

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНА МЕТАЛУРГІЙНА АКАДЕМІЯ УКРАЇНИ**

Г.Г. ШВАЧИЧ, О.А. ГУЛЯЄВА

**КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ
ТА ПРОГРАМУВАННЯ**

Затверджено на засіданні Вченої ради академії
як конспект лекцій. Протокол № 15 від 27.12.2010

Дніпропетровськ НМетАУ 2011

УДК 004(075.8)

Швачич Г.Г., Гуляєва О.А. Комп'ютерні технології та програмування: Конспект лекцій. – Дніпропетровськ: НМетАУ, 2011. – 78 с.

Розглянуто основні засоби й прийоми обчислень у середовищі Excel, а також методи програмування на мовах VBA, C, C++.

Призначений для студентів напряму 6.050202 – автоматизація та комп'ютерно-інтегровані технології.

Іл. 20. Бібліогр.: 7 найм.

Друкується за авторською редакцією.

Відповідальний за випуск Г.Г. Швачич, канд. техн. наук, проф.

Рецензенти: О.Ю. Потап, канд. техн. наук, доц. (НМетАУ)
О.Г. Холод, канд. техн. наук, доц. (Дніпропетровський університет економіки та права ім. А. Нобеля)

© Національна металургійна академія
України, 2011

© Г.Г. Швачич, О.А. Гуляєва, 2011

Зміст

| | |
|---|----|
| Тема 1. Комп'ютерні технології..... | 4 |
| Тема 2. Табличний процесор Excel..... | 6 |
| Тема 3. Excel. Організація обчислень..... | 14 |
| Тема 4. Excel. Обчислювальний процес, що розгалужується..... | 16 |
| Тема 5. Excel. Одномірні й двомірні масиви..... | 19 |
| Тема 6. Макроси | 23 |
| Тема 7. Середовище VBA (Visual Basic for Application)..... | 27 |
| Тема 8. Основні конструкції мови VBA | 34 |
| Тема 9. Середовище Borland C++ Builder | 42 |
| Тема 10. Елементи мови C++. Створення консольного додатка. Лінійний обчислювальний процес..... | 44 |
| Тема 11. Windows додатки у графічному середовищі. Операції C++ | 50 |
| Тема 12. Обчислювальний процес, що розгалужується..... | 55 |
| Тема 13. Операції над двійковим кодом..... | 60 |
| Тема 14. Циклічний обчислювальний процес. Генерація випадкових чисел. Одномірні масиви..... | 63 |
| Тема 15. Двомірні масиви. Алгоритми реалізації..... | 71 |
| Тема 16. Функції користувача в C++. Рекурсивні функції | 73 |
| Література..... | 77 |

Тема 1. Комп'ютерні технології

1.1. Дані

Кодування даних двійковим кодом

Завдяки кодуванню комп'ютер може обробляти інформацію різного роду. В обчислювальній техніці існує двійкове кодування, тобто дані представляються за допомогою двох знаків 0 і 1.

Біт - двійковий розряд, найменша порція інформації. Одним бітом може бути представлено два поняття 0 або 1 (так чи ні).

За допомогою 2-х бітів може бути представлено 4 різні поняття:

00 01 10 11 (тобто 2^2).

С допомогою 3-х бітів може бути представлено 8 різних понять (2^3) і т.д.

При збільшенні на 1 одиницю кількості розрядів ми в 2 рази збільшуємо кількість значень, які можуть бути виражені в даній системі.

Загальна формула має вигляд:

$N=2^m$, де N – кількість незалежних значень, що кодуються;
 m - розрядність процесора.

Перші ПК були 8-розрядними. Потім з'явилися 16, 32, 64, 128-розрядні процесори і як наслідок зросли можливості ПК.

Для кодування цілих чисел від 0 до 255 досить мати 8 розрядів двійкового коду (8 біт). 16 біт дозволяють закодувати цілі числа від 0 до 65535 і т.д.

Кодування текстової інформації

Якщо кожному символу алфавіту зіставити певне ціле число (наприклад, порядковий номер), то за допомогою двійкового коду можна кодувати й текстову інформацію. Восьми двійкових розрядів досить для кодування 256 різних символів.

За основу кодування символів узятя кодова таблиця ASCII (American Standard Code for Information Interchange). Таблиця складається з 2-х частин - базової й розширеної.

Базова таблиця закріплює значення кодів від 0 до 127. Це керуючі коди, коди символів англійської й російської мов, розділові знаки, цифри, арифметичні дії.

Розширена частина ставиться до символів з номерами від 128 до 255, це національний алфавіт.

Пристрій уведення перетворює символи (текст) у двійкові коди. При виводі інформації виконується зворотне перетворення: двійковий код → звичайне представлення.

Універсальна система кодування текстових даних UNICODE заснована на 16-розрядному кодуванні, це дозволяє забезпечити унікальні коди для $2^{16} = 65536$ символів, цього досить для розміщення в одній таблиці символів більшості мов планети.

Одиниці виміру даних

В інформатиці різні типи даних мають універсальне двійкове подання. Найменшою одиницею виміру є байт. Більші одиниці утворюються додаванням префіксів кіло-, мега-, гіга-, тера-.

1 Кбайт=1024 байт

1 Мбайт=1024 Кбайт

1 Гбайт=1024 Мбайт

1 Тбайт=1024 Гбайт

1.2. Створення комплексних текстових документів

Текстовий процесор **Microsoft Word** має великі можливості при роботі з об'єктами нетекстової природи. Програма дозволяє створювати й вбудовувати геометричні фігури, художні заголовки, діаграми, формульні вирази. Розглянемо прийоми створення *комплексних текстових документів*, тобто документів, що містять об'єкти, вбудовані в текст.

Робота з формулами

У програмі Microsoft Word засобом для введення формул є редактор формул **Microsoft Equation 3.0**. Для запуску редактора формул служить команда пункту меню **Вставка** → **Об'єкт**. У діалоговому вікні, що відкрилося, варто вибрати пункт **Microsoft Equation 3.0** у списку *Тип об'єкта* на вкладці *Створення*.

Введена формула автоматично вставляється в текст як об'єкт. Далі її можна перемістити в будь-яке інше місце документа через буфер обміну (CTRL+X - вирізати; CTRL+V - вставити).

Для редагування формули безпосередньо в документі досить двічі клацнути на ній. При цьому автоматично відкривається вікно редактора формул.

Робота з таблицями

Текстовий процесор MS Word має потужні засоби створення таблиць. Три основних засоби створення таблиць:

- кнопка **Додати таблицю** на панелі інструментів *Стандартна*;
- діалогове вікно *Вставка таблиці* (*Таблиця*→*Вставити*→*Таблиця*);

- засіб креслення таблиць *Таблиці та границі* (*Таблиця* → *Накреслити таблицю*).

Форматування таблиць можна виконувати, використовуючи діалогове вікно *Властивості таблиці* (*Таблиця* → *Властивості таблиці*). Вкладки вікна дозволяють:

- задати метод вирівнювання таблиці щодо сторінки документа (вирівнювання);
- задати метод взаємодії таблиці з навколишнім текстом (обтікання);
- визначити варіант оформлення зовнішніх і внутрішніх рамок таблиці (границі й заливання) і ін.

Створення й редагування рисунків

Для роботи з векторними рисунками служить панель інструментів *Рисование* (**Вид** → **Панелі інструментів** → **Малювання**). Основним засобом цієї панелі, призначеним для створення найпростіших об'єктів, є список, що розкривається, **Автофігури**.

Для автофігур є особливий засіб створення текстового оформлення — текст може розміщатися у полі автофігури. Це виконується командою **Додати текст** у контекстному меню автофігури.

Створення графічних заголовків

Для створення художніх графічних написів, наприклад, заголовків, текстовий процесор MS Word має спеціальний програмний засіб **WordArt**. Доступ до нього здійснюється двома способами:

- через панель інструментів **WordArt** (*Вид* → *Панелі інструментів* → *WordArt*);
- за допомогою кнопки **Додати об'єкт WordArt** на панелі інструментів *Малювання*.

Тема 2. Табличний процесор Excel

2.1. Основні поняття

Програма Microsoft Excel призначена для роботи з таблицями даних. Документ Excel називається книгою. Книга являє собою набір аркушів, кожний з яких має назву, що відображається на ярлику аркуша.

Імена стовпців А, В, С, IV (усього 256). Імена рядків від 1 до 65536.

Файли робочих книг мають розширення .xls.

2.2. Уведення, редагування даних

Уведення даних здійснюється безпосередньо у поточну комірку. Комірка може містити дані 3-х типів: текст, число, формула. Текстові дані за замовчуванням вирівнюються по лівому краю, а числа - по правому.

Щоб змінити формат відображення даних у поточній комірці, використають команду пункту меню **Формат** → **Комірка**. Вкладки цього діалогового вікна дозволяють:

вибирати формат запису даних:

- кількість знаків після коми;
- вказівка грошової одиниці;
- спосіб запису дати й ін.;

задавати напрямок тексту й метод його вирівнювання;

визначати шрифт і накреслення символів;

задавати фонові кольори та ін.

2.3. Копіювання, переміщення вмісту комірки

Копіювання й переміщення в Excel можна здійснити методом перетаскування або через буфер обміну.

Метод перетаскування:

- навести покажчик миші на рамку поточної комірки, або виділеного діапазону (покажчик прийме вид стрілки);
- зручно використати *спеціальне перетягування* за допомогою натиснутої правої кнопки миші;
- при відпусканні кнопки з'являється меню, у якому можна вибрати конкретну операцію.

Застосування буфера обміну:

- виділити копійований (що вирізаємо) діапазон;
- помістити його у буфер, виконавши команду *Правка* → *Копіювати* (*Правка* → *Вирізати*);
- вставити дані в аркуш, виконавши команду *Правка* → *Вставити*.

2.4. Автозаповнення

У правому нижньому куті рамки поточної комірки перебуває *маркер заповнення*.

Приклад 2.1. Заповнити діапазон комірок A1:A10 непарними числами за допомогою *маркера заповнення*.

Порядок дій:

- увести у комірку A1 число 1, у комірку A2 число 3;
- виділити діапазон A1:A2 при натиснутій лівій кнопці миші;
- простягнути за маркер заповнення вниз до комірки A10.

Приклад 2.2. Заповнити стовпець В значеннями з діапазону [-2;5] із кроком 0,5 за допомогою засобів прогресії.

Порядок дій:

- у комірку В1 ввести значення -2 (початкове);
- виконати команду *Правка* → *Заповнити* → *Прогресія*;
- у вікні, що з'явилося, установити перемикачі:
 - розташувати по стовпцях;
 - арифметична;
 - увести крок = 0,5;
 - увести граничне значення 5, клацнути ОК.

Аналогічним чином може бути побудована геометрична прогресія.

2.5. Формули

Обчислення в таблицях Excel здійснюються за допомогою формул.

Ознака формули — знак =. Формула може містити числові константи, посилання на комірки, а також функції Excel, з'єднані знаками математичних операцій.

Пріоритет виконання операцій у виразі:

| | |
|----------------|------------------------|
| () | дії в дужках |
| sin(),..... | обчислення функцій |
| — | унарний мінус |
| % | узяття % |
| ^ | піднесення в степінь |
| * / | множення, ділення |
| + — | додавання, вирахування |
| & | об'єднання тексту |
| = < > >= <= | операції порівняння |

2.6. Стандартні функції Excel

Виклик стандартної функції:

ім'я функції (список аргументів)

Функція може бути записана вручну або за допомогою **Майстра функцій**.

Виклик Майстра функцій: пункт меню *Вставка* → *Функція*. Інакше, щигликом на кнопки **f_x** у рядку формул.

У списку *Категорія* вибирається категорія, до якої ставиться функція (якщо визначити категорію важко, використовують пункт **Повний алфавітний перелік**). У списку *Виберіть функцію* — конкретна функція.

В якості аргументів можуть використовуватися число, адреса комірки або довільний вираз.

Приклади запису формул:

$$e^{2,4} - \sin 3,5 = \exp(2,4) - \sin(3,5)$$

$$\sin 30^0 = \sin(30 * \text{ПИ}() / 180)$$

$$\sqrt[3]{\cos 14,5} + 45,6^2 = \cos(14,5)^{(1/3)} + 45,6^2$$

**2.7. Копіювання формули на блок комірок.
Відносні й абсолютні посилання**

Посилання — це адреса комірки, що використовується. За замовчуванням посилання на комірки у формулах розглядаються як *відносні*. Це означає, що при копіюванні формули адреси автоматично змінюються відповідно до відносного розташування вихідної комірки.

При *абсолютній адресації* посилання при копіюванні не змінюються. Для зміни способу адресації треба виділити посилання на комірку і натиснути клавішу F4.

- \$A1 посилання абсолютне по стовпцю;
- A\$1 посилання абсолютне по рядку;
- \$A\$1 абсолютне посилання.

Приклад 2.3. Знаходження суми значень двох діапазонів-стовпців.

Порядок дій:

- заповнити числами діапазони A1:A4 і B1:B4;
- у комірку C1 занести формулу =A1+B1, простягнути маркер заповнення на діапазон C2:C4
- у комірку D1 занести формулу =\$A\$1+B1, простягнути маркер заповнення на діапазон D2:D4.

| | A | B | C | D |
|---|---|---|--------|------------|
| 1 | 1 | 4 | =A1+B1 | =\$A\$1+B1 |
| 2 | 2 | 5 | =A2+B2 | =\$A\$1+B2 |
| 3 | 3 | 6 | =A3+B3 | =\$A\$1+B3 |
| 4 | 7 | 8 | =A4+B4 | =\$A\$1+B4 |

Рис. 2.1. Абсолютні й відносні посилання

У режимі перегляду формул (Сервіс → Параметри) бачимо, що в діапазоні C2:C4 адреси комірок настроюються на нове місце розташування, у діапазоні D2:D4 адреса комірки A1 не змінюється.

Приклад 2.4. Заданий масив $X = (3,6; -4; 8; 1,6; 2)$. Обчислити

a) $y = \cos x + \frac{\sqrt[3]{x}}{2}$

b) $s = \sum_{i=1}^5 x_i$

Результати обчислень

| | A | B | C |
|---|----------|----------|----------|
| 1 | x | y | s |
| 2 | 3,6 | -0,13045 | 11,2 |
| 3 | -4 | -1,44734 | |
| 4 | 8 | 0,8545 | |
| 5 | 1,6 | 0,555604 | |
| 6 | 2 | 0,213814 | |

B2← = COS(A2)+A2^(1/3)/2

32← = СУМ(A2:A6)

Рис. 2.2. Одномірні масиви

2.8. Присвоєння ім'я комірці (діапазону комірок)

Якщо комірка має ім'я, то надалі, замість її адреси можна використовувати це ім'я.

Для того щоб **привласнити комірці A1 ім'я x**, треба

- зробити A1 активною;
- у *поле імені* ввести **x**, натиснути *Enter*.

Присвоєння імені діапазону комірок:

- виділити діапазон;
- у *поле імені* ввести *ім'я*, натиснути *Enter*.

Інакше: пункт меню *Вставка*→*Ім'я*→*Присвоїти*, ввести *ім'я*, клацнути кнопку *Добавити*, *ОК*.

Приклад 2.5. Обчислити значення функції $Y = ax^2 + bx$, якщо x належить відрізку $[-2;0]$, $\Delta x = 0,5$; $a = 8$, $b = -4,6$.

Результати обчислень

| | A | B | C | D |
|---|----------|----------|----------|----------|
| 1 | x | y | a | b |
| 2 | -2 | 41,2 | 8 | -4,6 |
| 3 | -1,5 | 24,9 | | |
| 4 | -1 | 12,6 | | |
| 5 | -0,5 | 4,3 | | |
| 6 | 0 | 0 | | |

B2 ← = \$C\$2*A2^2+\$D\$2*A2

Рис. 2.3. Обчислення значень функції

Привласнити коміркам C2 і B2 імена **a** й **b** відповідно, замінити абсолютні адреси комірок на їхні імена. Формула прийме вигляд $=a*A2^2+b*A2$

2.9. Побудова й редагування діаграм

У програмі Excel термін «діаграма» використовується для позначення всіх видів графічного подання числових даних. Побудова графічного зображення виробляється на основі *ряду даних*.

Діаграма зберігає зв'язок з даними, на основі яких вона побудована, і при відновленні цих даних негайно змінює свій вигляд.

Для побудови діаграми використовують **Майстер діаграм**, що запускає щигликом по кнопці *Майстер діаграм* на стандартній панелі інструментів.

Часто зручно заздалегідь виділити область, що містить дані, які будуть відображатися, але задати цю інформацію можна й у ході роботи *майстра*.

Під управлінням *Майстра діаграм* виконуються 4 кроки:

Крок 1: вибір типу діаграми.

Крок 2: завдання діапазону з даними, якщо вони не були виділені раніше. Якщо діапазон з даними був обраний заздалегідь, то в області попереднього перегляду з'явиться приблизне відображення майбутньої діаграми.

Крок 3: оформлення діаграми:

- назва діаграми, підпис осей (вкладка *Заголовки*);
- відображення й маркування осей координат (вкладка *Осі*);
- опис побудованих графіків (вкладка *Легенда*);
- відображення написів, що відповідають окремим елементам даних на графіку (вкладка *Підписи даних*) і ін.

Крок 4: вказується, чи варто використати для розміщення діаграми новий робочий аркуш або наявний.

Редагування діаграми

Діаграма складається з набору окремих елементів, таких як самі графіки (ряди даних), осі координат, заголовки, область побудови та ін.

При щиглику на елементі діаграми він виділяється маркерами. Відкрити вікно для форматування елемента діаграми можна через меню *Формат* (для виділеного елемента) або через контекстне меню (команда *Формат*).

Щоб видалити діаграму, потрібно її виділити й натиснути клавішу *Delete*.

Приклад 2.6. Побудувати графік функції $Y = \cos x$ для значень x з діапазону $[-\frac{p}{2}; \frac{3p}{2}]$ із кроком $\frac{p}{4}$.

Порядок дій:

1. Заповнити значеннями стовпці X и Y (див. рис. 2.4).
 - A2 ← = -ПІ()/2
 - A3 ← = A2+ПІ()/4, скопіювати формулу на діапазон A4:A10
 - B2 ← = cos(A2), скопіювати формулу на діапазон B3:B10
2. Виділити діапазон A1:B10.
3. Викликати *Майстер діаграм* і проробити розглянуті раніше 4 кроки.

При побудові графіка функції зручно користуватися типом **Точковий**. При цьому, за замовчуванням, 1-й стовпець даних являє собою вісь X, 2-й стовпець - вісь Y. Одержимо графік, представлений на рис. 2.4.

| | A | B |
|----|----------|--------------|
| 1 | X | Y |
| 2 | -1,5708 | 6,12574E-17 |
| 3 | -0,7854 | 0,707106781 |
| 4 | 0 | 1 |
| 5 | 0,785398 | 0,707106781 |
| 6 | 1,570796 | 6,12574E-17 |
| 7 | 2,356194 | -0,707106781 |
| 8 | 3,141593 | -1 |
| 9 | 3,926991 | -0,707106781 |
| 10 | 4,712389 | -1,83772E-16 |

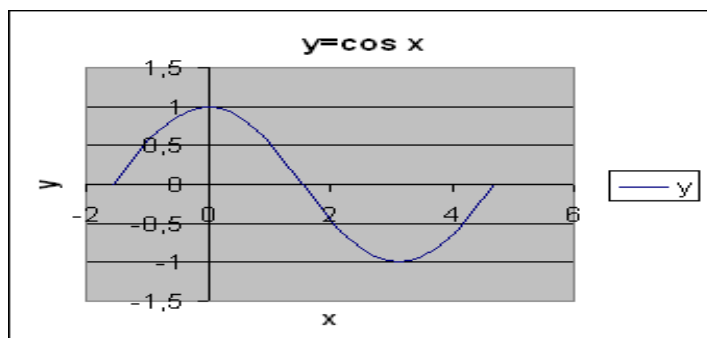


Рис. 2.4. Побудова графіка функції $Y = \cos x$

2.10. Побудова графіка функції, заданої параметрично

Приклад 2.7. Побудувати графік залежності $y(x)$, якщо

$$x = \sin^3 t;$$

$$y = \cos^3 t;$$

$$t \in [0; 2\pi], \quad \Delta t = 0,2.$$

Порядок дій (див. рис. 2.5):

| | A | B | C |
|-------|----------|----------|----------|
| 1 | t | x | y |
| 2 | 0 | 0 | 1 |
| 3 | 0,2 | 0,007841 | 0,941384 |
| 4 | 0,4 | 0,059054 | 0,781385 |
| 5 | 0,6 | 0,18002 | 0,562201 |
| 6 | 0,8 | 0,369151 | 0,338182 |
| 7 | 1 | 0,595823 | 0,157729 |
| 8 | 1,2 | 0,809659 | 0,047579 |
| 9 | 1,4 | 0,956981 | 0,00491 |
| 10 | 1,6 | 0,998721 | -2,5E-05 |
| 11 | 1,8 | 0,923577 | -0,01173 |
| | | | |
| 31 | 5,8 | -0,10029 | 0,694376 |
| 32 | 6 | -0,02181 | 0,885207 |
| 33 | 6,2 | -0,00057 | 0,989662 |

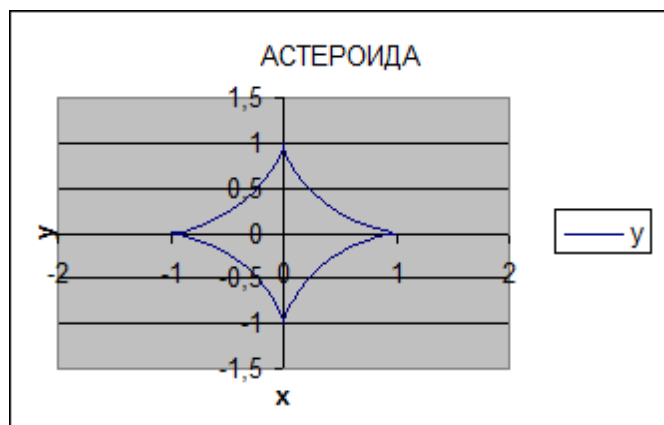


Рис. 2.5. Астероїда

- заповнити значеннями стовпець **t**;
- привласнити діапазону A2:A33 ім'я **t**;
- заповнити значеннями стовпець X
 $B2 \leftarrow =\sin(t)^3$
- заповнити значеннями стовпець Y

$$C2 \leftarrow =\cos(t)^3$$

- виділити діапазон В1:333, викликати *майстер діаграм*, проробити 4 кроки, тип графіка — *точковий*.

2.11. Побудова поверхні

Приклад 2.9. Побудувати графік функції двох змінних

$$z = x^2 - y^2;$$

$$x \in [-7,5;7,5]; \quad \Delta x = 1,5;$$

$$y \in [-5;5]; \quad \Delta y = 1.$$

Гіперболічний параболоїд (див. рис. 2.6.)

| | A | B | C | D | E | F | G | H | L |
|----|-----|-------|----|-------|-----|--------|-----|--------|-------|
| 1 | y x | -7,5 | -6 | -4,5 | -3 | -1,5 | 0 | 1,5 | 7,5 |
| 2 | -5 | 31,25 | 11 | -4,75 | -16 | -22,75 | -25 | -22,75 | 31,25 |
| 3 | -4 | 40,25 | 20 | 4,25 | -7 | -13,75 | -16 | -13,75 | 40,25 |
| 4 | -3 | 47,25 | 27 | 11,25 | 0 | -6,75 | -9 | -6,75 | 47,25 |
| 5 | -2 | 52,25 | 32 | 16,25 | 5 | -1,75 | -4 | -1,75 | 52,25 |
| 6 | -1 | 55,25 | 35 | 19,25 | 8 | 1,25 | -1 | 1,25 | 55,25 |
| 7 | 0 | 56,25 | 36 | 20,25 | 9 | 2,25 | 0 | 2,25 | 56,25 |
| 8 | 1 | 55,25 | 35 | 19,25 | 8 | 1,25 | -1 | 1,25 | 55,25 |
| 9 | 2 | 52,25 | 32 | 16,25 | 5 | -1,75 | -4 | -1,75 | 52,25 |
| 10 | 3 | 47,25 | 27 | 11,25 | 0 | -6,75 | -9 | -6,75 | 47,25 |
| 11 | 4 | 40,25 | 20 | 4,25 | -7 | -13,75 | -16 | -13,75 | 40,25 |
| 12 | 5 | 31,25 | 11 | -4,75 | -16 | -22,75 | -25 | -22,75 | 31,25 |

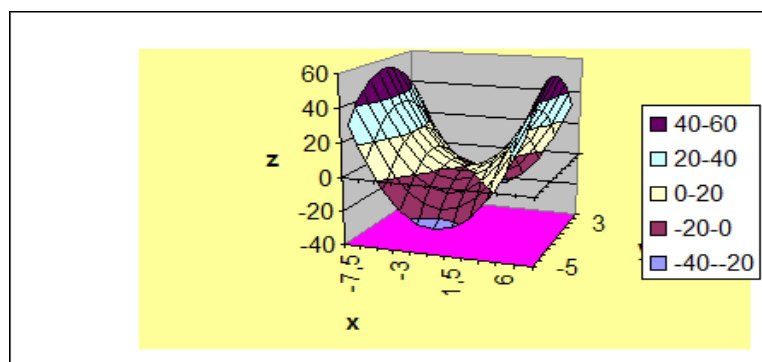


Рис. 2.6. Гіперболічний параболоїд

Порядок дій:

- заповнити 1-й рядок значеннями X;
- заповнити 1-й стовпець значеннями Y;
- $B2 \leftarrow =B1^2 - A2^2$, скопіювати формулу по стовпцю, а потім по рядку;
- виділити всю таблицю даних, діапазон A1:L12;
- викликати *Майстер діаграм*, проробити 4 кроки: тип - *поверхня*, ряди даних у рядках.

Тема 3. Excel. Організація обчислень

3.1. Наближене рішення нелінійних рівнянь. Підбір параметра

Вирішується рівняння виду $f(x)=0$. Питання про пошук наближеного рішення розпадається на 2 етапи:

- знаходження інтервалу ізоляції кореня (якщо він не заданий);
- уточнення кореня.

Приклад 3.1. Знайти корінь рівняння $8x^3 - 17x^2 + 8,5x - 8,25 = 0$.

Це рівняння має або 3, або 1 дійсний корінь.

Знаходження інтервалу ізоляції кореня:

- в області визначення функції вибрати довільним образом діапазон зміни x , наприклад, $[0;3]$ із кроком 0,5. Заповнити значеннями стовпець X (див. рис. 3.1).
- заповнити значеннями стовпець $Y=f(x)$. Для цього в комірку B2 занести формулу $B2 \leftarrow = 8*A2^3-17*A2^2+8,5*A2-8,25$. Скопіювати формулу на діапазон B3:B8.
- вибрати відрізок, на кінцях якого функція приймає значення різних знаків, це й буде інтервал ізоляції кореня. У нашому випадку це відрізок $[1,5; 2]$. Якщо такого відрізка немає, то змінити діапазон зміни x .

Уточнення кореня

- вибрати початкове наближення до кореня з відрізка $[1,5; 2]$, нехай це буде $x_0=1,7$;
 $C2 \leftarrow 1,7$
- обчислити значення функції в цій точці, тобто $f(x_0)$;
 $D2 \leftarrow =8*32^3-17*32^2+8,5*32-8,25$
- уточнити це значення за допомогою команди пункту меню **Сервіс** → **Підбір параметра**

У вікні діалогу

Установити в комірці

ввести D2

Значення

ввести 0

Змінюючи значення комірки

ввести $\$C\2 (щиглик на комірці C2)

У вікні діалогу **Результат підбору параметра:**

- наближене значення до кореня перебуває в комірці C2, $x \approx 1,851961$ в комірці D2 - погрішність кореня $8,8293E-0,6$.
- для збереження результату клацнути **ОК**.

При підборі параметра використовується ітераційний процес, при якому перевіряється одне значення за іншим, поки не отримуємо потрібне рішення.

| | A | B | C | D |
|---|----------|----------|---------------|--------------------|
| 1 | x | y | корінь | погрішність |
| 2 | 0 | -8,25 | 1,851961 | 8,8293E-06 |
| 3 | 0,5 | -7,25 | | |
| 4 | 1 | -8,75 | | |
| 5 | 1,5 | -6,75 | | |
| 6 | 2 | 4,75 | | |
| 7 | 2,5 | 31,75 | | |
| 8 | 3 | 80,25 | | |

Рис. 3.1. Наближене рішення нелінійного рівняння

3.2. Апроксимація функцій емпіричними формулами. Побудова лінії тренда

Лінія тренда звичайно використовується в задачах прогнозування, вона дозволяє графічно відображати тенденцію даних і прогнозувати їхні подальші зміни. Цей аналіз називається **регресійним аналізом**.

Приклад 3.2. Постановка задачі. Нехай у результаті досліджень отримані дані, які представлені у таблиці

| Місяць | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|-------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Продаж од. товару | 250 | 260 | 265 | 300 | 350 | 370 | 380 | 420 | 440 | 500 |

Необхідно підібрати формулу найкращого наближення для функції, заданої таблично.

Для рішення цієї задачі необхідно виконати наступне:

- у середовищі Excel побудувати таблицю вихідних даних;
- побудувати графік функції, заданої таблично (див. рис. 3.2.);
- з контекстного меню графіка вибрати **Додати лінію тренда**.

У діалоговому вікні, що відкрилося, у вкладці **тип** представлено 6 різних видів лінії тренду, які можуть бути додані в діаграму:

| | |
|----------------|-----------------|
| лінійна; | логарифмічна; |
| поліноміальна; | статечна; |
| експонентна; | ковзне середнє. |

Ми вибираємо вид залежно від типу даних. Для нашого випадку це *лінійна* лінія тренду.

На вкладці *Параметри* вибрати:

- ✓ Показати рівняння на діаграмі
- ✓ Помістити на діаграмі величину вірогідності апроксимації R^2

Поле **Прогноз** визначає продовження лінії тренду.

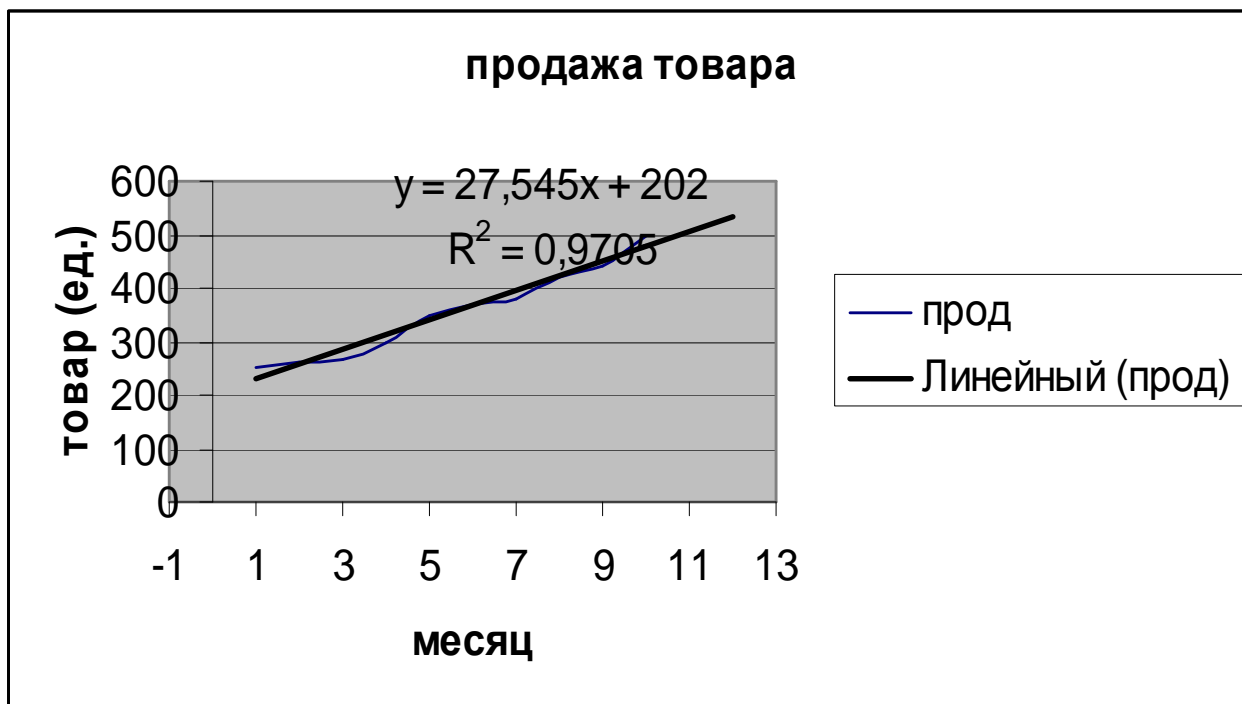


Рис. 3.2. Побудова лінії тренду

Лінія тренду може бути продовжена за межі реальних даних для подання майбутніх значень (див. рис. 3.2).

R^2 — число від 0 до 1, що позначає близькість лінії тренду до фактичних даних. Таким чином, формула найкращого наближення для функції заданої таблично має вигляд $Y = 27,545x + 202$.

Редагування лінії тренду може бути здійснене за допомогою контекстного меню.

Тема 4. Excel. Обчислювальний процес, що розгалужується

4.1. Функція ЕСЛИ

Функція **ЕСЛИ** використовується для перевірки умов і може бути викликана за допомогою *Майстра функцій* з категорії **Логічні**.

Загальний вигляд функції *если*:

ЕСЛИ (лог.вираз; значення якщо істина; значення якщо неправда)

Ця функція повертає одне значення, якщо *лог. вираз* при обчисленні дає значення *істина*, і інше значення — якщо *неправда*.

Приклад 4.1. Обчислити значення функції

$$z = \begin{cases} t \sin x, & \text{якщо } x \geq 0; \\ \cos y + x, & \text{якщо } x < 0; \end{cases}$$

$$x = 2a, \quad y = 3a, \quad a_H = -3, \quad a_K = 1, \quad \Delta a = 0,5; \quad t = 3,08.$$

Побудувати графік залежності $Z(a)$.

Послідовність дій (див. рис.4.1):

- увести значення t ;
заповнити значеннями стовпець L (один із варіантів заповнення):
 $A2 \leftarrow -3$; $A3 \leftarrow =A2+0,5$
Скопіювати формулу на діапазон $A4:A10$:
- привласнити діапазону $A2:A10$ ім'я L ;
- заповнити значеннями стовпець X :
 - $B2 \leftarrow = 2*L$
 - скопіювати формулу на $B3:B10$;
- заповнити значеннями стовпець Y :
 - $C2 \leftarrow = 3*L$
 - скопіювати формулу на $C3:C10$;
- заповнити значеннями стовпець Z : $D2 \leftarrow =ЯКЩО$

У діалоговому вікні, що відкрилося, заповнити поля:

| | |
|--------------------------------|-------------------------|
| Логічний вираз | $B2 \geq 0$ |
| Значення, якщо істина | $=ЕСЛИ(\$E\$2*\sin(B2)$ |
| Значення, якщо неправда | $\cos(C2)+B2$ |

У рядку формул:

$$=ЕСЛИ (B2 \geq 0; \$E\$2*\sin(B2); \cos(C2)+B2)$$

- скопіювати формулу на діапазон $D2:D10$.

Для побудови графіка залежності $Z(a)$ необхідно виділити діапазон $A1:A10$, при натиснутій клавіші [CTRL] виділити діапазон $D1:D10$.

Викликавши *Майстер діаграм* зручно вибрати тип *Точковий*, при цьому за замовчуванням вісь X буде заповнена значеннями a , а вісь Y — значеннями Z .

| | A | B | C | D | E |
|----|------|----|------|----------|------|
| 1 | L | X | Y | Z | T |
| 2 | -3 | -6 | -9 | -6,91113 | 3,08 |
| 3 | -2,5 | -5 | -7,5 | -4,65336 | |
| 4 | -2 | -4 | -6 | -3,03983 | |
| 5 | -1,5 | -3 | -4,5 | -3,2108 | |
| 6 | -1 | -2 | -3 | -2,98999 | |
| 7 | -0,5 | -1 | -1,5 | -0,92926 | |
| 8 | 0 | 0 | 0 | 0 | |
| 9 | 0,5 | 1 | 1,5 | 2,591731 | |
| 10 | 1 | 2 | 3 | 2,800636 | |

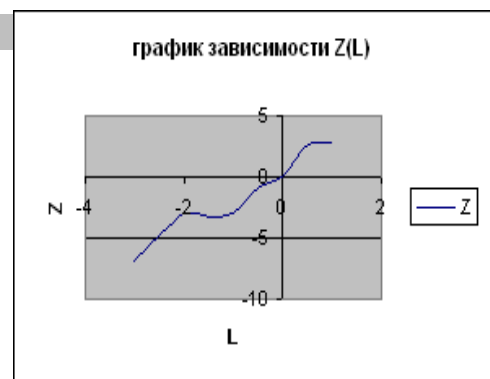


Рис. 4.1. Обчислювальний процес, що розгалужується. Графік залежності $Z(a)$

4.2. Функції И, ИЛИ

Функції И, ИЛИ як і функція ЕСЛИ використовуються при перевірці умов і можуть бути викликані за допомогою *Майстра функцій* з категорії *Логічні*.

Загальний вигляд функції И:

И (логічне вираження 1; логічне вираження 2; ...)

Функція повертає значення *істина*, якщо всі аргументи мають значення *істина*; повертає значення *неправда*, якщо хоча б один аргумент має значення *неправда*.

Загальний вигляд функції ИЛИ:

ИЛИ (логічне вираження 1; логічне вираження 2; ...)

Функція повертає значення *істина*, якщо хоча б один з аргументів має значення *істина*; повертає *неправда*, якщо всі аргументи мають значення *неправда*.

Таблиця істинності

| х | у | и | или |
|----------|----------|----------|------------|
| 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 |

1 - істина

0 – неправда

Приклад 4.2. математичний запис
 $x \in [0; 3]$

запис у середовищі Excel:
И($x \geq 0$; $x \leq 3$)

$x < -10$ **И** $x > 10$

ИЛИ ($x < -10$; $x > 10$)

Приклад 4.3. Обчислити значення функції

$$z = \begin{cases} \sin x, & x \leq -1 \\ x + 1, & -1 < x < 1 \\ x^2, & x \geq 1 \end{cases}$$

$$x \in [-5; 5], \Delta x = 1.$$

Привласнивши діапазону значень X ім'я x , формула може бути записана у вигляді:

інакше ЕСЛИ ($x \leq -1$; $\sin(x)$; ЕСЛИ (И($x > -1$; $x < 1$); $x + 1$; x^2))
ЕСЛИ ($x < -1$; $\sin(x)$; ЕСЛИ ($x \geq 1$; x^2 ; $x + 1$))

Тема 5. Excel. Одномірні й двомірні масиви

5.1. Матричні функції. Рішення системи лінійних алгебраїчних рівнянь матричним способом

Матричні функції вибираються за допомогою *Майстра функцій* з категорії *Математичні*. До цих функцій належать:

МОБР(масив)

Повертає зворотну матрицю для матриці, що зберігається в масиві.

Масив — числовий масив з рівною кількістю рядків і стовпців.

МОПРЕД(масив)

Повертає визначник матриці, що зберігається в масиві.

Масив — числовий масив з рівною кількістю рядків і стовпців.

МУМНОЖ(масив1;масив2)

Повертає добуток матриць (матриці зберігаються в масивах). Результатом є масив з таким же числом рядків, як *масив1* і з таким же числом стовпців, як *масив2*.

Масив1, масив2 — масиви, які перемножуються

Приклад 5.3. Знайти рішення системи лінійних алгебраїчних рівнянь матричним способом.

$$\begin{cases} 2x_1 + 3x_2 + 7x_3 + 6x_4 = 1 \\ 3x_1 + 5x_2 + 3x_3 + x_4 = 3 \\ 5x_1 + 3x_2 + x_3 + 3x_4 = 4 \\ 3x_1 + 3x_2 + x_3 + 6x_4 = 5 \end{cases} \quad (5.1)$$

Система (5.1) у матричній формі має вигляд:

$$\mathbf{AX}=\mathbf{B}, \quad \text{де } A = \begin{pmatrix} 2 & 3 & 7 & 6 \\ 3 & 5 & 3 & 1 \\ 5 & 3 & 1 & 3 \\ 3 & 3 & 1 & 6 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 3 \\ 4 \\ 5 \end{pmatrix}, \quad X = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

Рішення системи шукаємо у вигляді $X = A^{-1}B$

Для рішення цієї задачі необхідно виконати наступні дії:

1. Уведення початкових даних:

- у діапазон A3:D6 увести елементи матриці A;
- у діапазон E3:E6 увести елементи вектора B.

2. Обчислення зворотної матриці A^{-1} :

- виділити діапазон A11: D14, в якому буде перебувати зворотна матриця;
- за допомогою *Майстра функцій* викликати функцію **МОБР**;
- у поле *масив* увести діапазон A3:D6;
- натиснути **CTRL+SHIFT+ENTER**

формула обчислення зворотної матриці: $\{=МОБР(A3:D6)\}$

3. Знаходження рішення системи:

- виділити діапазон G3:G6, тут буде перебувати рішення системи;
- за допомогою *Майстра функцій* викликати функцію **МУМНОЖ**;
- у поле *Масив1* увести A11:D14;
- у поле *Масив2* увести E3:E6;
- натиснути **CTRL+SHIFT+ENTER**

формула знаходження рішення системи: {=МУМНОЖ(A11:D14;E3:E6)}

4. У результаті обчислень у діапазоні G3:G6 одержуємо координати вектора X - це й буде шукане рішення системи (5.1).

| | A | B | C | D | E | F | G |
|----|---|----------|----------|------|-----------------|------------------|------------------------|
| 1 | Рішення системи лінійних рівнянь | | | | | | |
| 2 | матриця А | | | | вектор В | | рішення системи |
| 3 | 2 | 3 | 7 | 6 | 1 | | 0,185714286 |
| 4 | 3 | 5 | 3 | 1 | 3 | | 0,778571429 |
| 5 | 5 | 3 | 1 | 3 | 4 | | -0,635714286 |
| 6 | 3 | 3 | 1 | 6 | 5 | | 0,457142857 |
| 7 | | | | | | | |
| 8 | | | | | | | |
| 9 | | | | | | | |
| 10 | зворотна матриця А⁻¹ | | | | | перевірка | |
| 11 | 0,028571 | -0,12857 | 0,385714 | -0,2 | | 1 | |
| 12 | -0,12381 | 0,307143 | -0,25476 | 0,2 | | 3 | |
| 13 | 0,171429 | -0,02143 | 0,064286 | -0,2 | | 4 | |
| 14 | 0,019048 | -0,08571 | -0,07619 | 0,2 | | 5 | |

Рис. 5.1. Рішення системи лінійних алгебраїчних рівнянь матричним способом

5. Зробимо перевірку, підставивши рішення системи у початкове матричне рівняння $AX=B$. Для цієї мети виконаємо наступні дії:

- виділити діапазон F11:F14, тут буде перебувати результат AX;
- за допомогою *Майстра функцій* викликати функцію **МУМНОЖ**;
- у поле *Масив1* увести A3:D6;
- у поле *Масив2* увести G3:G6;
- натиснути **CTRL+SHIFT+ENTER**.

У результаті підстановки знайденого рішення у початкове рівняння в діапазоні F11:F14 одержуємо праву частину системи (5.1).

Формула, за якою здійснюється перевірка: {=МУМНОЖ(A3:D6;G3:G6)}

5.2. Статистичні функції. Одномірні масиви

Статистичні функції вибираються за допомогою *Майстра функцій* з категорії *Статистичні*. До цих функцій належать:

МАКС(число1;число2; ...)

Ця функція повертає найбільше значення зі своїх аргументів.

Число1, число2,... - від 1 до 30 чисел, серед яких потрібно знайти найбільше. Логічні значення й текст ігноруються.

МИН(число1;число2; ...)

Ця функція повертає найменше значення зі своїх аргументів.

Число1, число2,... - від 1 до 30 чисел, серед яких потрібно знайти найменше.

СРЗНАЧ(число1; число2; ...)

Повертає середнє (арифметичне) своїх аргументів.

Число1, число2, ... - це від 1 до 30 аргументів, для яких обчислюється середнє значення.

СЧЁТ(значення1; значення2; ...)

Підраховує кількість чисел у списку аргументів.

Значення1, значення2, ... - це від 1 до 30 аргументів, які можуть містити дані різних типів, але в підрахунку беруть участь тільки числа.

СЧЁТЕСЛИ(діапазон; критерій)

Підраховує кількість комірок усередині діапазону, що задовольняють заданому критерію.

Діапазон — діапазон, у якому потрібно підрахувати комірки.

Критерій — критерій у формі числа, виразу або тексту, що визначає, які осередки треба підраховувати.

СУММЕСЛИ (діапазон; критерій; діапазон підсумовування)

(категорія *математичні*)

Підсумує комірки, що задовольняють заданому критерію.

Діапазон — діапазон комірок, що перевіряють.

Критерій — критерій у формі числа, виразу або тексту, що визначає, які комірки треба підраховувати

Діапазон підсумовування — фактичні комірки для підсумовування.

Приклад 5.1.

| | А | В |
|---|---------------------------------|---|
| 1 | Вартість майна | Комісійні |
| 2 | 100 000 | 7 000 |
| 3 | 200 000 | 14 000 |
| 4 | 300 000 | 21 000 |
| 5 | 400 000 | 28 000 |
| 6 | Формула | Опис (результат) |
| 7 | СУММЕСЛИ(A2:A5;">160000";B2:B5) | Сума комісійних для вартості майна більше 160000 (63 000) |

Приклад 5.2. У масиві $X = (-3; 16; 24; -0,1;-8)$ знайти кількість елементів, що належать відрізка $[-5; 5]$ (рис. 5.2).

B2 ← =ЕСЛИ (И(A2>=-5;A2<=5);A2), скопіювати формулу по стовпцю

C2 ← =СЧЁТ (B2:B6)

Результати обчислень:

| | A | B | C |
|---|---------|----------|-----------|
| 1 | масив X | [-5;5] | кількість |
| 2 | -3 | -3 | 2 |
| 3 | 16 | НЕПРАВДА | |
| 4 | 24 | НЕПРАВДА | |
| 5 | -0,1 | -0,1 | |
| 6 | -8 | НЕПРАВДА | |

Рис. 5.2. Знаходження кількості елементів, що належать відрізку [-5; 5]

5.3. Двомірні масиви

Приклад 5.3. Існує двомірний масив A(3,4) (мал. 12). Знайти:

1) суму позитивних елементів у кожному рядку масиву A

Прядок дій:

- E2← =СУММЕСЛИ(A2:D2,">0")
- скопіювати формулу на діапазон E3:E4

2) середнє арифметичне позитивних елементів

Прядок дій:

- сформувати масив з позитивних елементів
A8← = ЕСЛИ (A2>0; A2), скопіювати формулу на діапазон A8:D10
- знайти середнє значення
E8← =СРЗНАЧ(A8:D10)

| | A | B | C | D | E | F |
|----|-------------------------------------|------|------|------|-----------------|-----------------|
| 1 | Исходный массив | | | | S | |
| 2 | -2 | 3 | 7 | 0 | 10 | |
| 3 | 0 | 5 | -6 | 7,8 | 12,8 | |
| 4 | 7 | -6 | 5 | 3 | 15 | |
| 5 | | | | | | |
| 6 | | | | | | |
| 7 | массив из положит. элементов | | | | ср.знач. | ср.геом. |
| 8 | ЛОЖЬ | 3 | 7 | ЛОЖЬ | 5,4 | 5,069029 |
| 9 | ЛОЖЬ | 5 | ЛОЖЬ | 7,8 | | |
| 10 | 7 | ЛОЖЬ | 5 | 3 | | |

Рис. 5.3. Рішення задач із двомірним масивом

3) середнє геометричне позитивних елементів

Прядок дій:

- скористатися вже отриманим масивом з позитивних елементів
- знайти середнє геометричне елементів масиву
F8 ← =СРГЕОМ(A8:D10) (категорія *статистичні*)

Тема 6. Макроси

6.1. Технологія побудови макросів

Макрос (macro) - це послідовність команд, яка написана мовою VBA. Ці команди не пишуться вручну, вони виходять як результат роботи з мишею й клавіатурою. В MS Office існує вбудований засіб - макрорекодер, що дозволяє перетворювати всі наші дії в макроси.

Розглянемо на прикладі правила й послідовність дій при створенні макросів.

Приклад 6.1. Створення інформаційної системи в середовищі Excel.

Аркуш 1 перейменуємо в *Головне меню*

Аркуш 2 — у Завдання 1

Аркуш 3 — у Завдання 2

Аркуш 4 — у Завдання 3

Аркуш Головне меню. Розробка дизайну (один з варіантів)

ІНФОРМАЦІЙНА СИСТЕМА

Завдання 1

Завдання 2

Завдання 3

Порядок дій:

- за допомогою об'єкта WordArt (панель інструментів *Малювання*) увести заголовок;
- зобразити об'єкти (панель інструментів *Малювання*)
- дати назви об'єктам (контекстне меню об'єкта → *Додати текст*)

Проектування макросів

Прийmemo єдиний принцип створення макросів, що полягає в наступному:

- макрос повинен починатися й закінчуватися із щиглика на одній комірці, наприклад A1;
- кожна дія користувача повинна завершуватися натисканням клавіші *enter*;
- у процесі створення макросів не допускати зайвих дій.

1. Створення макросу для переходу на Аркуш Завдання 1:

- клацнути на комірці A1;
- вибрати пункт меню *Сервіс* → *Макрос* → *Почати запис*;
- у вікні *Запис макросу* заповнити поля (обов'язковим є тільки поле *Ім'я макросу*).

Ім'я макросу

увести *Завдання1* (за замовчуванням Макрос1)

Опис

увести коментар, наприклад, «Перехід на Аркуш2»

Сполучення клавіш CTRL+A

Зберегти в «ця книга» або може бути «нова книга»

Клацнути кнопку ОК

З'явиться плаваюча панель інструментів із кнопкою **Зупинити запис**.

Запис макросу (послідовність дій)

Клацнути на Аркуш *Завдання 1*.

2. Зупинити макрос

Виконати команду пункту меню *Сервіс* → *Макрос* → *Зупинити запис* (або клацнути на кнопку *Зупинити запис*).

3. Призначити макрос об'єкту *Завдання 1*:

- перейти на Аркуш *Головне меню*;
- контекстне меню об'єкта → *Призначити макрос*;
- вибрати ім'я *Завдання1*, **ОК**.

4. Перевірити роботу макросу

- клацнути на об'єкті *Завдання 1*

можливо інакше, за допомогою гарячих клавіш CTRL+A

або через верхнє меню *Сервіс* → *Макрос* → *Макроси*, вибрати потрібний, клацнути *виконати*.)

Аналогічним засобом створити макроси для переходів на Аркуші *Завдання 2* і *Завдання 3*, давши їм імена *Завдання2*, *Завдання3* відповідно.

Аркуш *Завдання 1*. Розробка дизайну

Завдання 1. Обчислити значення функції

$$z = \begin{cases} t \sin x, & \text{якщо } x \geq 0 \\ \cos y + x, & \text{якщо } x < 0 \end{cases}$$

$$x = 2a, \quad y = 3a, \quad a_n = -3; \quad a_k = 1; \quad \Delta a = 0,5; \quad t = 3,08$$

Побудувати графік залежності $Z(a)$.

| | A | B | C | D | E |
|----|------|------|----|------|----------|
| 12 | L | t | X | Y | Z |
| 13 | -3 | 3,08 | -6 | -9 | -6,91113 |
| 14 | -2,5 | | -5 | -7,5 | -4,65336 |
| 15 | -2 | | -4 | -6 | -3,03983 |
| 16 | -1,5 | | -3 | -4,5 | -3,2108 |
| 17 | -1 | | -2 | -3 | -2,98999 |
| 18 | -0,5 | | -1 | -1,5 | -0,92926 |
| 19 | 0 | | 0 | 0 | 0 |
| 20 | 0,5 | | 1 | 1,5 | 2,591731 |
| 21 | 1 | | 2 | 3 | 2,800636 |

обчислити x

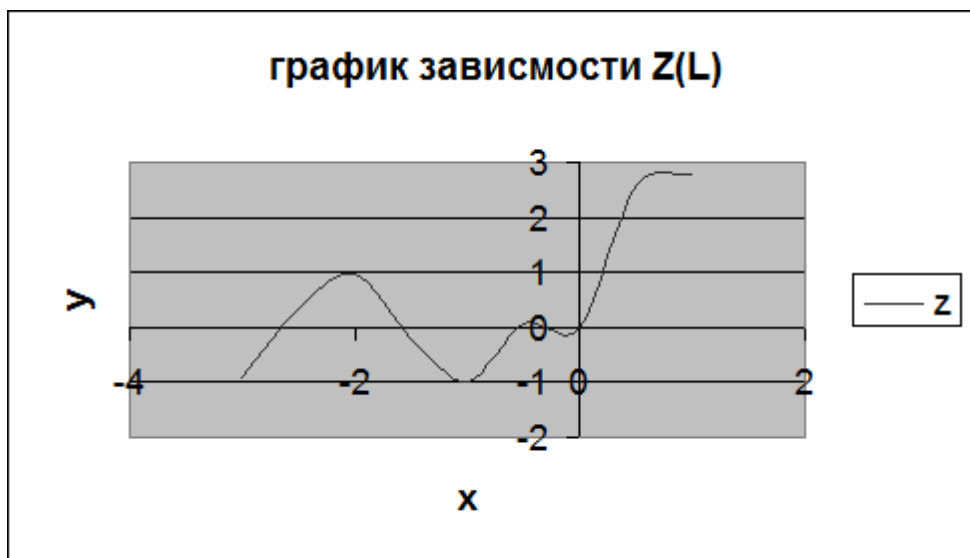
обчислити y

обчислити z і
побудувати графік

очистити

← головне меню

Рис. 6.1. Обчислення значення функції



Порядок дій

1. Увести умову завдання, викликавши редактор формул MS Equation 3.0.
2. Увести вихідні дані (див. рис. 6.2):
 - заповнити значеннями стовпець L, увести значення змінної t;
 - привласнити діапазону A13:A21 ім'я L.
3. Зобразити об'єкти, скориставшись панеллю інструментів *малювання*. Дати їм імена: *обчислити x*, *обчислити y*, *обчислити z* і *побудувати графік*, *очистити*, *головне меню*
 - контекстне меню об'єкта → *додати текст*.
4. Зобразити об'єкт Діаграма MS Graph:
 - виділити діапазон A12:A21, натиснути (утримувати) клавішу Ctrl, виділити діапазон E12:E21;
 - викликати *Майстер діаграм* і виконати 4 кроки.

Проектування макросів

1. Створення макросу для обчислення значень X:

- клацнути на комірці A1;
- вибрати пункт меню *Сервіс* → *Макрос* → *Почати запис*;
- у вікні *Запис макросу* заповнити поля, ім'я макросу *обчислити x*.

Виконати дії:

- C13 ← =2*L, скопіювати формулу по стовпці;
- клацнути в A1.

2. Зупинити макрос

Виконати команду пункту меню *Сервіс* → *Макрос* → *Зупинити запис* (або клацнути на кнопку *Зупинити запис*)

3. Призначити макрос об'єкту *Обчислити x*:

- з контекстного меню об'єкта вибрати *Призначити макрос об'єкту*

4. Перевірити роботу макросу:

- очистити діапазон C13:C21;
- клацнути на об'єкті *обчислити x*.

Макроси для обчислення значень Y, значень Z, очищення діапазону C13:C21, а також макрос для повернення в головне меню створюються аналогічно (див. пункти 1, 2, 3, 4), при цьому

D13 ← =3*I

E13 ← =ЕСЛИ(C13>=0; \$B\$13*sin(C13); cos(D13)+C13)

Графік будується автоматично.

6.2. Читання програми VB Script

Переглянути код макросу можна, виконавши команду *Сервіс* → *Макрос* → *Макроси*. У діалоговому вікні вибрати ім'я макросу й клацнути кнопку *Змінити*.

Дана дія викличе Редактор VB з текстом програми обраного Макросу.

Код макросу *Завдання1* має вигляд:

```
Sub Завдання1()  
' Завдання1 Макрос           'коментар  
  Sheets("Завдання 1").Select  'Аркуш Завдання 1 обране  
End Sub
```

Код макросу *Обчислити X* має вигляд:

```
Sub ВычислитьX ()  
' ВычислитьX Макрос  
  Range("C13").Select           'вибирається комірка C13  
  ActiveCell.Formula1C1 = "=2*L" 'в обрану комірку вводиться формула  
  Selection.AutoFill Destination:=Range("C13:C21"), Type:=xlFillDefault  
  Range("C13:C21").Select       'вибирається діапазон C13:C21  
  Range("A1").Select            'вибирається комірка A1  
End Sub
```

6.3. Друк документів в Excel

Перед друком робочого аркуша варто перейти в режим *Попереднього перегляду* (кнопка *Попередній перегляд* на панелі інструментів Стандартна). У діалоговому вікні вибрати кнопку *Сторінка*, потім зробити вибір на вкладках:

Сторінка

- орієнтація аркуша (книжкова, альбомна)

Аркуш

- сітка (включити або відключити сітку);
- заголовки рядків і стовпців.

Поля — дозволяє змінити величину полів сторінки (верхнє, нижнє, лїве, правє).

Для виведення на друк клацнути кнопку *Друк*.

Границі друкованої сторінки виділені на робочому аркуші дрібним пунктиром.

Тема 7. Середовище VBA (Visual Basic for Application)

Об'єктно-орієнтоване програмування (ООП) є однією із кращих технологій при створенні великих програмних проектів.

7.1. Інтерфейс редактора VBA

VBA функціонує тільки в складі Office додатків, таких як Excel, Word, Access та ін. Код VBA набирається в *редакторі VBA*. Запуск редактора VBA із середовища Excel можливий такими способами:

- пункт меню **Сервіс** → **Макрос** → **Редактор Visual Basic**;
- за допомогою гарячих клавіш [ALT+F11];
- панель інструментів **Visual Basic** → **Редактор Visual Basic**;
- § панель інструментів *Елементи керування* → **вихідний текст**.

Повернутися з редактора *Visual Basic* у робочу книгу можна натисканням кнопки **View Microsoft Excel** на панелі інструментів *Standard*.

Інтерфейс редактора **VBA** складається з наступних основних компонентів:

- | | |
|--------------------------|-----------------------------|
| • вікна проекту | вікно Project - VBA Project |
| • вікна редагування коду | вікно Code |
| • вікна властивостей | вікно Properties |
| • вікна форм | вікно UserForm |
| • панелі інструментів | панель ToolBox |

Виклик вікон може бути здійснений за допомогою пункту меню **View**.

Excel і VBA можуть перебувати у двох режимах:

Режим конструктора (дизайну). У цьому режимі ми розташовуємо об'єкти, редагуємо їхні властивості, пишемо програмний код. Перехід у режим конструктора здійснюється щикликом на відповідній піктограмі панелі інструментів *Елементи керування*.

Режим виконання — це звичайний режим, у якому виконуються обчислення.

7.2. Елементи керування.

Властивості, події й методи елементів керування

Елементи керування — це видимі об'єкти. Вони розташовані на панелі інструментів *Елементи керування* (середовище Excel), а також на панелі інструментів **ToolBox** редактора VBA. Деякі з них:

- **Кнопка (CommandButton)** – призначена для ініціалізації, закінчення або переривання яких-небудь дій.
- **Поле (TextBox)** – призначено для відображення інформації, що вводиться користувачем, а також для відображення результатів обчислень.

- **Перемикач (OptionButton)** – пропонує користувачеві зробити вибір одного елемента із групи альтернативних можливостей.
- **Напис (Label)** – призначена для відображення заголовків або короткого пояснювального тексту.
- **Малюнок (Image)** – призначений для виводу растрових зображень.
- **Смуга прокручування (ScrollBar)** – дозволяє встановити числові значення, ґрунтуючись на положенні повзунка

Об'єкти можуть бути розташовані як на робочому Аркуші, так і на **Формі**.

Форма – це об'єкт **UserForm**.

Додавання Форми в проект:

- перейти в редактор VB;
- пункт меню *Insert* → *UserForm*.

Кожний об'єкт має властивості (характеристики), події, на які він реагує й методи (оброблювачі подій).

Виклик вікна властивостей:

- у вікні *редактора коду* пункт меню *View* → *Properties*;
- на Аркуші Excel панель інструментів *Елементи керування* → *властивості*.

Деякі властивості об'єкта:

- **Name** – ім'я об'єкта;
- **Caption** – заголовок об'єкта;
- **BackColor** – колір об'єкта;
- **ForeColor** – колір заголовка;
- **Font** – шрифт та ін.

Події й оброблювачі подій

В VBA визначені спеціальні **процедури обробки подій**.

Наприклад, процедура обробки події щиглика по об'єкту:

```
Private Sub CommandButton1_Click()
' рядка програми, написані користувачем
End Sub
```

Доступні методи

Для кожного конкретного об'єкта в VBA визначені доступні методи, що реалізують ефективне керування.

Проект VBA за замовчуванням містить модуль всієї книги, а також кожний Аркуш Excel має власний модуль.

Всі елементи керування, установлені в Аркуш, будуть належати даному Аркушу, а значить їхні оброблювачі подій будуть реалізуватися в модулі даного аркуша.

7.3. Доступ до комірок Аркуша Excel, вивід даних у комірки, оформлення вигляду комірок

Об'єкт **Range** (діапазон) є одним із ключових об'єктів VBA. Комірка A2, як об'єкт, може бути записана двома способами

Range ("A2") або Cells(2,1) , де 2 — номер рядка
1 — номер стовпця

Приклад 7.1.

```
Cells(2,1) = 14.6  
Range("A1") = Sin(0.5)  
Range("A3") = Range("A1")*Cells(2,1)^2
```

Оформлення зовнішнього вигляду комірки або діапазону комірок може бути здійснене за допомогою властивостей **Interior** (інтер'єр), **Font** (шрифт) і вкладених у них властивостей:

| | |
|----------------|-----------|
| Color | колір |
| Pattern | візерунок |
| Size | розмір |
| Bold | жирний |
| Italic | курсив |
| і ін. | |

Приклад 7.2.

```
Range("A6") = "текст"  
Range("A6").Interior.Color = vbGreen 'заливання діапазону зеленим кольором  
Range("A6").Font.Color = vbBlue 'колір шрифту голубий  
Range("A6").Font.Bold=True 'шрифт жирний
```

Текст, що впливає за символом ('), ігнорується компілятором і являє собою коментар.

7.4. Типи даних.

Змінна типу Variant — основний тип даних середовища VBA

В VBA є наступні базові типи даних:

| | |
|----------------|---|
| Integer | цілий |
| Single | із плаваючою точкою звичайної точності |
| Double | із плаваючою точкою подвійної точності |
| String | строковий |
| Variant | тип, що використовується за замовчуванням |

Змінна повинна бути оголошена до її застосування. Якщо змінної не привласнене ніяке значення, то вона має значення нуль. Якщо тип змінної не заданий, то він за замовчуванням мається на увазі як **Variant**. Локальна змінна буде доступна тільки в тій процедурі, у якій вона оголошена.

Для оголошення змінної в VBA використовується оператор **Dim**.

Змінна може бути простою або масивом.

Приклад 7.3. Оголошення простих змінних.

```
Sub Test1()  
Dim a, c 'за замовчуванням тип Variant  
a = 100.5: c = 0  
Range("B1") = a: Range("B3") = c  
End Sub
```

Просту змінну типу *Variant* можна переголосити в одномірний масив із присвоєнням значень елементам даного масиву, використовуючи функцію **Array**.

Приклад 7.4.

```
Sub Test2()  
Dim A  
A = Array(10, 20, 30) 'нумерація елементів іде з нуля  
Range("A1")=A(0) 'відобразиться 10  
Range("A2")=A(2) 'відобразиться 30  
End Sub
```

Приклад 7.5. Оголошення змінної типу одномірний масив і присвоєння значень елементам масиву.

```
Dim A(2) 'верхнє значення індексу дорівнює 2  
A(0)= 3.6 : A(1)= 4.82  
A(2)= -12.7
```

Приклад 7.6. Оголошення змінної типу двомірний масив і присвоєння значень елементам масиву.

```
Dim B(1,1)  
B(0,0)=2: B(0,1)=4  
B(1,0)=1: B(1,1)=6
```

7.5. Математичні функції VBA

Середовище VBA надає користувачеві великий список убудованих математичні функцій. Деякі з них:

| функція | опис |
|------------|---|
| Abs | модуль (абсолютна величина) |
| Atn | арктангенс |
| Cos | косинус |
| Exp | експонента, тобто піднесення числа e (основа натурального логарифма) у зазначену степінь |
| Log | натуральний логарифм |
| Int | ціла частина числа |

| | |
|------------|--|
| Rnd | повертає випадкове число з інтервалу [0;1) |
| Sin | синус |
| Sgn | знак числа |
| Sqr | квадратний корінь |
| Tan | тангенс |
| Mod | Остача від ділення |

Приклад 7.7. Одержання випадкових чисел.

Dim Value1, Value2, Value3

Randomize ініціалізує генератор випадкових чисел
 Value1 = 100***Rnd()** 'повертає випадкове число з діапазону [0;100)
 Value2 = **Int**(6***Rnd()**)+1 'повертає ціле випадкове число з діапазону [1;6]
 Value3 = 100***Rnd()**-50 'повертає випадкове число з діапазону [-50;50)

7.6. Розробка Форми користувача для обчислення значень функції $Y=f(x)$

Розглянемо на прикладі функції $y(x) = a(x^2 + 1)$, a — задане число.

Порядок дій:

1. Завантажити середовище Excel. Зберегти проект.
1. Переіменувати Аркуш 1 в *Обчислення значення функції*.
2. На панелі інструментів *Елементи керування* перейти в режим *Конструктора*, установити об'єкт *Кнопка*, дати їй заголовок *Відкрити форму* (властивість *Caption*).
4. Перейти в редактор VB, створити Форму, дати їй заголовок та ім'я *Обчислення функції*
 - панель інструментів *Елементи керування* → *вихідний текст*
 - пункт меню **Insert** → **UserForm**
 - властивість **Name** → *Обчислення_ функції*
 - властивість **Caption** → *Обчислення_ функції_Y=f(x)*

5. Дизайн Форми

- збільшити розміри Форми за допомогою маркерів
 - скориставшись панеллю інструментів *ToolBox*, установити об'єкти і задати їм властивості у *вікні властивостей* (див. рис. 7.1).
- Label1.Caption → Зміна значення аргументу ползком
 Label2.Caption → Зміна значення параметра **a**
 Label3.Caption → Значення функції
 Label4 — для виводу значень функції
 Label5 — для виводу значень смуги прокручування
 ScrollBar1.min → -10 (ураховувати область визначення функції)
 ScrollBar1.max → 100

Command Button1. Caption → Clear
 Command Button2. Caption → Exit
 Image1 - для відображення функції

- **вставити формулу $y(x) = a(x^2 + 1)$ в об'єкт Image1**
 - увести формулу в текстовому редакторі Word
 - скопіювати через буфер обміну в графічний редактор Paint, зберегти у форматі *.bmp
 - вставити формулу в об'єкт Image1 за допомогою властивості *picture*.

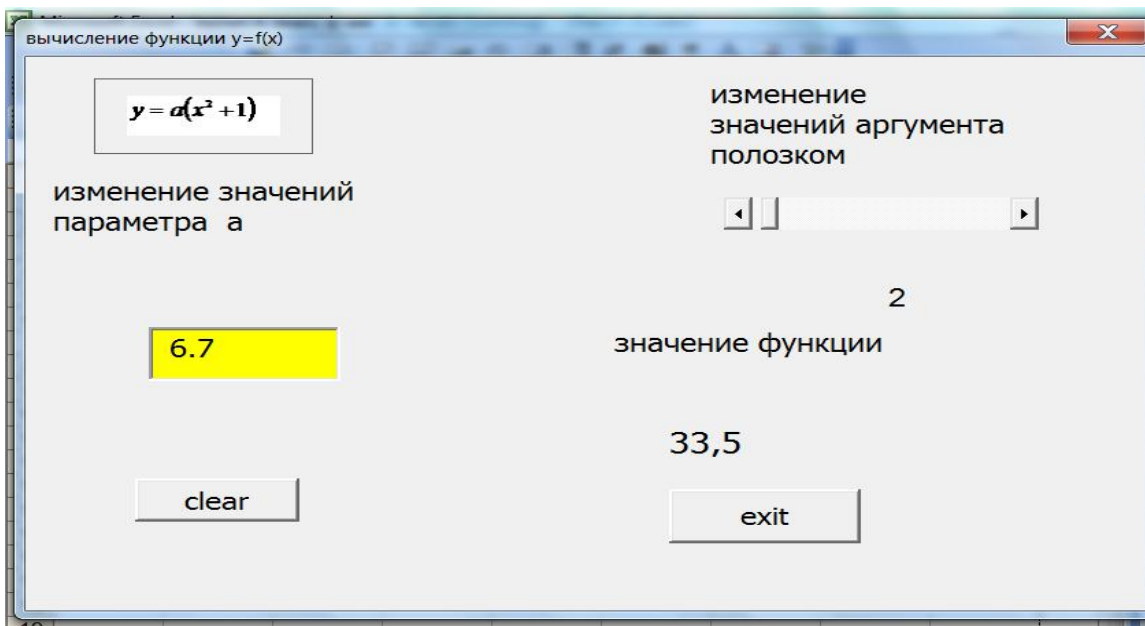


Рис. 7.1. Форма для обчислення значень функції

6. Перейти на Аркуш 1, увести оброблювач події *Щиглик по кнопці*

```
Private Sub CommandButton1_Click()
    Обчислення_функції.Show 'метод
End Sub
```

7. Перейти в модуль **Форми**.

- у вікні проекту зробити 2 клацання на об'єкті *Обчислення функції*
 У розділі **General Declaration** (загальна область) увести
 Dim y, a, x As Single 'змінні доступні в межах модуля

8. Увести оброблювач події **TextBox1_Change()**

```
Private Sub TextBox1_Change()
    a = Val(TextBox1.Text)
    y = a * (x ^ 2 + 1)
    Label4.Caption = y 'вивід значення функції
End Sub
```

Val() - функція, що перетворить строковий тип у числовий

9. Увести оброблювач події ScrollBar1_Change()

```
Private Sub ScrollBar1_Change()  
    Label5.Caption = ScrollBar1.Value  
    x = ScrollBar1.Value  
    y = a * (x ^ 2 + 1)  
    Label4.Caption = y           'вивід значень функції  
End Sub
```

10. Очищення полів

```
Private Sub CommandButton1_Click()  
    Label4.Caption = 0  
    TextBox1.Text = 0  
    Label5.Caption = 0  
End Sub
```

11. Закрити форму

```
Private Sub CommandButton2_Click()  
    End  
End Sub
```

12. Задати початкові властивості об'єктам.

Оброблювач події UserForm_Initialize()

```
Private Sub UserForm_Initialize()  
    Label1.Font.Size = 14  
    Label2.Font.Size = 14  
    Label3.Font.Size = 14  
    Label4.Font.Size = 16  
    Label5.Font.Size = 14  
    TextBox1.Font.Size = 14  
    CommandButton1.Font.Size = 13  
    CommandButton2.Font.Size = 13  
    TextBox1.BackColor = vbYellow  
End Sub
```

13. Запустити проект на виконання.

- перейти на Аркуш 1
- вийти з режиму конструктора
- клацнути на об'єкті *Відкрити форму*. Ввести значення параметра **a**, ползком на лінійці вибрати значення аргументу **x**. Результат роботи на рис. 7.1.

Тема 8. Основні конструкції мови VBA

8.1. Операції відносини. Логічні операції

При перевірці умов можуть використатися операції відносини:

| | | | |
|-----|-------------|----|---------------------|
| = | дорівнює | > | більше |
| < > | не дорівнює | >= | більше або дорівнює |
| < | менше | <= | менше або дорівнює |

а також логічні операції:

And (аналог И), **Or** (аналог ИЛИ), **Not**, що повертають логічні значення.

Таблиця істинності

| x | y | And | Or | Not x |
|---|---|-----|----|-------|
| 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 |

1 - істина

0 - неправда

Операція **And** повертає значення *істина*, якщо всі аргументи мають значення *істина*; повертає значення *неправда*, якщо хоча б один аргумент має значення *неправда*.

Операція **Or** повертає значення *істина*, якщо хоча б один з аргументів має значення *істина*; повертає *неправда*, якщо всі аргументи мають значення *неправда*.

Операція **Not** повертає значення *істина*, якщо була *неправда* й *неправда* в протилежному випадку.

Наприклад, приналежність елемента масиву проміжку [-1;1] може бути записана таким способом:

$x(i) > -1$ **And** $x(i) < 1$

8.2. Обчислювальний процес, що розгалужується.

Оператор умовного переходу **IF**

Оператор **IF** задає виконання груп операторів залежно від виконання умови. Існує кілька варіантів цього оператора:

а) **IF** умова **Then** оператори

У цьому випадку, якщо умова приймає значення *істина*, то виконуються оператори, що впливають за **Then**, у протилежному разі — оператор, що впливає за оператором **IF**.

б) **IF** умова **Then** оператори 1 **Else** оператори 2

У цьому випадку, якщо умова приймає значення *істина*, то виконуються *оператори 1*, що впливають за **Then**, у протилежному випадку — *оператори 2*, що впливають за **Else**.

в) блок **IF**

```

IF умова Then
    оператори1
[Else
    оператори2]
End IF

```

Виконується аналогічно випадку б). Група [**Else** оператори 2] може бути відсутня.

Приклад 8.1. Обчислити значення функції

$$z = \begin{cases} a + b, & \text{при } a > b \\ a - b, & \text{при } a < b \\ a * b, & \text{при } a = b \end{cases}$$

Нехай вихідні дані $a=8,7$ і $b=5,3$ перебувають в комірках аркуша Excel A1 і B1 відповідно.

Процедура на VBA має вигляд:

Sub f1()

Dim a, b, z

a = Range("A1"): b = Range("B1")

If a > b Then z = a + b

If a < b Then z = a - b

If a = b Then z = a * b

Range("C1") = z

End Sub

| | A | B | C |
|---|-----|-----|----|
| 1 | 8,7 | 5,3 | 14 |
| 2 | | | |

8.3. Функція **InputBox** для введення даних із клавіатури

Функція **InputBox** виводить на екран вікно, що містить повідомлення й поле введення. Вона встановлює режим очікування й повертає значення типу *String*, що містить текст. Для перетворення текстового типу в числовий використовується функція **Val**, наприклад:

a=val(InputBox("a"))

Приклад 8.2. Знайти дійсні корені квадратного рівняння $ax^2+bx+c=0$, $a \neq 0$. Значення **a**, **b**, **c** уводяться із клавіатури, результат виводиться в комірки робочого Аркуша.

Порядок дій:

- розмістити два об'єкти Кнопка для запуску проекту й для очищення діапазону, дати їм заголовки *Рішення рівняння* й *Очистити* відповідно.

| | A | B | C | D | E | F |
|---|----------------------|---|----|-------------------|---|---|
| 1 | a | b | c | | | |
| 2 | 2 | 3 | -4 | | | |
| 3 | | | | | | |
| 4 | x1=0,850781059358212 | | | Решение уравнения | | |
| 5 | x2=-2,35078105935821 | | | | | |
| 6 | | | | ОЧИСТИТЬ | | |
| 7 | | | | | | |
| 8 | | | | | | |

Рис. 8.1. Інтерфейс рішення квадратного рівняння

- оброблювач події *клацання по кнопці Рішення рівняння* має вигляд:


```

Private Sub CommandButton1_Click()
a = Val(InputBox("a"))
b = Val(InputBox("b"))
c = Val(InputBox("c"))
Range("A2") = a           'запис значення змінної в комірку Аркуша
Range("B2") = b
Range("C2") = c
d = b ^ 2 - 4 * a * c
If d >= 0 Then
  x1 = (-b + Sqr(d)) / (2 * a)
  x2 = (-b - Sqr(d)) / (2 * a)
  Range("A4") = "x1=" & x1
  Range("A5") = "x2=" & x2
Else
  Range("A4") = "дійсних коренів немає"
End If
End Sub

```
- запустити проект на виконання – пункт меню **Run (F5)**
- оброблювач події *клацання по кнопці Очиstitи* має вигляд:


```

Private Sub CommandButton2_Click()
Range("A4:C5").ClearContents
End Sub

```
- повернутися в Excel, вийти з режиму конструктора. Запустити проект на компіляцію й виконання. Із клавіатури ввести значення $a=2$, $b=3$, $c=-4$. Результат див. на рис. 8.1.

8.4. Циклічний обчислювальний процес. Оператори циклу FOR ... NEXT

За допомогою конструкції **FOR ... NEXT** можливе повторення групи операторів задане число раз.

Загальний вигляд:

```

FOR змінна = A1 TO A2 [step A3]
  [оператори]
[EXIT FOR]
  [оператори]
NEXT [змінна] ,
  
```

де *змінна* — параметр циклу

A1 - початкове значення параметра циклу;

A2 - кінцеве значення параметра циклу;

A3 - крок зміни параметра циклу (якщо крок дорівнює 1, то може бути відсутнім);

EXIT FOR — примусовий вихід із циклу.

Приклад 8.3. Обчислити значення функції Z. Вихідні дані перебувають в комірках Аркуша Excel. Результат вивести в комірки Аркуша.

| | | | | | | | | | | | | |
|----|---|------|--|--|----------|------|----------|---|---|---|---|---|
| 1 | Λ | | | | В | С | Д | Е | Ф | Г | Н | І |
| 2 | Висчислити значення функції | | | | | | | | | | | |
| 3 | $z = \begin{cases} \cos \alpha + 4x, & x > y \\ \sin^2 \alpha + y, & x < y \end{cases}$ | | | | | | | | | | | |
| 4 | | | | | | | | | | | | |
| 5 | $x = 3\alpha; \quad y = x - 2t$ | | | | | | | | | | | |
| 6 | $t = 3.6; \quad \alpha \in [-1.2; 3.2], \quad \Delta\alpha = 0.5$ | | | | | | | | | | | |
| 7 | <input type="button" value="ВЫЧИСЛИТЬ"/> | | | | | | | | | | | |
| 8 | | | | | | | | | | | | |
| 9 | | | | | | | | | | | | |
| 10 | | | | | | | | | | | | |
| 11 | ln | -1,2 | | | z | | l | | | | | |
| 12 | lk | 3,2 | | | -14,0378 | -1,2 | | | | | | |
| 13 | dl | 0,5 | | | -7,83518 | -0,7 | | | | | | |
| 14 | t | 3,6 | | | -1,41993 | -0,2 | | | | | | |
| 15 | | | | | 4,555336 | 0,3 | | | | | | |
| 16 | | | | | 10,29671 | 0,8 | | | | | | |
| 17 | | | | | 15,0675 | 1,3 | | | | | | |
| 18 | | | | | 21,3728 | 1,8 | | | | | | |
| 19 | | | | | 26,93372 | 2,3 | | | | | | |
| 20 | | | | | 32,66778 | 2,8 | | | | | | |

Рис. 8.2. Обчислення значення функції

Порядок дій:

- перейти в режим конструктора, установити елементи керування *CommandButton1* і *CommandButton2*, дати їм заголовки *Обчислити* й *Очистити* відповідно;
- оброблювач події клацання по кнопці *Обчислити* має вигляд:
Private Sub CommandButton1_Click()
Dim ln, lk, dl, a, z, l

```

ln = Range("b11")
lk = Range("b12")
dl = Range("b13")
t = Range("b14")
i = 12
For l = ln To lk Step dl
    x = 3 * l
    y = x - 2 * t
    If x >= y Then
        z = Cos(l) + 4 * x
    Else
        z = Sin(l) ^ 2 + y
    End If
    Cells(i, 4) = z
    Cells(i, 5) = 1
    i = i + 1
Next
End Sub

```

- оброблювач події *клацання по кнопці Очистити* має вигляд:

```

Private Sub CommandButton2_Click()
Range("d12:e20").ClearContents
End Sub

```
- повернутися в Excel, вийти з режиму конструктора. Перевірити роботу проекту клацанням по кнопках.

Приклад 8.4. Заповнення діапазону комірок випадковими числами

```

For i = 1 To 10
    Cells(2, i) = 100*Rnd()-50 'випадкові числа з діапазону [-50;50)
Next

```

8.5. Одномірні масиви

Якщо розмірність масиву в програмі не постійна, то такий масив називається *динамічним*. Розмірність масиву може задаватися на етапі виконання програми. Динамічні масиви оголошуються за допомогою оператора **ReDim**.

Приклад 8.5. Створити процедуру для генерування одномірного масиву цілих випадкових чисел будь-якої розмірності. Знайти середнє геометричне позитивних елементів цього масиву.

Порядок дій:

- розробити дизайн

| | A | B | C | D | E | F | G | H | I | J |
|----|------------------------------|-----|-----------|----|--------|-----|----------|----|-----|---|
| 1 | | | | | размер | 9 | | | | |
| 2 | | | | | | | | | | |
| 3 | | | | | | | | | | |
| 4 | | | ВЫЧИСЛИТЬ | | | | ОЧИСТИТЬ | | | |
| 5 | | | | | | | | | | |
| 6 | | | | | | | | | | |
| 7 | 20 | -46 | -9 | 36 | 29 | -13 | 46 | 37 | -45 | |
| 8 | | | | | | | | | | |
| 9 | sr. geom. = 32,3696504799439 | | | | | | | | | |
| 10 | | | | | | | | | | |

Рис. 8.3. Результат роботи проекту прикладу 8.5.

- перейти в режим конструктора, установити елементи керування *CommandButton1* і *CommandButton2* дати їм заголовки *обчислити* й *очистити* відповідно
- оброблювач події *клацання по кнопці Обчислити* має вигляд:

```
Private Sub CommandButton1_Click()
    Dim p, k, n, i
    n = Range("f1")
    ReDim x(n - 1)
    Randomize
    For i = 0 To n - 1
        x(i) = Int(100 * Rnd() - 50)
        Cells(7, i + 1) = x(i)
    Next
    p = 1: k = 0
    For i = 0 To n - 1
        If x(i) > 0 Then
            p = p * x(i): k = k + 1
        End If
    Next
    Range("a9") = "sr. geom. = " & p ^ (1 / k)
End Sub
```
- оброблювач події *клацання по кнопці Очистити* має вигляд:

```
Private Sub CommandButton2_Click()
    Range("A7:p7").ClearContents
    Range("A9").Clear
End Sub
```
- запуснути на виконання. Результат роботи на рис. 8.2.

Вихідний масив може бути заданий у програмі, у цьому випадку користуються функцією *Array*, наприклад:

$x = \text{Array}(-17, 8, -0.3, 6, -9, 5.5, 0, 7.8, -9, -4.5)$

Приклад 8.6. Комірки діапазону A60:F60 заповнені цілими числами і пофарбовані в різні кольори. Знайти суму значень комірок червоного й зеленого кольорів для комірок, що мають значення >1. Див. рис. 8.4.

| A | B | C | D | E | F |
|-----------------------|---|---|---|---|---|
| 2 | 3 | 1 | 2 | 1 | 5 |
| Сума значень комірок: | | | 5 | | |

Рис. 8.4. До прикладу 8.6.

```

Dim i, s
S=0
For i=1 To 6
  If Cells(60,i).Interior.Color=vbRed Or Cells(60,i).Interior.Color=vbGreen Then
    If Cells(60,i)>1 Then s=s+ Cells(60,i)
  End If
Range("D61")=s
Next i

```

8.6. Двомірні масиви

Приклад 8.7. Створити процедуру для генерування двомірного масиву цілих випадкових чисел будь-якої розмірності. Знайти суму елементів цього масиву.

Порядок дій:

- розробити дизайн (див. рис. 8.5)

| | A | B | C | D | E | F | G | H | | | |
|----|--|-----|-----|----|---------------------|----------|----------|---|--|--|--|
| 1 | Сгенерировать случайным образом двумерный массив целых чисел любой размерности. Найти сумму элементов массива. | | | | размерность | | | | | | |
| 2 | | | | | 3 | | 4 | | | | |
| 3 | | | | | | | | | | | |
| 4 | | | | | вывести и вычислить | | очистить | | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |
| 7 | | | | | | | | | | | |
| 8 | | | | | | | | | | | |
| 9 | | | | | | | | | | | |
| 10 | | | | | | | | | | | |
| 11 | | | | | | | | | | | |
| 12 | массив | | | | | сумма=87 | | | | | |
| 13 | | -4 | -21 | 12 | 14 | | | | | | |
| 14 | | -24 | -23 | 32 | 32 | | | | | | |
| 15 | | 8 | 48 | 41 | -28 | | | | | | |

Рис. 8.5. Результат роботи проекту прикладу 8.6.

- перейти в *режим конструктора*, установити елементи керування *CommandButton1* і *CommandButton2* дати їм заголовки *вивести й обчислити* та *очистити* відповідно
- оброблювач події *клацання по кнопці Очистити* має вигляд:


```
Private Sub CommandButton2_Click()
    Range("a13:D15").ClearContents
    Range("f12").Clear
End Sub
```
- оброблювач події *клацання по кнопці Вивести й обчислити* має вигляд:


```
Private Sub CommandButton1_Click()
    Dim i, j, n, m, s
    m = Range("F3")
    n = Range("G3")
    ReDim x(m-1, n-1)
    For i = 0 To m - 1
    For j = 0 To n - 1
    x(i, j) = Int(100 * Rnd() - 50)
    Cells(i + 13, j + 1) = x(i, j)
    s = s + x(i, j)
    Next
    Next
    Range("F12") = "сума=" & s
End Sub
```
- запустити проект на виконання. Результат роботи див. на рис. 8.5.

Тема 9. Середовище Borland C++ Builder

Builder — це середовище, у якому може бути здійснене об'єктно-орієнтоване програмування (ООП), тобто середовище із графічним інтерфейсом. Кожний об'єкт має властивості (характеристики), події, на які він реагує, й методи (оброблювачі подій).

На ринку програмних продуктів існує багато середовищ для автоматизації програмування. По потужності середовище Builder може суперничати тільки з Borland Delphi.

9.1. Історія створення мови C++

Предком мови C++ з'явилася мова C, створена в 2-ій половині 1970-х років Денісом Рітчі (США), як мова системного програмування.

В 80-х роках Б'єрном Струострупом створюється мова C++, як мова об'єктно-орієнтованого програмування. Символами ++ позначається операція збільшення на одиницю, тобто C++ задумана як мова C з розширеними можливостями.

Існує стандарт мови C и C++, тобто однозначне й машинно-незалежне визначення мови.

9.2. Головні складові частини середовища C++ Builder

Дизайнер Форм (Form1) — це порожня форма, що автоматично з'являється на екрані при створенні проекту. Вона заповнюється потрібними об'єктами, обраними в палітрі компонентів.

Палітра компонентів — тут розташовані компоненти середовища, використовується їхнє посторінкове угруповання. Існує бібліотека візуальних компонентів VCL. Основні візуальні компоненти перебувають на сторінках Standard, Addition, Win32.

Інспектор об'єктів (Object Inspector) — це вікно, що дозволяє побачити основні властивості й події об'єкта, поміщеного у форму. Інформація змінюється залежно від об'єкта, обраного у формі.

Інспектор об'єктів складається із двох сторінок. Перша сторінка — це список властивостей (*Properties*), друга — список подій (*Events*).

Вікно редактора коду (коротко Редактор) — при створенні нової форми, до неї створюється програмний модуль (за замовчуванням ім'я Unit1.cpp) для програм-оброблювачів подій. Потрапити в програмний модуль (редактор коду) з вікна Form1 можна за допомогою клавіші [F12].

9.3. Проект Builder

Всі користувальницькі програми в середовищі Builder оформляються у вигляді *проектів*. Результатом роботи є здійснений файл (додаток).

Розробка будь-якого проекту починається зі створення нової папки, у якій необхідно зберегти порожній проект.

Для цього виконуються наступні дії:

- пункт меню **File** → **Save Project As**
зберегти в новій папці два файли:
 - програмний модуль: за замовчуванням ім'я **Unit1** (.cpp) → Зберегти
 - модуль проекту: за замовчуванням ім'я **Project1** (.bpr) → Зберегти.

При першому збереженні проекту рекомендується привласнювати унікальні імена. Під час розробки додатка корисно робити проміжні збереження:

головне меню **File** → **Save All** — зберігаються всі вихідні файли під поточними іменами.

Папки, що містять файли проектів, можна переміщати з диска на диск, переносити на інші комп'ютери й перейменовувати, але файли, що ставляться безпосередньо до проекту, перейменовувати не можна.

Склад і структура проекту

Середовище Builder зв'язує з кожним своїм додатком наступні файли:

Unit1.cpp — C++ файл (текст програми). Якщо з'являться нові форми, то для кожної з них буде побудований свій програмний модуль: Unit2.cpp, Unit3.cpp і т.д.

Unit1.h — модуль із оголошеннями компонентів, розташованих у даній формі. Для кожної нової форми буде побудований свій модуль: Unit2.h, Unit3.h, і т.д.

Unit1.dfm — тут зберігається опис форми й всіх розташованих у ній компонентів.

Project1.cpp — файл проекту. Він містить код головної програми, написаної мовою C++. У файлі проекту втримуються посилання на всі форми проекту й стосовні до них модулі.

Уміст файлу Project1.cpp можна переглянути, виконавши команду **View** → **Project Source** головного меню.

Project1.exe — здійснений файл (додаток), готова до виконання програма, що може функціонувати під керуванням операційної системи Windows.

Project1.res — файл ресурсів проекту, являє собою двійковий файл. У ньому зберігаються різні значки, графічні зображення, що використовуються в проекті.

Project1.bpr — цей файл відповідає за проведення процесу зборки й компіляції проекту.

Якщо переглянути папку, у якій перебуває весь проект, то виявимо ще й інші файли, наприклад, файли тимчасового зберігання ~cpp, ~dfm.

Тема 10. Елементи мови C++.

Створення консольного додатка.

Лінійний обчислювальний процес

10.1. Символи мови

10.1.1. Великі букви латинського алфавіту A, B, C, D, ...X, Y, Z

Малі літери латинського алфавіту a, b, c,...x, y, z

Символ підкреслення _

10.1.2. Великі й малі літери російського алфавіту

Однакові великі й малі літери вважаються різними символами.

10.1.3. Арабські цифри 0, 1, 2, ...9

10.1.4. Спеціальні символи . , ; () < > / \ * = - % і ін.

10.1.5. Керуючі символи, що використовуються у функціях вводу-

виведення: \n - перехід на новий рядок;

\t - горизонтальна табуляція;

\o - нульовий символ;

\v - вертикальна табуляція й інші.

10.2. Константи

Розрізняють чотири види констант:

Цілі константи

- *десяткові*: наприклад 16; 240

- *восьмирічні*: цифри восьмирічної системи 0, 1, 2, 3, 4, 5, 6, 7.

Восьмирічні константи завжди починаються з нуля

- *Шістнадцятирічні*: як цифри використовуються 16 символів — 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. *Шістнадцятирічні* константи завжди починаються з пари символів 0x.

Наприклад:

*Шістнадцятирічні
константи*

0x10

0xFFFF

0x1F1A

*Восьмирічні
константи*

01

065

0777

Дійсні константи (із плаваючою точкою)

Наприклад: 3.45; 1.5E-2; -0,85E12; -.56

Символьні константи

Являють собою символи, взяті в апострофи. У таблиці кодів ASCII кожному символу ставиться у відповідність ціле позитивне число (код), тому значенням символьної константи є числовий код символу.

Приклади символьних констант: 'Q'; 'a'; '\n'.

Строкові константи — це послідовність символів, взятих в лапки. Наприклад: “ `Borland C++` ”.

Наприкінці строкової константи завжди існує ознака кінця рядка `\0`, що формує компілятор: “ `Borland C++\0` ”.

10.3. Коментарі

Якщо текст коментарів займає один рядок, то використовується `//.....`

Якщо текст коментарів займає більше одного рядка, то використовується

```
/*      */
```

10.4. Типи даних

Особливістю мови C є відсутність принципу умовчання. Тому типи всіх змінних повинні бути оголошені.

Базові типи даних:

| | |
|---------------|---|
| int | цілий тип; |
| float | тип із плаваючою точкою; |
| double | із плаваючою точкою подвійної точності; |
| char | символьний тип; |
| void | порожній тип. |

На основі цих типів будуються інші типи за допомогою модифікаторів:

| | |
|-----------------|------------|
| short | короткий; |
| long | довгий; |
| unsigned | без знака; |
| signed | знаковий. |

Таблиця основних типів даних

| ТИП | РОЗМІР | ЗНАЧЕННЯ | ПОЯСНЕННЯ |
|--------------|--------|--------------------------|----------------------------|
| bool | 1 | true, false | логічний тип |
| int | 2 | -32768 — 32768 | цілий тип |
| short int | 2 | -32768 — 32768 | короткий цілий тип |
| long int | 4 | -2147483648 — 2147483648 | довгий цілий зі знаком |
| unsigned int | 2 | 0 — 65535 | цілий без знака |
| float | 4 | -1,2e-38 — 34e38 | дійсний звичайної точності |
| double | 8 | 2,2e-308 — 1.8e308 | дійсний подвійної точності |
| char | 1 | 256 значень символів | символьний |

Ініціалізація змінних

а) **int** a = 24, i = 5; //змінним цілого типу **a**, **i** привласнюються значення;

б) **char** c='c'; //змінної символічного типу **c** привласнюється значення.

У цьому випадку змінна **c** містить один символ **c** (точніше його код по таблиці кодування ASCII), тип **char** і **int** взаємозамінні.

в) **String** a, b; // оголошуються змінні строкового типу **a** і **b**
a = "Програмування";
b = "на C++";

Змінна будь-якого типу може бути оголошена як не змінююча. Це досягається додаванням зарезервованого слова **const**.

Наприклад: **const double** A=2.12E-2

10.5. Масиви

Оголошення масивів:

а) **int** a[30]; //масив з 30 елементів, нумерація з нуля
Це елементи a[0], a[1], ...,a[29]. Кількість байт у пам'яті 2*30=60 байт.

б) **double** b[2][3]; //двовимірний масив з 6 елементів
Це елементи b[0][0], b[0][1], b[0][2], b[1][0], b[1][1], b[1][2]

Ініціалізація масивів

Одномірний масив

```
int m[2]={1,8};  
інакше int m[2];  
m[0]=1; m[1]=8;
```

Двовимірний масив $A = \begin{pmatrix} 2 & 1 & 3 \\ 4 & 5 & 6 \end{pmatrix}$

```
int a[2][3]={{2, 1, 3}, {4, 5, 6}};  
можливі варіанти int a[2][3]={2, 1, 3, 4, 5, 6};  
int a[ ][3]={2, 1, 3, 4, 5, 6};
```

Масив символів

```
char str[3]={'a','b','c'}; //одномірний масив символів
```

10.6. Арифметичні операції

| | | | |
|---|------------|---|-----------------------|
| + | додавання | / | цілочисельне ділення |
| - | віднімання | % | остача від ділення |
| * | множення | | унарні операції + і - |

| Наприклад: | операція | результат |
|------------|----------|--|
| | 11/3 | 3 |
| | 11./3.3. | 666..... |
| | 11%3 | 2 |
| | -x*y | (-x)*y, тому що пріоритет унарних операцій вище, ніж у бінарних. |

10.7. Математичні функції

| функція | опис |
|--------------------|---|
| abs | модуль (абсолютна величина) |
| sin | синус |
| cos | косинус |
| tan | тангенс |
| exp | експонента, тобто піднесення числа e (основа натурального логарифма) у зазначену степінь |
| log | натуральний логарифм |
| log10 | десятковий логарифм |
| sqrt | корінь квадратний |
| pow(x, y) | піднесення в степінь x^y |
| pow10(x) | піднесення в степінь 10^x |
| asin | арксинус |
| acos | арккосинус |
| atan | арктангенс |
| fmode(x, y) | остача від ділення x на y |
| M_PI | число π |

Всі функції, крім **abs**, повертають значення типу **double**, типи аргументів теж **double**.

Функція **abs** повертає значення типу **int**, тип аргументу теж **int**.

Приклади запису арифметичних виразів мовою C++ :

$$\frac{e^{2x} + \sin(x-y)^2}{\sqrt{xy} - \ln \frac{x}{2}} \quad (\exp(2*x) + \sin((x-y)*(x-y))) / (\sqrt{x*y} - \log(x/2.))$$

$$\sin^2 x \quad \text{pow}(\sin(x), 2)$$

$$\sqrt[3]{x} \quad \text{pow}(x, 1./3.)$$

10.8. Функції в C++

Всі програми на C++ будуються з функцій. Одна частина функцій перебуває в бібліотеках. Інша — створюється самим користувачем (функції користувача).

Відповідно до загального правила, кожна функція перед використанням повинна бути оголошена. Оголошення бібліотечних функцій (прототипи) перебувають у заголовних файлах, які підключаються до програми за допомогою директиви **#include** <ім'я файлу.h>.

Оголошення математичних функцій підключаються за допомогою директиви **#include** <math.h>.

10.9. Консольний режим. Можливості вводу-виведення C++

Консольний додаток — це програма мовою C++ у середовищі Builder, що запускається без графічного інтерфейсу в консольному вікні.

Ввід, що йде із клавіатури

>> операція вводу

Наприклад:

а) **cin** >> a;

cin — стандартне ім'я потоку уведення. По цій операції дані вводяться в змінну **a**.

б) **cin** >> i >> j >> s;

У цьому випадку дані із клавіатури вводяться в три змінні i, j, s.

Вивід, що йде на екран

<< операція виведення

Наприклад:

а) **cout** <<b;

cout — стандартне ім'я потоку виводу. Значення змінної **b** буде виводитися на екран.

Функції **cin**, **cout** підключаються до проекту за допомогою директиви **#include** <iostream.h>

10.10. Створення консольного додатка

Працюючи в консольному режимі, завжди повинна бути функція з ім'ям **main** (головна), саме з неї починається виконання програми, у якому би місці вона не перебувала.

Приклад 1. Розробити проект для обчислення значення функції y при різних значеннях x .

$$y = a \cdot e^x + \ln c, \text{ де}$$

$$c = \sqrt{b + s \cdot \sin\left(\frac{p}{4}\right)}$$

a = 3,112; b = 5,85; s = 9, 48; значення x вводяться із клавіатури.

Порядок дій:

- створити новий порожній проект, використовуючи пункт меню **File** → **New** у вікні, що відкрилося, вибрати *Console Wizard* → *OK*, в наступному діалоговому вікні активізувати перемикач C++, потім *Console Application* → *OK*

На екрані з'явиться вікно **Unit1.cpp** і заготовка для уведення функції:

```
int main(int argc, char* argv[])
{
    return 0;
}
```

- зберегти проект у новій папці: пункт меню **File** → **Save Project As**
 - ім'я програмного модуля за замовчуванням **Unit1**
 - ім'я модуля проекту **Project1**

На цьому організаційна частина закінчена, сформуємо модуль Unit1.

- внести зміни в заготовку, програма на C++ у консольному режимі має вигляд:

```
#include <math.h>           //для математичних функцій
#include <iostream.h>       //для cin, cout
#include <conio.h>          //для getch(),clrscr ()
//-----
main()
{
double a=3.12, b=5.85, s=9.48; //визначення констант
double x,y;                  //оголошення змінних
clrscr ();                  //функція очищення екрана
cout<<"уведіть x"<<endl;    //вивід на екран і спуск на новий рядок
cin>>x;                    //уведення із клавіатури значення x
c= sqrt(b+s*sin(M_PI/4));
y=a*exp(x)+log(c);
cout<<"y="<<y;
getch();
}
```

- запустити на виконання **RUN [F9]**
увести із клавіатури, наприклад, 5.5
результат $y=762,74$
- зберегти налагоджену програму *File*→ *Save All*.

Функція `getch()` робить затримку екрана виводу й чекає уведення будь-якого символу із клавіатури.

10.11. Стадії проходження програми

1. *Вихідний модуль* (*.cpp) - програма алгоритмічною мовою, передається препроцесору, що виконує директиви, що перебувають у тексті. У текст програми включається вміст зазначених файлів.

#include <ім'я файлу.h> - пошук файлу здійснюється в бібліотеці ОС (бібліотечні файли).

#include "ім'я файлу.h" - пошук файлу здійснюється в поточному каталозі (функції, створені користувачем).

Отриманий текст передається на вхід компілятора.

1. Компілятор виявляє синтаксичні помилки й у випадку їхньої відсутності будує *об'єктний модуль* (*.obj).

2. Компоновщик підключаючи до об'єктного модуля інші об'єктні модулі (бібліотечні файли), формує *здійснений модуль*.

Здійснений модуль має розширення *.exe - це готова до виконання програма.

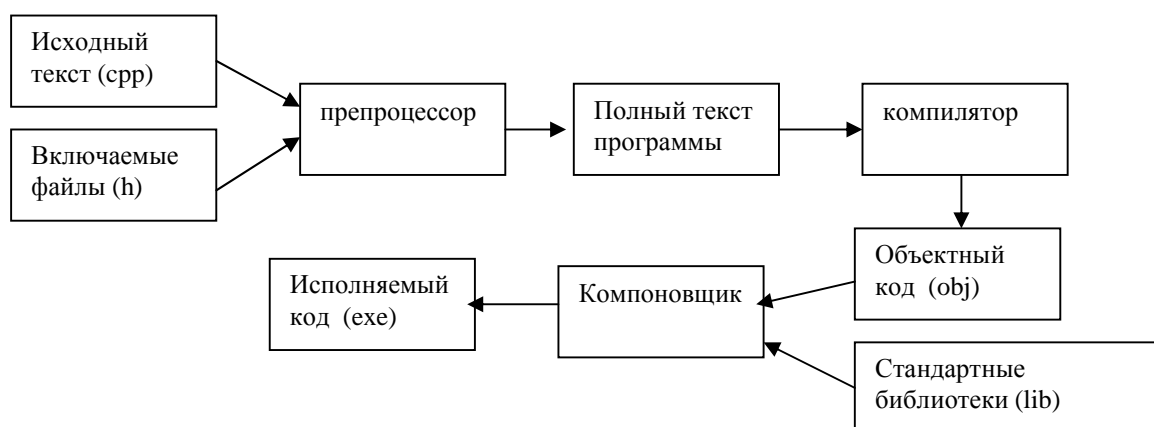


Рис. 10.1. Стадії проходження програми

Тема 11. Windows додатки в графічному середовищі. Операції C++

11.1. Функції приведення типів

StrToInt(String S) — перетворення рядка в ціле число

StrToFloat(String S) — перетворення рядка в число із плаваючою точкою

IntToStr(int Value) — перетворення цілого 10-го числа в строковий тип

FloatToStr(value) — перетворення значення із плаваючою точкою в строкове подання

IntToHex(int value) — перетворення цілого 10-го числа в 16-е

11.2. Операції з рядками

1. Злиття рядків

```
Label1->Caption="Мова" "- " "C++";
```

Виведеться: Мова - C++

2. Злиття рядка й числового виразу

```
Label2 ->Caption =" Результат=" + IntToStr(4*6);
```

Виведеться: Результат=24

10.3 Ввод даних із клавіатури

У графічному режимі ввод даних із клавіатури здійснюється за допомогою функції **InputBox**, що оголошується в заголовному файлі **Stdlib.h**.

Функція повертає значення типу **String**.

Загальний вигляд:

InputBox (“повідомлення”, “підказка”, “Default”)

Default — строковий вираз, що використовується за замовчуванням, якщо користувач не введе рядок. У загальному випадку може бути пробіл.

Наприклад, фрагмент коду:

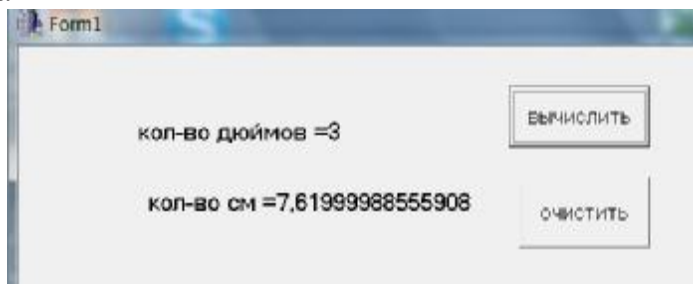
```
{  
float a,b;          // оголошення змінних  
a = StrToFloat(InputBox(" уведіть", " a ", " "));  
b = StrToFloat(InputBox(" уведіть", " b ", " "));  
Label1->Caption = a+b;  
}
```

Приклад 11.1. Створити проект для переведення кількості дюймів, що вводяться з клавіатури, у сантиметри. 1дюйм = 2,54см.

Порядок дій:

- створити новий порожній проект, використовуючи пункт меню **File > New Application**;
- зберегти проект;
- розмістити об'єкти, додати їм необхідні властивості

Дизайн



На сторінці **Standard** палітри компонентів:
об'єкт **Label1**— для виводу кількості дюймів;

об'єкт **Label2** — для виведення кількості см;

об'єкт **Button1** — для запуску проекту, властивість *Caption* → *обчислити*;

об'єкт **Button2** — для очищення полів виводу, властивість *Caption* → *очистити*;

- оброблювач події *циклик по кнопці обчислити* має вигляд

```
#include <Stdlib.h>
```

```
//=====
```

```
void __fastcall TForm1::Button1Click(TObject *Sender)
```

```
{
```

```
float a,inch;
```

```
inch=StrToFloat(InputBox("vv","kd"," "));
```

```
a=inch*2.54;
```

```
Label1->Caption="кількість дюймів "+FloatToStr(inch);
```

```
Label2->Caption="кількість см "+FloatToStr(a);
```

```
}
```

- оброблювач події *циклик по кнопці очистити* має вигляд

```
void __fastcall TForm1::Button2Click(TObject *Sender)
```

```
{
```

```
Label1->Caption="";
```

```
Label2->Caption="";
```

```
}
```

- запустити програму на компіляцію та виконання **[F9] (Run)**, у вікно, що відкрилося, увести з клавіатури кількість дюймів, результат з'явиться в об'єкті **Label2**
- сохоронити налагоджену програму, виконавши команду **Save All**.

Уведення значень у компонент **Edit** (редагуєме поле)

Внесемо зміни в проект:

- установити об'єкт **Edit1** для введення даних строкового типу (сторінка *Standard* палітри компонентів)
- внести зміни в код:
рядок `inch=StrToFloat(InputBox("vv","kd"," "));`
замінити на `inch=StrToFloat(Edit1->Text);`
- запустити проект на виконання *Run*, значення дюймів увести в об'єкт *Edit1* із клавіатури (наприклад 3,5), клацнути на кнопці *обчислити*.

Приклад 11.2. Розробити проект для обчислення значення функції

$$y = a \cdot e^x + \ln c$$

$$c = \sqrt{b + \sin\left(\frac{p}{4}\right)} \quad , \text{ де } a = 3,112; b = 5,85; x = 5,5$$

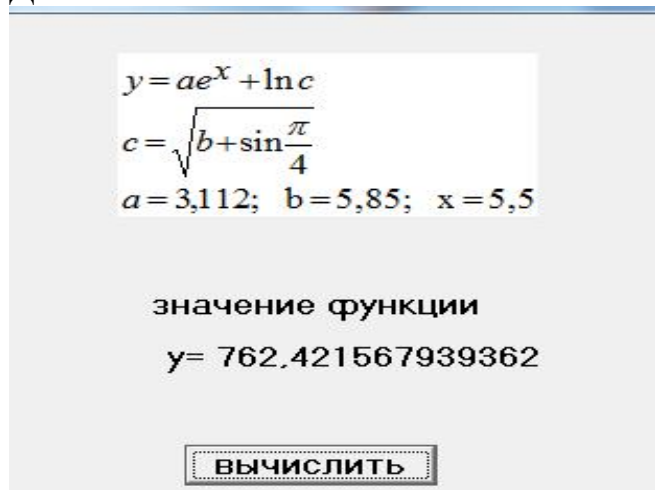
Порядок дій:

- створити новий проект, зберегти його

- розмістити об'єкти й задати їм необхідні властивості:
 об'єкт **Label1** — властивість *Caption* → значення функції;
 об'єкт **Label2** — для виводу результату;
 об'єкт **Button1** — для запуску проекту, властивість *Caption* → *обчислити*;
 об'єкт **Image1** для вставки малюнка (сторінка **Addition** палітри компонентів).

Малюнок вставляється у форматі .bmp у властивість **Picture** об'єкта Image1.

Дизайн



- оброблювач події *циклик по кнопці обчислити*

```
#include <math.h>
```

```
//-----
```

```
void __fastcall TForm1::Button1Click(TObject *Sender)
```

```
{
```

```
    double a=3.112,b=5.85,x=5.5;    //завдання констант
```

```
    double y,c;                    //оголошення змінних
```

```
    c = sqrt(b+sin(M_PI/4));
```

```
    y = a*exp(x)+log(c);
```

```
    Label2->Caption="y = "+FloatToStr(y);
```

```
}
```

- зберегти налагоджену програму, виконавши команду **Save All**.

11.4. Операції відносин

| | | | |
|---|----------|----|---------------------|
| = | дорівнює | != | нерівно |
| > | більше | >= | більше або дорівнює |
| < | менше | <= | менше або дорівнює |

Якщо дві змінні зрівнюються за допомогою операцій відносин, то результат завжди буде логічного типу.

Істина (true) — виробляється значення, відмінне від нуля, найчастіше одиниця.

Неправда (false) — виробляється значення, що дорівнює нулю.

11.5. Логічні операції

Логічні операції в C++ відповідають класичним логічним операціям

&& логічне І
|| логічне АБО
! логічне НІ

Таблиця істинності

| x | y | x && y | x y | !x |
|----------|----------|-----------------------|---------------|-----------|
| 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 |

1 - істина

0 – неправда

Пріоритет логічних операцій нижче пріоритету операцій відносини.

Наприклад, запис мовою C++ умови $x \in [0;100]$ має вигляд

$x \geq 0 \ \&\& \ x \leq 100$

11.6. Операція присвоювання

Позначається знаком =

1. Вираз виду $i = i + 2$ може бути записаний у вигляді $i += 2$

$i *= 2$ еквівалентно $i = i * 2$

$i /= 10$ еквівалентно $i = i / 10$

$i \% = 10$ еквівалентно $i = i \% 10$

2. Багаторазове присвоювання $a = b = c = x * y$ відбувається праворуч-ліворуч. Спочатку обчислюється $x * y$, потім його значення привласнюється змінній **c**, потім **b**, і лише потім змінній **a**.

11.7. Перетворення типів

Якщо у програмі використовуються змінні різних типів, то компілятор автоматично здійснює перетворення.

Правила перетворень

1. В операторі присвоювання тип значення правої частини завжди перетвориться в тип значення лівої частини.

Наприклад,

$\text{int } i = 3.14; // 3.14$ перетвориться до цілого типу, у результаті $i = 3$.

2. В арифметичному виразі нижчий тип завжди перетвориться до вищого, наприклад,

$\text{float} \rightarrow \text{double}, \text{int} \rightarrow \text{long int}$

3. Будь-який вираз може бути наведений до бажаного типу за допомогою конструкції **(тип) вираз**

Наприклад, `int x=5;`
Значенням виразу `(float) x/2;` буде 2.5

11.8. Операції збільшення й зменшення на одиницю

`++` збільшити на одиницю
`--` зменшити на одиницю

Приклад 11.3.

```
int c;  
++ c; //значення c збільшується на одиницю, а потім використовується;  
c++; //значення c використовується, а потім збільшується на одиницю.
```

Примр 11.4. Що буде результатом виконання програми?

```
main ()  
{ int x=5;  
  int y=60;  
  x++;  
  ++y;  
  cout<<x<<y;  
  cout <<x++<<++y;  
}
```

Результат 6 61
 6 62

Приклад 11.5. `c=5;`
`x=c++;` //у результаті `x=5, c=6`

Тема 12. Обчислювальний процес, що розгалужується

12.1. Умовна операція

Загальний вигляд **e1? e2:e3**, де e1, e2, e3 — вираз.

Якщо e1 приймає значення *істина* (тобто $\neq 0$), то значенням цієї конструкції буде e2, у противному випадку e3.

Наприклад, знаходження найбільше з 2-х чисел a і b може бути записано у вигляді `max = (a > b) ? a : b;`

Порожній оператор ;

Цей оператор використовується там, де по синтаксису мови потрібен оператор, а за змістом ніяких дій не виконується.

Складений оператор (блок) береться у фігурні дужки { }. Він може зустрічатися скрізь, де повинен бути один оператор, інакше кажучи, блок еквівалентний одному операторові.

; після блоку не ставиться.

12.2 Умовний оператор IF

Розглянемо два варіанти оператора IF:

а) **IF (умова) оператор1;**

Якщо умова приймає значення *істина* (тобто $\neq 0$), то виконується *оператор1*, у противному випадку — наступний оператор програми.

б) **IF (умова) оператор1;
else оператор2;**

Якщо умова приймає значення *істина* (тобто $\neq 0$), то виконується *оператор1*, у противному випадку виконується *оператор2*.

Приклад 12.1. Знайти дійсні корені квадратного рівняння $ax^2 + bx + c = 0$, де a, b, c — задані числа, $a \neq 0$.

У випадку $D < 0$ вивести повідомлення *дійсних коренів немає*.

Програма в консольному режимі:

```
#include<iostream.h>
#include<conio.h>          //для clrscr ()
#include<math.h>
//-----
main()
{
double a, b, c, d, x1, x2;      //оголошення змінних
clrscr ();                    //функція очищення екрана
cout << " уведіть значення a, b, c " << endl;
cin >> a >> b >> c;
d=b*b-4*a*c;                  //обчислення дискримінанти
if (d >= 0)
{ x1=(-b + sqrt(d))/( 2*a );    //знаходження кореня
  x2=(-b - sqrt(d))/( 2*a );
cout << "x1=" << x1 << endl;
cout << "x2=" << x2;
}
else cout << "дійсних коренів немає";
getch(); }
```


Приклад 12.2. Обчислити значення функції y при різних значеннях x

$$y = \begin{cases} \cos x, & x \leq 0 \\ \arcsin x, & 0 < x \leq \frac{P}{2} \\ \log_4 x, & \frac{P}{2} < x \leq 64 \\ \frac{1}{x^2}, & x > 64 \end{cases}$$

Програма в консольному режимі:

```
#include<math.h>
#include<conio.h>
#include<iostream.h>
//-----
void main()
{
    double x,y;
    clrscr();
    cout<<"уведіть x";
    cin>>x;
    if (x <= 0) y = cos(x);
    if (x > 0 && x <= M_PI_2) y = asin(x);
    if (x > M_PI_2 && x <= 64) y=log(x)/log(4);
    if (x > 64) y=1/pow(x, 2);
    cout << "\n y=" << y;
    getch();
}
```

Запустити програму на компіляцію.

при $x = -3$ одержимо результат $y = -.989972$

при $x = 1$ одержимо результат $y = 1.5702$

Приклад 12.3. Створити в графічному середовищі Builder проект для обчислення значення функції y при різних значеннях x

$$y = \begin{cases} a + b - x, & x \geq 1,5 \\ \sin^2 x + \cos(a + b), & x < 1,5 \end{cases}$$

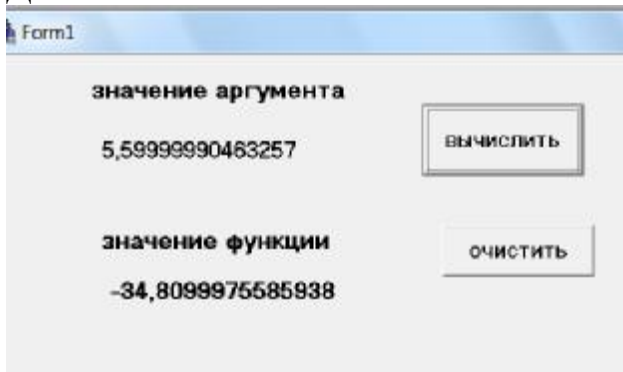
$$a = 1,3 - x^2; \quad b = 0,85$$

Порядок дій:

- створити новий проект і зберегти його;
- розмістити об'єкти, задати їм необхідні властивості на сторінці Standard палітри компонентів:
об'єкт **Label1**, властивість *Caption* → значення аргументу;

- об'єкт **Label2** для виводу значень x ;
- об'єкт **Label3**, властивість *Caption* → значення функції;
- об'єкт **Label4**, для виводу значень y ;
- об'єкт **Button1** для запуску проекту, властивість *Caption* → *обчислити*;
- об'єкт **Button2** для очищення полів, властивість *Caption* → *очистити*.

Дизайн



- оброблювач події *клацання на об'єкті Обчислити*

```

#include <math.h>
//-----
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    float b=0.85;
    float x,y,a;
    x=StrToFloat(InputBox("vvedite","x",""));
    a=1.3-x*x;
    if(x>=1.5) y=a+b-x;
    else
    y=sin(x)*sin(x)+cos(a+b);
    Label2->Caption=x;
    Label4->Caption=y;
}

```
- оброблювач події *клацання на об'єкті Очистити*

```

void __fastcall TForm1::Button2Click(TObject *Sender)
{
    Label2->Caption="";
    Label4->Caption="";
}

```

12.3 Оператор вибору switch

Синтаксис:

```
switch (вираз)
{
    case const1: послідовність операторів; break;
    case const2: послідовність операторів; break;
    .....
    case const: послідовність операторів; break;
    [ default: послідовність операторів; break;]
}
```

Правила виконання оператора:

- обчислюється вираз у заголовку й зрівнюється послідовно з *const1*, *const2*, ... *const*. Як тільки буде знайдена відповідність, виконуються оператори, що випливають за двокрапкою;
 - якщо відповідність ніде не встановлена, то виконуються оператори, що випливають за ключовим словом *default*.
- break** — вихід з оператора *switch* і перехід до наступного оператора програми.

Приклад 12.4. Приклад програми, що забезпечує виведення назв днів тижня по їхньому номеру.

Консольний режим:

```
#include <conio.h>
#include <iostream.h>
//-----
void main()
{
    int dn;
    cout<<"уведіть номер дня тижня";
    cin>> dn;          //у змінну dn уводиться номер дня тижня
    switch (dn)
    {
        case 1 : cout<<"понеділок"; break;
        case 2 : cout<<"вівторок"; break;
        case 3 : cout<<"середа"; break;
        case 4 : cout<<"четвер"; break;
        case 5 : cout<<"п'ятниця"; break;
        case 6 : cout<<"субота"; break;
        case 7 : cout<<"неділя "; break;
        default: cout<<" потрібно ввести ціле число 1-7 "; break;
    }
    getch();}
```

Тема 13. Операції над двійковим кодом

13.1. Порозрядне (побітове) додавання

В обчислювальній техніці інформація кодується двійковим кодом $\{0; 1\}$. Кожному символу ставиться у відповідність код з таблиці ASCII кодів. Будь-який символ у пам'яті комп'ютера займає один байт.

Коди чисел від 1 до 16 в 10-ій, 2-ій, 16-ій системах числення представлені в таблиці.

| Decimal | Binary | Hex |
|---------|-----------|-----|
| 1 | 0000 0001 | 01 |
| 2 | 0000 0010 | 02 |
| 3 | 0000 0011 | 03 |
| 4 | 0000 0100 | 04 |
| 5 | 0000 0101 | 05 |
| 6 | 0000 0110 | 06 |
| 7 | 0000 0111 | 07 |
| 8 | 0000 1000 | 08 |

| Decimal | Binary | Hex |
|---------|-----------|-----|
| 9 | 0000 1001 | 09 |
| 10 | 0000 1010 | 0A |
| 11 | 0000 1011 | 0B |
| 12 | 0000 1100 | 0C |
| 13 | 0000 1101 | 0D |
| 14 | 0000 1110 | 0E |
| 15 | 0000 1111 | 0F |
| 16 | 0001 1111 | 10 |

Приклад 13.1. Додавання двійкових чисел. Перехід від двійкової системи числення в десяткову.

$$0110\ 1101 \quad 2^6 + 2^5 + 2^3 + 2^2 + 2^0 = 109_{10}$$

$$\underline{0010\ 1110} \quad 2^5 + 2^3 + 2^2 + 2^1 = 46_{10}$$

$$1001\ 1011 \quad 2^7 + 2^4 + 2^3 + 2^1 + 2^0 = 155_{10}$$

Коли оперують великими числами, то користуватися двійковою системою незручно, тому переходять до більш стислої системи числення - 16-ій.

Цифри 16-ої системи числення: $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$

Наприклад $0101\ 0001\ 1111_2 \rightarrow 0x51F_{16}$

13.2. Порозрядні логічні операції

Порозрядні логічні операції виконуються над відповідними бітами чисел, що мають цілий тип. Кожний біт має значення 0 або 1.

Порозрядними логічними операціями є:

- &** логічне множення (І)
- |** логічне додавання (АБО)
- ^** виключаюче АБО
- ~** заперечення (НІ)
- <<** зсув вліво
- >>** зсув вправо

Порозрядні логічні операції дозволяють забезпечити доступ до кожного біта інформації.

Таблиця істиності

| значення бітів | | результат операції | | | |
|----------------|----|--------------------|---------|--------|-----|
| e1 | e2 | e1 & e2 | e1 e2 | e1 ^e2 | ~e1 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 |

1 - істина

0 - неправда

Приклад 13.2. Логічне множення (&).

Результат приймає значення 1, якщо обидва біти мають значення 1.

В 2-ій системі числення

$$\begin{array}{r} 1100\ 0001 \\ 0100\ 0010 \\ \hline 0100\ 0000 \end{array}$$

В 16-ій системі числення: $0x0 \& 0xFF \rightarrow F0_{16}$

Із прикладів видно, якщо необхідно встановити значення розряду, що дорівнює нулю, то користуються операцією &.

Приклад 13.3. Логічне додавання (|).

Результат приймає значення 1, якщо хоча б один з бітів має значення 1.

В 2-ій системі числення

$$\begin{array}{r} 1100\ 0001 \\ 0100\ 0010 \\ \hline 1100\ 0011 \end{array}$$

В 16-ій системі числення $0x0 | 0xFF \rightarrow FF_{16}$

Якщо необхідно встановити значення розряду, що дорівнює 1, то користуються операцією |.

Приклад 13.4. Виключаюче АБО (^)

Результат приймає значення 1, якщо значення тільки одного з бітів дорівнює 1 (один і тільки один).

В 2-ій системі числення

$$\begin{array}{r} 1100\ 0001 \\ 0100\ 0010 \\ \hline 1000\ 0011 \end{array}$$

В 16-ій системі числення $0x0 \wedge 0xFF \rightarrow 0F_{16}$

Приклад 13.5. Логічне заперечення (~).

В 2-ій системі числення: ~0100 0001 → 1011 1110

13.3. Таблиця пріоритетів операцій

| Операція | Призначення |
|--------------------|--------------------------------------|
| ++ -- | збільшення (зменшення) на 1 |
| ~ | побітове НІ |
| ! | логічне НІ |
| + - | унарний +, унарний - |
| () | приведення типу |
| * / % | множення, ділення, обчислення остачі |
| + - | додавання, вирахування |
| >> << | зсуви |
| < <= > >= == != | операції відносини |
| & | побітове І |
| ^ | побітове виключаюче АБО |
| | побітове АБО |
| && | логічне І |
| | логічне АБО |
| ?: | умовна операція |
| = += *= | присвоювання |

13.4. Порозрядні зсуви

Операції зсувів застосовуються тільки до цілочисленних змінних.

При використанні операцій зсувів може відбуватися втрата старших або молодших розрядів, відсутні значення бітів при цьому доповнюються нулями, значення змінних змінюються.

Загальний вигляд операції: **змінна >> число позицій**

Всі біти змінної зсуваються вправо на зазначене число позицій (SHR — загальноприйняте позначення).

змінна << число позицій

У цьому випадку всі біти змінної зсуваються вліво на зазначене число позицій (SHL - загальноприйняте позначення).

Приклад 13.6. Ліве зсув на 1 позицію

до засуву 0010 0000 $\rightarrow 2^5 = 32_{10}$

після засуву 0100 0000 $\rightarrow 2^6 = 64_{10}$

Лівий зсув на 1 розряд множить число на 2.

Приклад 13.7. Правий зсув на 1 позицію

до засуву 0010 0000 $\rightarrow 2^5 = 32_{10}$

після засуву 0001 0000 $\rightarrow 2^4 = 16_{10}$

Правий зсув на 1 розряд ділить число на 2.

Приклад 13.8. Фрагмент програми на C++

```
unsigned char bits =9;    //000010012
```

```
bits=bits << 3;         //0100 10002
```

```
bits=bits >> 5;         //0000 00102
```

Тема 14. Циклічний обчислювальний процес. Генерація випадкових чисел. Одномірні масиви

Цикл — це фрагмент програми, що може повторюватися багаторазово. Існує три види взаємозамінних операторів циклу: **for**, **while**, **do...while**

14.1. Оператор циклу FOR

Загальний вигляд оператора:

for (вираз 1; вираз 2; вираз 3) тіло циклу;

вираз 1 — привласнює початкове значення параметра циклу;

вираз 2 — умовний вираз, що задає умову продовження циклу;
 вираз 3 — задає зміну параметра циклу;
 тіло циклу — може бути або простий, або складений оператор.

Алгоритм роботи циклу:

- 1) привласнюється початкове значення параметру циклу
- 2) перевіряється умова продовження циклу.

Якщо умова приймає значення *істина* ($\neq 0$), то виконується тіло циклу, у противному випадку виконується оператор, що впливає за оператором **for**.

Таким чином, перед кожним повторенням циклу відбувається зміна параметра циклу й перевірка умови.

Приклад 14.1. `for (i= 0; i < 10; i ++) cout << i << "\n";`

У результаті роботи циклу в стовпець виведуться цифри від 0 до 9.

Приклад 14.2. `for (i = 9; i >= 0; i --) cout << i << "\n";`

У цьому випадку в стовпець виведуться ті ж цифри від 0 до 9, але у зворотному порядку.

Приклад 14.3. Приклади нескінченних циклів (можуть бути отримані випадково).

`for (i = 1; 1; i ++) оператор;`
`for (i=10; i>6; i++) оператор;`

Приклад 14.4. Обчислити значення функції

$$z = \begin{cases} ax^2 + 1, & x > 1 \\ ay + x, & x \leq 1 \end{cases}$$

$$x = 3I + \cos I$$

$$y = 2 \sin I, \quad a = 3,1; \quad I \in [-5;5]; \quad \Delta I = 0,5$$

Результат вивести у вигляді таблиці

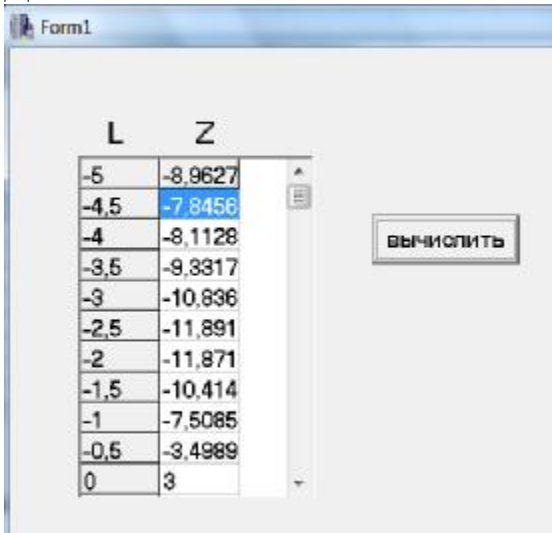
| | |
|----------|----------|
| L | Z |
| | |

Порядок дій:

- для виводу результатів у вигляді таблиці необхідно встановити на формі компонент **StringGrid** (сторінка **Additional**). Визначити кількість рядків у таблиці за формулою $\left[\frac{I_k - I_n}{\Delta I} \right] + 1 = 21$;
- задати властивості компонента *StringGrid*:
ColCount → 2 (кількість стовпців);
RowCount → 21 (кількість рядків);

- установити у форму два компоненти Label, дати їм заголовки **L** і **Z** відповідно;
- установити компонент **Button** для запуску проекту, властивість *Caption* → *обчислити*.

Дизайн:



- оброблювач події *клатання по кнопці Обчислити* має вигляд:


```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    float x, y, l, z, a;           //оголошення змінних
    int i = 0;                    //завдання початкового номера рядка
    a=3.0;
    for (l = -5;l <= 5;l += 0.5)
    {
        x=3*l+cos(l);
        y=2*sin(l);
        if (x > l) z = a*x*x+l;
        else
            z = a*y+x;
        StringGrid1->Cells[0][i]=l;           //Cells[int Col][int Row]
        StringGrid1->Cells[1][i]=z;         //вивод результатів у таблицю
        i++;                                 //зміна номера рядка в таблиці
    }
}
```

Всі результати обчислень можуть бути переглянуті за допомогою вертикальної лінійки компонента StringGrid.

14.2. Оператор циклу WHILE

Загальний вигляд:

while (умова продовження циклу) тіло циклу;

Тіло циклу - простий оператор або складений.

Цикл виконується доти, поки умова приймає значення *істина* ($\neq 0$). Якщо умова приймає значення *неправда*, то керування передається наступному операторові програми.

Так само як і в циклі *FOR* спочатку перевіряється умова, а потім виконується тіло циклу. Це цикли із передумовою.

Приклад 14.5. Для заданого рядка (прізвище користувача, задане латинськими буквами) відобразити ASC коди кожного символу.

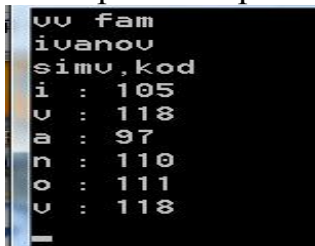
У C++ рядок — це масив символів, що закінчується нульовим байтом. Нульовий байт автоматично додається компілятором.

Алгоритм рішення задачі. Консольний режим.

```
#include<iostream.h>
#include<conio.h>
//-----
void main()
{
    char name [10];           //довжина масиву з обліком '\0'
    cout << "уведіть прізвище " << endl;
    cin >> name;
    cout << " символи прізвища й відповідні їм коди " << "\n";
    int i=0;
    while (name[i]!='\0')
    {
        cout << name[i] << ":" << int(name[i]) << "\n";
        i++;
    }
    getch();}
}
```

Цикл буде працювати доти, поки не зустрінеться ознака кінця рядка '\0'.

Результат роботи проекту:



```
ivanou
i : 105
v : 118
a : 97
n : 110
o : 111
u : 118
```

14.3. Оператор циклу DO...WHILE

Загальний вигляд:

```
do
    тіло циклу
while ( умова продовження циклу );
```

Це цикл із постумовою, тобто спочатку виконується тіло циклу, а потім оцінюється умова продовження циклу. Якщо в результаті оцінки виходить значення *істина*, то цикл повторюється. Якщо отримано значення *неправда*, то цикл завершується.

Таким чином, цикл виконується хоча б один раз.

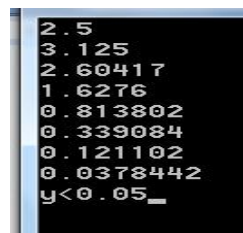
Приклад 14.6. Обчислити значення членів нескінченного ряду

$$x; \frac{x^2}{2!}; \frac{x^3}{3!}; \dots \frac{x^n}{n!}, \dots \quad \text{де } x - \text{ задане число}$$

Обчислення проводити, поки не виконається умова $\frac{x^n}{n!} \leq \epsilon$, де ϵ – задана мала величина (точність обчислення).

Алгоритм рішення в консольному режимі:

```
#include<iostream.h>
#include<conio.h>
//-----
main()
{
    float x=2.5,eps=0.05,y=1;
    int n=1;
    do
    {
        y=y*x/n;
        cout<<y<<endl;
        n=n+1;
    }
    while(y>eps);
    cout<<"y<0.05";
    getch();}
```



```
2.5
3.125
2.60417
1.6276
0.813802
0.339084
0.121102
0.0378442
y<0.05
```

14.4. Генерація випадкових чисел

Функція **rand()** генерує ціле число в діапазоні між **0** і **RAND_MAX** (**RAND_MAX** – символічна константа, яка оголошена в заголовному файлі **<stdlib.h>**). Для того щоб вибрати цілі числа в діапазоні, наприклад від 0 до 5, використовується операція обчислення остачі **%** у сполученні з **rand()**:

Наприклад: **rand() % 6** цілі числа в діапазоні від 0 до 5
 1+rand() % 6 цілі числа в діапазоні від 1 до 6
 rand() % 9 - 4 цілі числа в діапазоні від -4 до 4

Разом з функцією **rand** використовується бібліотечна функція **srand**, що одержує цілий аргумент, і при кожному виконанні програми задає точку входу в таблицю випадкових чисел.

Наприклад **srand (time (NULL));**

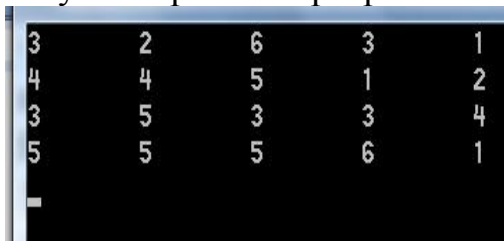
Функція **time** повертає поточний календарний час у секундах.

У результаті, при кожному запуску програми генеруються різні числа.

Приклад 14.6. Моделювання кидання гральної кістки.

```
#include <iostream.h>
#include <stdlib.h>
#include <conio.h>
//-----
void main()
{
srand (time (NULL) );             //вхід у таблицю випадкових чисел
for ( int i = 1; i <= 20 ; i++)    //цикл працює 20 разів
{
cout << 1+rand() % 6 << "\t";    //вивід випадкових чисел від 1 до 6 у рядок
if ( i%5 == 0 )                 //числа виводяться по 5 у рядок
cout<<endl;
}
getch();
}
```

Результат роботи програми:



```
3      2      6      3      1
4      4      5      1      2
3      5      3      3      4
5      5      5      6      1
_
```

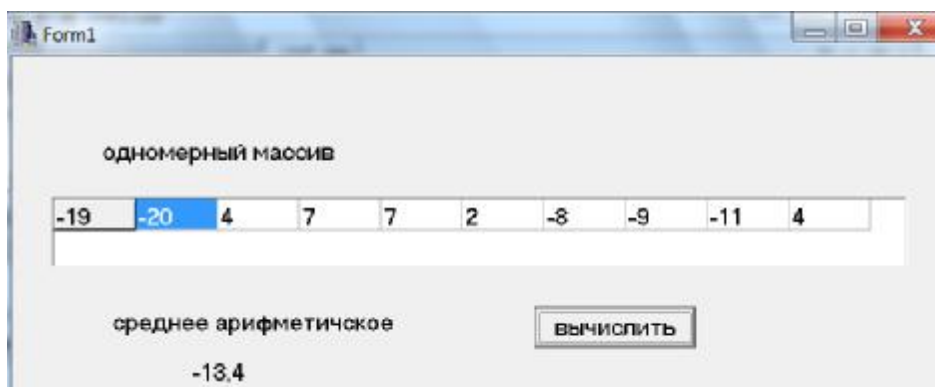
14.5. Одномірні масиви

Приклад 14.7. Згенерувати випадковим чином одномірний масив цілих чисел розмірністю 7 у діапазоні [-10;10]. Графічний режим:

```
srand(time(NULL));
for (i=0; i<7; i++)
{ x[i]=rand()%21-10;           //заповнення масиву випадковими числами
StringGrid1->Cells[i][0] =x[i]; //вивід масиву в рядок
}
```

Приклад 14.8. Згенерувати випадковим чином одномірний масив цілих чисел у діапазоні [-20;20] розмірністю 10. Знайти середнє арифметичне від'ємних елементів масиву.

Реалізація алгоритму в графічному середовищі.
Інтерфейс



Порядок дій:

- розмістити об'єкти, задати їм необхідні властивості:
 - об'єкт **Label1**, властивість *Caption* → *Одномірний масив*
 - об'єкт **StringGrid1** (сторінка **Addition**)
 - властивості **ColCount** → 10 (кількість стовпців)
 - RowCount** → 1 (кількість рядків);
 - об'єкт **Button1** для запуску проекту, властивість *Caption* → *обчислити*
 - об'єкт **Label2**, властивість *Caption* → *Середнє арифметичне*
 - об'єкт **Label3** для виводу результату
- оброблювач події *кляцання по кнопці Обчислити* має вигляд:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    int x[10];
    double sa;
    int i, sum=0, k=0;
    srand(time(NULL));
    for (i=0;i<=9;i++)
```

```

{
x[i]=rand()%41-20;           //заповнення масиву випадковими числами
StringGrid1->Cells[i][0] =x[i]; //вивід масиву
if(x[i]<0)
{
sum+=x[i];
k++;
}}
sa=(double)sum/(double)k;    //пошук середнього арифметичного
Label3->Caption=sa;
}

```

14.6 Динамічний розподіл пам'яті

У мові C++ змінні й масиви можуть ініціалізуватися динамічно під час виконання програми. Це дозволяє створювати саме той об'єкт, що потрібен, використовуючи інформацію, що стає відомою тільки на етапі виконання.

Динамічне виділення пам'яті для об'єкта здійснюється за допомогою оператора **new**, при цьому використовується операція покажчика на адресу об'єкта (*).

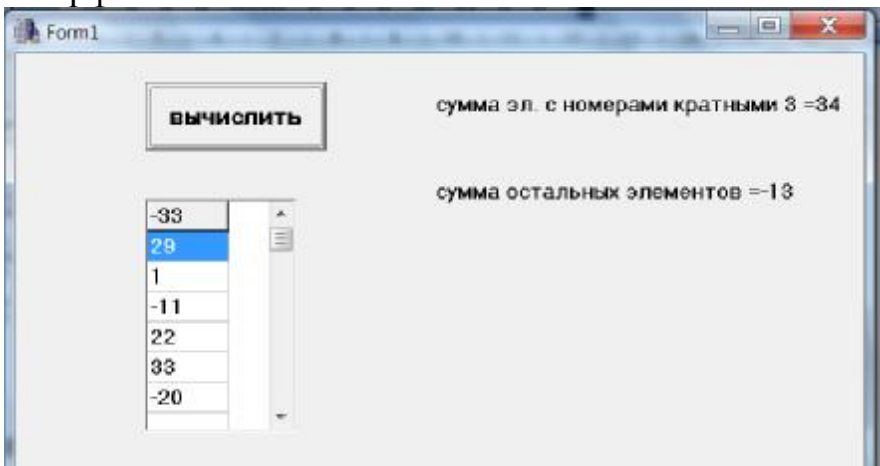
Наприклад

```
int*arr=new int[size]; //*arr - адреса масиву arr
```

У цьому випадку, за адресою масиву з ім'ям *arr* розмірністю *size*, оператором **new** буде виділена пам'ять.

Приклад 14.6. Згенерувати випадковим чином масив цілих чисел будь-якої розмірності. Знайти суму елементів з номерами, кратними 3, і суму інших елементів.

Інтерфейс



Порядок дій:

- установити об'єкти Label1, Label2 для виводу результатів

об'єкт Button1 для запуску проекту, властивість *Caption*→*обчислити*
об'єкт StringGrid1 для виводу елементів масиву

- оброблювач події *кляцання по кнопці Обчислити* має вигляд:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    int i;
    float s1,s2;
    int size;
    size=StrToFloat(InputBox("vv","size","")); //уведення розмірності масиву
    int*arr=new int[size]; //динамічне виділення пам'яті
    srand (time(NULL));
    for (i=0;i<size;i++)
    {
        arr[i]=50-rand()%100; //заповнення масиву випадковими числами
        StringGrid1->Cells[0][i]=arr[i]; //вивід масиву
    }
    s1=0;s2=0;
    for (i=0;i<size ;i++)
    if((i+1) %3==0 )
    s1=s1+arr[i];
    else
    s2=s2+arr[i];
    Label1->Caption="сума элем. з ном. кратними 3="+FloatToStr(s1);
    Label2->Caption="сума інших елементів="+FloatToStr(s2);
}
```
- запустити проект на виконання. Увести із клавіатури розмірність масиву, наприклад, 7. Результат див. на Формі.

Тема 15. Двомірні масиви. Алгоритми реалізації

Приклад 15.1. Ініціалізація двомірного масиву A(4,5). Значення генеруються випадковим чином у діапазоні від 0 до 10.

```
int a[4][5];
for ( i = 0; i<4; i++)
for ( j = 0; j<5; j++)
a[i][j] = rand() % 11;
```

Приклад 15.2. Уведення двомірного масиву A(2,3) із клавіатури. Консольний режим.

```
double a[2][3];
for ( int i=0; i<2; i++)
```

```
for ( int j=0; j<3; j++)
cin >> a[i][j];
```

Приклад 15.3. Вивід двомірного масиву A(4,5) по рядках. Консольний режим.

```
for ( i = 0; i<4; i++)
{
for ( j = 0; j<5; j++)
cout<<a[i][j];
cout<<endl;
}
```

Приклад 15.4. Знайти суму позитивних елементів двомірного масиву

$$A = \begin{pmatrix} 1 & 0 & 3 \\ 5 & -4 & 9 \end{pmatrix}$$

Алгоритм рішення. Консольний режим.

```
#include <iostream.h>
#include <conio.h>
//-----
{ int a[2][3]={ {1,0,3},{5,-4,9}};
int i, j, s=0;
for (i=0;i<2;i++)
{
for (j=0;j<3;j++)
if (a[i][j]>0) s+=a[i][j];
}
cout <<"s="<<s<<"\n";
getch();
}
```

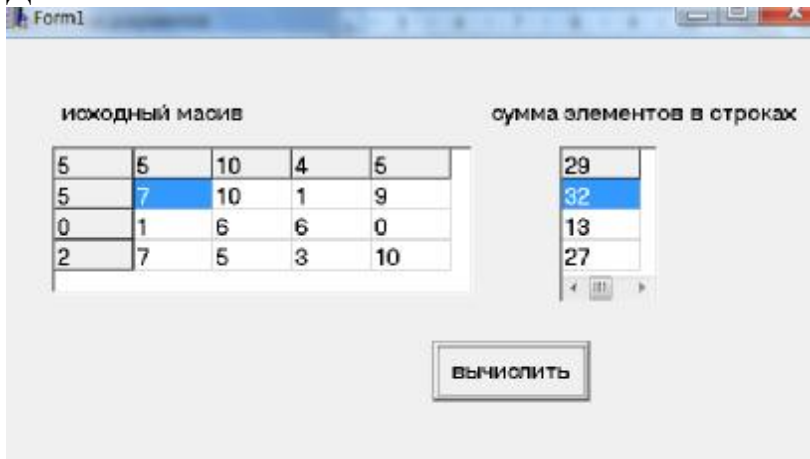
Приклад 15.5. Заповнити двомірний масив A(4,5) випадковими цілими числами з діапазону [0;10]. Знайти суму елементів у кожному рядку. Створити проект у графічному середовищі.

Порядок дій:

- розмістити об'єкти, задати їм необхідні властивості:
 - ü об'єкт **Label1**, властивість *Caption* → *Вихідний масив*;
 - ü об'єкт **Label2**, властивість *Caption* → *Сума елементів у рядках*;
 - ü об'єкт **StringGrid1** для виводу елементів масиву:
 - властивості **ColCount** → 5 (кількість стовпців);
 - RowCount** → 4 (кількість рядків);
 - ü об'єкт **StringGrid2** для виводу результатів:
 - властивості **ColCount** → 1;
 - RowCount** → 4;

ü об'єкт **Button1** для запуску проекту, властивість *Caption*→ *обчислити*.

Дизайн



Оброблювач події *циклик по кнопці обчислити* має вигляд:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    float s,a[4][5];
    int i,j;
    srand(time(NULL));           //ініціалізація генератора випадкових чисел
    for (i=0;i<4;i++)
    for (j=0;j<5;j++)
    {
        a[i][j]=rand()%11;       //заповнення масиву випадковими числами
        StringGrid1->Cells[j][i]=a[i][j]; //заповнення таблиці випадковими числами
    }
    for (i=0;i<4;i++)
    {
        s=0;
        for (j=0;j<5;j++)
            s+=a[i][j];          //обчислення суми в рядку
        StringGrid2->Cells[0][i]=s; //вивід результатів у таблицю
    }
}
```

- запустити на виконання RUN

Тема 16. Функції користувача в C++. Рекурсивні функції

З використанням функції зв'язано три поняття:

- опис функції — опис дій, які виконує функція;
- оголошення функції;
- звертання до функції.

Опис функції

Загальний вигляд оператора:

```
[тип] ім'я функції ([список формальних параметрів])  
{  
    тіло функції  
}
```

Перший рядок — заголовок функції.

Тип — це тип значення, що функція повертає за допомогою оператора **return**. Якщо тип не заданий, то за замовчуванням мається на увазі *int*. Якщо функція не повертає значення, то тип *void*.

Формальні параметри — це змінні, які використовуються усередині тіла функції. Вони одержують значення при виклику функції шляхом копіювання в них значень відповідних фактичних параметрів.

Для кожного формального параметра повинен бути зазначено тип.

Оператор **return**

Загальний вид оператора: **return** [вираження];

Цей оператор забезпечує:

- вихід з функції й повернення в визивну програму;
- повернення значення функції.

У тілі функції може бути кілька операторів *return* або жодного.

Приклад 16.1. Функція користувача для знаходження найбільшого з 2-х цілих чисел. Можливі варіанти:

| | | | |
|----|--|----|---|
| a) | <pre>max (int a, int b) { if (a>b) return a; else return b; }</pre> | б) | <pre>max (int a, int b) { int m; if (a>b) m=a; else m=b; return m; }</pre> |
|----|--|----|---|

Якщо оператор *return* відсутній, то повернення в визивну програму відбувається після виконання останнього оператора тіла функції, а значення, що повертається не визначено.

Оголошення функції

Особливістю стандарту мови C є те, що для створення правильного машинного коду до першого виклику функції необхідно повідомити тип результату, що повертається, а так само кількість і типи аргументів.

Оголошення функції:

```
[тип] ім'я функції ([список формальних параметрів]);
```

Оголошення функції може збігатися із заголовком в описі функції, хоча це й не завжди так. При оголошенні функції імена формальних параметрів не грають ніякої ролі й ігноруються компілятором.

Тому оголошення функції (див. приклад 16.1.) може виглядати так:

```
max (int a, int b)
```

або так:

```
max (int , int )
```

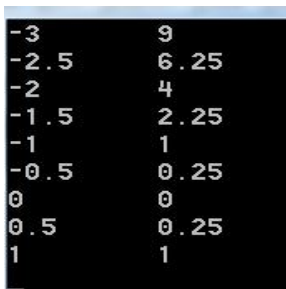
Звертання до функції має вигляд:

ім'я функції ([список фактичних параметрів])

Приклад 16.2. Створити функцію для обчислення квадрата числа. Скористатися нею при обчисленні квадратів чисел з діапазону [-3; 1] із кроком 0,5. Алгоритм рішення задачі. Консольний режим.

```
#include <iostream.h>
#include <conio.h>
//-----
float sqr(float a);           //це оголошення функції (прототип)
main ()                       //це головна функція
{
    float b;
    for (b=-3; b<=1;b+=0.5)
        cout << b << "\t " <<sqr(b)<<"\n";
    getch();
}
float sqr(float a)           //це опис функції
{
    return a*a;
}
```

Результат роботи програми



```
-3      9
-2.5    6.25
-2       4
-1.5    2.25
-1       1
-0.5    0.25
0        0
0.5     0.25
1        1
```

У даному прикладі оголошення функції *sqr* перебуває перед головною функцією *main*, а опис функції — після.

Приклад 16.3. Знаходження найбільшого з 2-х цілих чисел. Алгоритм рішення задачі. Консольний режим.

```
#include<iostream.h>
#include<conio.h>
```

```

int max(int x, int y)      //опис функції max збігається з оголошенням
{
    return (x>y)? x:y;
}
void main()                //це головна функція
{ int a,b;
  cin>>a>>b;
  cout<<max(a,b);        //звертання до функції
  getch();
}

```

У даному прикладі опис функції перебуває перед головною функцією, тобто збігається з оголошенням .

16.3. Область дії змінних

Існує три типи змінних:

- локальні (внутрішні)
- глобальні
- формальні параметри

Локальна змінна існує тільки в тому блоці, у якому оголошена. При виході із блоку ця змінна і її значення губляться.

Область дії локальної змінної — блок.

Наприклад:

```

for ( int i=0; i<10; i++)
{ тіло циклу }
i=0;
.....

```

У цьому випадку перша змінна *i* відома тільки в циклі *for* , а у виразі *i=0*; це буде вже інша змінна, тобто їй компілятор привласнить зовсім іншу адресу.

Глобальна змінна — це змінна, оголошена поза якою-небудь функцією й може бути використана в будь-якому місці програми.

Область дії глобальної змінної — вся програма.

Формальні параметри — це також локальні параметри. Їхня область дії — блок, що є тілом функції.

16.4. Рекурсивні функції

У мові С++ функції можуть викликати самі себе. Функція називається рекурсивною, якщо оператор у тілі функції містить виклик цієї ж функції.

Приклад 16.4. Класичний приклад рекурсивної функції — обчислення факторіала числа, $N!=1*2*3*...*N$

Нехай ця функція називається **factorial**.

Програма на С++ у консольному режимі має вигляд:

```
#include<iostream.h>
```

```

#include<conio.h>
//.....
long int factorial (int n);           //це оголошення функції
void main()                          //це головна функція
{
    for (int i=0; i<=10; i++)
        cout<<i<<"!="<<factorial(i)<<"\n"; //звертання до функції
    getch();
}
long int factorial(int n)            //це опис функції
{
    if (n==0 | n==1) return 1;
    return factorial(n-1)*n;
}

```

Результат роботи програми

```

0! = 1
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800

```

Література

1. Шалин П. А. Енциклопедия Windows XP. - СПб.:Питер, 2003. - 688 с.
2. Гарнаев А. Ю. Самоучитель VBA. 2-е изд., перераб. и доп. - СПб.:БХВ-Петербург, 2004. -500 с.:ил.
3. Швачич Г. Г., Овсянников О. В. та ін. Информатика та комп'ютерна техніка. Елементи об'єктно-орієнтованого програмування. Розділ «Реалізація концепції об'єктно-орієнтованого програмування в мові Visual Basic for Application»: Навчальний посібник. - Дніпропетровськ: НМетАУ, 2006.-52 с.
4. Информатика. Базовый курс. 2-е издание / Под ред. С. В. Симоновича. - СПб.:Питер, 2009. - 640 с.:ил.
5. Березин Б.И., Березина С.Б. Начальный курс С и С++. - М.:Диалог-МИФИ, 2001. - 288 с.
6. Павловская Т.А. С/С++. Программирование на языке высокого уровня. Учебник для вузов. - СПб.:Питер, 2003. - 400 с.
7. Пахомов Б.И. С/С++ и Borland С++ Builder для студентов. - СПб.: БХВ-Петербург, 2006. - 448 с.

Навчальне видання

Швачич Геннадій Григорович

Гуляєва Олена Анатоліївна

КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ ТА ПРОГРАМУВАННЯ

Конспект лекцій

Тем. план 2011, поз. 221

Підписано до друку 11.05.2011. Формат 60x84 1/16. Папір друк. Друк плоский.
Облік.-вид. арк. 4,59. Умов. друк. арк. 4,52. Тираж 100 пр. Замовлення №

Національна металургійна академія України
49600, м. Дніпропетровськ-5, пр. Гагаріна, 4

Редакційно-видавничий відділ НМетАУ