові ЕОМ ракетно-космічних комплексів – made in Ukraine.

Олександр Миколайович Щукар'єв – людина, що бачила майбутнє.

Контр-адмірал ВМС США Грейс Мюррей Хоппер – життєвий шлях і місце в історії кібернетики.

Микола Михайлович Амосов – кардіохірург, філософ, кібернетик.



ЗМІСТ

ЯК ПРАЦЮВАТИ З ЦІЄЮ КНИГОЮ.	4
РОЗДІЛ 1. КОДУВАННЯ ДАНИХ	5
1.1. Отримання, кодування і декодування даних	5
1.2. Текстові повідомлення та їх кодування	10
<i>Практична робота №1. Розв'язування задач на визначення довжини двійкового коду текстових даних.</i>	<i>15</i>
СЛОВНИЧОК	17
РОЗДІЛ 2. АПАРАТНО-ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРА	18
2.1. Історія засобів опрацювання інформаційних об'єктів	18
2.2. Види сучасних комп'ютерів та їх застосування	23
2.3. Архітектура комп'ютера	26
2.4. Мультимедійні пристрої введення та виведення	34
2.5. Класифікація та загальні характеристики програмного забезпечення	40
<i>Практична робота №2. Конфігурування комп'ютера під потреби користувача</i>	<i>50</i>
2.6. Службове програмне забезпечення	51
<i>Практична робота № 3. Архівування та розархівування даних.</i>	<i>62</i>
СЛОВНИЧОК	62
РОЗДІЛ 3. ОПРАЦЮВАННЯ ТЕКСТОВИХ ДОКУМЕНТІВ	64
3.1. Формати файлів текстових документів. Форматування символів і абзаців	64
3.2. Списки, таблиці, символи і формули	68
3.3. Створення та редагування графічних об'єктів у текстовому документі.	74
3.4. Сильове оформлення абзаців	76
3.5. Структура документа. Розділи. Колонтитули	81
3.6. Посилання. Автоматизоване створення змісту та покажчиків	84
3.7. Опрацювання складного текстового документа	86
<i>Практична робота № 4. Створення текстового документа, що містить об'єкти різних типів</i>	<i>89</i>
<i>Практична робота № 5. Структура документа. Автоматизоване створення змісту і покажчиків</i>	<i>90</i>
СЛОВНИЧОК	90
РОЗДІЛ 4. СТВОРЕННЯ ТА ОПРАЦЮВАННЯ ОБ'ЄКТІВ МУЛЬТИМЕДІЯ	92
4.1. Формати аудіо- та відеофайлів. Конвертація аудіо- та відеофайлів	92
4.2. Програмне забезпечення для опрацювання об'єктів мультимедія.	98

4.3. Створення й опублікування мультимедія	100
<i>Практична робота № 6. Створення відеокліпу.</i>	
Додавання відеоефектів, налаштування часових параметрів аудіо- та відеоряду	103
<i>Практична робота № 7. Розміщення аудіо- та відеоматеріалів у Інтернеті</i>	103
СЛОВНИЧОК	105
РОЗДІЛ 5. ОСНОВИ ПОДІЙНО- ТА ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ.	106
5.1. Загальний порядок розв'язування задач на комп'ютері	106
5.1.1. Підготовчий етап	106
5.1.2. Етап виконання	109
5.2. Мови програмування	111
5.3. Основні відомості про мову Free Pascal і середовища програмування	114
5.4. Загальні відомості про середовище візуального програмування Lazarus	117
5.5. Основи візуального програмування в середовищі Lazarus	125
5.6. Створення проекту в режимі візуального програмування середовища Lazarus	131
5.7. Розроблення проекту з простим кодом	136
<i>Практична робота № 8. Створення об'єктно-орієнтованої програми, що відображає вікно повідомлення</i>	140
<i>Практична робота № 9. Створення програми з кнопками та написами</i>	141
СЛОВНИЧОК	141
РОЗДІЛ 6. АЛГОРИТМИ РОБОТИ З ОБ'ЄКТАМИ ТА ВЕЛИЧИНАМИ.	142
6.1. Консольний режим роботи середовища Lazarus	142
6.2. Дані, змінні, константи	146
6.3. Уведення та виведення даних	154
6.3.1. Уведення та виведення даних у консольному режимі	154
6.3.2. Уведення і виведення даних у віконному режимі.	157
6.4. Вирази, операнди та операції	161
6.4.1. Арифметичні вирази.	161
6.4.2. Логічні вирази.	162
6.4.3. Операції над рядковими даними	164
6.5. Розроблення програм для реалізації лінійних алгоритмів	167
<i>Практична робота № 10. Описання та виконання лінійних алгоритмів опрацювання величин у середовищі програмування</i>	170
<i>Практична робота № 11. Налаштування програми</i>	170
СЛОВНИЧОК	171
6.6. Розроблення програм для реалізації алгоритмів із розгалуженнями	171

6.7. Розроблення програм для реалізації алгоритмів із повтореннями для опрацювання величин	175
<i>Практична робота № 12. Описання та виконання алгоритмів з повтореннями та розгалуженнями для опрацювання величин</i>	<i>179</i>
6.8. Розроблення програм для реалізації алгоритмів із графічним відображенням даних	179
6.8.1. Створення графічних зображень у середовищі Lazarus	180
6.8.2. Інструменти й методи для створення графічних зображень	182
6.8.3. Графічні інструменти	182
6.8.4. Методи класу TCanvas	184
6.8.5. Відображення рисунків із зовнішніх файлів.	191
<i>Практична робота № 13. Описання та виконання алгоритмів з графічним відображенням даних</i>	<i>195</i>
СЛОВНИЧОК	195
РОЗДІЛ 7. ТЕХНОЛОГІЇ ОПРАЦЮВАННЯ ЧИСЛОВИХ ДАНИХ У СЕРЕДОВИЩІ ТАБЛИЧНОГО ПРОЦЕСОРА	196
7.1. Обчислення в середовищі табличного процесора	196
7.2. Призначення та використання основних математичних функцій табличного процесора	205
7.3. Призначення та використання основних логічних функцій табличного процесора	211
7.4. Призначення й використання основних статистичних функцій табличного процесора	214
7.5. Умовне форматування	221
7.6. Створення і налагодження діаграм	224
7.7. Упорядковування даних у таблицях. Автоматичні та розширені фільтри	232
7.8. Підсумки і проміжні підсумки	239
7.9. Надання значень параметрам сторінки. Друкування електронної таблиці	242
<i>Практична робота № 14. Розв’язування задач на обчислення</i>	<i>246</i>
<i>Практична робота № 15. Використання математичних, логічних і статистичних функцій табличного процесора. Умовне форматування.</i>	<i>247</i>
<i>Практична робота № 16. Упорядковування даних у таблицях. Автоматичні та розширені фільтри</i>	<i>247</i>
СЛОВНИЧОК	248
Розділ 8. РОЗВ’ЯЗУВАННЯ КОМПЕТЕНТІСНИХ ЗАДАЧ	249
Розділ 9. ВИКОНАННЯ ІНДИВІДУАЛЬНИХ І ГРУПОВИХ НАВЧАЛЬНИХ ПРОЕКТІВ	251

НАВЧАЛЬНЕ ВИДАННЯ

ГУРЖІЙ Андрій Миколайович
КАРТАШОВА Любов Андріївна
ЛАПІНСЬКИЙ Віталій Васильович
РУДЕНКО Віктор Дмитрович

ІНФОРМАТИКА

Підручник для 8 класу
загальноосвітніх навчальних закладів

Рекомендовано Міністерством освіти і науки України

Редактор *Л.В. Дячишин*
Художній редактор *І.Б. Шутурма*
Коректор *О.А. Тростянчин*

Макет, підбір та колажування ілюстрацій *В.В. Лапінський*
Обкладинка *О.В. Шингур*

Формат 70×100¹/₁₆.
Ум. друк. арк. 20,736. Обл.-вид. арк. 19,5.
Зам.

Державне підприємство “Всеукраїнське спеціалізоване видавництво “Світ”
79008 м. Львів, вул. Галицька, 21
Свідоцтво суб’єкта видавничої справи ДК № 4826 від 31.12.2014
www.svit.gov.ua
e-mail: office@svit.gov.ua
svit_vydav@ukr.net

Друк ТДВ «Патент»
88006 м. Ужгород, вул. Гагаріна, 101
Свідоцтво суб’єкта видавничої справи ДК № 4078 від 31.05.2011