

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ ТА ПРОГРАМУВАННЯ

1. ОСНОВИ АЛГОРИТМІЗАЦІЇ

**МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ
ЛАБОРАТОРНИХ І ПРАКТИЧНИХ РОБІТ ТА САМОСТІЙНОЇ РОБОТИ
ДЛЯ СТУДЕНТІВ НАПРЯМУ ПІДГОТОВКИ**

6.050202 «АВТОМАТИЗАЦІЯ ТА КОМП'ЮТЕРНО-ІНТЕГРОВАНІ ТЕХНОЛОГІЇ»

НАВЧАЛЬНЕ ЕЛЕКТРОННЕ ВИДАННЯ

Київ 2014

Комп'ютерні технології та програмування 1. Основи алгоритмізації: метод. вказівки до викон. лаб. і практ. робіт та самост. роботи для студ. напряму підготовки 6.050202 «Автоматизація та комп'ютерно-інтегровані технології» (Навч. електронне видання)/ Автори: О.О. Квітка, А.М. Шахновський, С.Л. Мердух – К.: 2014. – 94 с.

Гриф надано Вченою радою ХТФ,
протокол № 5
від «26» травня 2014 р.

Навчальне електронне видання

КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ ТА ПРОГРАМУВАННЯ

1. ОСНОВИ АЛГОРИТМІЗАЦІЇ

методичні вказівки до виконання

лабораторних і практичних робіт та самостійної роботи

для студентів напряму підготовки

6.050202 «Автоматизація та комп'ютерно-інтегровані технології»

Автори: Квітка Олександр Олександрович
Шахновський Аркадій Маркусович
Мердух Світлана Леонідівна

Відповідальний О.В. Сангінова, к.т.н., доц.
редактор:

© О.О. Квітка, А.М. Шахновський, 2014 р.

ЗМІСТ

ПЕРЕДМОВА	4
1. СТВОРЕННЯ ТА ВИКОНАННЯ ПРОГРАМ У СЕРЕДОВИЩІ РЕДАКТОРА VISUAL BASIC	5
2. БЕЗПОСЕРЕДНЄ ПРОГРАМУВАННЯ	12
3. РОЗРОБКА ТА ПРОГРАМУВАННЯ АЛГОРИТМІВ РОЗГАЛУЖЕНОЇ СТРУКТУРИ	28
4. ПРОГРАМУВАННЯ РОЗГАЛУЖЕНЬ ЗА ДОПОМОГОЮ ОПЕРАТОРА БАГАТОВАРІАНТНОГО ВИБОРУ	38
5. ОРГАНІЗАЦІЯ АРИФМЕТИЧНИХ ЦИКЛІВ	43
6. ОРГАНІЗАЦІЯ ВКЛАДЕНИХ ЦИКЛІВ	51
7. ОДНОВИМІРНІ МАСИВИ. ХАРАКТЕРНІ ПРИЙОМИ ПРОГРАМУВАННЯ	58
8. ВИКОНАННЯ ОПЕРАЦІЙ З ДВОВИМІРНИМИ МАСИВАМИ	74
9. ВИДІЛЕННЯ ОКРЕМИХ ОБЛАСТЕЙ ДВОВИМІРНИХ МАСИВІВ	81
СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ	87
ДОДАТОК А.	88
ДОДАТОК Б.	91
ДОДАТОК В.	92

ПЕРЕДМОВА

Розвиток науки та промисловості вимагає від сучасних фахівців із комп'ютерно-інтегрованих технологій, хімічної технології та інженерії високого рівня професійної підготовки в області інформаційних технологій.

Кваліфікований фахівець з напрямку підготовки “Автоматизація та комп'ютерно-інтегровані технології” має успішно застосовувати у своїй професійній діяльності знання та навички «прикладного програмування», зокрема алгоритмізації практичних задач, поставлених замовником, розробки раціональної архітектури та створення засобами сучасних середовищ програмування програмних продуктів з метою виконання технологічних та техніко-економічних обчислень, програмування засобів автоматизації, тощо.

Створення ефективних програмних продуктів у сучасних середовищах програмування, крім володіння синтаксисом відповідної мови програмування, специфічними засобами середовища програмування, та знання візуальних компонентів, вимагає від програміста насамперед вміння алгоритмізувати вихідну задачу.

Оскільки мова програмування Visual Basic використовується, зокрема, для автоматизації додатків MS Office, при програмуванні багатьох засобів в складі автоматизованих систем керування технологічними процесами, тощо, то можна сміливо твердити, що використання MS Visual Basic for Applications у якості навчального середовища програмування не тільки дає можливість опанувати базові навички алгоритмізації, а й надає можливість здобути навички автоматизації задач, що виникають при роботі у поширених прикладних програмах (в першу чергу – в MS Excel), а також дає також змогу студентам засвоїти синтаксис мови програмування сімейства Visual Basic, яка є однією з найпоширеніших в складі засобів створення прикладних програмних продуктів.

Методичні вказівки розроблено відповідно до програми підготовки бакалаврів за напрямом підготовки «Автоматизація та комп'ютерно-інтегровані технології» і є складовою частиною дисципліни «Комп'ютерні технології та програмування». Ця дисципліна є базовою у підготовці бакалаврів вказаного напрямку.

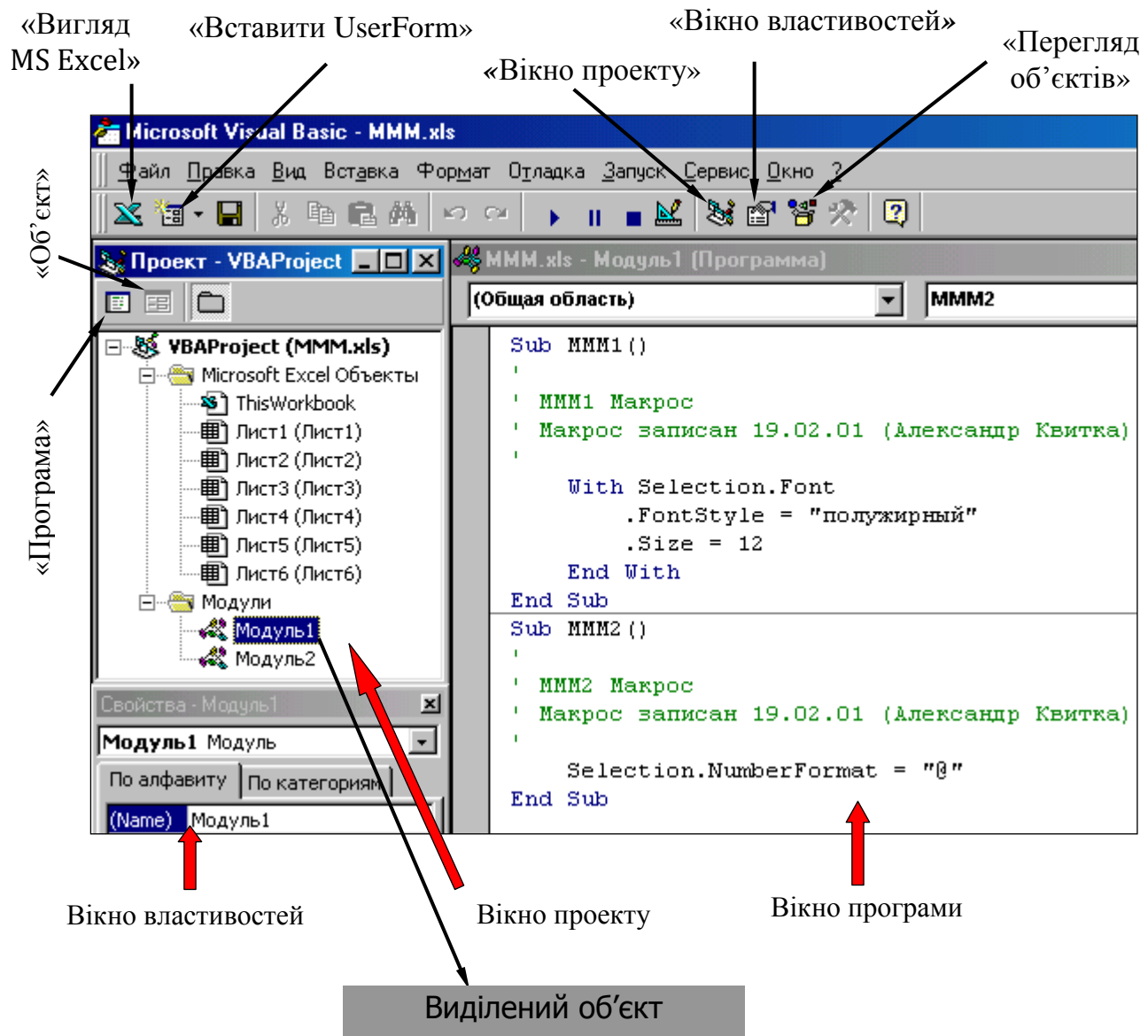
У методичних вказівках наведені мета та завдання для лабораторних робіт, стисло викладений необхідний теоретичний матеріал та зазначено літературу для поглибленого вивчення, надано порядок виконання робіт та обробки результатів, перелічено вимоги до оформлення звітів та порядку їх подання, сформульовано контрольні питання для самопідготовки студентів та визначені засоби безпеки, яких слід дотримуватися при виконанні робіт.

1. Створення та виконання програм у середовищі редактора Visual Basic

Мета та основні завдання роботи: дослідити особливості організації середовища програмування Visual Basic. Набути досвіду роботи з інтерфейсом середовища Visual Basic for Application (VBA).

1.1 Основні теоретичні відомості

Інтерфейс редактора Visual Basic



Створення процедури

Для створення процедури спочатку необхідно вставити в робочу книгу «*Модуль*», де буде записана процедура:

1. Відкриємо вікно редактора «Visual Basic».
2. У «**Вікні проекту**» (якщо воно відсутнє, його треба відкрити) необхідно вико-

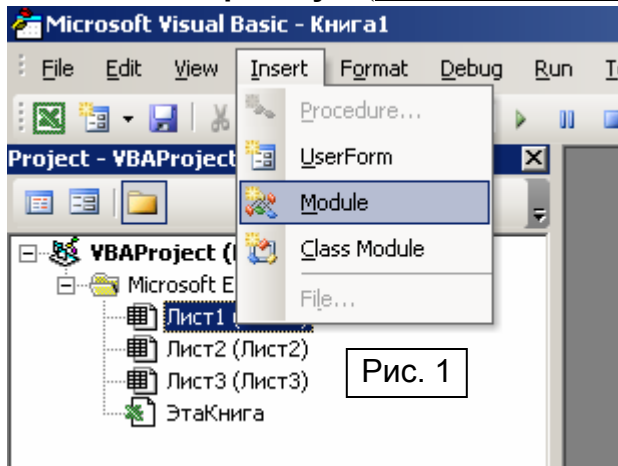
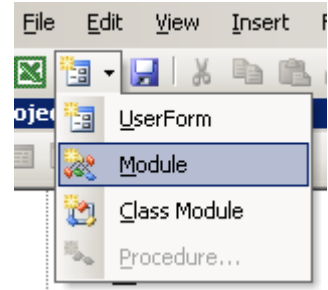
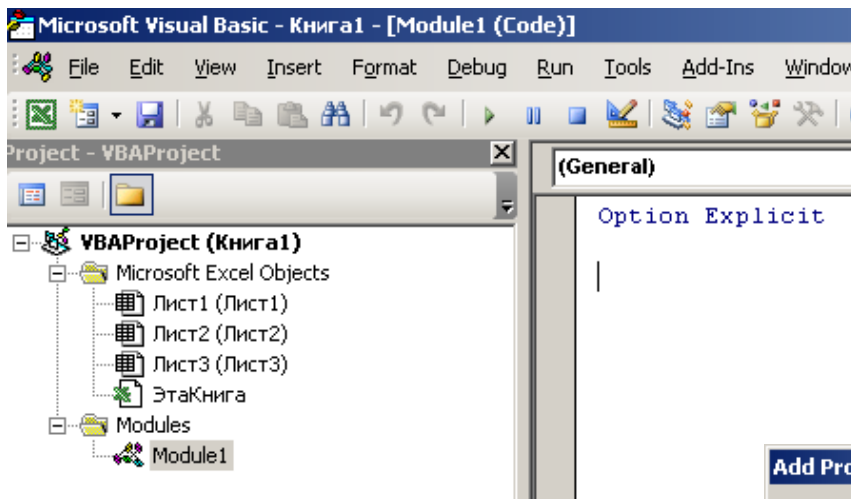


Рис. 1

нати команду «*Insert*» → «*Module*» (рис. 1). Можна також скористатися контекстним меню або кнопкою на стандартній панелі інструментів зі списком «*Insert*» → «*UserForm/ Module/...*». В результаті у «Вікні проекту» додається елемент «*Module1*» та



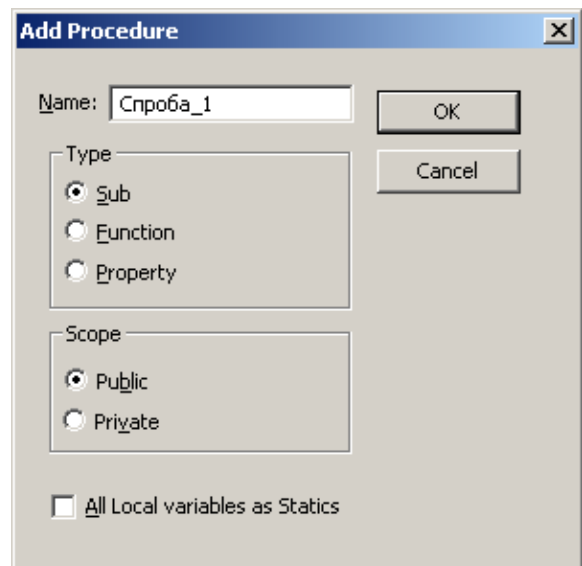
відкривається «Вікно програми», яке відповідає цьому модулю (рис. 2).



Оператор Option Explicit, з якого починається текст програми, потребує явного опису змінних у програмі. Він може бути присутнім або бути відсутнім залежно від налаштування параметрів в редакторі «Visual Basic». Якщо цей оператор відсутній, його треба

набрати вручну.

3. За допомогою команди «*Insert*» → «*Procedure*» або користуючись вже відомою кнопкою зі списком «*Insert*» → «*Procedure/UserForm/Module/...*» відкриваємо діалогове вікно «*Add Procedure*». В цьому вікні необхідно задати ім'я процедури (перші три символи – букви), тип – Sub (інші параметри приймаються за умовчанням).



Option Explicit

```
Public Sub Спроба_1()  
|  
End Sub
```

Потім натиснути «ОК». Після цього в текст модуля додається процедура з введеним ім'ям (оператори заголовку та закінчення).

4. В рядку, наступному після заголовку процедури, починаємо введення її програмного коду. Нехай програмний код буде таким: MsgBox "Це пробна процедура".

При наборі програмного коду після введення імені функції (**MsgBox**) з'являється підказка з форматом цієї функції:

```
Public Sub Спроба_1()  
    msgbox  
End Sub
```

MsgBox(**Prompt**, [Buttons As VbMsgBoxStyle = vbOKOnly], [Title], [HelpFile], [Context]) As VbMsgBoxResult

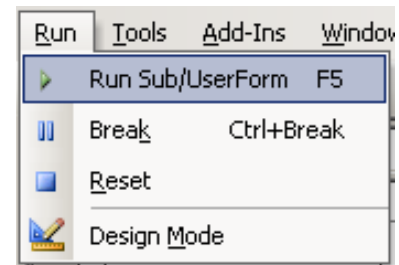
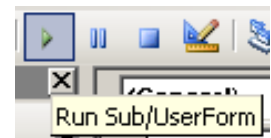
Остаточний текст процедури виглядає так:

```
Public Sub Спроба_1()  
    MsgBox "Це пробна процедура"  
End Sub
```

Виконання процедури

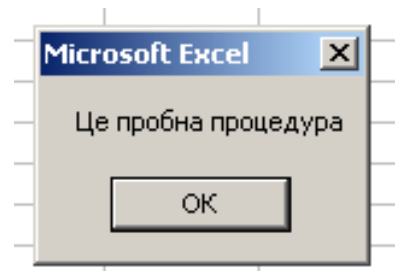
Ініціювати виконання процедури можна різними способами:

- ✓ Клацнути по кнопці «**Run Sur/UserForm**» на стандартній панелі інструментів.
- ✓ Виконати команду з основного меню «**Run**» → «**Run Sur/UserForm**».
- ✓ Натиснути клавішу **F5**.



Розглянемо як це відбувається на прикладі створеної процедури. Виконання процедури приведе до переходу в «Книгу» MS Excel та виведенню вікна із запрограмованим повідомленням.

Після натиснення на кнопку **ОК** вікно закриється, і ми знову повернемося в середовище редактора «Visual Basic».



Збереження модуля з програмним кодом

Усі модулі, які входять в структуру VBA-проекту зберігаються в складі «Книги» MS Excel, до складу якої вони входять. Тому для збереження модуля (тексту створеної програми) необхідно зберегти дану «Книгу» MS Excel.

Приклад редагування процедури

Створимо нову процедуру, користуючись створеною вище.

Для цього скопіюємо три рядки існуючої процедури і вставимо їх у цьому ж модулі. Відразу відредагуємо назву нової процедури – **Спроба_2**.

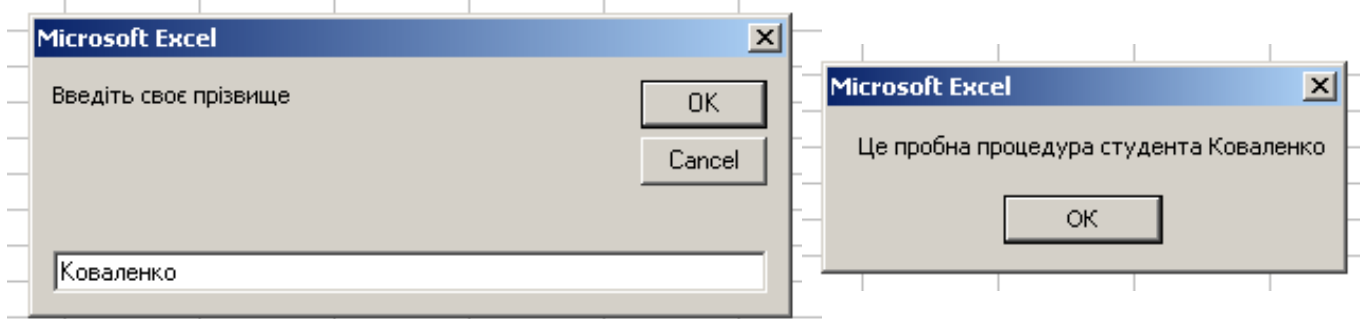
Додамо відразу після заголовку процедури новий рядок для введення прізвища користувача за допомогою функції **InputBox**. Як і у випадку з функцією **MsgBox**, при введенні функції **InputBox** з'являється підказка з форматом цієї функції:

```
sName = inputbox(  
MsgBox " InputBox(Prompt, [Title], [Default], [XPos], [YPos], [HelpFile], [Context]) As String
```

Після закінчення редагування процедура матиме такий вигляд:

```
Public Sub Спроба_2()  
    Dim sName As String  
    sName = InputBox("Введіть своє прізвище")  
    MsgBox "Це пробна процедура студента " & sName  
End Sub
```

При виконанні цієї процедури спочатку з'явиться вікно для введення прізвища користувача, а потім – «вікно результату»:




1.2 Опис лабораторних засобів та обладнання

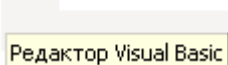
Лабораторна робота №1 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

1.3 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

1.4 Послідовність виконання лабораторної роботи № 1.

1. Відкрити нову книгу MS Excel, включити панель інструментів «*Visual Basic*» та за її допомогою (кнопка ) відкрити вікно редактора Visual Basic.



2. Ознайомитись з інтерфейсом редактора Visual Basic:
 - призначення вікон та способи переключення між ними;
 - призначення основних пунктів меню;
 - можливості використання довідкової системи.
3. За допомогою або команди «*Insert*»→«*Module*», або відповідної кнопки на панелі інструментів вставити в проект модуль для запису програмного коду.
4. У вікні створеного модуля вставити нову процедуру, для чого використати команду «*Insert*»→«*Procedure...*», або відповідну кнопку на панелі інструментів. У вікні «*Add Procedure*», що відкриється, ввести в якості назви процедури своє прізвище і натиснути кнопку **OK**¹. Після цього у вікні модуля з'являються оператори заголовку та закінчення процедури (**Public Sub <ім'я процедури>()** та **End Sub**, відповідно).
5. Після заголовку процедури (**Public Sub <ім'я процедури>()**) набрати текст програми² розв'язання квадратного рівняння³:

Option Explicit

Public Sub QuadraticEquation()

Dim a As Single, b As Single, c As Single, d As Single

Dim x As Single, x1 As Single, x2 As Single, sText As String, sName As String

' Приклад програми розв'язку квадратного рівняння

' Студент _____, гр. _____

sName = InputBox("Введіть своє прізвище")

Введення коефіцієнтів квадратного рівняння

¹ Всі інші параметри приймаються за умовчанням.

² Якщо перед заголовком процедури немає оператора **Option Explicit**, його слід набрати.

³ Коментарі, надруковані жирним шрифтом на темному фоні, набирати не треба. Заголовок процедури створює програма, а рядки коментарів (виділені сірим фоном) надані лише для пояснень.

```

a = InputBox("Введіть коефіцієнт а квадратного рівняння" & vbNewLine & "a*x^2 + b*x + c = 0")
b = InputBox("Введіть коефіцієнт b квадратного рівняння" & vbNewLine & "a*x^2 + b*x + c = 0")
c = InputBox("Введіть коефіцієнт c квадратного рівняння" & vbNewLine & "a*x^2 + b*x + c = 0")

```

'Поступове формування у змінній **sText** довгого тексту для виведення результатів
'знак & використовується для додавання наступної частини інформації, що виводиться
'інформації, що записана після константи **vbNewLine** буде виводитись з нового рядка

```

sText = "Розв'язок квадратного рівняння"
sText = sText & vbNewLine & a & "x^2 + " & b & "x + " & c & " = 0"
sText = sText & vbNewLine & "виконав студент " & sName

```

'Обчислення дискримінанту

```
d = b * b - 4 * a * c
```

'Розгалуження розрахунку на три гілки в залежності від знаку дискримінанту

```
If d < 0 Then
```

'Виведення результату

```

MsgBox sText & vbNewLine & "Нема дійсних коренів"
ElseIf d = 0 Then
    x = -b / a / 2

```

'Виведення результату

```

MsgBox sText & vbNewLine & " - Корінь один: x=" & x
Else
    d = Sqr(d)
    x1 = (-b + d) / (2 * a)
    x2 = (-b - d) / (2 * a)


```

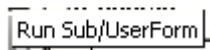
'Виведення результату

```

MsgBox sText & vbNewLine & "Коренів два: x1=" & x1 & " x2=" & x2
End If
End If
End Sub

```

6. Налаштувати набрану програму. Для запуску програми на виконання слід або натиснути клавішу **F5** на клавіатурі, або натиснути на панелі інструментів кнопку , або виконати команду «**Run**» → «**Run Sub/UserForm**».



7. Підібрати необхідні значення коефіцієнтів квадратного рівняння для того щоб, перевірити роботу кожної гілки програми.
 8. Розв'язати три підібрані квадратні рівняння за допомогою цієї програми. В ході розв'язку необхідно зробити скріншоти вікон⁴, що виводять результати розрахунку. Скріншоти роздрукувати.
 9. Зберегти файл MS Excel з набраною програмою.
 10. Продемонструвати викладачу роботу програми.
- Оформити звіт про виконання лабораторної роботи⁵.

⁴ Щоб зберегти скріншот активного вікна слід використати клавіші **Alt + PrintScreen**.

⁵ В звіті повинні бути наведені скріншоти вікон з результатами розрахунків.

1.5 Обробка та аналіз результатів. Оформлення звіту

Звіти з усіх лабораторних робіт виконуються на листах формату А4⁶. Протокол кожної лабораторної роботи починається з номера та теми роботи, прізвища студента та шифру групи. Наприклад:

Лабораторна робота № 1

Тема: Середовище редактора Visual Basic

Виконав студент гр. ХА-11

Козаченко В.І.

До протоколу лабораторної роботи слід включити:

- мету роботи;
- короткі теоретичні відомості;
- опис інтерфейсу редактора Visual Basic (з необхідними скріншотами);
- роздрукований текст програми;
- вихідні дані та отримані результати;
- висновки.

Після захисту лабораторної роботи протокол, підписаний викладачем, зберігається студентом в окремому файлі. По закінченні циклу лабораторних робіт протоколи всіх робіт збираються разом в окрему папку з титульним листом (додаток Б) і здаються викладачеві.

1.6 Контрольні запитання

1. Що таке «вікно проекту»? Як його відкрити?
2. Навіщо потрібно вікно «Свойства»?
3. Як використовується вікно «*Immediate*»?
4. Які є режими налаштування редактора VBA?
5. Для чого потрібно редагування макросів? У чому воно полягає?

⁶ З дозволу викладача допускається оформлення протоколів лабораторних робіт в зошиті. В такому випадку всі роздруковані тексти програм, розрахунків, блок-схем повинні бути вклеєні у відповідні місця протоколів.

2. Безпосереднє програмування

Мета та основні завдання роботи: дослідити особливості створення елементарних програмних засобів нерозгалуженої структури. Вивчити особливості оперування змінними та константами і програмування арифметичних виразів у мові програмування Visual Basic.

2.1 Основні теоретичні відомості

Основи мови програмування Visual Basic

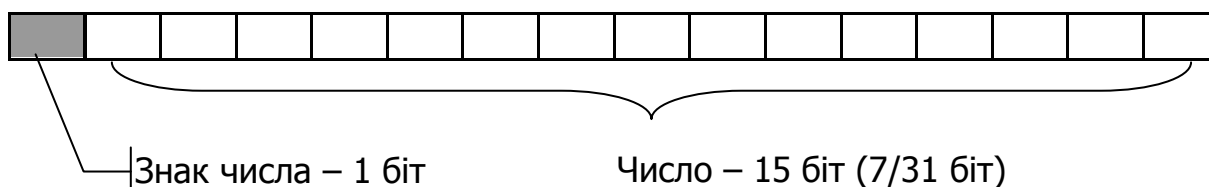
Використовувані символи.

Символи, які використовуються у мові Visual Basic, служать для формування ключових слів, операторів, імен змінних і міток, з яких будуються програми. Мова Visual Basic включає наступні символи: **латинські літери, цифри та спеціальні символи**. Літерні символи представлені великими та малими буквами латинського алфавіту (**A – Z**), (**a – z**). Цифрові символи представлені діапазоном цифр (**0 – 9**).

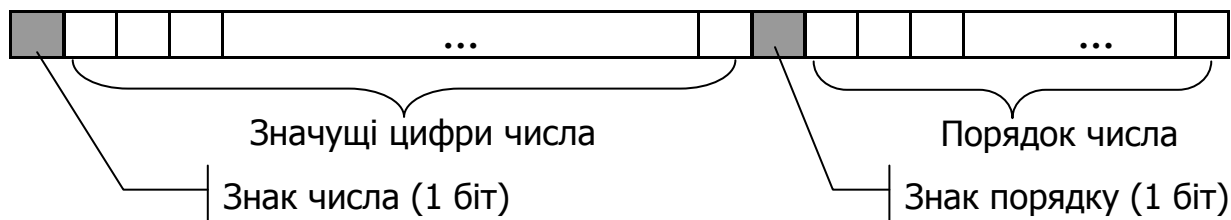
Типи даних мови Visual Basic

Тип даного визначає його внутрішнє представлення в пам'яті комп'ютера та кількість байт, яке потрібно для його збереження. За умовчанням, визначається тільки один тип даних – універсальний (**Variant**). При виборі типу даних (див. табл. 2.1), слід враховувати, що обробка даних, які мають меншу довжину, здійснюється значно швидше, ніж даних великої довжини.

Збереження в пам'яті даних типу Integer (Byte/Long)



Збереження в пам'яті даних типу Single



Таблиця 2.1 Типи даних VBA

Тип даних	Розмір (байт)	Діапазон значень
Boolean	2	ІСТИНА (ИСТИНА/TRUE) або ХИБНІСТЬ (ЛОЖЬ/FALSE)
Byte	1	Цілі числа від 0 до 255
Integer	2	Цілі числа від – 32 768 до 32 767
Long	4	Цілі числа від – 2 147 483 648 до 2 147 483 647
Single	4	Від –3,402823E38 до –1,401298E-45 (від’ємні) та від 1,401298E-45 до 3,402823E38 (додатні)
Double	8	Від –1,79769313486232E308 до –4,94065645841247E-324 (від’ємні) та від 4,94065645841247E-324 до 1,79769313486232E308 (додатні)
Currency	8	Від – 922 337 203 685 477,5808 до 922 337 203 685 477,5807
Date	8	Значення дат від 1 січня 100 року до 31 грудня 9999 року та значення часу від 0:00:00 до 23:59:59 .
Decimal	14	±79 228 162 514 264 337 593 543 950 335 (без десяткових знаків) або ±7,9228162514264337593543950335 (з 28 десятковими розрядами)
Object	4	<i>Посилання на будь-який об’єкт</i>
String (змінної довжини)	10 + 1*n	<i>Від 0 до приблизно 2 млрд. символів</i>
String (фіксованої довжини)	Довжина рядка	<i>Від 0 до приблизно 65 400 символів</i>
Variant (універсальний числовий)	16	Будь-яке число з діапазону Double
Variant (універсальний текстовий)	22 + 1*n	Той же діапазон, що і для строкового типу із змінною довжиною

Константи і змінні.

Константа – це попередньо визначена величина, значення якої не може бути змінено в процесі виконання програми. Константи можуть представляти будь-який з п'яти перерахованих вище типів даних. Числові константи можуть бути позитивними або негативними, цілими і з плаваючою комою, одинарної довжини і подвійної довжини. Найчастіше використовуються константи з плаваючою комою одинарної точності. **приклад:**

5 -28 0 3.14 -8.57 .9154 -36. -2.6014E12 5.039E-8 5.078E+10 22!

Символьні (рядкові, текстові) константи – дані типу **STRING** представляють собою послідовність алфавітно-цифрових символів (не більше 32 767), укладених в подвійні лапки. У число цих символів не можуть входити інші подвійні лапки (") і символ повернення каретки. **Приклади:**

"Лабораторну роботу виконав студ. Иванченко С." "Введіть "

"Рішення:" "х=" "Невірне введення! Повторіть."

Змінна – це величина, до якої звертаються в програмі за допомогою імені (ідентифікатора) і яка може набувати різних значень в процесі виконання програми.

Приклади констант:

Цілі числа (**Integer**) 5 -28 0

Дійсні числа (**Single**) 3.14 -8.572 .9154 -36. -2.601E1 5.039E-8

Символьні (строкові, текстові) константи (**String**) "Введіть а "

"х=" "Невірне значення! Повторіть!" "Розв'язок:"

"Лабораторну роботу виконав студ. Иванченко С."

Приклади іменованих констант:

Формат: **Const** <Ім'я_константи1> = вираз1 [, <Ім'я константи2> = вираз2]...

Const Pi AS Single = 3.14, Step AS Integer= 5, sFileName1 AS String= "prog.dat"

Приклади опису змінних:

Dim iK As Integer, iNumb As Integer, A As Single, Alfa As Single

Dim Beta As Single, sWrd As String, sName1 As String, Sum As Double

Примітка. Іменами констант та змінних не можуть бути службові слова мови Visual Basic. В список службових (або ключових) слів входять команди, оператори та імена стандартних функцій.

Арифметичні вирази.

Арифметичний вираз – це комбінація операндів, пов'язаних між собою знаками арифметичних операцій. В якості операндів можуть використовуватися: константи, змінні (прості і індексовані), звернення до функцій (стандартним (див. табл.2.2) і певним користувачем), а також висловлювання, укладені в круглі дужки.

Знаки операцій:

додавання (+), віднімання (-), множення (*), ділення (/), цілочисельне ділення⁷ (\), зведення в ступінь (^)

Пріоритет операцій – відповідає правилам математики:

0 (...) - обчислення в дужках;

1 f (...) - обчислення стандартних функцій;

2 ^ - піднесення до степеня;

3 * / - множення, ділення;

**4 ** - цілочисельне ділення;

5 MOD - обчислення залишку від цілочисельного ділення;

6 + - - додавання, віднімання.

Приклади запису арифметичних виразів⁸:

$$\frac{a \times b}{c \times d} \rightarrow a * b / (c * d) \quad \boxed{a * b / c * d} \rightarrow \frac{a \times b}{c} \times d$$

$$\frac{a \times \sin^2 x + \cos z^3}{2 - \ln z} \rightarrow (a * \text{SIN}(x) ^ 2 + \text{COS}(z ^ 3)) / (2 - \text{LOG}(z))$$

$$\sqrt{\frac{\alpha \sin^3 x \times \cos^3 x}{2e^{-a \times x}}} \rightarrow \text{SQR}(\text{alfa} * \text{SIN}(x) ^ 3 * \text{COS}(x) ^ 3 / (2 * \text{EXP}(-a * x)))$$

$$\sqrt[3]{\ln |\sin(-x)|} \rightarrow \text{LOG}(\text{ABS}(\text{SIN}(-x))) ^ (1 / 3)$$

⁷ Обидва числа округлюються до найближчого цілого.

⁸ У кожному з прикладів слід розібратися, чому проставлені або не проставлені дужки.

Таблиця 2.2 Стандартні математичні функції

Abs(x)	— абсолютне значення;
Exp(x)	— обчислення значення експоненти;
Log(x)	— обчислення <u>натурального логарифму</u> ⁹ ;
Sqr(x)	— обчислення квадратного кореня;
Fix(x)	— виділення цілої частини числа (2.73 -> 2);
Sin(x)	$\left. \begin{array}{l} \text{Sin}(x) \\ \text{Cos}(x) \\ \text{Tan}(x) \\ \text{Atn}(x) \end{array} \right\} \begin{array}{l} \text{тригонометричні функції синуса, косинуса, тангенса і} \\ \text{арктангенса (кут слід задавати в радіанах)}^{10}. \text{ Для} \\ \text{перетворення значення кута з градусної заходи в радіан,} \\ \text{можна скористатися формулою:} \\ \textbf{град = рад * 180 / \pi}; \end{array}$
Cos(x)	
Tan(x)	
Atn(x)	
Cint(x)	— округлення числа x (Наприклад:: 2.73 -> 3);
Int(x)	— округлення найбільшого цілого, не перевершує x (Наприклад: 2.73 -> 2 -2.73 -> -3);
Sgn(x)	— функція, визначення знака числа (виразу) якщо x > 0, то функція = 1, якщо x = 0, то функція = 0, якщо x < 0, то функція = -1;
Rnd[(X)]	— генерація випадкового числа (0 < x < 1).

Деякі оператори мови Visual Basic.

Оператор присвоювання.

Загальний вид оператора присвоювання:

<Ім'я_змінної> = <вираз>

В залежності від типу виразу, оператори присвоювання бувають арифметичні або символічні. приклад:

```

c = a + b
s = pi * r ^ 2
z = Log(Abs(Sin(-x))) ^ (1 / 3)
x = x + h
k = k + 1
a = 12

```

Коментар.

Запис коментарів у тексті програми починається з одинарної лапки <'> (апострофа).

⁹Для обчислення логарифмів за іншою основою (у тому числі десятковою) слід використовувати відповідну математичну формулу.

¹⁰ Обчислення всіх інших тригонометричних функцій слід виконувати, використовуючи відповідні математичні формули.

Наприклад:

'Лабораторна робота N1 Студ. Вовк Є., гр. ХА-51

'Розрахунок температурного поля

Коментарем можна закінчувати програмний рядок. Однак слід пам'ятати, що після коментаря не повинні розташовуватися інші оператори.

Наприклад:

$k = k + 1$ 'поточне значення змінної k збільшується на 1

Функції введення-виведення

Функція **MsgBox** призначена для виводу на екран діалогових вікон, що містять повідомлення. При цьому вона встановлює режим очікування натиснення кнопки користувачем, а потім повертає значення типу **Integer**, яке залежить від того, яка кнопка була натиснута.

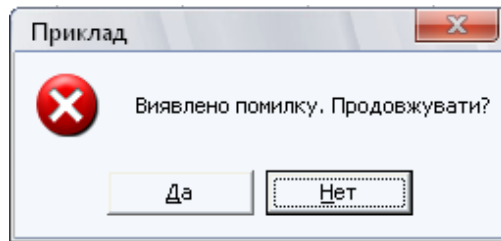
Синтаксис: `MsgBox(prompt[, buttons] [, title] [, helpfile, context])`

Синтаксис функції **MsgBox** містить наступні іменовані аргументи:

Аргумент	Описание	Статус
prompt	Строковий вираз, який відображується як повідомлення в діалоговому вікні. Максимальна довжина рядка складає приблизно 1024 символів. Строкове значення prompt може містити декілька фізичних рядків.	Обов'язковий.
buttons	Числовий вираз, який представляє суму значень, що вказують число і тип кнопок у діалоговому вікні, тип використовуваного значка, основну кнопку і модальність вікна повідомлення. <i>Значення цього аргументу за умовчанням дорівнює 0.</i>	Необов'язковий.
title	Строковий вираз, що відображується в рядку заголовку діалогового вікна. <i>Якщо цей аргумент відсутній, в рядок заголовку записується ім'я додатку.</i>	Необов'язковий.
helpfile	Строковий вираз, який визначає ім'я файлу довідки, що містить довідкові відомості про це діалогове вікно. <u>Якщо цей аргумент вказаний, необхідно вказати також аргумент context.</u>	Необов'язковий.
context	Числовий вираз, що визначає номер відповідного розділу довідкової системи. <u>Якщо цей аргумент вказаний, необхідно вказати також аргумент helpfile.</u>	Необов'язковий.

Нижче наведено приклад вікна, створеного функцією **MsgBox**:

Ans = MsgBox("Виявлено помилку. Продовжувати?", 276, "Приклад", "DEMO.HLP", 1000)



Значення, які може приймати аргумент **buttons**:

Константа ¹¹	Значення	Опис
VbOKOnly	0	Відображується тільки кнопка "ОК".
VbOKCancel	1	Відображується кнопки "ОК" та "Отмена" (Cancel).
VbAbortRetryIgnore	2	Відображується кнопки "Прервать" (Abort), "Повторить" (Retry) та "Пропустить" (Ignore).
VbYesNoCancel	3	Відображується кнопки "Да" (Yes), "Нет" (No) та "Отмена" (Cancel).
VbYesNo	4	Відображується кнопки "Да" (Yes) та "Нет" (No).
VbRetryCancel	5	Відображується кнопки "Повторить" (Retry) та "Отмена" (Cancel).
VbCritical	16	Використовується значок "Критическое сообщение".
VbQuestion	32	Використовується значок "Предупреждающий запрос".
VbExclamation	48	Використовується значок "Предупреждение".
VbInformation	64	Використовується значок "Информационное сообщение".
VbDefaultButton1	0	Основною ¹² є перша кнопка.
VbDefaultButton2	256	Основною є друга кнопка.
VbDefaultButton3	512	Основною є третя кнопка.
VbDefaultButton4	768	Основною є четверта кнопка.
VbApplicationModal	0	Модальне вікно на рівні застосування: щоб продовжити роботу з поточним застосуванням, необхідно відповісти на це повідомлення.
VbSystemModal	4096	Модальне вікно на рівні системи: усі застосування будуть недоступні до тих пір, поки користувач не дасть відповідь на це повідомлення.

Перша група значень (0-5) вказує число і тип кнопок, що відображуються у вікні діалогу, друга група (16, 32, 48, 64) задає тип використовуваного значка, третя група (0, 256, 512) визначає кнопку, яка є основною, а четверта група (0, 4096) – модальність вікна повідомлення. При визначенні значення аргументу **buttons** слід

¹¹ Ці константи визначені в мові VBA. Використання імен цих констант замість їх значень можливе в будь-якому місці програми.

¹² Основною є кнопка, яка спрацьовує за умовчанням при натисканні клавіши Enter.

підсумовувати не більш за одне значення з кожної групи. Отже, аргумент **buttons** дозволяє управляти наступними параметрами вікна, яке створюється функцією **MsgBox**:

- ✓ Набір кнопок (типи та кількість) і їх розміщення у вікні;
- ✓ Піктограма, що відображується у вікні;
- ✓ Яка кнопка призначається за умовчанням;
- ✓ Режим (модальність) вікна повідомлення.

Значення, які повертає функція MsgBox

Константа	Значення	Натиснута кнопка
vbOK	1	ОК
vbCancel	2	Отмена (Cancel)
vbAbort	3	Прервать (Abort)
vbRetry	4	Повторить (Retry)
vbIgnore	5	Пропустить (Ignore)
vbYes	6	Да (Yes)
vbNo	7	Нет (No)

Приклад. Наступна програма ілюструє роботу функції **MsgBox**. На рисунку 2.1 наведені вікна, що створюється при роботі програми.

Public Sub Приклад_вікна_1()

```
Dim Msg As String, Style As Single, Title As String
Dim Help As String, Ctxt As Single, Response As Integer
Msg = "Виявлено помилку. Продовжувати?" ' Повідомлення.
Style = vbYesNo + vbCritical + vbDefaultButton2 ' Кнопки.
Title = "Приклад" ' Заголовок.
Help = "DEMO.HLP" ' Файл довідки
Ctxt = 1000 ' Контекст
```

```
' Функція MsgBox виводить вікно з повідомленням
```

```
Response = MsgBox(Msg, Style, Title, Help, Ctxt)
```

```
If Response = vbYes Then ' Натиснута кнопка "Да" (Yes)
```

```
    ...
    ... Операції, які треба виконати, коли натиснуто кнопку "Да"
    ...
```

```
    MsgBox "Натиснута кнопка ""Да"" ", vbInformation, "Відповідь"
```

```
Else ' Натиснута кнопка "Нет" (No)
```

```
    ...
    ... Операції, які треба виконати, коли натиснуто кнопку "Нет"
    ...
```

```
    MsgBox "Натиснута кнопка ""Нет"" ", vbInformation, "Відповідь"
```

```
End If
```

```
MsgBox "Значення, повернуте функцією MsgBox: " & Response, ,  
        "Результат"
```

```
End Sub
```

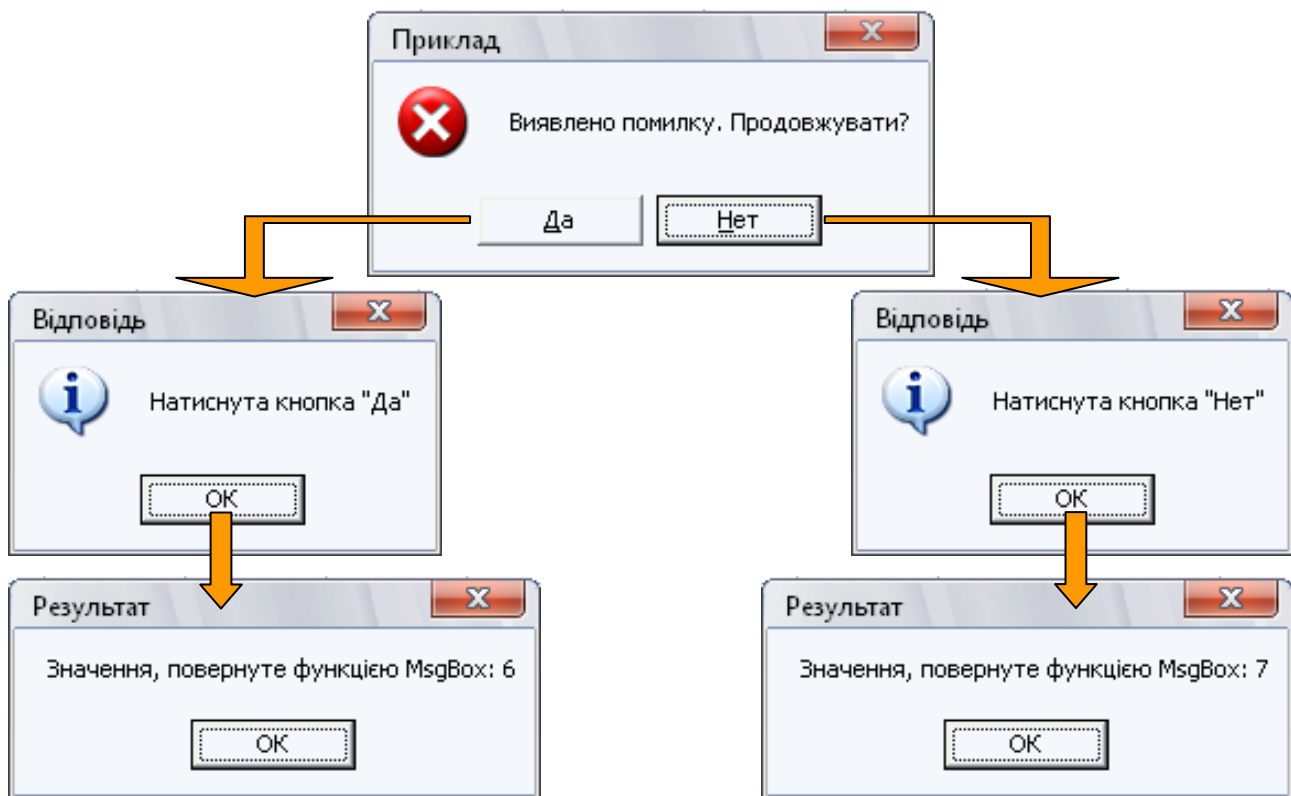


Рис. 2.1 – Результат роботи програми **Приклад_вікна_1**

Функція **InputBox** виводить на екран діалогове вікно, що містить повідомлення і поле для введення значення, встановлює режим очікування введення тексту користувачем або натиснення кнопки, а потім повертає значення типу **String**, яке містить текст, введений в полі. Функція **InputBox** має такий синтаксис:

InputBox(prompt[, title] [, default] [, xpos] [, ypos] [, helpfile, context])

Синтаксис функції **InputBox** містить наступні іменовані аргументи:

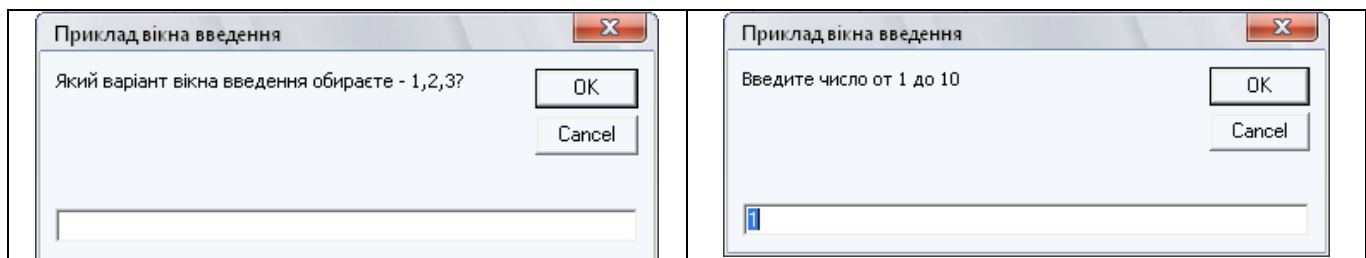
Аргумент	Опис	Статус
prompt	Строковий вираз, який відображується як повідомлення в діалоговому вікні. Максимальна довжина рядка складає приблизно 1024 символів. Строкове значення prompt може містити декілька фізичних рядків.	Обов'язковий.
title	Строковий вираз, що відображується в рядку заголовку діалогового вікна. Якщо цей аргумент відсутній, в рядок заголовку записується ім'я додатку.	Необов'язковий.
default	Строковий вираз, який відображується в полі введення як значення використовуване за умовчанням, якщо користувач не введе іншу строку. Якщо цей аргумент відсутній, поле введення відображується порожнім.	Необов'язковий.

Аргумент	Опис	Статус
xpos	<u>Числовий вираз</u> , що задає відстань по горизонталі між лівою межею діалогового вікна і лівим краєм екрану (у твіпах). <i>Якщо цей аргумент пропущений, діалогове вікно вирівнюється по центру екрану по горизонталі.</i>	Необов'язковий.
ypos	<u>Числовий вираз</u> , що задає відстань по вертикалі між верхньою межею діалогового вікна і верхнім краєм екрану (в твіпах). <i>Якщо цей аргумент пропущений, діалогове вікно розташовується по вертикалі приблизно на одну третину висоти екрану.</i>	Необов'язковий.
helpfile	<u>Строковий вираз</u> , який визначає ім'я файлу довідки, що містить довідкові відомості про це діалогове вікно. <u>Якщо цей аргумент вказаний, необхідно вказати також аргумент context.</u>	Необов'язковий.
context	<u>Числовий вираз</u> , що визначає номер відповідного розділу довідкової системи. <u>Якщо цей аргумент вказаний, необхідно вказати також аргумент helpfile.</u>	Необов'язковий.

Якщо користувач натискає кнопку "ОК" або клавішу **ENTER**, функція **InputBox** повертає вміст поля введення. Якщо користувач натискає кнопку "Відміна", функція поверне порожній рядок (« »).

Приклади функції **InputBox**:

- 1) var = InputBox("Який варіант вікна введення обираєте - 1,2,3?", Title)**
- 2) MyValue = InputBox("Введіть число від 1 до 10", "Приклад вікна введення", "1")**
- 3) Наступний приклад ілюструє помилку, що виникає при використанні функції **InputBox**, коли у вікні вводу замість числа введено текст. Це призводить до помилки, яка припиняє виконання програми.



Public Sub Приклад_вікна_3()

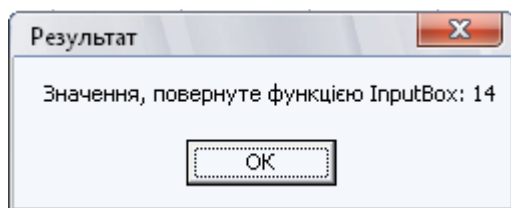
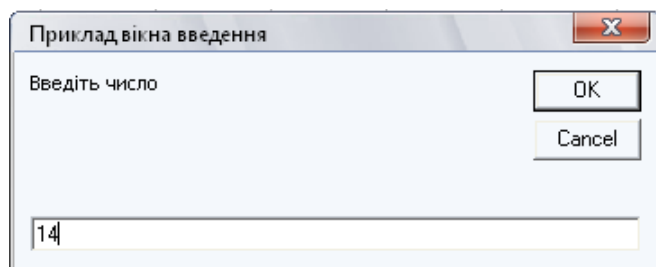
Dim iMyValue As Integer

iMyValue = InputBox("Введіть число", "Приклад вікна введення", 1)

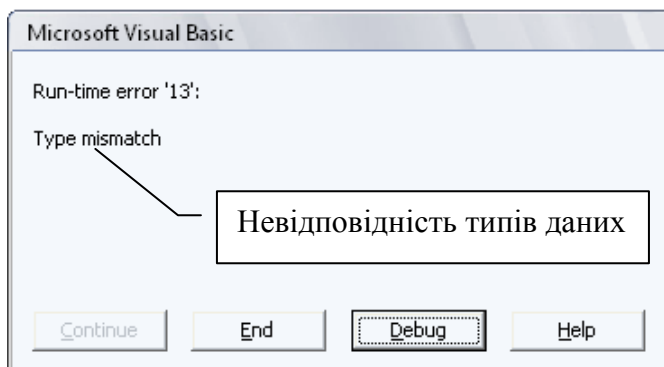
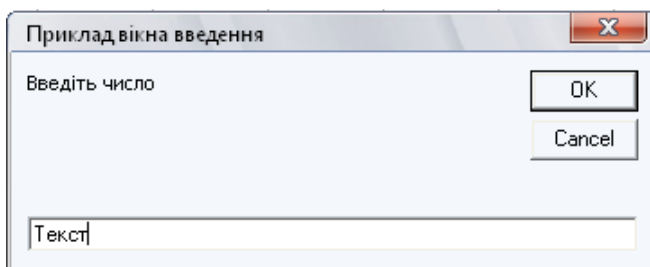
MsgBox "Значення, повернуте функцією InputBox: " & iMyValue, , "Результат"

End Sub

а) у вікно введено число
(програма працює нормально)



б) у вікно введено текст (виникає помилка,
робота програми припиняється)



Метод ***InputBox*** є альтернативою функції InputBox. Він виводить на екран діалогове вікно, що містить повідомлення і поле для введення значення, встановлює режим очікування введення тексту користувачем або натиснення кнопки, а потім повертає значення заданого типу, що містить інформацію, введену в поле. Синтаксис методу ***InputBox*** такий:

Application.InputBox(prompt[, title] [, default] [, xpos] [, ypos] [, helpfile, context] [, type])

Аргументи методу ***InputBox*** аналогічні аргументам функції ***InputBox*** за винятком останнього аргументу **type**. Цей аргумент явно задає тип значення, що повертається:

Значення	Значення що повертається
0	Формула
1	Число
2	Текст (рядок)
4	Логічні значення (ІСТИНА або ХИБНІСТЬ)

Значення	Значення що повертається
8	Посилання на комірку
16	Значення помилки
64	Масив значень

При незаданому значенні аргументу за умовчанням повертається текст.

Приклад, що наведено нижче, ілюструє як метод **InputBox** відпрацьовує помилку, аналогічну тій, що виникає при використанні функції InputBox, коли у вікні вводу замість числа введено текст. В даному випадку з'являється повідомлення про помилку, але виконання програми не припиняється.

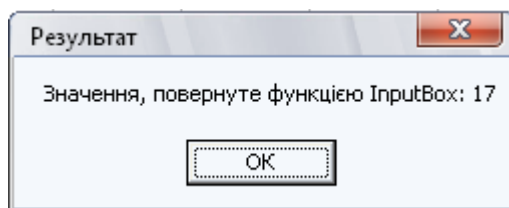
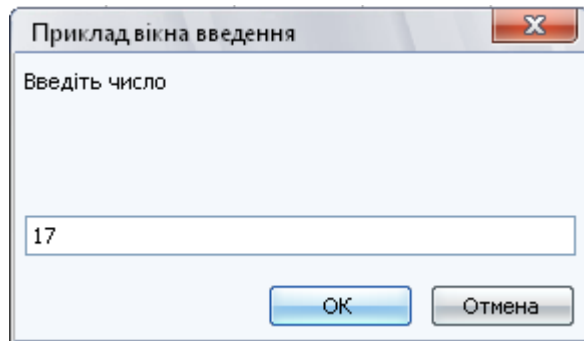
Public Sub Приклад_вікна_4()

```
Dim iMyValue As Integer
```

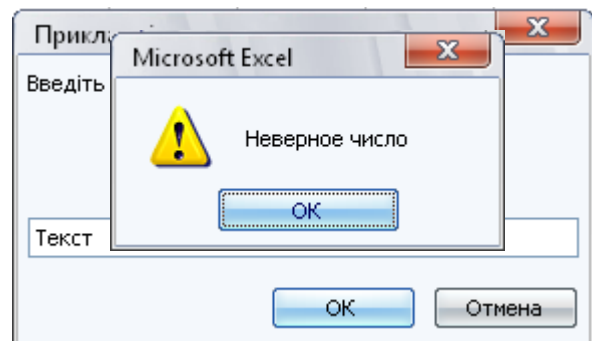
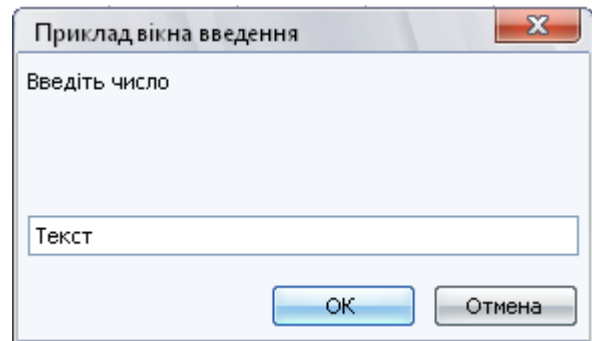
```
iMyValue = Application.InputBox("Введіть число", "Приклад вікна введення", 1,,  
,,, 1)
```

```
MsgBox "Значення, повернуте функцією InputBox: " & iMyValue, , "Результат"  
End Sub
```

а) у вікно введено число
(програма працює нормально)



б) у вікно введено текст (виникає помилка,
але робота програми не припиняється)



Приклади найпростіших програм:

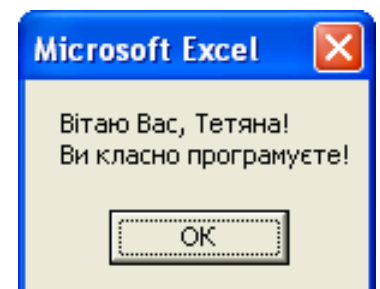
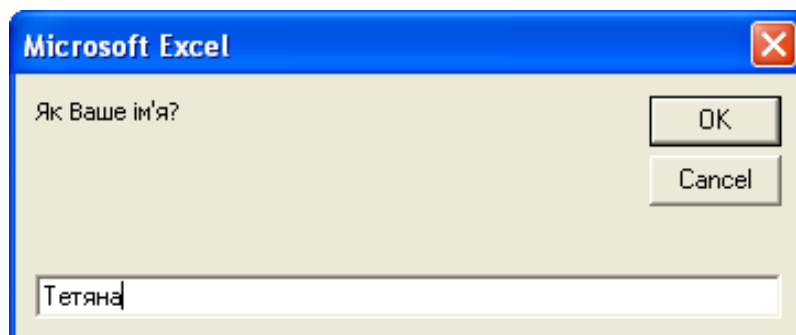
Public Sub Pryklad_1()

```
Dim sName As String
```

```
sName = InputBox("Як Ваше ім'я?")
```

```
MsgBox "Вітаю Вас, " & sName & "!" & vbNewLine & "Ви класно програмуєте!"
```

```
End Sub
```



```
Public Sub Pryklad_2()
```

```
    Dim a As Single, b As Single, x As Single, y As Single
```

```
    a = InputBox("Введіть значення змінної a")
```

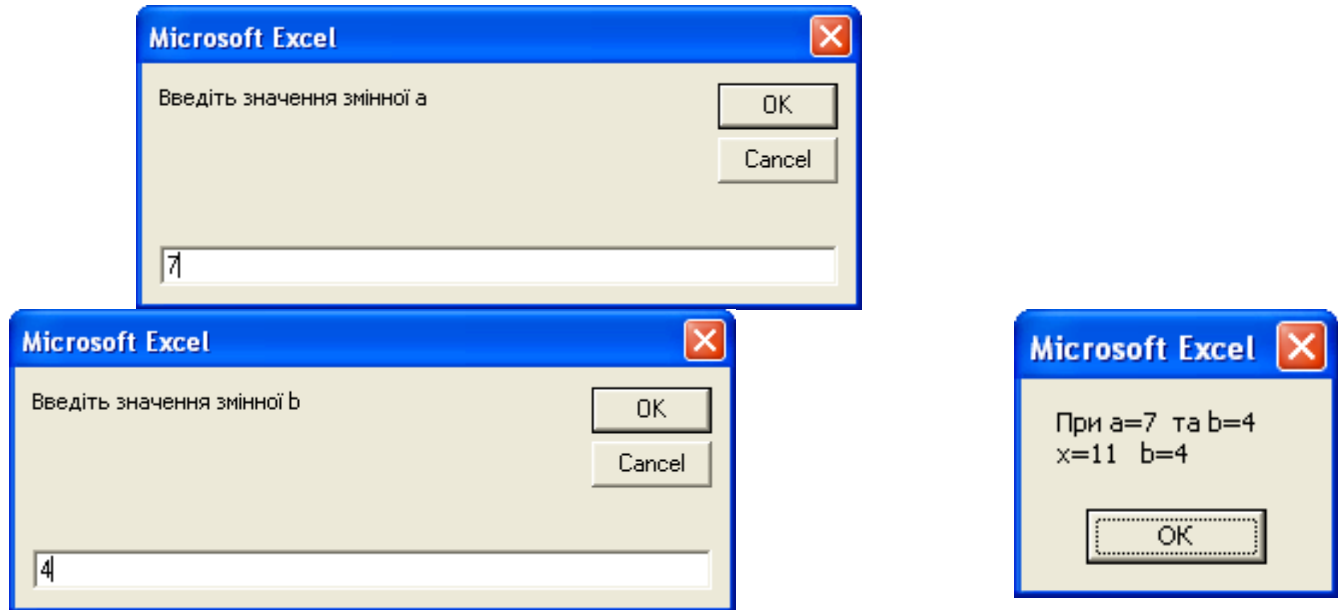
```
    b = InputBox("Введіть значення змінної b")
```

```
    x = a + b
```

```
    y = a - b
```

```
    MsgBox "При a=" & a & " та b=" & b & vbNewLine & "x=" & x & " b=" & b
```

```
End Sub
```



Зручним способом зберігання, як початкової інформації, так і результатів роботи програм є файли. При цьому до даних, записаних у файлі, можна багаторазово звертатися. Введення початкових даних з файлу дозволяє уникнути помилок, які досить часто бувають при діалоговому введенні даних з клавіатури. Результати розрахунку будь якої програми, записані у файл, можуть бути використані, наприклад, в якості початкових даних для іншої програми.

Щоб програма могла обмінюватися даними з файлом(прочитувати або записувати), в програмі має бути оператор, відкриваючий цей файл, тобто організуючий канал зв'язку між дисковим файлом і програмою. Для цього використовується оператор виду:

```
OPEN "Ім'я_файлу" FOR <режим роботи> AS#<номер_каналу>
```

де «Ім'я_файлу» – ім'я файлу з розширенням (можливе повне ім'я файлу – диск:\шлях\ім'я_файлу);

<режим роботи> – встановлює дозволений режим доступу до даних, що зберігаються у файлі (для файлу послідовного доступу можливі режими: INPUT, OUTPUT, APPEND);

INPUT – файл відкривається тільки для читання (для введення даних);

OUTPUT – файл відкривається тільки для запису (для виведення даних);

APPEND – файл відкривається для запису (нові пропозиції записуються в кінець вже існуючого файлу – дозапис в наявний файл).

<номер_каналу> – ціле число між 1 і 255. Звернення до файлу з програми відбувається під цим номером.

Після того, як обмін даними з файлом закінчений, канал зв'язку необхідно закрити за допомогою оператора **CLOSE**:

CLOSE #<номер_каналу>

Якщо в операторі **CLOSE** не вказаний номер файлу, то будуть закриті усі відкриті файли. Доступ до дискового файлу (між операторами **OPEN** і **CLOSE**) організується в тексті програми за допомогою наступних операторів:

INPUT #<номер_каналу> [, список вводу]

(використовується для читання інформації з файлу даних);

PRINT #<номер_каналу> [, список виведення]

(використовується для виведення інформації в дисковий файл).

2.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи вивчити та описати в протоколі лабораторної роботи такі поняття (визначення) і операції (команди):

- правила запису констант, змінних, основні типи даних VBA та правила їх опису; стандартних функцій та арифметичних виразів;
- правила запису коментарів;
- оператор присвоювання та оператори введення-виведення.

2.3 Опис лабораторних засобів та обладнання

Лабораторна робота №2 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

2.4 Заходи безпеки під час виконання практичної і лабораторної робіт

Заходи безпеки, яких треба дотримуватись при виконанні даної роботи, наведені у додатку А.

2.5 Вказівки до виконання лабораторної роботи №2

1. На листі MS Excel виконати розрахунок свого варіанту індивідуального завдання, для чого записати формули для обчислення величин, які мають бути

пораховані, та підібрати такі значення вихідних змінних, при яких розрахунок виконується¹³.

2. Створити в редакторі Visual Basic процедуру для обчислення двох заданих величин та їх суми відповідно до варіанту індивідуального завдання. При виведенні отриманих результатів передбачити також і виведення вихідних даних¹⁴.
3. Налаштувати програму. При виконанні розрахунків використати підібрані значення вихідних змінних (п.1). Порівняти результати отримані у програмі з результатами, отриманими в MS Excel.
4. Продемонструвати результати викладачу.
5. Програму зберегти на дискеті (або іншому носії).
6. Оформити протокол лабораторної роботи.

2.6 Обробка та аналіз результатів. Оформлення звіту з лабораторної роботи

До протоколу лабораторної роботи крім мети роботи та коротких теоретичних відомостей¹⁵ звітуючи про виконану роботу слід включати:

- номер індивідуального варіанту;
- повну умову поставленої індивідуальної задачі;
- алгоритм (блок-схему)¹⁶ розв'язку задачі (починаючи з лабораторної роботи № 3);
- опис ідентифікаторів – всіх змінних, що використовуються в програмі (якщо їх назви відрізняються від назв параметрів в постановці задачі);
- роздрукований текст програми¹⁷;
- роздруковані результати виконання розрахунків та вихідних даних, при яких вони отримані¹⁸;
- висновки.

2.7 Контрольні запитання

1. Які символи використовуються у мові Visual Basic?
2. Що таке константи і змінні?

¹³ Тобто значення змінних повинні бути такі, щоб розрахунки не приводили до ділення на нуль, обчислення квадратного кореню або логарифму з від'ємного числа, тощо.

¹⁴ Використати функцію **MsgBox**.

¹⁵ Ця частина протоколу готується при підготовці до лабораторної роботи і перевіряється викладачем перед виконанням роботи згідно РСО.

¹⁶ Правила запису блок-схем наведені в Додатку В.

¹⁷ Текст кожної програми повинен починатися з рядків коментарів, де вказано прізвище студента та назва групи, тема роботи та № індивідуального варіанту.

¹⁸ Вихідні дані та результати розрахунків при роздрукуванні повинні супроводжуватись текстовими поясненнями (наприклад, <Температура = 315 K> або <y = -14.53>).

3. Які типи даних існують в мові Visual Basic?
4. Які існують правила запису імен змінних?
5. Як розрізняються змінні різних типів?
6. Який порядок виконання операцій в арифметичному виразі?
7. Правила запису стандартних функцій в Visual Basic?
8. Як обчислити значення виразу?
9. Що таке коментар?
10. Як описуються змінні в програмі?
11. Як працює програма, якщо вона містить функцію InputBox?
12. Які роздільники використовуються в операторові MsgBox?
13. Як здійснити читання з файлу і запис у файл?
14. Яким чином можна дописати інформацію у вже існуючий файл?
15. Яким чином здійснюється зв'язок програми з файлами?

3. Розробка та програмування алгоритмів розгалуженої структури

Мета та основні завдання роботи: дослідити особливості створення елементарних програмних засобів розгалуженої структури у мові програмування Visual Basic.

3.1 Основні теоретичні відомості

Більшість задач, що зустрічаються в практичній діяльності людини, не вкладаються в лінійну схему, а мають варіанти розрахунків, які залежать від виконання (або не виконання) тих чи інших умов. Наприклад, при розв'язанні квадратного рівняння спосіб обчислення коренів залежить від знаку дискримінанта. Або в наступному прикладі:

$$y = \begin{cases} x^2 - 4 * x^{0.5}, \text{при } x > 0 \\ x^3 + 1.8 * x, \text{при } x \leq 0 \end{cases}$$

в залежності від знаку змінної x , змінна y обчислюється за різними формулами.

Блочний оператор If. При необхідності виконання великої групи операторів в залежності від якої-небудь умови найбільш зручно використовувати блочний умовний оператор **If**. За допомогою блочного оператора **If** може бути перевірено не одну умову, а будь яку кількість умов, які є альтернативами та перевіряються послідовно до «першого виконання». Для цього в операторі **If** використовуються оператори **ElseIf** (для перевірки другого і подальших умов). Виконання будь якої з альтернативних гілок приводить к виходу з блоку. Тому в загальному випадку оператор **If** має такий формат:

```
If < умова 1 > Then
    1-а група операторів
[ ElseIf < умова 2 > Then
    2-а група операторів ]
...
[ Else
    n-а група операторів ]
End If
```

Слід пам'ятати, що кожен блочний оператор **If** має бути «закритий» оператором **End If**. Частини **Else** і **ElseIf** є необов'язковими¹⁹. У скороченому варіанті оператор має такий формат:

```
If < умова > Then
    перша група операторів
Else
    друга група операторів
End If
```

¹⁹ Оператори або їх фрагменти, «взяті в квадратні дужки», не є обов'язковими.

або такий:

```
If <умова> Then  
    група операторів  
End If
```

де *перша група операторів* (*група операторів*), та *друга група операторів* – оператори, які виконуються у разі дотримання або не дотримання умови відповідно. В «*умові*» записуються вирази (або змінні), пов'язані знаками відношень: =, <, >, <=, >=, <>. Вони називаються логічними операціями порівняння.

Умова, в операторові **If**, також може бути складною. Складні умови будуються з простих відношень за допомогою знаків логічних операцій:

- **And** – логічне множення (*кон'юнкція*) "**I**". В цій операції результат є істиною, коли кожна з простих умов істинна;
- **Or** – логічне складання (*диз'юнкція*) "**АБО**". В цій операції результат є істиною, коли хоч одна з складових умов істинна;
- **Not** – логічне заперечення (*інверсія*) "**НІ**". В цій операції хибне значення перетворюється на істинне і навпаки.

Приклад 1: **If** x > 5 **And** x <= 50.5 **Then** ...

Приклад 2: **If** x <= 5 **Or** x > 50.5 **Then** ...

Порядок виконання операцій при перевірці умов (при обчисленні логічних виразів) наступний:

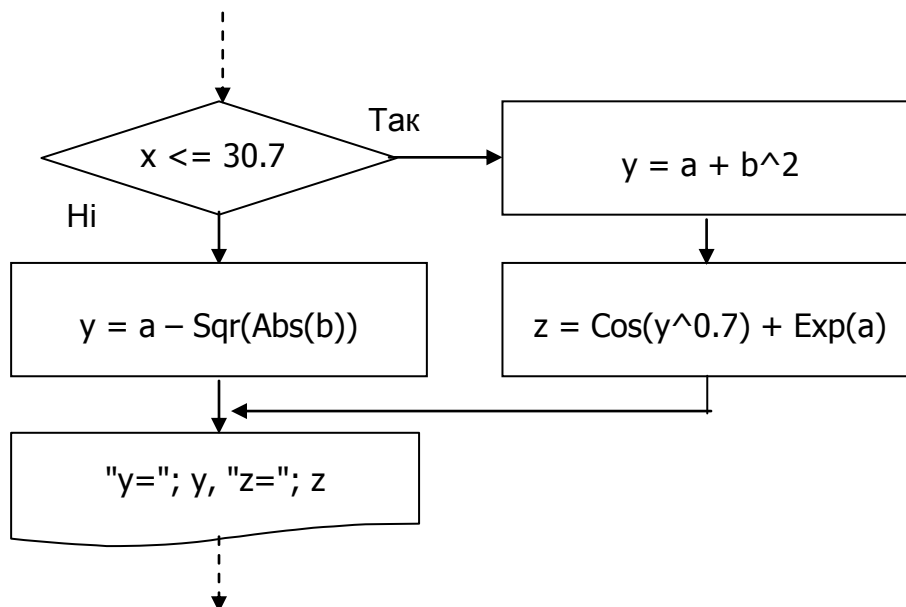
1. Обчислюються усі арифметичні вирази;
2. Виконуються усі операції відношень;
3. Виконуються усі операції **Not**;
4. Виконуються усі операції **And**;
5. Виконуються усі операції **Or**

Робота оператора: Якщо умова істинна, то виконується "*перша група операторів*", розташована в рядках після оператора **If**, а якщо хибна, – то виконується "*друга група операторів*", розташована в рядках після оператора **Else**. Після виконання однієї з гілок – «**Then**» або «**Else**» дія програми триває з оператора що йде за оператором **End If**, тобто завжди виконується тільки одна з гілок алгоритму розгалуження. Оператор **If** можна використовувати в скороченій формі (якщо частина «**Else**» не потрібна – приклад 4).

Приклад 3:

```
If x <= 30.7 Then  
    y = a + b^2  
    z = Cos(y^0.7) + Exp(a)  
    MsgBox "y=" & y & "z=" & z  
Else  
    y = a - Sqr(Abs(b))  
    MsgBox "y=" & y  
End If
```

Цей оператор працює за таким алгоритмом:



Приклад 4:

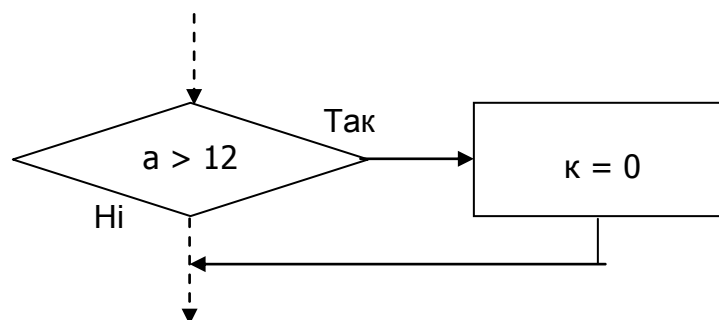
If a > 12 Then

к = 0

'Якщо a > 12, коефіцієнт к обнуляється

End If

працює за алгоритмом:



Приклад 5.

Визначити значення функції Y при різних значеннях аргументу x , за наступних умов:

$$Y = \begin{cases} \text{EXP}(x), & \text{при } x = 1 \\ \text{COS}(x^2)^2, & \text{при } 2 \leq x \leq 5 \\ \text{TG}(\text{SQR}(x)), & \text{при } 1 < x < 2 \\ a + b^2, & \text{при } x < 1, x > 5 \end{cases}$$

Варіант алгоритму для розв'язання цього прикладу наведений на рис. 3.1

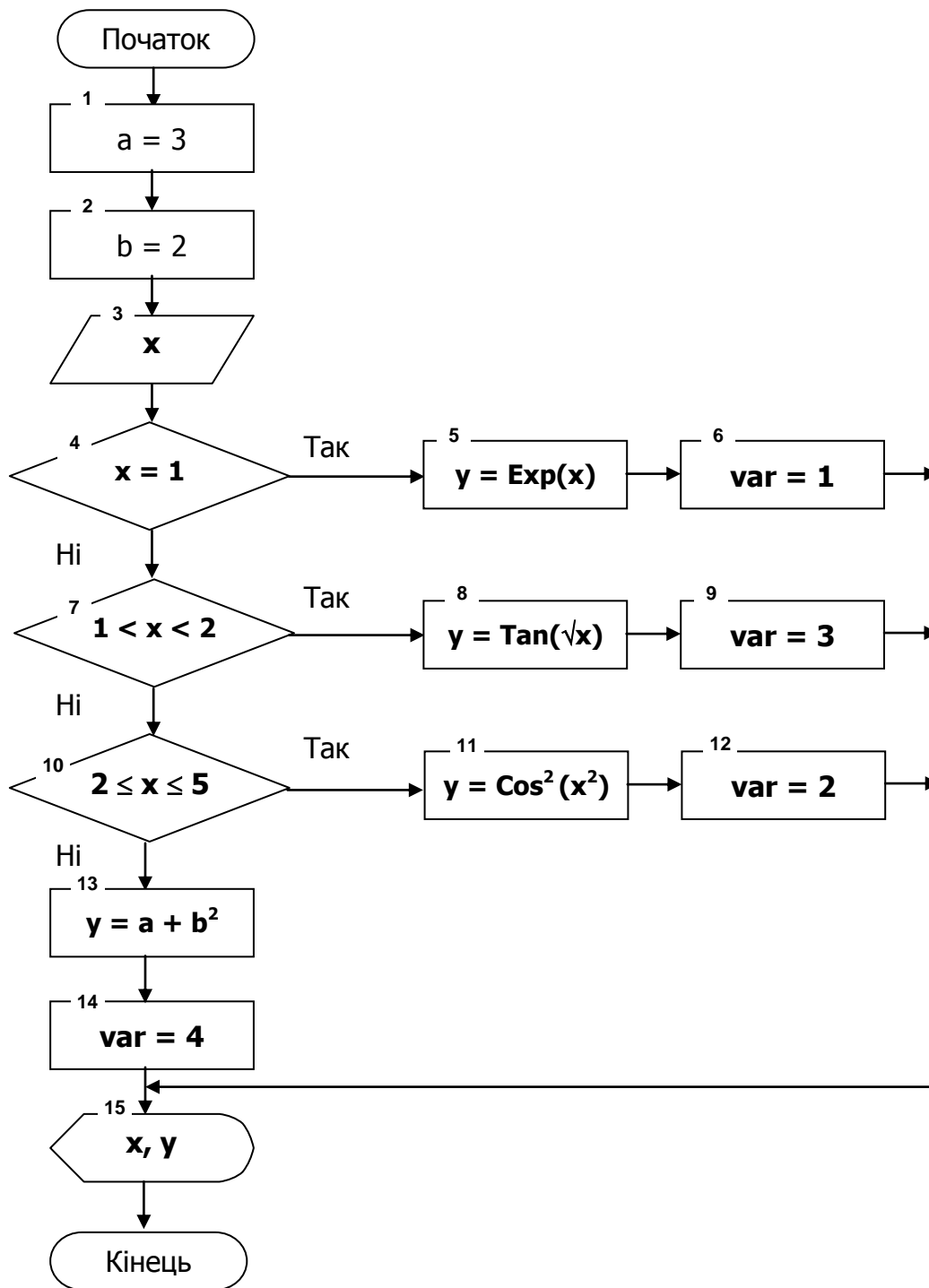


Рис. 3.1. Алгоритм розв'язання прикладу 5

Приклад розв'язання:

Option Explicit

Public Sub Розгалуження1()

```
Dim x As Single, y As Single, a As Single, b As Single, var As Integer
a = 3: b = 2
x = InputBox("Введіть x")
If x = 1 Then
    y = Exp(x)
    var = 1
ElseIf x > 1 And x < 2 Then
    y = Tan(Sqr(x))
    var = 3
ElseIf x >= 2 And x <= 5 Then
    y = Cos(x ^ 2) ^ 2
    var = 2
Else
    y = a + b ^ 2
    var = 4
End If
MsgBox "x=" & x & " y=" & y & vbCrLf & "варіант" & var
End Sub...
```

Одна умова може бути вложена усередині іншої, тобто перевірятися тільки, якщо виконалася перша (зовнішня) умова. Наприклад:

Приклад 6.

```
If x > 0 Then
    If y > 0 Then                ` перевіряється тільки, якщо x > 0
        Debug.Print " x > 0 та y > 0 "
    Else
        Debug.Print " тільки x > 0 "
    End If
ElseIf x = 0 Then
    If y = 0 Then                ` перевіряється тільки, якщо y = 0
        Debug.Print " x = y = 0 "
    Else
        Debug.Print " тільки x = 0 "
    End If
Else
    Debug.Print " x < 0 "
End If
```


Для наочності програми (тобто візуального визначення, до якого **If** відносяться частини **Else** та **End If**, а також чітко бачити всі гілки розгалуження) необхідно використовувати елементи структурного програмування, виділяючи кожний блок (в даному випадку – **If-Else-End If**), який є в тексті програми. Для цього, використовуючи певну кількість проміжків, слід зміщувати залежні рядки оператора **If** (оператори кожної з гілок) праворуч. Див. приклад 6.

Рядковий оператор If є історично більш старим інструментом і має менше можливостей у порівнянні з блочним оператором **If**. Його доцільно використовувати в разі перевірки нескладних (не громіздких) окремих умов. Він записується в один рядок і має такий формат:

If <умова> **Then** <оператор 1> [**Else** <оператор 2>]

де *оператор 1*, *оператор 2* – оператори мови Visual Basic, які виконуються у випадку виконання або не виконання умови. "*Умова*" записується за тими самими правилами, що і для блочного оператора **If**.

Робота оператора: Якщо умова істинна, то виконується "оператор 1", записаний після слова **Then**, а якщо хибна, – то виконується "оператор 2", записаний після слова **Else**. Після виконання однієї з гілок (**Then** або **Else**) виконання програми продовжується з оператора наступного за оператором **If**. Оператор **If** можна використовувати у скороченій формі (частина "**Else**" відсутня – приклади 2 і 3). Приклади:

- 1) **If** a > 0 **Then** y = Sin(a-2) **Else** y = 0.5 * a - 1
- 2) **If** b > 0 **Then** y = Cos(b)
- 3) **If** z > 0 **Then** k%=1

В двох останніх випадках якщо умова хибна, то фактично нічого не відбувається – виконання програми продовжується з оператора програми, який слідує за оператором **If** оскільки відсутня частина **Else**.

Нижче наведений варіант програми з використанням рядкового оператора **If** для розв'язання прикладу №5.

```
' ПРОГРАМА РОЗРАХУНКУ y
a = InputBox("Введіть значення змінних a")
b = InputBox("Введіть значення змінних b")
x = InputBox("Введіть значення змінних x")
If x = 1 Then y = Exp(x)
If x > 1 And x < 2 Then y = Tan(Sqr(x))
If x >= 2 And x <= 5 Then y = Cos(x ^ 2) ^ 2
If x < 1 Or x > 5 Then y = a + b ^ 2
Debug.Print " При x="; x, "y="; y
```

3.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи вивчити та описати в протоколі лабораторної роботи такі поняття (визначення) і операції (команди):

- види логічних операцій та правила їх запису;
- правила запису та роботи блочного умовного оператора **If**;
- правила запису та роботи рядкового умовного оператора **If**;
- створити блок-схему для розв'язання прикладу 6;
- порівняти алгоритм рядкового оператора із алгоритмом, наведеним на рис. 3.1. Чим відрізняється алгоритм даної програми?
- розробити та програмно реалізувати алгоритми розв'язання двох задач зі списку наведеного нижче²⁰:

1. Визначити, в якому квадранті, чи на якій осі координат розташована довільна точка A з координатами x_1, y_1 .
2. Визначити, чи належать довільні точки $A(x_1, y_1)$, та $B(x_2, y_2)$ одному чи різним квадрантам.
3. Визначити, чи можливо помістити одну фігуру в іншу (круг із площею S_1 та квадрат з площею S_2) при суміщенні їх центрів симетрії.
4. Розрахувати площу квадрату та ромбу (гострий кут A) зі стороною a та визначити, яка з фігур має більшу площу.
5. Розрахувати площу кіл вписаних до квадрату та ромбу (гострий кут A) зі стороною a , та визначити, площа якого кола менша та на скільки.
6. На площині задані координати вершин трикутника $A(x_1, y_1)$, $B(x_2, y_2)$, $C(x_3, y_3)$. Визначити, чи є він рівнобічним, рівнобедреним, різнобічним. (Можливо використати ϕ -лу 3).
7. На площині задані координати вершин трикутника. Визначити, чи є він прямокутним. (Можливо використати формулу 3).
8. Визначити, чи знаходиться точка з координатами x_1, y_1 в площі трикутника, утвореного осями координат та прямою $y = 5 - x$. (Можливо використати формули 2, 3 та 4).
9. Визначити, які з точок $A(x_1, y_1)$, $B(x_2, y_2)$, та $C(x_3, y_3)$ знаходяться в області, що утворена прямими $y = 2x - 4$ та $y = x - 2$.
10. Визначити координати четвертої вершини x_4, y_4 прямокутника, якщо відомі координати інших вершин (x_1, y_1) , (x_2, y_2) , та (x_3, y_3) ,
 - а) якщо сторони прямокутника паралельні осям координат;
 - б) якщо сторони прямокутника не паралельні осям координат.

²⁰ Вибір задач узгодити з викладачем.

11. Визначити, чи належить точка A з координатами x, y прямокутнику з координатами вершин $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$.
12. Точка A з координатами x, y знаходиться в площині трикутника, вершини якого мають координати $(0, 5); (-3, 0); (5, -2)$. Визначити, до якої із сторін трикутника точка A знаходиться ближче всього. (Можливо використати формулу 3).
13. Визначити, яка з точок $A(x_1, y_1); B(x_2, y_2); C(x_3, y_3)$ найближча до початку координат. (Можливо використати формулу 4).
14. Визначити, чи пройде цеглина розміром $a \times b \times c$ в прямокутний отвір розміром $d \times f$ при умові, що одна із сторін цеглини паралельна краю отвору.
15. Визначити, чи пройде пластина в формі трикутника зі сторонами a, b, c та висотою h у прямокутне вікно $f \times d$, при умові, що одна із сторін трикутника паралельна краю вікна.

Примітка. При розв'язанні задач можна використати наступні формули:

1. Площа трикутника: $S = \frac{a \times h_a}{2}$, де h_a – висота трикутника до сторони a .
2. Формула Герона: $S = \sqrt{p \cdot (p-a) \cdot (p-b) \cdot (p-c)}$, де $p = \frac{a+b+c}{2}$.
3. Довжина відрізка AB ($A(x_1, y_1), B(x_2, y_2)$) дорівнює: $l_{AB} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.
4. Якщо точка $D(x, y)$ знаходиться у площі трикутника ABC , де $A(x_1, y_1), B(x_2, y_2), C(x_3, y_3)$, то $S_{ABC} = S_{ADC} + S_{CDB} + S_{ADB}$.

3.3 Опис лабораторних засобів та обладнання

Лабораторна робота №3 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

3.4 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

3.5 Послідовність виконання практичної роботи

У рамках практичної роботи виконати наступні дії:

- виконати вправи з програмування розгалужених алгоритмів (відповідно до прикладів, наданих на лекції).

3.6 Вказівки до виконання лабораторної роботи №3

Завдання №1

1. На листі MS Excel виконати розрахунок свого варіанту індивідуального завдання, для чого записати всі формули для обчислення шуканих величин. Підібрати такі значення вихідної змінної t (або інших змінних за вказівкою викладача), щоб перевірити всі варіанти обчислення змінної Y .
2. Розробити розгалужений алгоритм розв'язання індивідуального завдання та створити в редакторі Visual Basic процедуру, що його реалізує за допомогою блочного умовного оператора. Передбачити виведення як отриманих результатів (кінцевого і проміжних), так і вихідних даних²¹.
3. Налаштувати програму²². При виконанні розрахунків використати підібрані значення вихідних змінних (п.1). Порівняти результати отримані у програмі з результатами, отриманими в MS Excel.
4. Продемонструвати роботу програми викладачу.

Завдання №2

5. У відповідності до свого варіанту завдання розробити варіант розгалуженого алгоритму для використання рядкового умовного оператора та написати відповідну процедуру в редакторі Visual Basic. Передбачити виведення як отриманих результатів (кінцевого і проміжних), так і вихідних даних²³.
6. Налаштувати програму²⁴. При виконанні розрахунків використати підібрані значення вихідних змінних (п.1). Порівняти результати отримані у програмі з результатами, отриманими першому варіанті програми (п.2,3) та в MS Excel.
7. Продемонструвати роботу програми викладачу.
8. Оформити протокол лабораторної роботи²⁵.

3.7 Обробка та аналіз результатів. Оформлення звіту з лабораторної роботи

Протокол лабораторної роботи оформлюється згідно умов наведених вище (див. лабораторні роботи №1 та №2). Необхідно проаналізувати результати, отримані за програмами, розробленими при виконанні двох завдань, та порівняти їх. Обов'язково необхідно перевірити правильність роботи кожної з гілок алгоритмів. Слід порівняти алгоритми, використані в кожному з завдань, – якщо в різних завданнях застосовано один і той же алгоритм, то наводити його треба лише при першому застосуванні.

²¹ Використати функцію **MsgBox**.

²² Перевірити роботу всіх гілок розробленого алгоритму.

²³ Використати функцію **MsgBox**.

²⁴ Перевірити роботу всіх гілок розробленого алгоритму.

²⁵ Алгоритм задачі навести у вигляді блок-схеми.

3.8 Контрольні запитання

1. Який алгоритм називають таким, що розгалужується?
2. Які існують варіанти умовного оператора? В чому їх відмінність?
3. Що відбувається, якщо умова, що перевіряється в операторі IF, виявляється хибною?
4. Які існують логічні операції?
5. Що таке складна умова? Як вона записується?
6. Який порядок дій при виконанні повного оператора IF?
7. Який пріоритет арифметичних і логічних операцій?

4. Програмування розгалужень за допомогою оператора багатоваріантного вибору

Мета та основні завдання роботи: дослідити особливості створення елементарних програмних засобів розгалуженої структури у мові програмування Visual Basic.

4.1 Основні теоретичні відомості

Оператор багатоваріантного вибору SELECT CASE подібно до блочного оператора **IF** може реалізовувати алгоритми розгалуженої структури. Але його відмінність полягає в тому, що всі варіанти розгалуження в ньому залежать від значення тільки одного параметру, який визначається в головному операторі **Select Case**.

```
Select Case <параметр>
    Case K1
        <1-а група операторів>
    Case K2
        <2-а група операторів>
    ...
    Case Else
        <n-а група операторів>
End Select
```

Варіанти форматів оператора **Case**:

1. Аргументи в списку оператора **Case** задаються переліком значень (розділяються комами)

Приклади (перевіряється чи дорівнює значення параметру заданим числам):

```
Case 1,3,5
Case 2,4,
Case 0
```

2. Аргументи в списку оператора **Case** задаються діапазоном значень:

```
Case <вираз 1> TO < вираз 2>
```

(значення < вираз 2> повинно бути більше значення < вираз 1>).

Приклади (перевіряється чи знаходиться значення параметру у вказаному діапазоні):

```
Case 1 To 9
Case 2.5 To 273.2
Case -7 To -3
```

3. Аргументи в списку оператора **Case** задаються за допомогою умови:

```
Case Is < знак відношення> <вираз>
```

Приклади (перевіряється чи влаштовує значення параметру заданій умові):

Case is < 1

Case is > 9

Case Is < 2.1 + b ^ 2

Нижче наведений приклад програми із використанням оператора **SELECT CASE**:

Public Sub Vybir1()

' Оператор багатоваріантного вибору. **Приклад 1.**

Dim a As Single

a = Application.InputBox("Задайте число", , , , , , 1)

Select Case a

Case 1, 3, 5, 7, 9

MsgBox "a=" & a & " - непарне в інтервалі [1,9]"

Case 2, 4, 6, 8

MsgBox "a=" & a & " - парне в інтервалі [1,9]"

Case Is < 1

MsgBox "a=" & a & " < 1"

Case Is > 9

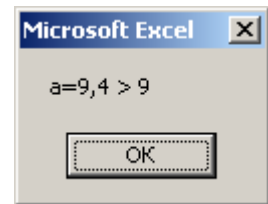
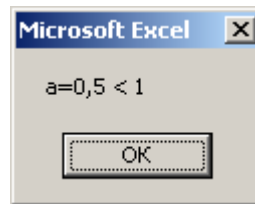
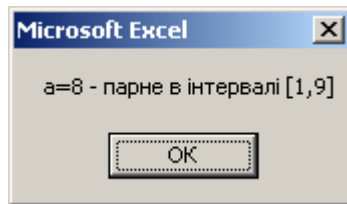
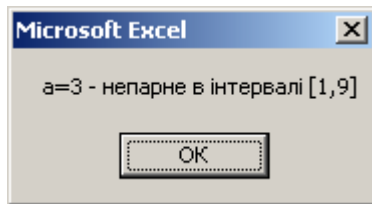
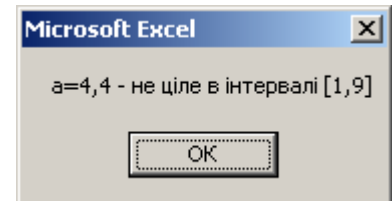
MsgBox "a=" & a & " > 9"

Case Else

MsgBox "a=" & a & " - не ціле в інтервалі [1,9]"

End Select

End Sub



Далі наведений варіант програми із використанням оператора **SELECT CASE** для розв'язання прикладу №5.

Sub TestRezalt5()

' Оператор багатоваріантного вибору. **Приклад 2.**

Dim sName As String, sMessage As String, iMark As Integer

sName = InputBox("Ваше прізвище?")

iMark = Application.InputBox("Скільки балів Ви набрали?", , , , , , 1)

sMessage = sName & ". Ви набрали " & iMark & " балів." & vbCrLf & "Ваша оцінка - "

Select Case iMark 'Блок вибору оцінки залежно від кількості набраних балів

Case Is >= 82

sMessage = sMessage & "5 (A)"

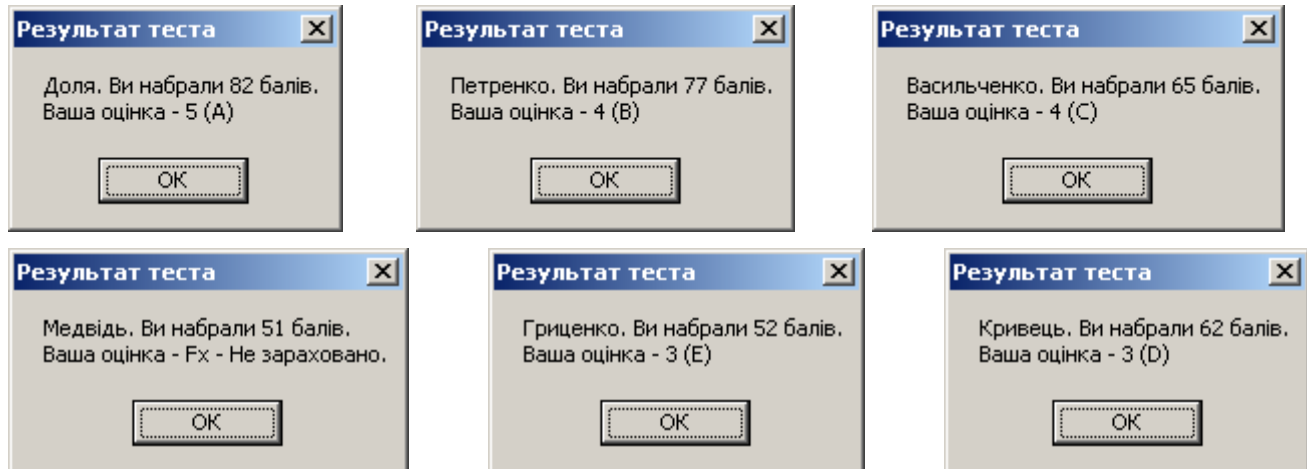
Case 73 To 81

```

    sMessage = sMessage & "4 (B)"
Case 65 To 72
    sMessage = sMessage & "4 (C)"
Case 56 To 64
    sMessage = sMessage & "3 (D)"
Case 52 To 55
    sMessage = sMessage & "3 (E)"
Case Is < 52
    sMessage = sMessage & "Fx - Не зараховано."
End Select
    MsgBox sMessage, , "Результат теста"
End Sub

```

Приклади результатів, отриманих за програмою **Sub TestRezalt5()**



Sub Vybir2()

' Оператор багатоваріантного вибору. **Приклад 3.**

```
Dim x As Single, y As Single, a As Single, b As Single, var As Integer
```

```
a = 3: b = 2
```

```
x = InputBox("Введіть x")
```

```
Select Case x
```

```
  Case 1
```

```
    y = Exp(x)
```

```
    var = 1
```

```
  Case Is < 1
```

```
    y = a + b ^ 2
```

```
    var = 4
```

```
  Case Is < 2
```

```
    y = Tan(Sqr(x))
```

```
    var = 3
```

```
  Case Is <= 5
```

```
    y = Cos(x ^ 2) ^ 2
```

```
    var = 2
```


Case Else

y = a + b ^ 2

var = 4

End Select

MsgBox "x=" & x & " y=" & y & vbNewLine & "варіант" & var
End Sub

4.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи вивчити та описати в протоколі лабораторної роботи такі поняття (визначення) і операції (команди):

– правила запису та роботи оператора **SELECT CASE**;

4.3 Опис лабораторних засобів та обладнання

Лабораторна робота №3 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

4.4 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

4.5 Вказівки до виконання лабораторної роботи №4

У рамках практичної роботи виконати наступні дії:

1. Розробити розгалужений алгоритм розв'язання індивідуального завдання (використати завдання з лабораторної роботи №3) та створити в редакторі Visual Basic процедуру, що його реалізує за допомогою оператора **SELECT CASE**. Передбачити виведення як отриманих результатів (кінцевого і проміжних), так і вихідних даних²⁶.
2. Налаштувати програму²⁷. При виконанні розрахунків використати підібрані значення вихідних змінних. Порівняти результати отримані у програмі з результатами, отриманими в лабораторній роботі №3.
3. Продемонструвати роботу програми викладачу.
4. Оформити протокол лабораторної роботи²⁸.

²⁶ Використати функцію **MsgBox**.

²⁷ Перевірити роботу всіх гілок розробленого алгоритму.

²⁸ Алгоритм задачі навести у вигляді блок-схеми.

4.6 Обробка та аналіз результатів. Оформлення звіту з лабораторної роботи

Протокол лабораторної роботи оформлюється згідно умов наведених вище (див. лабораторні роботи №1 та №2). Необхідно проаналізувати результати, отримані за програмами, розробленими при виконанні всіх трьох завдань отриманими в лабораторній роботі №3 та №4, та порівняти їх. Обов'язково необхідно перевірити правильність роботи кожної з гілок алгоритмів. Слід порівняти алгоритми, використані в кожному з завдань, – якщо в різних завданнях застосовано один і той же алгоритм, то наводити його треба лише при першому застосуванні.

4.7 Контрольні запитання

1. Який порядок дій при виконанні оператора SELECT CASE?
2. Коли і чому можна використати інструкцію SELECT CASE?
3. Які умови можуть бути записані після ключового слова CASE
4. Що загального в роботі оператора IF і оператора вибору SELECT CASE?

5. Організація арифметичних циклів

Мета та основні завдання роботи: дослідити особливості створення елементарних програмних засобів циклічної структури у мові програмування Visual Basic.

5.1 Основні теоретичні відомості

Циклічним називають алгоритм, в якому певна група операцій повторюється декілька разів. Арифметичні цикли (цикли з лічильником) – це цикли, в яких кількість повторень циклу відома (може бути розрахована) до його виконання. Для програмування таких циклів використовується оператор **FOR...NEXT**. Синтаксис цього оператора має вигляд:

```
FOR <змінна>=<початок> TO <кінець> [ STEP <крок> ]  
    ...  
    [ EXIT FOR ]  
    ...  
NEXT <змінна>
```

} Тіло циклу

де <змінна> – це змінна – *параметр циклу*, <початок> та <кінець> – відповідно початкове та кінцеве значення цієї змінної, а <крок> – крок її змінення. Якщо крок не задано, він за умовчанням приймає значення +1.

Виконання циклу можна перервати достроково за допомогою оператора **Exit For**. В цьому випадку програма продовжує свою роботу з оператора, що слідує за оператором **Next**.

Приклади заголовків циклів:

For i = 1 To 28

For i = m To n

For k = 23 To -44 Step -3

For x = xn To xk Step dx

For y = 0.15 To 4.5 Step 0.25

For z = 1.9 To 0.3 Step -0.05

Алгоритм, за яким працює оператор **For...Next**, наведено на рис. 4.1²⁹. Тобто, слід пам'ятати, що в операторі циклу **For...Next** умова завершення циклу перевіряється на його початку (цикл з передумовою). Тому, якщо <початок> > <кінець> (<крок> > 0) або, якщо <початок> < <кінець> (<крок> < 0), то цикл не буде виконаний жодного разу – програма продовжить своє виконання з оператора, який знаходиться після оператора **Next**. Тому значення параметру циклу в операторі **For** слід ретельно перевіряти.

Одним з типових прикладів циклічних алгоритмів є задача табулювання функції. Табулювання функції – це обчислення значень функції при зміні аргументу від деякого початкового значення до деякого кінцевого значення з певним кроком.

²⁹ Наведено приклад циклічного алгоритму, коли кінцеве значення більше за початкове.

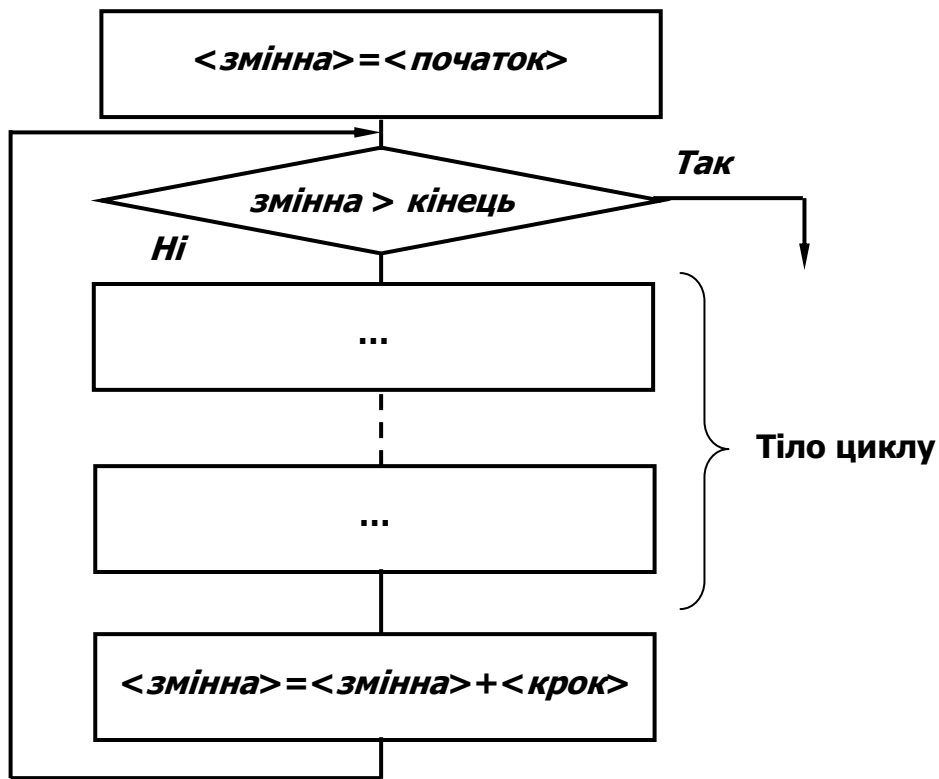


Рис. 4.1 – Алгоритм роботи оператора циклу (коли **<крок>** додатний)

Нижче наведено приклад програми табулювання функції $y = f(x)$, де $y = \alpha \sin x^2 + e^{\beta x}$:

Public Sub Tabul1()

'

' Приклад циклічної програми табулювання функції $y = f(x)$

'

Dim alfa As Single, beta As Single, xn As Single, xk As Single, dx As Single

Dim x As Single, y As Single

alfa = InputBox("Введіть коефіцієнт alfa", "Введення коефіцієнтів")

beta = InputBox("Введіть коефіцієнт beta", "Введення коефіцієнтів")

xn = InputBox("Введіть початкове значення x – xn", "Введення даних")

xk = InputBox("Введіть кінцеве значення x – xk", "Введення даних")

dx = InputBox("Введіть крок змінення x – dx", "Введення даних")

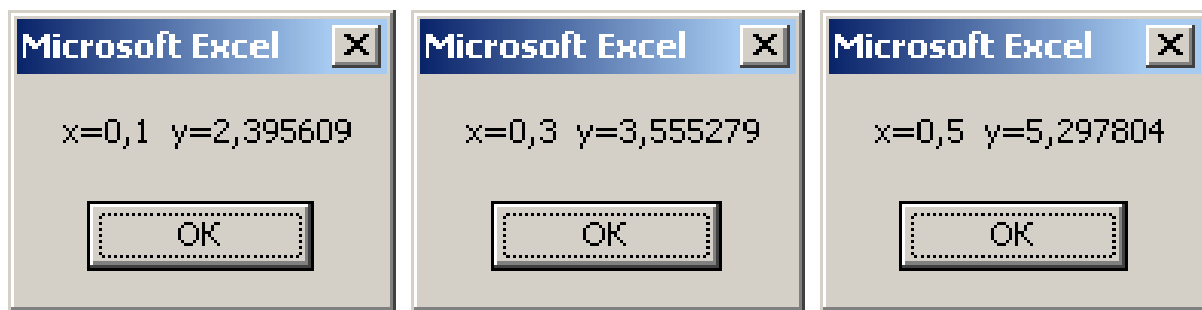
For x = xn To xk Step dx

$y = \alpha \sin(x^2) + 2 * \exp(\beta * x)$

MsgBox "x=" & x & " y=" & y

Next x

End Sub



Public Sub Tabul2()

' Приклад циклічної програми табулювання функції $y = f(x)$
 ' Формування таблиці для виведення у вікні функції MsgBox

```
Dim alfa As Single, beta As Single, xn As Single, xk As Single, dx As Single
Dim x As Single, y As Single, sTab As String
alfa = InputBox("Введіть коефіцієнт alfa", "Введення коефіцієнтів")
beta = InputBox("Введіть коефіцієнт beta", "Введення коефіцієнтів")
xn = InputBox("Введіть початкове значення x - xn", "Введення даних")
xk = InputBox("Введіть кінцеве значення x - xk", "Введення даних")
dx = InputBox("Введіть крок змінення x - dx", "Введення даних")
sTab = "Таблиця залежності функції  $y=f(x)$ "
sTab = sTab & vbCrLf & " x " & " y "
For x = xn To xk Step dx
    y = alfa * Sin(x ^ 2) + 2 * Exp(beta * x)
    sTab = sTab & vbCrLf & x & " " & y
Next x
MsgBox sTab, , "Таблиця результатів"
End Sub
```

Public Sub Tabul3()

' Приклад циклічної програми - табулювання функції $y = f(x)$
 ' Виведення таблиці у вікні Immediate

```
Dim alfa As Single, beta As Single, xn As Single, xk As Single, dx As Single
Dim x As Single, y As Single, sTab As String
alfa = InputBox("Введіть коефіцієнт alfa", "Введення коефіцієнтів")
beta = InputBox("Введіть коефіцієнт beta", "Введення коефіцієнтів")
xn = InputBox("Введіть початкове значення x - xn", "Введення даних")
xk = InputBox("Введіть кінцеве значення x - xk", "Введення даних")
dx = InputBox("Введіть крок змінення x - dx", "Введення даних")
Debug.Print "Таблиця залежності функції  $y=f(x)$ "
Debug.Print "x", "y"
For x = xn To xk Step dx
    y = alfa * Sin(x ^ 2) + 2 * Exp(beta * x)
    Debug.Print x, y
Next x
End Sub
```

Результати виконання програм **Tabul2** та **Tabul3**:

Таблиця результатів	
Таблиця залежності функції $y=y(x)$	
x	y
0,1	2,395609
0,3	3,555279
0,5	5,297804
0,7	7,750727
0,9	11,04707
1,1	15,31563
1,3	20,7137

Immediate	
Таблиця залежності функції $y=y(x)$	
x	y
0,1	2,395609
0,3	3,555279
0,5	5,297804
0,7	7,750727
0,9	11,04707
1,1	15,31563
1,3	20,7137

```
Debug.Print "Таблиця залежності функції  $y=y(x)$ "
```


```
Debug.Print "x"; "y"
```

```
For x = xn To xk Step dx
```


```
    y = alfa * Sin(x ^ 2) + 2 * Exp(beta * x)
```

```
    Debug.Print x; y
```

```
Next x
```



Immediate	
Таблиця залежності функції $y=y(x)$	
xу	
0,1	2,395609
0,3	3,555279
0,5	5,297804
0,7	7,750727
0,9	11,04707
1,1	15,31563
1,3	20,7137



Immediate	
Таблиця залежності функції $y=y(x)$	
x	y
0,1	2,395609
0,3	3,555279
0,5	5,297804
0,7	7,750727
0,9	11,04707
1,1	15,31563
1,3	20,7137

```
Debug.Print "Таблиця залежності функції  $y=y(x)$ "
```

```
Debug.Print Tab(3); "x"; Tab(9); "y"
```

```
For x = xn To xk Step dx
```

```
    y = alfa * Sin(x ^ 2) + 2 * Exp(beta * x)
```

```
    Debug.Print Tab(2); x; Tab(8); y
```

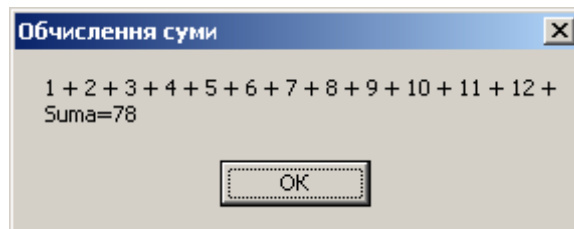
```
Next x
```

Обчислення сум та добутків

Обчислення сум та добутків відбувається шляхом накопичення. Як видно з подальшого прикладу, сума накопичується у змінній **Sum**, який попередньо надано нульове значення. В циклі на кожному кроці до змінної **Sum** додається один доданок (для даного прикладу – одне з чисел натурального ряду – **i**), а цикл повторюється стільки разів, скільки доданків в сумі.

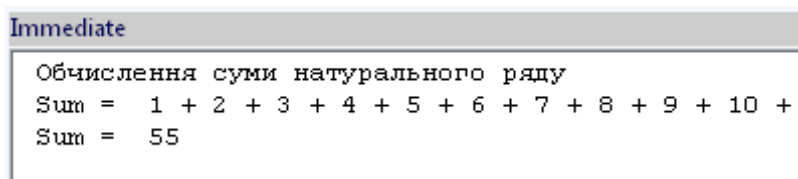
Public Sub Suma()

```
' Приклад накопичення суми чисел натурального ряду
Dim i As Integer, n As Integer, sRez As String, Sum As Single
n = InputBox("Введіть значення n")
Sum = 0
For i = 1 To n
    Sum = Sum + i
    sRez = sRez & i & " + "
Next i
sRez = sRez & vbCrLf & "Suma=" & Sum
MsgBox sRez, , "Обчислення суми"
End Sub
```



Public Sub Suma_D()

```
' Приклад накопичення суми чисел натурального ряду
Dim i As Integer, n As Integer, sRez As String, Sum As Single
n = InputBox("Введіть значення n")
Sum = 0
Debug.Print "Обчислення суми натурального ряду"
Debug.Print "Sum = ";
For i = 1 To n
    Sum = Sum + i
    Debug.Print i; "+";
Next i
Debug.Print
Debug.Print "Sum = "; Sum
End Sub
```



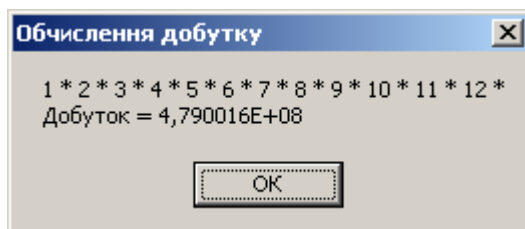
Public Sub Dobutok()

```
' Приклад накопичення добутку чисел натурального ряду
Dim i As Integer, n As Integer, sRez As String, Dob As Integer
n = InputBox("Введіть значення n")
Dob = 1
For i = 1 To n
    Dob = Dob * i
```

```

sRez = sRez & i & " * "
Next i
sRez = sRez & vbCrLf & "Добуток = " & Dob
MsgBox sRez, , "Обчислення добутку"
End Sub

```



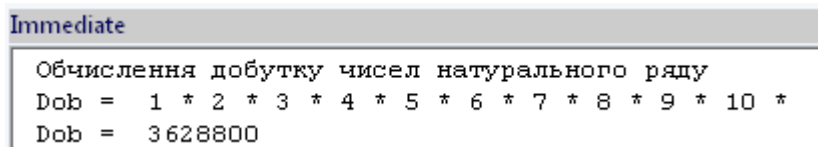
Примітка. Якщо **Dob As Integer**, при **n>7** виникає переповнення (**Overflow**)

Якщо **Dob As Single**, при **n>34** виникає переповнення (**Overflow**)

```

Public Sub Dobutok_D()
' Приклад накопичення добутку чисел натурального ряду
Dim i As Integer, n As Integer, sRez As String, Dob As Double
n = InputBox("Введіть значення n")
Dob = 1
Debug.Print "Обчислення добутку чисел натурального ряду"
Debug.Print "Dob = ";
For i = 1 To n
    Dob = Dob * i
    Debug.Print i; "*";
Next i
Debug.Print
Debug.Print "Dob = "; Dob
End Sub

```



5.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи вивчити та описати в протоколі лабораторної роботи такі поняття (визначення) і операції (команди):

- види циклічних алгоритмів, правила запису та роботи оператора циклу FOR... NEXT;
- можливості виведення таблиць:
 - за допомогою функції **MsgBox**;

- за допомогою інструкції **Debug.Print**;
 - у комірці листа MS Excel;
- властивості та методи діапазонів (комірок), які використовуються у прикладах, розглянутих на лекції;
 - користуючись завданням для лабораторної роботи № 3 («Розробка та програмування алгоритмів розгалуженої структури») розробити циклічний алгоритм і програму розрахунку заданої функції для 20 значень аргументу. Набрати і налаштувати програму в середовищі VBA. Передбачити такий інтервал зміни значень незалежної змінної, щоб при виконанні розрахунків були використані всі гілки алгоритму (по можливості – приблизно в рівній мірі). Друк результатів організувати у вигляді таблиці (в одному зі стовпців навести номер використаної формули);
 - самостійно вивчити правила запису функцій **Tab()** та **Spс()** та можливості їх використання при форматуванні таблиць. Перевірити їх роботу при створенні таблиць в практичній та лабораторній роботах.

5.3 Опис лабораторних засобів та обладнання

Лабораторна робота №5 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

5.4 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

5.5 Послідовність виконання практичної роботи

У рамках практичної роботи виконати наступні дії:

- розглянути приклади циклічних алгоритмів і програм, що наведені у лекції та перевірити їх роботу. Проаналізувати можливості різних засобів формування таблиці. Перевірити операції програмного занесення та зчитування інформації в комірках листа MS Excel.
- розглянути правила роботи вкладених циклів та перевірити їх роботу на прикладі.

5.6 Вказівки до виконання лабораторної роботи №5

1. На новому листі MS Excel виконати розрахунок таблиці для вказаних значень аргументу відповідно до індивідуального варіанту завдання. Відформатувати побудовану таблицю.
2. У відповідності до свого варіанту завдання розробити циклічний алгоритм та програму розрахунку заданої функції в середовищі VBA. Друк результатів

організувати у вигляді таблиці, яка виводиться у комірки листа MS Excel³⁰, у вікно **Immediately** та за допомогою функції **MsgBox**. Значення констант, які використовуються при розрахунку таблиці, слід також вивести на лист MS Excel.

3. Створити макрос для форматування створюваної на листі таблиці. Після перевірки та видалення зайвих рядків макросу вставити необхідні оператори до тексту програми.
4. Порівняти результати отримані за програмою з результатами, з отриманими в MS Excel.
5. Продемонструвати роботу програми викладачу³¹.
6. Оформити протокол лабораторної роботи, до складу якого треба включити теоретичні відомості, опис та результати виконання індивідуального завдання (блок-схему алгоритму, текст програми, роздруковки листа з таблицями (пп.1–4), скріншот вікон **Immediately** та **MsgBox** з таблицями).

5.7 Обробка та аналіз результатів. Оформлення звіту

Протокол лабораторної роботи оформлюється згідно умов наведених вище (див. лабораторні роботи №1 та №2). Зробити висновки відносно доцільності використання функцій TAB() та SPC() при друкуванні таблиць.

5.8 Контрольні запитання

1. Що таке цикли і для чого вони призначені?
2. Які бувають цикли?
3. З яких основних елементів складається цикл?
4. Як записується і як працює оператор циклу FOR...NEXT?
5. Для організації яких циклів використовується оператор FOR...NEXT?
6. Коли можна опустити ключове слово STEP в чи циклі FOR...NEXT?
7. Чи можна організувати цикл з від'ємним кроком?
8. Як здійснити достроковий вихід з циклу?
9. Чи можна передати управління всередину циклу, минувши заголовок циклу?
10. При якій умові тіло циклу жодного разу не спрацює?

³⁰ Використати інший діапазон того ж листа, що і попередньому в завданні (п.8). Значення констант, які використовуються при розрахунку таблиці, слід також вивести на лист MS Excel.

³¹ Зберігати копії розроблених програм на своїй дискеті або іншому носії (до кінця семестру).

6. Організація вкладених циклів

Мета та основні завдання роботи: дослідити особливості використання складних вкладених циклів з параметром при створенні програмних засобів у мові програмування Visual Basic.

6.1 Основні теоретичні відомості

Цикли, як і інші блоки, можна вкладати один в одного. При цьому необхідно виконувати наступні основні рекомендації:

- параметри зовнішнього і внутрішнього циклів мають бути різними і не змінюватися одночасно;
- цикли не повинні «перетинатися» – внутрішній цикл обов'язково має бути закритий раніше зовнішнього (див. приклади нижче);
- якщо для закриття кожного циклу використовується свій оператор **NEXT**, то їх кількість повинна співпадати з кількістю операторів **FOR**;
- якщо для закриття декількох циклів використовується один оператор **NEXT**, то імена параметрів циклів мають бути вказані в ньому в порядку вкладеності циклів (першим вказується ім'я змінної самого внутрішнього циклу, потім наступного і так далі). В цьому випадку кількість змінних циклу повинна співпадати з кількістю операторів **FOR**. Вказівка імені змінній циклу в частині **NEXT** необов'язково. Проте, бажано ставити їх завжди, оскільки при вкладеності циклів така вказівка допомагає стежити за правильністю вкладення циклів і уникати значної кількості помилок.

Приклади:

```
For i = 1 To 10
    For j = -5 To 0 Step 0.5
        ....
    Next j
Next i
```

```
For i = 1 To 5
    k = InputBox("k = ")
    For j = -10 To 10 Step 2
        z = (j - k)^2
        Debug.Print j, z
    Next j
Next i
```

```
For i = 1 To 3
    For j = 4 To 7
        For k = 3 To 8
            Debug.Print i, j, k
        Next k
    Next j
Next i
```

```
For i = 1 To 3
    For j = 4 To 7
        For k = 3 To 8
            Debug.Print i, j, k
        Next k, j, i
    Next j
Next i
```

Нижче наведено приклад програми табулювання функції двох змінних $z = f(x, y)$, де $z = \alpha \sin x^2 + e^{\beta y}$:

Public Sub Tabul4()

' Приклад табулювання функції двох змінних $z=f(x, y)$

```
Dim alfa As Single, beta As Single, xn As Single, xk As Single, dx As Single
Dim x As Single, y As Single, z As Single, sTab As String
Dim yn As Single, yk As Single, dy As Single
alfa = InputBox("Введіть коефіцієнт alfa", "Введення коефіцієнтів")
beta = InputBox("Введіть коефіцієнт beta", "Введення коефіцієнтів")
xn = InputBox("Введіть початкове значення x - xn", "Введення даних")
xk = InputBox("Введіть кінцеве значення x - xk", "Введення даних")
dx = InputBox("Введіть крок змінення x - dx", "Введення даних")
yn = InputBox("Введіть початкове значення y - yn", "Введення даних")
yk = InputBox("Введіть кінцеве значення y - yk", "Введення даних")
dy = InputBox("Введіть крок змінення y - dy", "Введення даних")
sTab = "Таблиця залежності функції  $z=z(x,y)$ "
sTab = sTab & vbCrLf & " x " & " y " & " z "
For x = xn To xk Step dx
    For y = yn To yk Step dy
         $z = \text{alfa} * \text{Sin}(x^2) + 2 * \text{Exp}(\text{beta} * y)$ 
        sTab = sTab & vbCrLf & x & " " & y & " " & z
    Next y
Next x
MsgBox sTab, , "Таблиця результатів"
End Sub
```

Public Sub Tabul4d()

' Приклад вкладених циклів - табулювання функції від двох змінних $z = f(x, y)$

```
Dim alfa As Single, beta As Single, xn As Single, xk As Single, dx As Single
Dim x As Single, y As Single, z As Single, sTab As String
Dim yn As Single, yk As Single, dy As Single
alfa = InputBox("Введіть коефіцієнт alfa", "Введення коефіцієнтів")
beta = InputBox("Введіть коефіцієнт beta", "Введення коефіцієнтів")
xn = InputBox("Введіть початкове значення x - xn", "Введення даних")
xk = InputBox("Введіть кінцеве значення x - xk", "Введення даних")
dx = InputBox("Введіть крок змінення x - dx", "Введення даних")
yn = InputBox("Введіть початкове значення y - yn", "Введення даних")
yk = InputBox("Введіть кінцеве значення y - yk", "Введення даних")
dy = InputBox("Введіть крок змінення y - dy", "Введення даних")
Debug.Print "Таблиця залежності функції  $z=z(x,y)$ "
Debug.Print "x", "y", "z"
For x = xn To xk Step dx
```

```

For y = yn To yk Step dy
    z = alfa * Sin(x ^ 2) + 2 * Exp(beta * y)
    Debug.Print x, y, z
Next y
Next x
End Sub

```

Результати виконання програм **Tabul4** та **Tabul4d**:

Таблиця результатів		
Таблиця залежності функції $z=z(x,y)$		
x	y	z
0,2	0,4	3,732214
0,2	0,6	5,007183
0,2	0,8	6,72821
0,2	1	9,051354
0,2	1,2	12,18727
0,5	0,4	4,188526
0,5	0,6	5,463495
0,5	0,8	7,184523
0,5	1	9,507667
0,5	1,2	12,64358
0,8	0,4	4,958068
0,8	0,6	6,233037
0,8	0,8	7,954064
0,8	1	10,27721
0,8	1,2	13,41313
1,1	0,4	5,702593
1,1	0,6	6,977561
1,1	0,8	8,698589
1,1	1	11,02173
1,1	1,2	14,15765

Immediate		
Таблиця залежності функції $z=z(x,y)$		
x	y	z
0,2	0,4	3,732214
0,2	0,6	5,007183
0,2	0,8	6,72821
0,2	1	9,051354
0,2	1,2	12,18727
0,5	0,4	4,188526
0,5	0,6	5,463495
0,5	0,8	7,184523
0,5	1	9,507667
0,5	1,2	12,64358
0,8	0,4	4,958068
0,8	0,6	6,233037
0,8	0,8	7,954064
0,8	1	10,27721
0,8	1,2	13,41313
1,1	0,4	5,702593
1,1	0,6	6,977561
1,1	0,8	8,698589
1,1	1	11,02173
1,1	1,2	14,15765

Запис інформації в комірки листа MS Excel

Option Explicit

' **Приклад 4.**

' Вкладені цикли - табулювання функції від двох змінних $z = f(x,y)$

Public Sub Tabul4()

 Range("A1").Select

 ActiveCell.Value = "Таблиця залежності функції $z=z(x,y)$ "

 Range("A2").Select

 ActiveCell.Value = "x"

 ActiveCell.Offset(0, 1).Range("A1").Select

 ActiveCell.Value = "y"

 ActiveCell.Offset(0, 1).Range("A1").Select

 ActiveCell.Value = "z"

 sTab = "Таблиця залежності функції $z=z(x,y)$ "

```

sTab = sTab & vbNewLine & " x " & " y " & " z "
Range("A3").Select
For x = xn To xk Step dx
    For y = yn To yk Step dy
        z = alfa * Sin(x ^ 2) + 2 * Exp(beta * y)
        ActiveCell.Value = x
        ActiveCell.Offset(0, 1).Range("A1").Select
        ActiveCell.Value = y
        ActiveCell.Offset(0, 1).Range("A1").Select
        ActiveCell.Value = z
        sTab = sTab & vbNewLine & x & " " & y & " " & z
        ActiveCell.Offset(1, -2).Range("A1").Select
    Next y
Next x
MsgBox sTab, , "Таблиця результатів"
End Sub

```

	A	B	C	D	E	F
1	Таблиця залежності функції $z=z(x,y)$				alfa =	2,2
2	x	y	z		beta =	1,5
3	0,2	0,4	3,732214			
4	0,2	0,6	5,007183			
5	0,2	0,8	6,72821			
6	0,2	1	9,051354			
7	0,2	1,2	12,18727			
8	0,5	0,4	4,188526			
9	0,5	0,6	5,463495			
10	0,5	0,8	7,184523			
11	0,5	1	9,507667			
12	0,5	1,2	12,64358			
13	0,8	0,4	4,958068			
14	0,8	0,6	6,233037			
15	0,8	0,8	7,954064			
16	0,8	1	10,27721			
17	0,8	1,2	13,41313			
18	1,1	0,4	5,702593			
19	1,1	0,6	6,977561			
20	1,1	0,8	8,698589			
21	1,1	1	11,02173			
22	1,1	1,2	14,15765			

x	y	z
0,2	0,4	3,732214
0,2	0,6	5,007183
0,2	0,8	6,72821
0,2	1	9,051354
0,2	1,2	12,18727
0,5	0,4	4,188526
0,5	0,6	5,463495
0,5	0,8	7,184523
0,5	1	9,507667
0,5	1,2	12,64358
0,8	0,4	4,958068
0,8	0,6	6,233037
0,8	0,8	7,954064
0,8	1	10,27721
0,8	1,2	13,41313
1,1	0,4	5,702593
1,1	0,6	6,977561
1,1	0,8	8,698589
1,1	1	11,02173
1,1	1,2	14,15765

6.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи повторити правила запису та роботи оператора циклу FOR... NEXT; вивчити та описати в протоколі лабораторної роботи:

- принципи побудови вкладених циклів (блоків);
- намалювати блок-схему алгоритму табулювання функції двох змінних для прикладу наведеного вище;
- організацію циклічних алгоритмів, правила запису та роботи вкладених циклів;
- розглянути приклад програми, наведеної в лекції, перевірити її роботу, проаналізувати порядок занесення інформації в комірки листа MS Excel при побудові таблиці.

6.3 Опис лабораторних засобів та обладнання

Лабораторна робота №6 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

6.4 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

6.5 Вказівки до виконання лабораторної роботи №6

1. На листі MS Excel виконати розрахунок таблиці (значень *доданків/співмножників*³²) для вказаних значень аргументів (параметру *суми/добутку* та змінної *a*) відповідно до індивідуального варіанту завдання. Відформатувати побудовану таблицю.
2. Розробити циклічний алгоритм розрахунку суми або добутку (у відповідності до свого варіанту завдання), що включає два вкладені цикли (в залежності від значення змінної *a* та від параметру *суми/добутку*).
3. Реалізувати розроблений алгоритм у вигляді процедури в середовищі VBA, яка при розрахунку *суми/добутку* для кожного значення змінної *a* створює на листі MS Excel³³ таблицю (див. приклад), в якій наведені значення параметру *суми/добутку*, *доданків/співмножників* та поточних *сум/добутків*, та виводить значення результату (*суми/добутку*). Крім того, програма повинна в залежності від бажання користувача³⁴ вивести вікно функції **MsgBox** з такою ж таблицею, як і на листі MS Excel, для заданого значення змінної *a*.

Примітка. В програмі слід передбачити, що у випадках, якщо значення чисельника або знаменника *доданку/співмножника* таке, що не дозволяє його розрахувати (наприклад, рівність нулю знаменника або чисельника)³⁵, слід не

³² Для зручності краще додатково окремо порахувати чисельник та знаменник кожного *доданку/співмножника*.

³³ Зручно використати інший діапазон того ж листа, що і в попередньому в завданні (п.3), так, щоб обидва результати можна було роздрукувати на одній сторінці для порівняння.

³⁴ Використати функцію **MsgBox** з кнопками «Да» та «Нет» або «Yes», «No» та «Cancel».

³⁵ Запрограмувати необхідні перевірки.

враховувати цей доданок/співмножник при розрахунку результату, а в таблиці проти відповідного значення аргументу (параметру суми/добутку) надрукувати відповідне повідомлення (див. приклад табл.).

4. Для форматування таблиці, що виводиться на листі MS Excel при роботі програми (п.3), створити макрос. Зробити копію процедури під змінним ім'ям. Після вилучення зайвих рядків макросу скопіювати його оператори до тексту копії програми. Перевірити роботу нового варіанту програми на іншому листі MS Excel.
5. Порівняти результати, отримані у програмі (п.3), з результатами, з отриманими в MS Excel (п.1).
6. Продемонструвати результати викладачу.
7. Програму зберегти на дискеті (або іншому носії).
8. Оформити протокол лабораторної роботи, до складу якого треба включити теоретичні відомості, опис та результати виконання індивідуального завдання (блок-схему алгоритму, текст програми (або після додання макросу, або програму та макрос окремо), роздруківки листа (листів) з таблицями (пп.1, 3, 4), скріншоти вікон з таблицею).

Табл. 2. $a = 4$

i	Dod_i	S_i
-5	0,1667	0,1667
-4	0,0667	0,2333
-3	0,0000	0,2333
-2	0,3333	0,5667
-1	знаменник =0	
0	-9,0000	-8,4333
1	знаменник =0	
2	8,3333	-0,1000
3	4,5000	4,4000
4	3,2667	7,6667

Рис. 6.1 – Приклад таблиці:

6.6 Обробка та аналіз результатів. Оформлення звіту

Протокол лабораторної роботи оформлюється згідно умов наведених вище (див. лабораторні роботи №1 та №2). Зробити висновки відносно взаємодії циклів та обґрунтувати доцільність кожної з використаних перевірок в алгоритмі.

6.7 Контрольні запитання

1. Наведіть правила запису вкладених циклів та їх графічне зображення.
2. Скласти блок-схему вкладених циклів.
3. Скільки разів виконується тіло вкладеного циклу?
4. Чи можна одним оператором NEXT закрити декілька циклів?
5. Чи можуть вкладені цикли складатися з операторів циклу різного типу?
6. Яка помилка виникає, якщо блочний умовний оператор, відкритий у циклі, закривається після ключового слова NEXT?

7.1.2. Операції з масивами

Задання масиву

1) За допомогою операторів введення:

```
n = InputBox("Розмір масиву n?")
ReDim a(n) As Single
For i = 1 To n
    x(i) = InputBox("Введіть значення a(" & i & ")", "Введення масиву")
Next i
...
```

2) за правилом арифметичної прогресії:

```
xn = InputBox("Введіть значення xn")
xk = InputBox("Введіть значення xk")
dx = InputBox("Введіть значення dx")
n = (xk - xn) / dx + 1
ReDim x(n)
For i = 1 To n
    x(i) = xn + dx*(i - 1)
Next i
...
```

3) за допомогою датчика випадкових чисел³⁶

```
n = InputBox("Розмір масиву n?")
ReDim x(n), y(n)
Randomize Timer
For i = 1 To n
    x(i) = Rnd * 100
    y(i) = Int(x(i))
Next i
...
```

Примітка. Перевірьте чим відрізняються масиви **x** та **y**

Якщо необхідно, щоб значення були і додатні, і від'ємні (*наприклад, в межах від -100 до 100*), то:

```
x(i) = Int(Rnd * 200) - 100
...
```

Public Sub Array4()

'Приклад задання масиву за допомогою датчика випадкових чисел
Dim i As Integer, n As Integer, sArr As String

³⁶ Датчик випадкових чисел **Rnd** генерує випадкове число в межах від 0 до 1.

```

Dim a() As Single, b() As Single
n = InputBox("Розмір масиву n")
ReDim a(n) As Single, b(n) As Single
Range("A1").Select
sArr = "Масив:" & vbNewLine
ActiveCell.Value = "Масив A"
Range("A2").Select
For i = 1 To n
    a(i) = Rnd * 100
    ActiveCell.Value = a(i)
    ActiveCell.Offset(0, 1).Range("A1").Select
    b(i) = Int(a(i))
    ActiveCell.Value = b(i)
    sArr = sArr & b(i) & ", "
    ActiveCell.Offset(1, -1).Range("A1").Select
Next i
MsgBox sArr
End Sub

```

Public Sub Array1()

' **Приклад 1** використання одновимірних масивів

```

Dim i As Integer, n As Integer, Sum As Single, sArr As String

```

```

Dim a() As Single
n = InputBox("Розмір масиву n?")

```

```

ReDim a(n) As Single

```

```

Range("A1").Select

```

```

sArr = "Масив A:" & vbNewLine

```

```

ActiveCell.Value = "Масив A"

```

```

Range("A2").Select

```

```

For i = 1 To n

```

```

    a(i) = InputBox("Введіть значення a(" & i & ")", "Введення масиву")

```

```

    sArr = sArr & a(i) & ", "

```

```

    ActiveCell.Value = a(i)

```

```

    ActiveCell.Offset(1, 0).Range("A1").Select

```

```

Next i

```

```

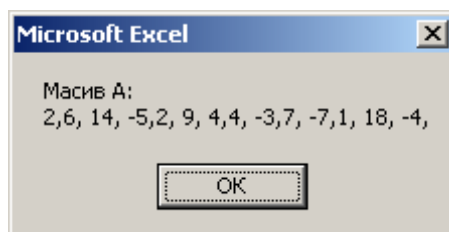
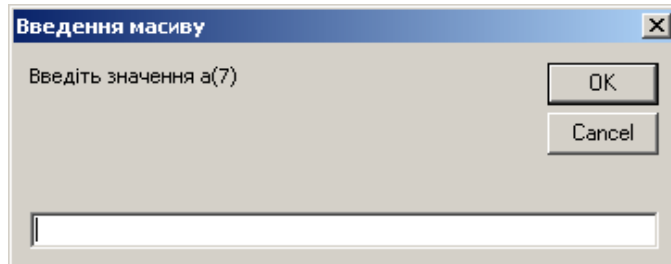
MsgBox sArr

```

```

End Sub

```



	A	B	C
1	Масив A		
2	2,6		
3	14		
4	-5,2		
5	9		
6	4,4		
7	-3,7		
8	-7,1		
9	18		
10	-4		
11	28	СУММ(A2:A10)	

Друкування масиву

Public Sub Array4b()

Dim i As Integer, n As Integer, sArr As String

Dim a() As Single

n = InputBox("Розмір масиву n")

ReDim a(n) As Single

Randomize Timer

Debug.Print "Масив A"

For i = 1 To n

a(i) = Int(Rnd * 200) - 100

Debug.Print a(i)

Next i

End Sub

Immediate
Масив A
-54
-84
10
0
-43
87
-70
-5
-41
-23

Immediate
Масив A
28
-64
72
70
99
29
-75
-30
73
-67

Public Sub Array4b()

Dim i As Integer, n As Integer, sArr As String

Dim a() As Single

n = InputBox("Розмір масиву n")

ReDim a(n) As Single

Randomize Timer

Debug.Print "Масив A"

For i = 1 To n

a(i) = Int(Rnd * 200) - 100

Debug.Print a(i);

Next i

Debug.Print

End Sub

Immediate
Масив A
-51 11 7 -37 -1 -32 69 32 78 -73 58 -86 -67 -96 -60

7.1.3. Характерні прийоми програмування

Обчислення суми та добутку здійснюється шляхом накопичення (див. розділ 4). Алгоритми наведені на рис. 7.1.

Кількість елементів розраховується так само, як і сума, з тією різницею, що доданком в процесі накопичення є **1**:

$$k = k + 1$$

Пошук елементів масиву за необхідною ознакою є однією з розповсюджених задач. В якості таких ознак може бути знак числа (додатне, від'ємне), кратність числа, якщо елементи масиву цілі, якомусь дільнику, найменший або найбільший елемент, тощо.

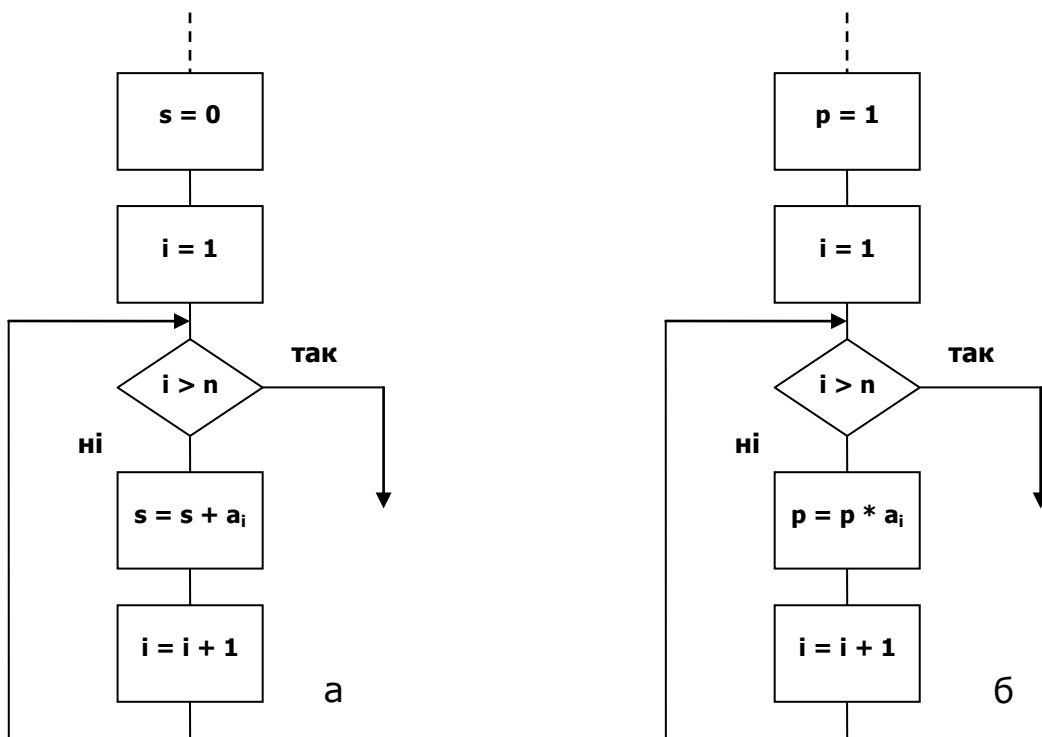


Рис. 7.1. – Алгоритми обчислення суми (а) та добутку (б)

Приклад 1. Обчислення окремо сум додатних та від’ємних елементів масиву:

Public Sub Find1b()

Dim i As Integer, n As Integer, sArr As String

Dim a() As Single, SumP As Single, SumM As Single

n = InputBox("Розмір масиву n")

ReDim a(n) As Single

Randomize Timer

sArr = "Масив:" & vbCrLf

Range("A1").Select

ActiveCell.Value = "Масив A"

Range("A2").Select

Debug.Print "Масив A"

For i = 1 To n

a(i) = Int(Rnd * 200) - 100

sArr = sArr & a(i) & ", "

Debug.Print a(i);

ActiveCell.Value = a(i)

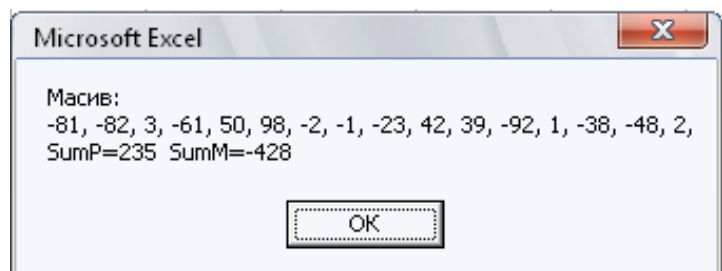
ActiveCell.Offset(0, 1).Range("A1").Select

Next i

SumP = 0: SumM = 0

For i = 1 To n

If a(i) > 0 Then



```

        SumP = SumP + a(i)
    Else
        SumM = SumM + a(i)
    End If
Next i
Debug.Print
Debug.Print "SumP="; SumP, "SumM="; SumM
MsgBox sArr & vbNewLine & "SumP=" & SumP & " SumM=" & SumM
Range("A3").Select
ActiveCell.Value = "SumP="
ActiveCell.Offset(0, 1).Range("A1").Select
ActiveCell.Value = SumP
ActiveCell.Offset(0, 1).Range("A1").Select
ActiveCell.Value = "SumM="
ActiveCell.Offset(0, 1).Range("A1").Select
ActiveCell.Value = SumM
End Sub

```

Immediate

```

Масив А
-81 -82 3 -61 50 98 -2 -1 -23 42 39 -92 1 -38 -48 2
SumP= 235 SumM=-428

```

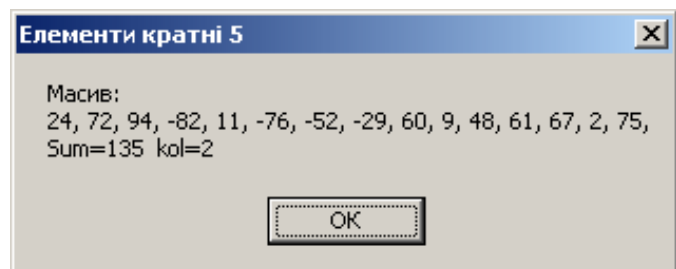
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Масив А															
2	-81	-82	3	-61	50	98	-2	-1	-23	42	39	-92	1	-38	-48	2
3	SumP=	235	SumM	-428												

Приклад 2. Обчислення сум та кількостей елементів масиву, кратних п'яти:

```

Public Sub Find2()
...
...
    Sum = 0: kol = 0
    For i = 1 To n
        If a(i) Mod 5 = 0 Then
            Sum = Sum + a(i)
            kol = kol + 1
        End If
    Next i
    MsgBox sArr & vbNewLine & "Sum=" & Sum & " kol=" & kol, , "Елементи кратні 5"
End Sub

```



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Масив А														
2	24	72	94	-82	11	-76	-52	-29	60	9	48	61	67	2	75

Примітка. Перевірка на кратність виконується шляхом обчислення залишку від ділення ($z(i) \text{ MOD } 5$) і перевірки чи дорівнює він нулю. Для перевірки на кратність також може використовуватись порівняння результатів звичайного та цілочисельного ділення:

IF $z(i) / 5 = z(i) \setminus 5$ THEN

Приклад 3. Розрахунок середнього значення елементів, кратних п'яти:

Public Sub Find2a()

...

...

Sum = 0: kol = 0

For i = 1 To n

If a(i) Mod 5 = 0 Then

Sum = Sum + a(i)

kol = kol + 1

End If

Next i

Aver = Sum / kol

Debug.Print

Debug.Print "Sum="; Sum, "kol="; kol, "Aver="; Aver

...

Масив А													
-66	30	98	66	-69	-35	-39	-85	-86	73	-79	35	5	96
Sum	-50												
kol =	5												
Aver	-10												

Immediate

Масив А

-66 30 98 66 -69 -35 -39 -85 -86 73 -79 35 5 96

Sum=-50 kol= 5 Aver=-10

Варіант даних, що призводять до помилки:

Microsoft Visual Basic

Run-time error '6':

Overflow

Continue

End

Debug

Help

Immediate

Масив А

36 49 -99 -37


```

Public Sub Find2a()
...
...
Sum = 0: kol = 0
For i = 1 To n
    If a(i) Mod 5 = 0 Then
        Sum = Sum + a(i)
        kol = kol + 1
    End If
Next i
Debug.Print
Range("A40").Select
If kol <> 0 Then
    Aver = Sum / kol
    Debug.Print "Sum="; Sum, "kol="; kol, "Aver="; Aver
    ActiveCell.Value = "Sum = "
    ActiveCell.Offset(0, 1).Range("A1").Select
    ActiveCell.Value = Sum
    ActiveCell.Offset(1, -1).Range("A1").Select
    ActiveCell.Value = "kol = "
    ActiveCell.Offset(0, 1).Range("A1").Select
    ActiveCell.Value = kol
    ActiveCell.Offset(1, -1).Range("A1").Select
    ActiveCell.Value = "Aver = "
    ActiveCell.Offset(0, 1).Range("A1").Select
    ActiveCell.Value = Aver
Else
    Debug.Print "Відсутні елементи кратні 5"
    ActiveCell.Value = "Відсутні елементи кратні 5"
End If

```

Immediate					Масив А				
Масив А					66	62	36	-32	
66 62 36 -32									
Відсутні елементи кратні 5					Відсутні елементи кратні 5				

Пошук мінімального та максимального елементів масиву є однією з найбільш типових задач. Алгоритми цих задач наведені на рис. 7.2. Слід наголосити, що при розв'язанні цих задач дуже важливим є знаходити не тільки значення мінімального (максимального) елементу, а і його порядковий номер.

Примітка. При необхідності поміняти місцями якісь елементи масиву, необхідно знати, що для обміну між собою значеннями двох змінних (наприклад, *a* і *b*), необхідно використати три оператори привласнення³⁷:

c = a

a = b

b = c

де додаткова змінна (*c*) відіграє роль проміжного буфера.

³⁷ Необхідно пам'ятати, що всі змінні мають бути одного типу.

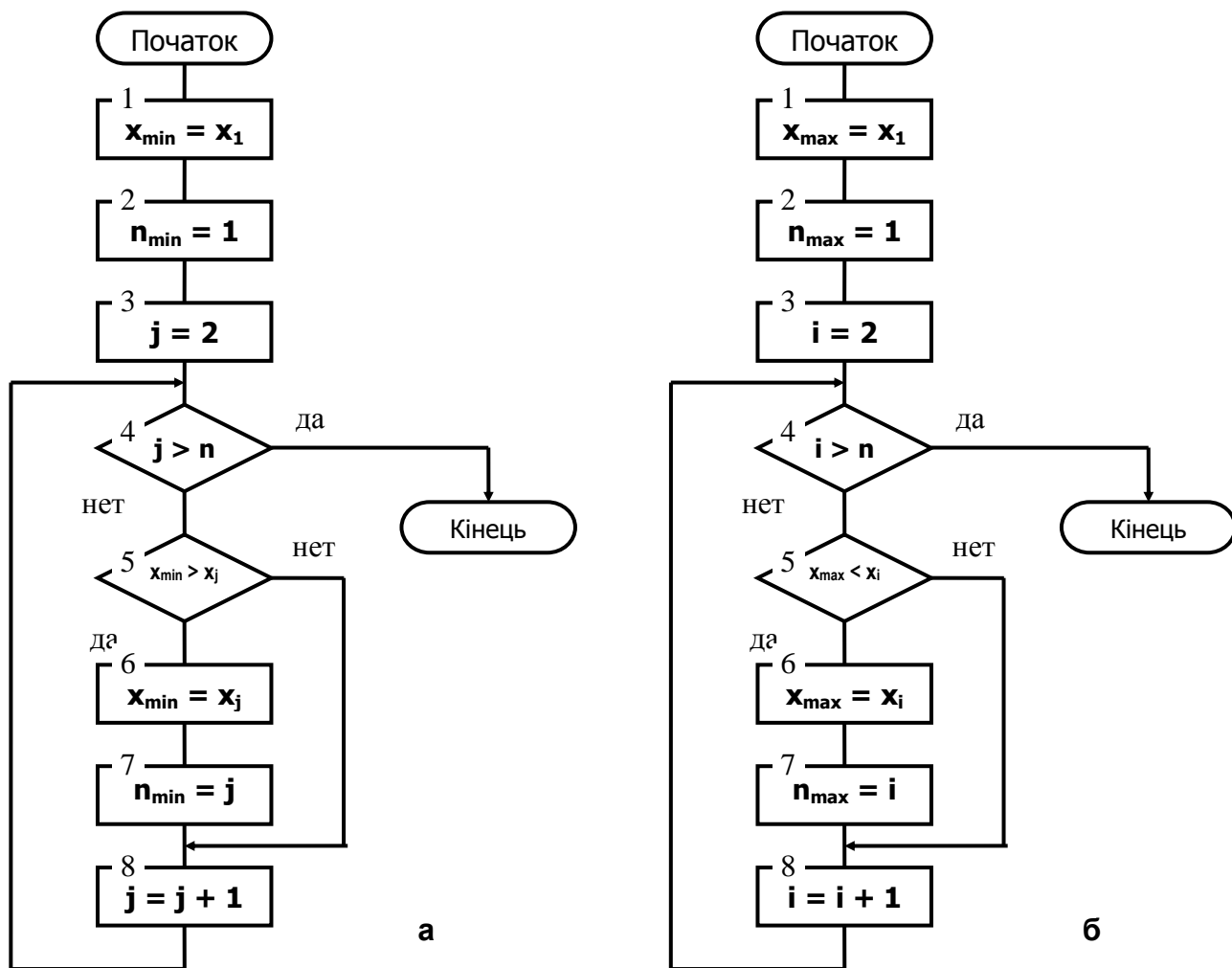


Рис. 7.2. – Алгоритми пошуку мінімального (а) та максимального (б) елементів у масиві

Приклад 4. Задача табулювання функції (на відміну від розглянутої в розділі 4) відрізняється тим, що крім друкування таблиці завдяки використанню масивів всі значення і аргументу, і функції зберігаються в пам'яті і можуть бути використані для подальших розрахунків.

Public Sub Array3()

' Приклад циклічної програми - табулювання функції $y = f(x)$

Dim alfa As Single, beta As Single

Dim xn As Single, xk As Single, dx As Single

Dim sTab As String, sRez As String, i As Integer, n As Integer

Dim SumX As Single, SumY As Single,

Dim x() As Single, y() As Single

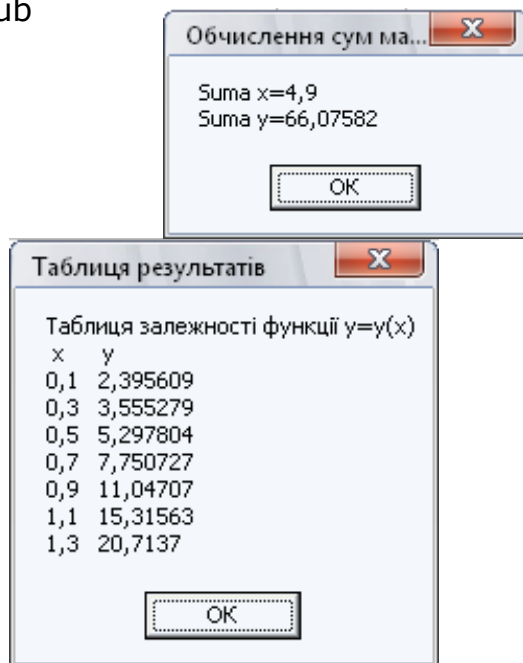
alfa = InputBox("Введіть коефіцієнт alfa", "Введення коефіцієнтів")

beta = InputBox("Введіть коефіцієнт beta", "Введення коефіцієнтів")

```

xn = InputBox("Введіть початкове значення x - xn", "Введення даних")
xk = InputBox("Введіть кінцеве значення x - xk", "Введення даних")
dx = InputBox("Введіть крок змінення x - dx", "Введення даних")
n = (xk - xn) / dx + 1
ReDim x(n) As Single, y(n) As Single
sTab = "Таблиця залежності функції y=y(x)"
sTab = sTab & vbCrLf & "x" & "    y"
Debug.Print "Таблиця залежності функції y=y(x)"
Debug.Print "x", "y"
For i = 1 To n
    x(i) = xn + dx * (i - 1)
    y(i) = alfa * Sin(x(i) ^ 2) + 2 * Exp(beta * x(i))
    Debug.Print x(i), y(i)
    sTab = sTab & vbCrLf & x(i) & "    " & y(i)
Next i
MsgBox sTab, , "Таблиця результатів"
Debug.Print
SumX = 0: SumY = 0
For i = 1 To n
    SumX = SumX + x(i)
    SumY = SumY + y(i)
Next i
Debug.Print "Suma x="; SumX, "Suma y = "; SumY
sRez = "Suma x=" & SumX & vbCrLf & "Suma y=" & SumY
MsgBox sRez, , "Обчислення сум масивів"
End Sub

```



Immediate	
Таблиця залежності функції y=y(x)	
x	y
0,1	2,395609
0,3	3,555279
0,5	5,297804
0,7	7,750727
0,9	11,04707
1,1	15,31563
1,3	20,7137
Suma x= 4,9 Suma y = 66,07582	

Алгоритми вставки в масив додаткового елементу та видалення елементу з масиву наведені на рис.7.3 та 7.4, відповідно. Слід обов'язково пам'ятати, що

задача збільшення кількості елементів в масиві може бути розв'язана лише за умови, що масив описано з запасом (кількість елементів масиву, вказана в операторі **DIM**, не буде перевищено при додаванні нового елементу). Невиконання цієї умови призведе до виникнення помилки: «Індекс вне диапазона».

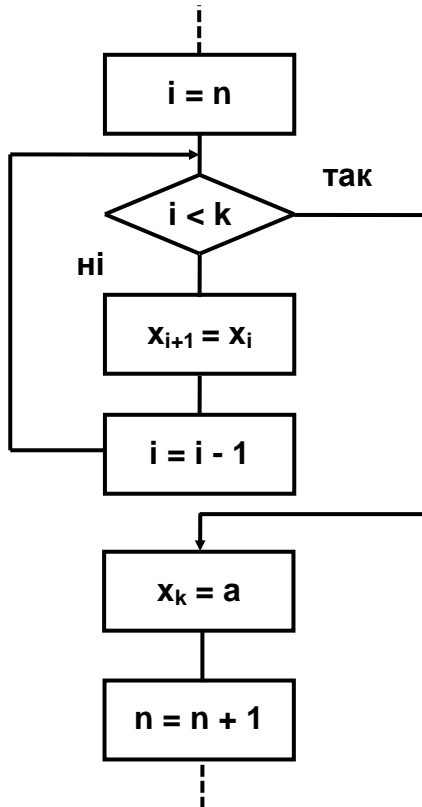


Рис. 7.3 – Алгоритм вставки в масив $x(n)$ додаткового елементу (під номером k)

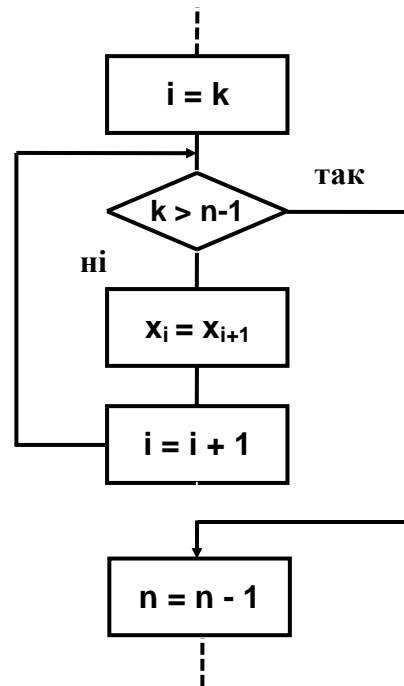
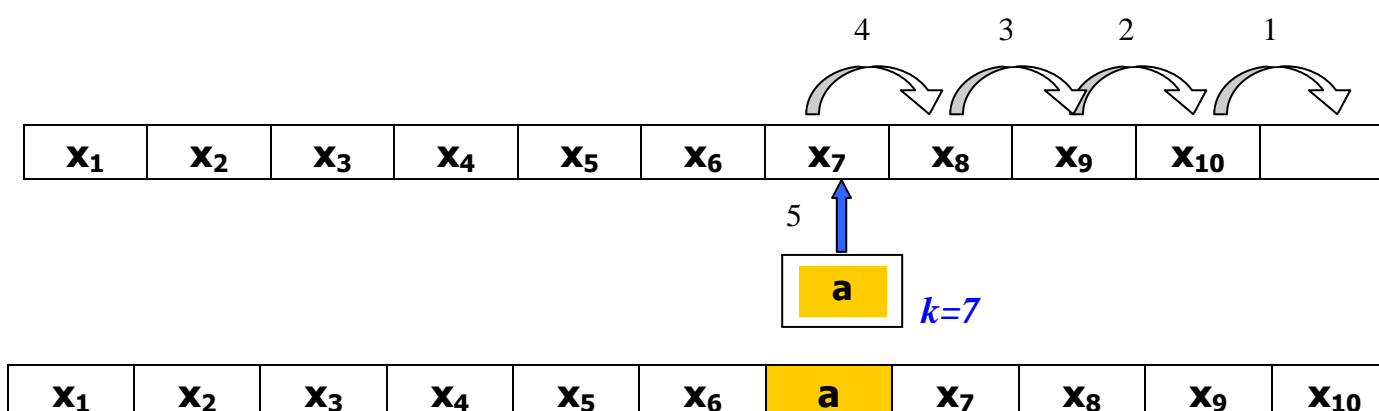


Рис. 7.4– Алгоритм видалення з масиву $x(n)$ k -го елементу

Алгоритм вставки додаткового елементу (під номером k) в масив $X(n)$

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Масив А								30						
2	-73	-55	8	-49	62	95	-60	59	74	-96	29	93	-77	-48	
3	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
4															
5	Масив новий														
6	-73	-55	8	-49	62	95	-60	59	30	74	-96	29	93	-77	-48
7	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15



Public Sub ArrayPlus()

' Приклад вставки додаткового елементу в масив

Dim i As Integer, n As Integer, sArr As String

Dim a() As Single, ANew As Single, iNumb As Integer

n = InputBox("Розмір масиву n")

ReDim a(n) As Single

Randomize Timer

Range("A1").Select

sArr = "Масив:" & vbNewLine

ActiveCell.Value = "Масив A"

Range("A2").Select

For i = 1 To n

a(i) = Int(Rnd * 200) - 100

ActiveCell.Offset(0, i - 1).Value = a(i)

ActiveCell.Offset(1, i - 1).Value = i

sArr = sArr & a(i) & ", "

Next i

MsgBox sArr

ANew = InputBox("Яке значення треба вставити в масив?")

iNumb = InputBox("Під яким номером значення " & ANew & vbNewLine & "слід вставити в масив?")

ActiveCell.Offset(-1, iNumb - 1).Value = ANew

ReDim Preserve a(n + 1) As Single

For i = n To iNumb Step -1

a(i + 1) = a(i)

Next i

a(iNumb) = ANew

Range("A5").Select

ActiveCell.Value = "Масив новий"

sArr = "Масив новий:" & vbNewLine

Range("A6").Select

For i = 1 To n + 1

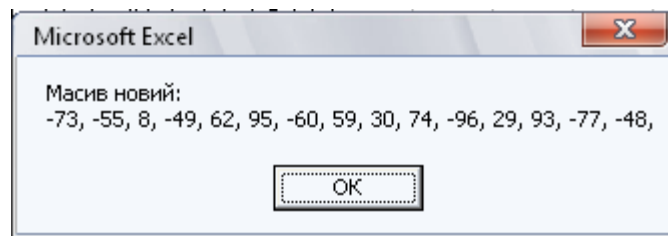
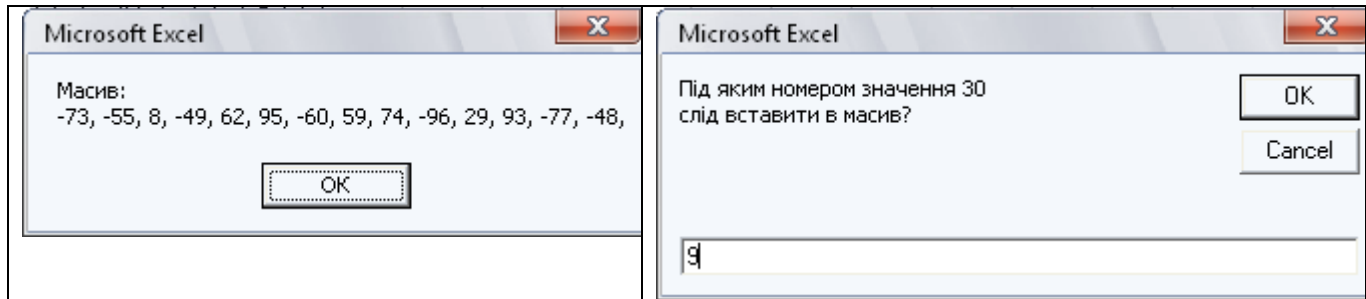
ActiveCell.Offset(0, i - 1).Value = a(i)

ActiveCell.Offset(1, i - 1).Value = i

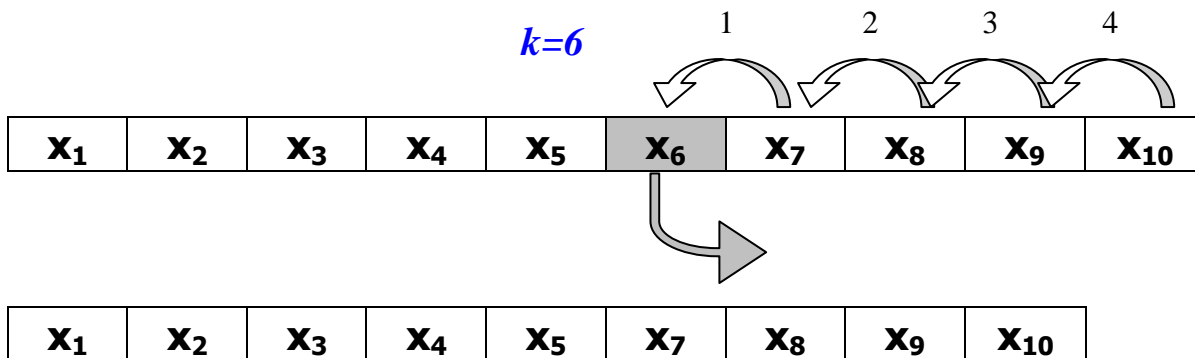
```

        sArr = sArr & a(i) & ", "
    Next i
    MsgBox sArr
End Sub

```



Алгоритм видалення k-го елементу з масиву X(n)



Public Sub ArrayMinus()

```

' Приклад Приклад видалення заданого елементу з масиву
Dim i As Integer, n As Integer, sArr As String
Dim a() As Single, iNumb As Integer
n = InputBox("Розмір масиву n")
ReDim a(n) As Single
Randomize Timer
Range("A1").Select
sArr = "Масив:" & vbCrLf
ActiveCell.Value = "Масив A"
Range("A2").Select
For i = 1 To n
    a(i) = Int(Rnd * 200) - 100
    ActiveCell.Offset(0, i - 1).Value = a(i)

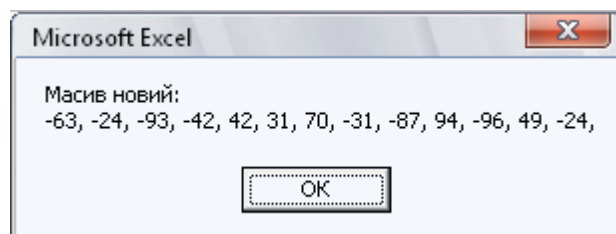
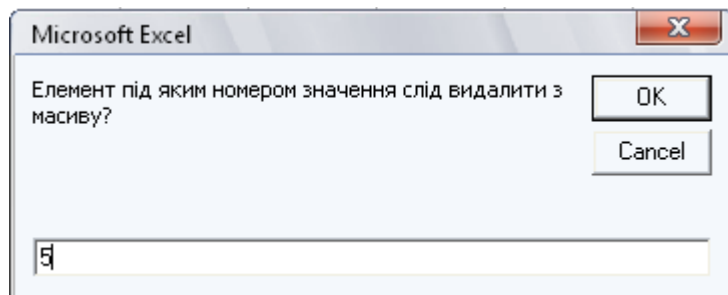
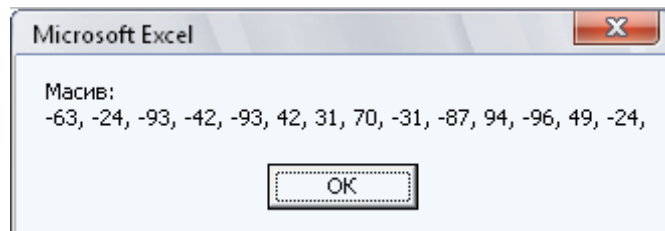
```

```

ActiveCell.Offset(1, i - 1).Value = i
    sArr = sArr & a(i) & ", "
Next i
    MsgBox sArr
iNumb = InputBox("Елемент під яким номером значення слід видалити з
                                     масиву?")

For i = iNumb To n - 1
    a(i) = a(i + 1)
Next i
'ReDim Preserve a(n - 1) As Single
Range("A5").Select
ActiveCell.Value = "Масив новий"
    sArr = "Масив новий:" & vbNewLine
Range("A6").Select
For i = 1 To n - 1
    ActiveCell.Offset(0, i - 1).Value = a(i)
    ActiveCell.Offset(1, i - 1).Value = i
    sArr = sArr & a(i) & ", "
Next i
    MsgBox sArr
End Sub

```



7.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи вивчити та описати в протоколі лабораторної роботи такі поняття (визначення) і операції (команди):

- поняття масиву, прийоми роботи з масивами (операції опису, введення та виведення масивів, звернення до окремих елементів масиву при виконанні з ними різних операцій);
- поняття датчика випадкових чисел і можливості його використання для задання значень довільного масиву
- типові алгоритми:
 - обчислення суми або добутку елементів масиву;
 - пошук найменшого і найбільшого елементів масиву;

- пошук елементів масиву, що задовольняють заданим умовам;
- розширення масиву шляхом додавання в нього нового елементу із заданим номером;
- скорочення масиву за рахунок видалення з нього елементу із заданим номером.

Написати програми, які реалізують алгоритми розширення масиву шляхом додавання в нього нового елементу із заданим номером та скорочення масиву за рахунок видалення з нього елементу із заданим номером (рис 7.3. та 7.4). Дослідити можливості програмного керування процесом форматування комірок при виводі результатів на лист MS Excel на прикладі розглянутих програм.

7.3 Опис лабораторних засобів та обладнання

Лабораторна робота №7 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

7.4 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

7.5 Вказівки до виконання лабораторної роботи №7

1. Розробити алгоритм для обробки одновимірною масиву у відповідності до свого варіанту завдання.
2. Реалізувати розроблений алгоритм у вигляді процедури в середовищі VBA. Програма повинна:
 - створювати масив довільного розміру за допомогою датчика випадкових чисел;
 - записувати на лист MS Excel створений масив та результати розрахунків. В разі, коли в масиві відбувається перестановка елементів, змінений масив слід також надрукувати;
 - в залежності від бажання користувача³⁸ виводити у вікно функції **MsgBox** ті ж самі результати, що і на лист MS Excel.
 - виводити ту ж саму інформацію у вікно **Immediate**.
3. Застосувати форматування до комірок листа MS Excel, в які виводяться результати.
4. Перевірити та налаштувати роботу програми на кількох прикладах. Масив повинен складатися з 12 – 20 довільних³⁹ цілих чисел.

³⁸ Використати функцію **MsgBox** з кнопками «Да» та «Нет» або «Yes», «No» та «Cancel».

5. Продемонструвати результати викладачу.
6. Оформити протокол лабораторної роботи, до складу якого треба включити теоретичні відомості, результати виконання самостійної роботи, опис та результати виконання індивідуального завдання (блок-схему алгоритму, текст програми, роздруківки листа (листів) з MS Excel), скріншоти вікон *MsgBox* та *Immediate*).

7.6 Обробка та аналіз результатів. Оформлення звіту

Протокол лабораторної роботи оформлюється згідно умов наведених вище (див. лабораторні роботи №1 та №2). Необхідно проаналізувати різні способи введення та виведення масивів та характерні прийоми програмування при використанні масивів.

7.7 Контрольні запитання

1. Що таке масив?
2. Як виконується опис масиву? Що є результатом цього опису?
3. Чи завжди треба оголошувати масив?
4. Як здійснюється введення і виведення масивів?
5. Як називається номер елемента одновимірного масиву?
6. Для чого служить оператор DIM?
7. Яким оператором встановлюється нижня межа індексу масиву, що приймається за замовчуванням?
8. Чому межі індексів масиву не можуть виходити за межі діапазону типу INTEGER?

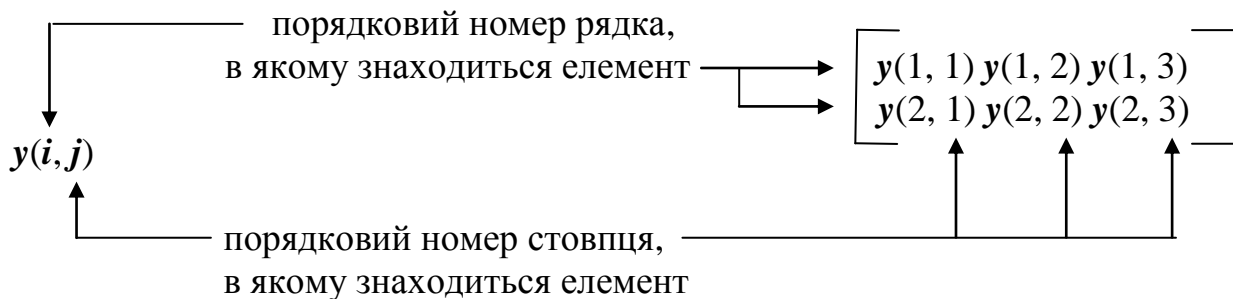
³⁹ Якщо в завданні вказано на використання доданих або від'ємних значень, масив обов'язково повинен включати як ті так і інші.

8. Виконання операцій з двовимірними масивами

Мета та основні завдання роботи: дослідити особливості використання індексованих змінних (багатовимірних масивів) при створенні програмних засобів у мові програмування Visual Basic.

8.1 Основні теоретичні відомості

Використання двовимірних та багатовимірних масивів відбувається так само, як і одновимірних, але необхідно пам'ятати, що для звернення до елемента такого масиву необхідно вказувати вже не один, а два (або більше) індексів, які визначають положення елемента в масиві:



Кількість індексів для багатовимірних масивів в Visual Basic не перевищує 60, кількість елементів в масиві не більше 32768.

Опис масиву:

Dim a(<кількість рядків>, <кількість стовпців>) [As <тип>]

Dim a([<початок> To] <кінець>, [<початок> To] <кінець>) [As <тип>]

Приклади:

Dim a(5, 7) As Single, k(10, 12) As Integer

Dim x(10 To 50, 0 To 4) As Single

Dim x(10 To 50, 0 To 4) As Single

Варіанти введення та виведення масиву

Public Sub Matr1()

Dim i As Integer, j As Integer, iRows As Integer, iCols As Integer,

Dim a() As Single, sArr As String

iRows = InputBox("кількість рядків")

iCols = InputBox("кількість стовпців")

ReDim a(iRows, iCols) As Single

sArr = "Масив:" & vbCrLf

Range("A1").Select

ActiveCell.Value = "Масив A"

Range("A2").Select

For i = 1 To iRows

For j = 1 To iCols

a(i, j) = Int(Rnd * 100) ' [= Int(Rnd * 200) - 100]

```

ActiveCell.Value = a(i, j)
ActiveCell.Offset(0, 1).Range("A1").Select
sArr = sArr & a(i, j) & ", "

```

Next j

```

ActiveCell.Offset(1, - iCols).Range("A1").Select
sArr = sArr & vbNewLine

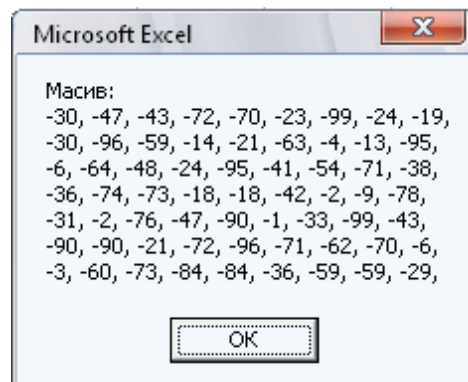
```

Next i

MsgBox sArr

End Sub

	A	B	C	D	E	F	G	H	I
1	Масив А								
2	-30	-47	-43	-72	-70	-23	-99	-24	-19
3	-30	-96	-59	-14	-21	-63	-4	-13	-95
4	-6	-64	-48	-24	-95	-41	-54	-71	-38
5	-36	-74	-73	-18	-18	-42	-2	-9	-78
6	-31	-2	-76	-47	-90	-1	-33	-99	-43
7	-90	-90	-21	-72	-96	-71	-62	-70	-6
8	-3	-60	-73	-84	-84	-36	-59	-59	-29



Public Sub Matr1A()

' Варіант 2

```

Dim i As Integer, j As Integer
Dim iRows As Integer, iCols As Integer, a() As Single

```

iRows = InputBox("кількість рядків")

iCols = InputBox("кількість стовпців")

ReDim a(iRows, iCols) As Single

Range("A1").Select

ActiveCell.Value = "Масив А"

Range("A2").Select

Debug.Print "Масив А"

For i = 1 To iRows

For j = 1 To iCols

a(i, j) = Int(Rnd * 200) - 100

ActiveCell.Offset(0, j - 1).Value = a(i, j)

Debug.Print a(i, j);

Next j

Debug.Print

ActiveCell.Offset(1, 0).Range("A1").Select

Next i

End Sub

Immediate							
Масив А							
-35	26	-59	-63	16	-84	-9	81
-48	57	-25	-43	83	26	25	-15
-81	12	38	82	66	-96	8	83
-14	35	0	2	-8	-30	-20	-47
-89	-52	95	-88	-22	-28	-3	-69
-6	-49	25	8	-69	87	30	1

	A	B	C	D	E	F	G	H
1	Масив А							
2	-35	26	-59	-63	16	-84	-9	81
3	-48	57	-25	-43	83	26	25	-15
4	-81	12	38	82	66	-96	8	83
5	-14	35	0	2	-8	-30	-20	-47
6	-89	-52	95	-88	-22	-28	-3	-69
7	-6	-49	25	8	-69	87	30	1
8								

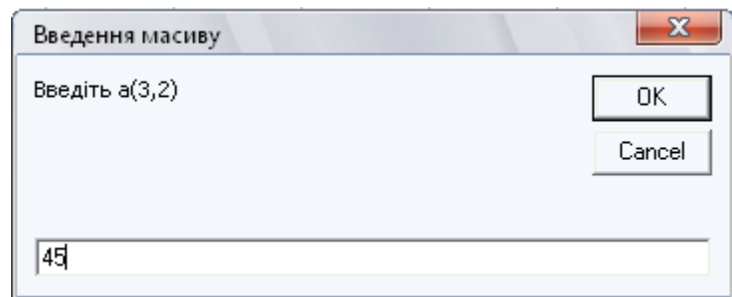
Public Sub Matr1B()

' Варіант 3

```
Dim i As Integer, j As Integer
Dim iRows As Integer, iCols As Integer, a() As Single
iRows = InputBox("кількість рядків")
iCols = InputBox("кількість стовпців")
ReDim a(iRows, iCols) As Single
Debug.Print "Масив А"
For i = 1 To iRows
    For j = 1 To iCols
        a(i, j) = InputBox("Введіть a(" & i & "," & j & ")",
            "Введення масиву")
        Debug.Print Tab(5 * (j - 1)); a(i, j);
    Next j
    Debug.Print
Next i
End Sub
```

Immediate

Масив А					
-22	-79	56	-9	50	19
66	-97	-58	-86	-79	-34
-75	-100	7	31	8	65
-84	-62	35	-10	-29	-71
40	85	6	-83	51	-20
-8	-2	-59	-35	-81	17
-67	85	-81	-12	-46	74



Обробка елементів масиву на прикладі обчислення їх суми

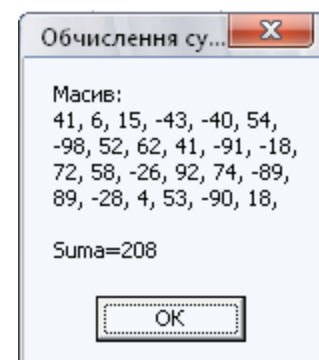
```
...
Sum = 0
For i = 1 To iRows
    For j = 1 To iCols
        Sum = Sum + a(i, j)
    Next j
Next i
Debug.Print "Suma="; Sum
MsgBox sArr & vbNewLine & "Suma=" & Sum, ,
    "Обчислення суми"
```

...

Immediate

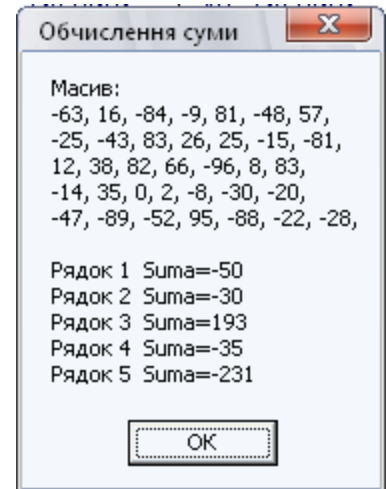
Масив А					
41	6	15	-43	-40	54
-98	52	62	41	-91	-18
72	58	-26	92	74	-89
89	-28	4	53	-90	18

Suma= 208



Обробка елементів в окремих рядках матриці
(на прикладі обчислення сум)

```
...  
For i = 1 To iRows  
    Sum = 0  
    For j = 1 To iCols  
        Sum = Sum + a(i, j)  
    Next j  
    Debug.Print "Рядок "; i; "Suma="; Sum  
    sArr = sArr & vbNewLine & "Рядок " & i & " Suma=" & Sum  
Next i  
MsgBox sArr, , "Обчислення суми"  
...
```



```
Immediate  
Масив A  
-63 16 -84 -9 81 -48 57  
-25 -43 83 26 25 -15 -81  
12 38 82 66 -96 8 83  
-14 35 0 2 -8 -30 -20  
-47 -89 -52 95 -88 -22 -28  
Рядок 1 Suma=-50  
Рядок 2 Suma=-30  
Рядок 3 Suma= 193  
Рядок 4 Suma=-35  
Рядок 5 Suma=-231
```

Обробка елементів в окремих стовпцях матриці
(на прикладі обчислення сум)

```
...  
For j = 1 To iCols  
    Sum = 0  
    For i = 1 To iRows  
        Sum = Sum + a(i, j)  
    Next i  
    Debug.Print "Стовпець "; j; "Suma="; Sum  
    sArr = sArr & vbNewLine & "Стовпець " & j & " Suma=" & Sum  
Next j  
MsgBox sArr, , "Обчислення суми"  
...
```

Immediate						
Масив А						
41	6	15	-43	-40	54	-98
52	62	41	-91	-18	72	58
-26	92	74	-89	89	-28	4
53	-90	18	-7	-41	24	29
-48	-45	65	64	17	97	82
-55	39	96	-52	6	-79	99
Стовпець	1	Suma= 17				
Стовпець	2	Suma= 64				
Стовпець	3	Suma= 309				
Стовпець	4	Suma=-218				
Стовпець	5	Suma= 13				
Стовпець	6	Suma= 140				
Стовпець	7	Suma= 174				

Обчислення суми	
Масив:	
41, 6, 15, -43, -40, 54, -98,	
52, 62, 41, -91, -18, 72, 58,	
-26, 92, 74, -89, 89, -28, 4,	
53, -90, 18, -7, -41, 24, 29,	
-48, -45, 65, 64, 17, 97, 82,	
-55, 39, 96, -52, 6, -79, 99,	
Стовпець 1	Suma=17
Стовпець 2	Suma=64
Стовпець 3	Suma=309
Стовпець 4	Suma=-218
Стовпець 5	Suma=13
Стовпець 6	Suma=140
Стовпець 7	Suma=174
OK	

8.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи вивчити та описати в протоколі лабораторної роботи такі поняття (визначення) і операції (команди):

- особливості опису двовимірних масивів і масивів більших розмірностей
- особливості введення та виведення масивів на лист MS Excel, у вікно функції *MsgBox* або у вікно *Immediate*;
- особливості організації звернення до елементів таких масивів при виконанні з ними різних операцій (в тому числі, – операцій з елементами масиву, що знаходяться в окремих рядках або стовпцях).

8.3 Опис лабораторних засобів та обладнання

Лабораторна робота №8 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

8.4 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

8.5 Вказівки до виконання практичної роботи

У рамках практичної роботи вивчити приклади алгоритмів та програм, які наведені в конспекті лекцій. Перевірити роботу алгоритмів, які вивчаються, на практичному занятті, та проаналізувати їх роботу. Оформити звіт з практичної роботи.

8.6 Вказівки до виконання лабораторної роботи №8

Завдання №1

1. Розробити алгоритм для обробки двовимірного масиву згідно своєму варіанту завдання.
2. Реалізувати розроблений алгоритм у вигляді процедури в середовищі VBA. Програма повинна:
 - створювати масив довільного розміру за допомогою датчика випадкових чисел або зчитувати його з листа MS Excel (в залежності від бажання користувача⁴⁰);
 - записувати на лист MS Excel створений масив та результати розрахунків. Передбачити запис результатів, які отримані для певного рядка або стовпця, у комірки, що знаходяться у відповідному рядку або стовпці;
 - застосовувати форматування до комірок листа MS Excel, в які виводяться результати;
 - в залежності від бажання користувача² виводити у вікно функції **MsgBox** ті ж самі результати, що і на лист MS Excel.
 - виводити ту ж саму інформацію у вікно **Immediate**.
3. Перевірити та налаштувати роботу програми на кількох прикладах.
4. Продемонструвати результати викладачу.

Завдання №2

5. Розробити алгоритм для обробки двовимірного масиву згідно своєму варіанту завдання.
6. Реалізувати розроблений алгоритм у вигляді процедури в середовищі VBA. Програма повинна:
 - створювати масив довільного розміру за допомогою датчика випадкових чисел або зчитувати його з листа MS Excel (в залежності від бажання користувача²);
 - записувати на лист MS Excel створений масив та результати розрахунків. Передбачити запис результатів, які отримані для певного рядка або стовпця, у комірки, що знаходяться у відповідному рядку або стовпці;
 - застосовувати форматування до комірок листа MS Excel, в які виводяться результати;
 - в залежності від бажання користувача² виводити у вікно функції **MsgBox** ті ж самі результати, що і на лист MS Excel.
 - виводити ту ж саму інформацію у вікно **Immediate**;
 - записувати у дисковий файл створений масив із відповідним заголовком та результати його обробки (передбачити можливість запису кількох варіантів розрахунку).
7. Перевірити та налаштувати роботу програми на кількох прикладах.

⁴⁰ Використати функцію **MsgBox** з кнопками «Да» та «Нет» або «Yes», «No» та «Cancel».

8. Продемонструвати результати викладачу.
9. Оформити протокол лабораторної роботи, до складу якого треба включити теоретичні відомості, результати виконання самостійної роботи, опис та результати виконання індивідуальних завдань (№1 та №2) – блок-схеми алгоритмів, тексти програм, роздруківки листів з MS Excel, скріншоти вікон *MsgBox* та *Immediate*, роздруківку файлів результатів.

8.7 Обробка та аналіз результатів. Оформлення звіту

Протокол лабораторної роботи оформлюється згідно умов наведених вище (див. лабораторні роботи №1 та №2). Необхідно проаналізувати різні способи введення та виведення двовимірних масивів та характерні прийоми програмування при їх використанні.

8.8 Контрольні запитання

1. Для чого в програмах використовуються двовимірні масиви?
2. Як оголошуються двовимірні масиви?
3. Скільки індексів характеризують конкретний елемент двовимірного масиву?
4. Як здійснюється введення і виведення двовимірних масивів?
5. Який з індексів елементу двовимірного масиву відповідає за номер рядка і який за номер стовпця?
6. Як знайти добутки елементів в кожному з парних стовпців двовимірного масиву?
7. Які функції повертають верхню або нижню межу (найменший або найбільший доступний індекс) для вказаного розміру масиву?

9. Виділення окремих областей двовимірних масивів

Мета та основні завдання роботи: дослідити специфічні особливості взаємодії операторів організації арифметичних циклів та операторів організації розгалуження на прикладі задач виділення окремих областей двовимірних масивів при створенні програмних засобів у мові програмування Visual Basic.

9.1 Основні теоретичні відомості

Якщо в масиві розміром $m \times n$ елементи, що підлягають обробці, знаходяться в області (рис. 9.1), яка обмежена певними рядками (m_1 та m_2) і стовпцями (n_1 та n_2), то для їх виділення застосовується така пара циклів:

		n_1				n_2	n	
m_1								
m_2								
m								

Рисунок 9.1 – елементи масиву, що підлягають обробці

```
For i = m1 To m2
    For j = n1 To n2
        <обробка a(i, j)>
    Next j
Next i
```

Примітка. Під *обробкою* розуміють будь-яку операцію з вказаним елементом. Наприклад:

$s = s + a(i, j)$

або:

Debug.Print a(i, j);

В квадратних матрицях – $A(n, n)$ межами області для виділення певних елементів можуть бути діагоналі. Слід пам'ятати, що координати елементів, які знаходяться на головній діагоналі, завжди однакові: $i = j$ (де i – номер рядка, $i = 1, 2, \dots, n$; j – номер стовпця, $j = 1, 2, \dots, n$). А координати елементів, які знаходяться на побічній діагоналі, пов'язані умовами $j = n - i + 1$ (або $i = n - j + 1$).

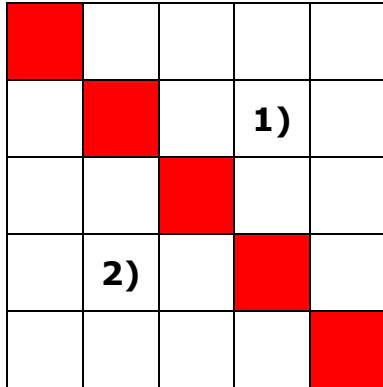
Отже для виділення елементів матриці, які знаходяться над головною діагоналлю, або під нею (рис. 9.2а), необхідно застосовувати такі пари циклів:

```
1) For i = 1 To n
    For j = i To n
        <обробка a(i, j)>
    Next j
Next i
```

```
2) For i = 1 To n
    For j = 1 To i
        <обробка a(i, j)>
    Next j
Next i
```

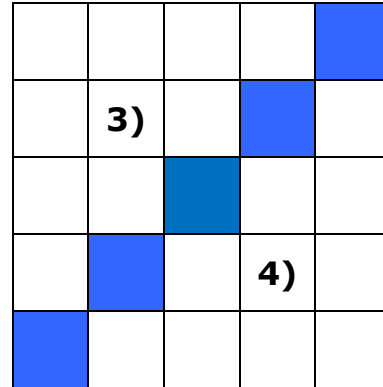
А якщо йдеться про виділення елементів матриці, які знаходяться над побічною діагоналлю, або під нею (рис. 9.2б), пари циклів мають такий вигляд:

3) **For i = 1 To n**
 For j = 1 To n - i + 1
 <обробка a(i, j)>
 Next j
 Next i



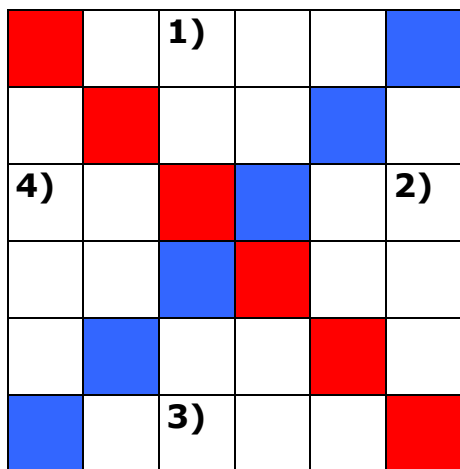
a

4) **For i = 1 To n**
 For j = n - i + 1 To n
 <обробка a(i, j)>
 Next j
 Next i

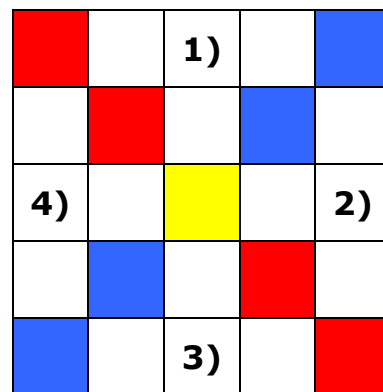


б

Рисунок 9.2 – виділення елементів масиву відносно головної діагоналі (а) та побічної (б)



a



б

Рисунок 9.3 – виділення областей масиву за допомогою обох діагоналей (де масив має парну (а) та непарну (б) кількість рядків і стовпців)

Цикли для виділення областей масиву, які знаходяться між обох діагоналей (рис. 9.3), мають такий вигляд:

1) **For i = 1 To n/2**
 For j = i To n - i + 1
 <обробка a(i, j)>
 Next j
 Next i

3) **For i = n/2 To n**
 For j = n - i + 1 To i
 <обробка a(i, j)>
 Next j
 Next i

```

2)  For j = n/2 To n
      For i = n - i + 1 To j
        <обробка a(i, j)>
      Next i
    Next j

```

```

4)  For j = 1 To n/2
      For i = j To n - i + 1
        <обробка a(i, j)>
      Next i
    Next j

```

Таким чином, – при виділенні областей «1» і «3» (рис. 9.3) зовнішнім буде цикл «по рядках» (від початку до середини або від середини до кінця), а внутрішнім – цикл «по стовпцях» (від однієї діагоналі до іншої). Аналогічно, для областей «2» і «4» (рис. 9.3) зовнішнім буде цикл «по стовпцях», а внутрішнім – «по рядках». Приклади алгоритмів для виділенні областей «1» і «2» наведені на рис. 9.4.

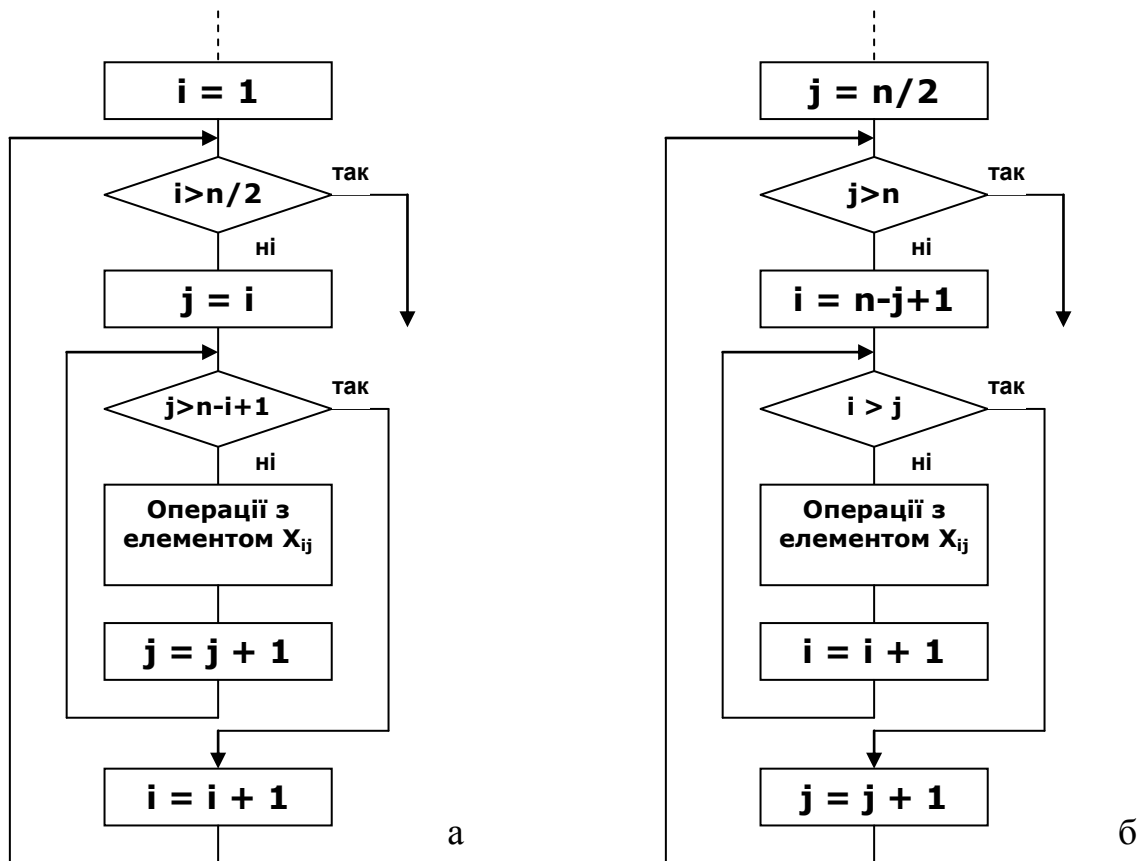


Рисунок 9.4 – Алгоритми перебору елементів X_{ij} квадратної матриці $X(n, n)$, розташованих в областях «1» (а) і «2» (б)

Необхідно також враховувати, що є різниця у визначенні середини матриці ($n/2$ – координати центру матриці). Вона відрізняється для матриці парного і непарного розміру – n (кількості елементів в рядку / стовпці матриці). Слід пам'ятати, що координати елементів (номери рядків / стовпців) завжди є цілими числами. Отже, коли розмір матриці непарний, це значення не дорівнює $n/2$. Наприклад, коли $n = 5$, середнім є рядок / стовець з номером 3 (а не 2,5!).

Примітка. Способи обчислення значення $n/2$ у наведених вище циклах, розібрати самостійно.

9.2 Вказівки до самостійної роботи

Самостійна робота є необхідним елементом підготовки до практичної і лабораторної робіт в комп'ютерному класі. Під час підготовки до виконання роботи вивчити та описати в протоколі лабораторної роботи особливості виділення різних областей двовимірних масивів для виконання типових операцій з елементами, що в них знаходяться. Вивчити особливості виведення окремих областей масивів на лист MS Excel, у вікно функції *MsgBox* та у вікно *Immediate*;

9.3 Опис лабораторних засобів та обладнання

Лабораторна робота №9 виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows.

9.4 Заходи безпеки під час виконання лабораторної роботи

Заходи безпеки, яких треба дотримуватись при виконанні даної лабораторної роботи, наведені у додатку А.

9.5 Вказівки до виконання практичної роботи

У рамках практичної роботи вивчити приклади алгоритмів і програм, наведених в конспекті лекцій та перевірити їх роботу. Проаналізувати особливості виділення різних областей двовимірних масивів та їх друку. Оформити звіт з практичної роботи.

9.6 Вказівки до виконання лабораторної роботи №9

Завдання № 1

1. Розробити алгоритм для обробки елементів двовимірного масиву у вказаній області відповідно до свого варіанту завдання.
2. Реалізувати розроблений алгоритм у вигляді процедури в середовищі VBA. Програма повинна:
 - створювати масив довільного розміру за допомогою датчика випадкових чисел або зчитувати його з листа MS Excel (в залежності від бажання користувача⁴¹⁾⁴²;
 - записувати на лист MS Excel створений масив та результати розрахунків;
 - застосовувати форматування до комірок листа MS Excel, в які виводяться результати⁴³;

⁴¹ Використати функцію *MsgBox* з кнопками «Да» та «Нет» або «Yes», «No» та «Cancel».

⁴² В завданнях №2 та №3 необхідно застосувати різні способи введення значень матриці.

- в залежності від бажання користувача¹ виводити у вікно функції **MsgBox** ті ж самі результати, що і на лист MS Excel.
 - виводити ту ж саму інформацію у вікно **Immediate**.
 - записувати інформацію в файл.
3. Перевірити та налаштувати роботу програми на кількох матрицях різного розміру.
 4. Продемонструвати результати викладачу.

Завдання № 2

5. Розробити алгоритм для обробки елементів двовимірного масиву у вказаній області відповідно до свого варіанту завдання.
6. Реалізувати розроблений алгоритм у вигляді процедури в середовищі VBA. Програма повинна:
 - створювати масив довільного розміру за допомогою датчика випадкових чисел або зчитувати його з листа MS Excel (в залежності від бажання користувача¹)²;
 - записувати на лист MS Excel створений масив та результати розрахунків;
 - застосовувати форматування до комірок листа MS Excel, в які виводяться результати³;
 - в залежності від бажання користувача¹ виводити у вікно функції **MsgBox** ті ж самі результати, що і на лист MS Excel.
 - виводити ту ж саму інформацію у вікно **Immediate**.
 - записувати інформацію в файл.
7. Перевірити та налаштувати роботу програми на кількох матрицях різного розміру.
8. Продемонструвати результати викладачу.
9. Оформити протокол лабораторної роботи, до складу якого треба включити теоретичні відомості, результати виконання самостійної роботи, опис та результати виконання індивідуальних завдань (№1 та №2) – блок-схеми алгоритмів, тексти програм, роздруківки листів з MS Excel, скріншоти вікон **MsgBox** та **Immediate**, роздруківку файлів результатів.

9.7 Обробка та аналіз результатів. Оформлення звіту

Протокол лабораторної роботи оформлюється згідно умов наведених вище (див. лабораторні роботи №1 та №2). Необхідно проаналізувати способи виділення різних областей двовимірних масивів та їх друку. Набути досвіду використання типових алгоритмів при обробці елементів масивів.

⁴³ Передбачити виділення потрібної області та комірок, які відповідають умовам задачі.

9.8 Контрольні запитання

1. Які умови відповідають елементам матриці, що знаходяться, відповідно, на головній або на побічній діагоналі?
2. Запишіть умову знаходження елементів матриці під головною діагоналлю.
3. Запишіть умову знаходження елементів матриці над головною діагоналлю.
4. Запишіть умову знаходження елементів матриці під побічною діагоналлю.
5. Запишіть умову знаходження елементів матриці над побічною діагоналлю.
6. Запишіть фрагмент програми знаходження елементів розташованих у верхній чверті матриці матриці з урахуванням граничних елементів та без них.
7. Запишіть фрагмент програми знаходження елементів розташованих в нижній чверті матриці матриці з урахуванням граничних елементів та без них.
8. Запишіть фрагмент програми знаходження елементів розташованих в правій чверті матриці матриці з урахуванням граничних елементів та без них.
9. Запишіть фрагмент програми знаходження елементів розташованих в лівій чверті матриці матриці з урахуванням граничних елементів та без них.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Основна література

1. Кашаев С.М. Офисные решения с использование MS Excel 2007 и VBA. [Текст] / СПб.: Питер, 2009. – 352 с.
2. Уокенбах Дж. Профессиональное программирование на VBA в Excel 2002. [Текст] / М.: Изд. дом "Вильямс", 2003. – 764 с.

Додаткова література

3. Эйткен П. Разработка приложений на VBA в среде office XP. [Текст] / – М: "Вильямс", 2003. – 496 с.
4. Эйткен П. Интенсивный курс программирования в Excel за выходные. [Текст] / М.: Изд. дом «Вильямс», 2006. 432 стр.
5. Сергеев А.П. Использование MS Office Excel 2007. - М.: ООО "Изд. Дом Вильямс", 2007. 288 с.
6. Кузьменко В.Г. VBA 2000. Самоучитель. [Текст] / – М.: Бином, 2000. – 408 с.
7. Гарнаев А. Самоучитель VBA. Технология создания пользовательских приложений. [Текст] / – СПб.: BHV, 1999. 512 с.
8. Глушаков С.В., Сурядный А.С., Мачула О.В. MS Excel XP для профессионала. - Харьков, Фолио, 2007. 319 с.
9. Коттингхэм М. Excel 2000. Руководство разработчика. [Текст] / – К.: BHV, "Ирина", 2000. – 704 с.
10. Каммингс С. VBA для чайников. [Текст] / 2-е изд. – М.: Вильямс, 2000. – 384 с.
11. Биллиг В.А. VBA в Office 2000. Офисное программирование. [Текст] / – М.: Издательско-торговый дом «Русская Редакция», 1999. — 480 с.: ил.
12. Биллиг В.А. VBA и Office 97. Офисное программирование. [Текст] / В.А. Биллиг, М.И. Дехтярь – М.: «Русская Редакция» ТОО «Channel Trading Ltd.», 1998. – 720 с.
13. Харрис М. Освой самостоятельно программирование MS Excel 2000 за 21 день. – М.: Вильямс (SAMS), 2000. – 880 с.
14. Кергаль И. Методы программирования на Бейсике (с упражнениями). – М.: Мир, 1991. – 288 с.
15. Безносик Ю.А. Программирование на языке Qbasic для персональной ЭВМ [Текст] / Ю.А. Безносик, С.Г. Бондаренко, А.А. Квитка. – К.: ИСИО, 1996. – 204 с.
16. Бобровский С.И.. Программирование на языке Qbasic для школьников и студентов [Текст] / М.: ДЕСС КОМ, 2000. – 208 с.

ЗАХОДИ БЕЗПЕКИ ПІД ЧАС ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

Цикл лабораторних робіт виконуються в комп'ютерних класах кафедри кібернетики хіміко-технологічних процесів хіміко-технологічного факультету та КБ ІС. Обладнання живиться електричним струмом напругою 220 В. Тому при виконанні лабораторних робіт слід дотримуватися заходів безпеки наступних інструкцій.

ІНСТРУКЦІЯ

з техніки безпеки при навчанні студентів на ПЕОМ в учбових лабораторіях кафедри кібернетики хіміко-технологічних процесів інженерно-хімічного факультету

Знання і суворе дотримання цих правил є обов'язковим для всіх осіб, допущених до роботи на ПЕОМ. Доведення їх до кожного зі студентів підтверджується особистим підписом кожного з них у контрольному листі з техніки безпеки. Особи, які не одержали такого інструктажу та не поставили підпис у контрольному листі з техніки безпеки, до роботи на ПЕОМ не допускаються.

Всі роботи в учбових лабораторіях кафедри кібернетики ХТП проводяться лише з дозволу викладача або співробітника кафедри.

Під час проведення занять в учбовій лабораторії не повинні знаходитися сторонні особи, в тому числі студенти інших груп. Студенти не повинні самовільно залишати учбову лабораторію під час занять.

При роботі на ПЕОМ треба пам'ятати, що в них використовується напруга, небезпечна для життя.

Всі особи, працюючі в учбових лабораторіях кафедри КХТП повинні бути ознайомлені з правилами надання першої медичної допомоги при ураженні електричним струмом.

Перед вмиканням ПЕОМ кожен з працюючих повинен отримати дозвіл викладача або співробітника кафедри.

У випадках виникнення короткого замикання, горіння, диму, вогню в апаратурі, пристрій необхідно негайно вимкнути з мережі та доповісти викладачеві або співробітникові кафедри. Самостійні дії по усуненню пошкодження забороняються.

У випадку виходу з ладу обладнання або програмного забезпечення, що зумовлені іншими причинами, доповісти викладачеві або співробітникові кафедри. Вимикати апаратуру при цьому не дозволяється. Самостійні дії по усуненню пошкодження забороняються.

Працюючі в учбових лабораторіях кафедри кібернетики ХТП несуть майнову та адміністративну відповідальність за збереження та використання обладнання, наданого для їх праці.

Категорично забороняється:

- самостійно вмикати та вимикати тумблери на щитку електроживлення;
- несанкціоновано вмикати електрообладнання;
- приносити та вмикати своє обладнання та пристрої, встановлювати власне програмне забезпечення;
- залишати без нагляду увімкнені пристрої та лабораторію;
- пересувати обладнання та комплектуючі;
- підключати та відключати інформаційні кабелі та кабелі живлення;
- використовувати власні носії інформації без дозволу викладачів або співробітників кафедри;
- знаходитись у учбовій лабораторії у верхньому одязі.

Після закінчення занять обладнання не вимикається. Робоче місце має бути прибрано працюючим та перевірене викладачем чи співробітником кафедри

ІНСТРУКЦІЯ

про міри пожежної безпеки у лабораторіях, учбових та робочих приміщеннях кафедри кібернетики хіміко-технологічних процесів хіміко-технологічного факультету

Всі студенти повинні знати та ретельно виконувати «Загальні правила пожежної безпеки в НТУУ «КПІ».

Завідуючий кафедрою та завідуючий лабораторією відповідають за забезпечення пожежної безпеки всіх приміщень кафедри та за справність протипожежного обладнання та сигналізації.

Все електричне обладнання, яке знаходиться в лабораторіях та приміщеннях кафедри, повинно мати заземлення.

В усіх приміщеннях повинно дотримуватись чистоти, не займати приміщення непотрібними меблями, обладнанням та матеріалами.

Всі двері основних та додаткових виходів утримувати у стані швидкого відкривання.

Зберігання та використання горючих та легкоспалахуючих рідин у приміщеннях кафедри забороняється.

Ремонт електричного обладнання проводити у строгій відповідності з правилами пожежної безпеки

Всі електрозахисти повинні знаходитися у закритому положенні, не займаними сторонніми предметами.

Коридори, проходи, тамбури, евакуаційні виходи та підходи до першочергових засобів пожежогасіння, а також комунікаційні ніші повинні бути постійно вільними, чистими та нічим не займаними.

Відповідальні особи перед закриттям приміщень повинні ретельно оглянути їх, забезпечити прибирання виробничих відходів, перевірити якість перекриття води, газу, відключити напругу електромережі, перевірити стан пожежної сигналізації та засобів пожежогасіння.

Від усіх приміщень мати два комплекти ключів. Один комплект здавати черговому, а інший – зберігати в певному місці, яке відомо обслуговуючому персоналу.

Студенти повинні знати та ретельно виконувати «Загальні правила техніки безпеки в НТУУ «КПІ», про що вони ставлять свій підпис у відповідному контрольному листі з техніки безпеки перед початком проведення циклу лабораторних робіт. Студенти, які не пройшли інструктаж і не поставили підпис у контрольному листі, до роботи не допускаються.

Зразок титульного листа.

Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут"

Кафедра кібернетики хіміко-технологічних процесів

Звіт
з лабораторних робіт
Дисципліна: *"Комп'ютерні технології та програмування"*

Виконав: студент 1-го курсу ХТФ

гр. ХА-31

Козаченко П.М.

Керівник: доц. Квітка О.О.

Київ 2013 р.

Блок-схеми.

Блоки, що відповідають усім операціям, які використовуються в програмах, мають стандартизовані форму і розміри. Блоки, які використовуються найчастіше наведені в таблиці 1. Блоки розташовуються у блок-схемі згідно з логікою алгоритму. Блок-схема розпочинається з блоку із словом «Початок» і закінчується блоком «Кінець» (див. табл. 1 п.6). Усі блоки послідовно нумеруються. Номер проставляється в розриві лінії в лівому верхньому кутку кожного блоку (окрім блоків «Початок» і «Кінець»).

Блоки з'єднуються між собою горизонтальними і/або вертикальними лініями (лінії, розташовані під кутом, не допускаються) із стрілками на кінці, що відображають напрям обчислювального процесу. При цьому стрілки, спрямовані "вниз" і "праворуч" можуть бути опущені. Перетину стрілок бажано не допускати. При неможливості провести нерозривну стрілку між двома блоками, використовуються або "з'єднувач", – якщо блоки розташовані на одній сторінці, або "міжсторінковий з'єднувач" – якщо вони розташовані на різних сторінках (див. табл. 1, відповідно п.7 і п.8). При використанні "з'єднувача" на одному кінці розриву в "колі" проставляється номер наступного блоку, а на іншому – номер попереднього блоку. Коли необхідно з'єднати стрілкою блоки на різних сторінках у "міжсторінковому з'єднувачі" аналогічним чином проставляються не лише номери *подальшого і попереднього* блоків, але і номери сторінок, на яких вони розташовані (рис. 1).

У кожному "розрахунковому блоці" (див. табл. 1 п.1) допускається написання не більше однієї розрахункової формули, але можливе "об'єднання" декількох формул в одному блоці під загальною назвою (рис. 2), якщо вони об'єднані загальним змістом або однаковим призначенням.

Алгоритм, що реалізовується у вигляді окремого програмного модуля, є автономним і зображується у вигляді окремої блок-схеми (окремого малюнка). Зв'язок між різними модулями однієї програми і, відповідно, – між блок-схемами алгоритмів які в них реалізовані, здійснюється за допомогою "блоку підпрограми" (див. табл. 1 п.5), який відповідає операторові звернення до підпрограми в "викликаючому модулі".

Алгоритм наводиться у вигляді графічних фігур (блоків), кожний з яких має певне смислове значення і є частиною рішення задачі.

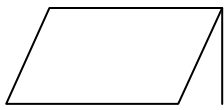
Розглянемо основні блоки:

$$b=1,5a$$



b

а 1. обчислювальний блок (виконання операції або групи операцій).

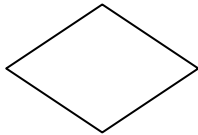


2. блок введення-виведення
 $b = 0.25 a$



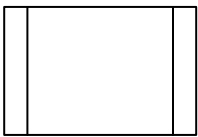
3. виведення на паперовий носій

b



4. логічний блок (вибір напрямку виконання алгоритму залежно від деяких умов)

b



5. "підпрограма"

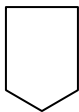
b



$d = 0.5a$

6. з'єднувач (зв'язок між перерваними лініями в межах однієї сторінки).

$0,5a$

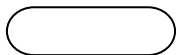


$0.6a$

7. міжсторінковий з'єднувач

-----[a

8. коментар (зв'язок між елементом схеми і поясненням).



$0.5a$

9. початок, кінець

b

Приклад.

