МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра програмного забезпечення систем

О.І. ШПАК, М.І. РОЛЬ, І.І. ШПАК

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОФІСНИХ СИСТЕМ

НАВЧАЛЬНО-МЕТОДИЧНИЙ ПОСІБНИК

УЖГОРОД – 2023

Розробка програмного забезпечення для офісних систем: навчально-методичний посібник до вивчення курсу для студентів спеціальностей: 121 Інженерія програмного забезпечення, 122 Комп'ютерні науки / О.І. Шпак, М.І. Роль, І.І. Шпак. – Ужгород: 2023. 88 с.

Навчально-методичний посібник з курсу «Розробка програмного забезпечення для офісних систем» містить теоретичні та практичні відомості щодо основ офісного програмування для вирішення практичних задач, перелік використаних джерел.

Укладачі:

Шпак О.І., к.ф-м.н., доцент кафедри програмного забезпечення систем факультету інформаційних технологій УжНУ;

Роль М.І., старший викладач кафедри програмного забезпечення систем факультету інформаційних технологій УжНУ.

Шпак І.І., к.ф.-м.н., ст.н.с., доцент кафедри програмного забезпечення систем факультету інформаційних технологій УжНУ;

Рецензенти:

- Кут В.І., к.т.н., доцент, завідувач кафедри інформатики та фізико-математичних дисциплін факультету інформаційних технологій ДВНЗ "УжНУ";
- Лях І.М., к.т.н., доцент, доцент кафедри інформатики та фізико-математичних дисциплін факультету інформаційних технологій ДВНЗ "УжНУ".

Рекомендовано кафедрою програмного забезпечення систем.
Протокол № 2 від 21 вересня 2023 року.

Рекомендовано Вченою радою факультету інформаційних технологій.
Протокол № 2 від 25 вересня 2023 року.

ЗМІСТ

ПЕРЕДМОВА

У навчальному посібнику розглядаються питання основ офісного програмування в пакеті MS Office. Викладаються технології роботи з мовою об'єктно-орієнтованого програмування Visual Basic for Application (VBA). Подається докладне роз'яснення структури елементів програмування мови VBA, короткі теоретичні відомості, необхідні для освоєння логіки мови програмування, і комплекси завдань для виконання на прикладі різних предметних областей. Наводяться приклади розв'язку завдань за допомогою об'єктно-орієнтованого програмування і їх використання для вирішення практичних задач.

Навчальний посібник розроблений згідно вимог навчальної програми з дисципліни «Розробка програмного забезпечення для офісних систем».

Невід'ємним завданням Microsoft Office є оптимізація процесів обчислення, пошуку, редагування, форматування, засобом виконання яких може виступати об'єктно-орієнтоване програмування на мові Visual Basic for Application (VBA). Використовуючи офісне програмування на мові VBA можна оптимізувати часте виконання користувачем однієї тієї ж самої операції, створивши для неї функцію Microsoft Office. Функції, які додані користувачем дуже часто використовуються при обробці великих масивів даних і вирішення питань по автоматизації систем при керуванні організаціями.

Даний навчальний посібник призначений для засвоєння основ офісного програмування на мові VBA з ціллю закріпити практичні навички оптимізації роботи користувача в Microsoft Office, а також ознайомити з логікою об'єктно-орієнтованого програмування на мові VBA для вирішення практичних завдань.

В посібнику наводяться приклади вирішення практичних завдань по програмуванню на мові VBA. Програми реалізуються в середовищі MS Excel з врахуванням наступних можливостей об'єктно-орієнтованого програмування на мові VBA і технології створення об'єктів:

- розробка програм з використанням:
  - процедур і функцій;
  - керуючих структур;
  - математичних і тригонометричних процедур и функцій;
  - алгоритмів сортування масивів і пошуку в масивах;
  - рядкових змінних і функцій обробки рядків;
- додавання користувацьких форм і елементів керування на користувацькі форми;
- використання та запис макросів, як єдиної команди для автоматичного виконання завдання.

В кожній темі наводяться теоретичні відомості, які необхідні для засвоєння наведених технологій і комплекс прикладів. Для кращого сприйняття матеріалу окремі теоретичні відомості супроводжуються ілюстраціями вікон програм у VBA. Для закріплення отриманих знань розроблені контрольні питання і завдання для самостійної роботи, а також тести по темам, по яким рекомендується проведення проміжного і поточного контролю знань студентів.

ТЕМА 1. ВСТУП ДО РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОФІСНИХ СИСТЕМ

1.1. Загальні положення

Поняття офісного програмування і його можливості

Розробка програмного забезпечення для офісних систем (надалі «Офісне програмування») являє собою процес розробки програм, які призначені для автоматизації офісної діяльності. Результатом офісного програмування можуть бути як достатньо прості рішення для обробки документів, так і масштабні проекти, що пов'язані з автоматизацією систем, які використовуються в управлінні організацією.

Офісне програмування має ряд особливостей у порівнянні з класичним програмуванням. По-перше, ціллю розробки у офісному програмуванні є не програма, а документ або сукупність документів. Навіть збереження проекту окремо від документу в офісному програмуванні неможливе. По-друге, середовище розробки орієнтоване в першу чергу на користувачів, а не на програмістів. Це означає, що в ньому передбачена можливість створення програм без написання коду за допомогою макросів. По-третє, це наявність вбудованої мови. Для всіх продуктів MS Office це мова є VBA – мова програмування, яка вбудована в пакети прикладних програм Microsoft Office, а також деяких допоміжних програмних продуктів. Вона дозволяє розширити функціональні можливості програми, в якій вона використовується.

За допомогою мови VBA можливо вирішити ряд завдань, в тому числі:

- автоматизація часто відтворюваних операцій;
- створення нових команд і закріплення зручних способів для їх виклику;
- створення нових кнопок на панелі інструментів;
- створення користувацьких діалогових вікон і форм з інтерфейсом;
- прискорення процедури редагування, форматування і т.п.

Являючись достатньо простою у засвоєнні, VBA може стати базою початкового вивчення програмування. В той же час глибокі знання можливостей офісного програмування на її основі дозволяють створити досить серйозні програмні продукти і документи, які можуть використовуватись в масштабах великих організацій.

Об'єктно-орієнтоване програмування: основні поняття і принципи

Однією з переваг мови VBA є те, що вона відноситься до об'єктно-орієнтованих мова програмування.

Об'єктно-орієнтоване програмування – це метод, при використанні якого головними елементами програм є об'єкти. Об'єктом виступає окремий елемент певного класу. Він характеризується сукупністю властивостей (структур даних, які характерні для даного об'єкту), методів їх обробки (підпрограм зміни їх властивостей) і подій на які даний об'єкт може реагувати і які призводять, як правило, до зміни властивостей об'єкта.

Основними поняттями і термінами мови VBA є:

- **атрибут** – характеристика, що призначена елементу класу (наприклад, властивості або методу);
- **клас** – визначення структури і поведінки об'єктів певного типу;
- **конструктор** – особливий метод, який викликається при створенні екземпляру класу;
- **тип даних** – певні види даних, які VBA зберігає і якими може маніпулювати;
- **оператор «крапка»** – знак крапки (.), який служить для вказівки того, що ім'я відноситься до дочірньому елемента об'єкту (наприклад, до властивості або методу);
- **екземпляр класу** – фактичний об'єкт, створений в програмі.

Основними принципами об'єктно-орієнтованого програмування є інкапсуляція, успадкування і поліморфізм.

Інкапсуляція являє собою об'єднання даних і властивих їм процедур обробки в одному об'єкті. В результаті користувач взаємодіє з інтерфейсом, не стикаючись з реалізацією програмного компоненту.

Спадкування – створення нових класів на базі існуючих з частковим або повним запозиченням функціональності.

Поліморфізм надає можливість об'єктам з однаковою специфікацією мати різну реалізацію.

Елементи проекту мови програмування VBA

Незалежно від програми, в якій створюється проект (MS Word, MS Excel, MS Access т. ін.), мова VBA включає в себе інтегроване середовище розробки – редактор Visual Basic. Зовнішній вигляд даного редактора наведено на рис. 1.1. Його виклик здійснюється за допомогою поєднання натискання клавіш Alt+F11, або через меню Розробник → Visual Basic (сама вкладка «Розробник» прихована за замовчуванням, активувати її можна в налаштуваннях).

Загальний вигляд редактора Visual Basic включає три основні типи вікон: *вікно проекту (1)* (містить вказівники на вікна програмних кодів і користувацьких форм), *вікно властивостей (2)* (дозволяє змінювати будь які властивості активного об'єкту), *вікно програми (3)* (дозволяє створювати, переглядати і редагувати програмний код). Крім цього, по бажанню користувача на екрані можуть бути розміщені додаткові вікна (4) (наприклад, вікно налагодження і т.п.).

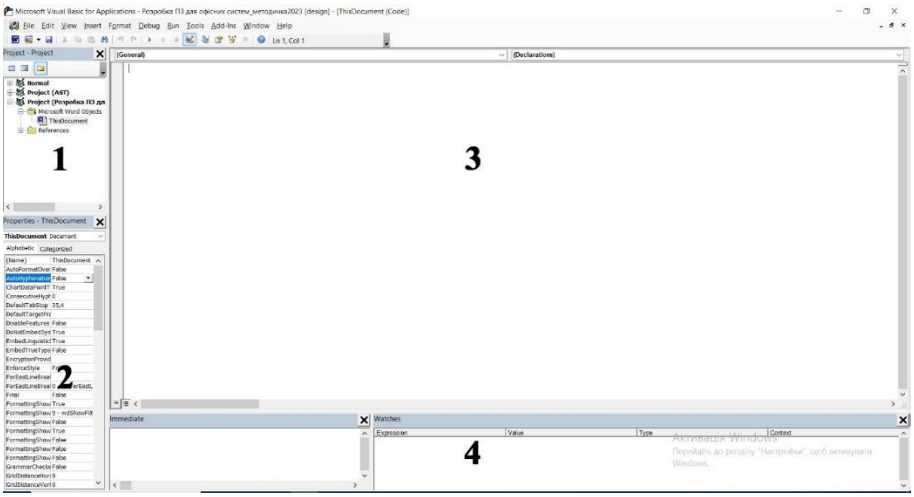


Рис.1.1. Загальний вигляд редактора

Контрольні питання і завдання

1. Що являє собою мова програмування VBA?
2. Яким чином використовується редактор Visual Basic?
3. Які особливості об’єктно-орієнтованого програмування?
4. Опишіть принципи офісного програмування.
5. Якими атрибутами характеризується об’єкт в ООП?

Практичні завдання

1. Запустіть програму Visual Basic з MS Word. Розгляньте основні вікна, команди меню.
2. Запустіть програму Visual Basic з MS Excel. Розгляньте основні вікна, команди меню.

1.2. Елементи мови програмування VBA

Структура програми

Основними поняттями, які відображають сукупність команд являється вихідний код, програма, процедура та макрос.

Процедура – послідовність команд, які виконуються разом, і яка має ім'я.

У мові VBA розрізняють два види процедур: підпрограма і функція.

Процедура-підпрограма – це самостійна програмна одиниця VBA, що включає різні типи команд. Вона починається з ключового слова `Sub` і унікального імені процедури. Далі в дужках містяться аргументи. Закінчення підпрограми позначається словами `End Sub`.

Синтаксис запису процедури-підпрограми наступний:

```
Sub <ім'я> (<аргументи> As <тип даних>)
<Команди>
End Sub
```

Після імені процедури в дужках можуть зазначатися аргументи – сукупність вихідних даних, що передаються в процедуру, а також тип цих даних.

> **Якщо в процедурі задані аргументи, то їх значення повинні бути перелічені в операторі, який викликає процедуру!**

Процедура-функція – це процедура, виконання якої формує деяке значення. Вона починається з ключового слова `Function` і імені функції. Далі в дужках вказуються аргументи. Закінчення підпрограми позначається словами `End Function`. Перед закінченням процедури необхідно задати значення, що повертається через команду ім'я = значення.

Синтаксис запису процедури-функції:

```
Function <ім'я> (<аргументи> As <тип даних>)
<Команди>
ім'я = <значення>
End Function
```

Сукупність створених користувачем процедур називається **модулем**.

Також процедури розрізняють по області їх дії. Процедура `Private Sub` доступна для всіх процедур в активному модулі і недоступна для процедур інших модулів.

Для позначення процедур, що доступні з будь якого модуля або проєкту, використовується словосполучення `Public Sub`. За замовчуванням процедури, описані без вказівки області дії, вважаються доступними з будь-яких модулів.

Макрос – послідовність команд, що дозволяє автоматично виконувати певну послідовність дій. Послідовність операторів мови VBA називається *кодом макросу*. Макроси можуть бути написані користувачем або створені в режимі протоколювання.

> **Документи, що містять макроси, повинні обов’язково зберігатися в файлах з підтримкою макросів! В MS Word 2010 і вище – це файли з розширенням .docm, в MS Excel 2010 і вище - .xlsm.**

Ідентифікатор, змінна, константа

Обов'язковими складовими програмного коду є ключові слова, ідентифікатори, змінні та константи. Крім того, можуть бути присутніми оператори, вирази, коментарі і т. д.

Оператор – це синтаксично завершена конструкція, що представляє окрему дію, опис або визначення.

Ключові слова – це слова або символи, які розпізнаються як елемент мови VBA. Ключові слова не можуть використовуватися в якості імен програм і

змінних. За замовчуванням в мові VBA ключові слова виділяються синім кольором.

Змінні – іменована область пам'яті, відведена для тимчасового зберігання даних, які можуть змінюватися під час виконання програми.

Виклик змінної здійснюється за іменем (ідентифікатору). Ідентифікатором є ім'я програми або програмного об'єкта (константи, змінної і т. д.). В якості імені змінної використовується послідовність літер і цифр (**першою обов'язково повинна бути буква**). Імена не повинні включати крапок, пробілів і символів **%, &, @, #, $, !**. Регістр написання в програмному коді не враховується. Ідентифікатори повинні бути унікальними в поточній видимій області і не збігатися з назвами функцій і операторів мови VBA. Максимальна довжина імені 255 символів.

Оголошення всіх змінних в мові VBA бажаним, але не є обов'язковим. Для оголошення змінних використовується ключове слово `Dim`, після чого вказується ім'я змінної, ключове слово `As` і тип даних. В одному рядку може бути перераховано кілька змінних. Синтаксис оператора має наступний вигляд:

```
Dim <змінна> As <тип даних>
```

Константи – іменований елемент, який зберігає постійне незмінне значення в процесі виконання програми. В ході виконання програми значення константи, на відміну від змінної, змінити не можна. Виділяють вбудовані константи і користувацькі. Користувацькі константи повинні бути оголошені в програмі за допомогою ключового слова `Const`.

Синтаксис оголошення наступний:

```
Const <ім'я> As <тип даних> = <значення>
```

> **Для запису констант в тексті програми використовується крапка, а не кома!**

Визначені в мові VBA константи не вимагають оголошення. Вони починаються з символів `vb`. Наприклад, константа `vbmonday` (константа, що позначає понеділок), яка використовується при заданні форматів часових даних – задає перший день тижня.

Коментар – пояснення до програми, які ігноруються при її виконанні. Коментарем вважається текст від одинарної лапки до кінця рядка. У редакторі VBA коментарі виділяються зеленим кольором. Бажано коментувати основні змінні, їх дії і ключові місця програми.

Типи даних і їх сумісність

Оголошення даних визначає тип даних, діапазон допустимих значень і розмір пам'яті, що відводиться під змінну. Мова VBA включає вбудовані і визначені користувачем типи даних. Вбудовані типи даних поділяються на числові, рядкові, логічні, дати та часу.

Числових типів даних в мові VBA є декілька:

- *цілі числа*:

`Byte` - від 0 до 255 (кількість пам'яті 1 байт);

`Integer` - від -32 768 до 32 767 (кількість пам'яті 2 байти);

`Long` - від -2 147 483 648 до 2 147 483 647 (розмір займаної пам'яті 4 байти);

- *числа з дробовою частиною:*

`Currency` - від -922 337 203 685 477,580 8 до 922 337 203 685 477,580 7 (грошовий формат з фіксованою крапкою, розмір займаної пам'яті 8 байт);

`Single` - від'ємні числа від -3,4028235E + 38 до -1,401298E - 45 і додатні числа від 1,401298E - 45 до 3,4028235E + 38 (з плаваючою крапкою, розмір займаної пам'яті 4 байти);

`Double` - від'ємні числа від -1,79769313486231570E + 308 до -4,94065645841246544E - 324 і додатні числа від 4,94065645841246544E - 324 до 1,79769313486231570E + 308 (з плаваючою крапкою, кількість пам'яті 8 байт).

Для збереження тексту використовуються ***рядкові змінні***. У мові VBA для оголошення рядкових змінних використовується тип `String`, який зберігає до 2 млрд символів. Рядкові дані в тексті програми кладуться в лапки. Розмір займаної пам'яті залежить від довжини рядка. Також в даній мові відсутній тип даних для символів, все це рядкові змінні.

Для ***логічних змінних*** використовується тип `Boolean`. Він приймає одне з двох значень: `True` (істина) або `False` (Брехня). Кількість пам'яті 2 байти.

Для представлення ***дати і часу*** використовується тип `Date`. Він дозволяє зберігати і здійснювати операції над датами з 01.01.100 до 31.12.9999. Кількість пам'яті 8 байт.

Тип `Variant` дозволяє зберігати інформацію будь-якого типу. Якщо в програмі змінна не оголошена або при оголошенні не вказано тип даних, то вона сприймається як тип `Variant`. Даний тип даних вимагає більшого обсягу пам'яті, ніж інші, тому рекомендуємо його використання по можливості уникати.

Не всі типи даних сумісні один з одним. Використання несумісних типів даних в одному і тому ж виразі неможливо. При обробці таких виразів з різними типами змінних мова VBA буде намагатися усунути відмінність типів шляхом перетворення значення у виразі в сумісні типи даних. Якщо усунути відмінності перетворенням типів не вдається, то відображається помилка і процедура припиняє виконуватися.

Спроба привласнити змінній значення, яке виходить за діапазон допустимих для даного типу, також призводить до помилки.

Дізнатися поточний тип змінної можна за допомогою наступної конструкції: `TypeName <ім'я змінної>.`

В програмі також можна створити власні користувацькі типи даних, які позначаються як `Type`. Користувацький тип даних дозволяє об’єднати декілька змінних з різними типами даних в єдину структуру. Синтаксис запису користувацького типу даних наступний:

```
Type <ім'я>
<Ім'я_змінної1> As <тип даних>
<Ім'я_змінної2> As <тип даних>
...
End Type
```

Використання таких типів використовується для перевірки введених значень (наприклад, номер IBAN розрахункового рахунку в банку). Зручно використовувати користувацькі типи для задання параметрів групи змінної (наприклад, відомості про фізичну особу в деякій базі даних містять інформацію про його прізвище та ім’я – рядкові змінні, дату народження – змінна дати і т. д.).

Типи операторів

Найпростішим способом здійснення і запису операцій зі змінними в мові VBA є використання операторів.

Традиційно одним з найбільш використовуваних є оператор присвоєння. Він призначений для задання початкових значень, запису результату обчислень в змінну, зміни значень і має наступний синтаксис:

```
<Змінна> = <вираз>
```

В якості змінної може виступати звичайна змінна, елемент масиву або властивість об'єкту. Виразом може бути як конкретне значення (змінна, константа), так і ті, що вимагають додаткових розрахунків (операції, функції).

Приклад. Написати програму для перетворення довжини з сантиметрів в дюйми.

```
Sub sm_v_dujm()
Dim lsm, ldujm As Single
Const dujm As Currency = 2.54
lsm = 20 'задає довжину в сантиметрах
ldujm = lsm / dujm 'розраховує довжину в дюймах
End Sub
```

Примітка. *Для запуску програми необхідно, перебуваючи у вікні програмного коду, натиснути кнопку F5.*

Оператор присвоювання може бути записаний ключовим словом `Let` в наступному вигляді:

```
Let <змінна> = <вираз>
```

тобто на відміну від JavaScript ключове слово `Let` є необов'язковим і зазвичай його опускають.

Арифметичні операторі використовуються для позначення математичних операцій. Основних арифметичних операторів в мові VBA сім (табл. 1.1).

Таблиця 1.1. Арифметичні оператори

Оператор мови VBA	**Математична операція**
+	Додавання
-	Віднімання, зміна знаку
*	Множення
/	Ділення
^	Зведення до степені
\	Залишок від ділення
Mod	Залишок від ділення цілих чисел (результат – ціле число)

Порядок виконання арифметичних дій відповідає загальноприйнятим правилам обчислень. Для зміни порядку виконання операцій можна використовувати круглі дужки.

Слід враховувати, що на результат виконання операцій може вплинути також тип змінної, яка використовується. Наприклад, якщо в результаті ділення отримано дробове число, то воно буде заокруглюватися до одного з цілих типів і записуватися дробовим для типів з дробовою частиною.

Ще одна група операторів, які використовуються в мові VBA – ***оператори порівняння***. Вони можуть використовуватися як для числових, так і для рядкових змінних. Результатом порівняння буде логічна (Boolean) змінна, яка приймає значення `True`, якщо порівняння вірне, і `False` – якщо порівняння невірне.

Основні оператори порівняння наведені у таблиці 1.2.

Таблиця 1.2. Оператори порівняння

Оператор мови VBA	**Опис**
=	Рівне
>	Більше
<	Менше
>=	Більше або рівне

<=	Менше або рівне
<>	Не рівно
Like	Відповідність шаблону

Операція порівняння застосовується тільки до даних одного типу. Даними можуть виступати значення змінних, констант і т.п.

При порівнянні рядкових змінних відбувається послідовне, по символьне порівняння зліва направо в алфавітному порядку. Рівність двох рядків буде при повному співпадінні їх довжини і кожного з символів двох змінних (в даному випадку регістр букв враховується).

Оператор `Like` порівнює рядкову змінну з шаблоном і визначає їх відповідність. Шаблон задається спеціальними підстановочними символами, значення яких наведено в таблиці 1.3.

Таблиця 1.3. Підстановочні символи, які використовуються для створення шаблонів оператора `Like`

Підстановочний символ	Опис
#	Одна будь яка цифра (від 0 до 9)
*	Будь яка кількість символів (включаючи нульове)
?	Будь який один символ
[a,b,c]	Будь який символ зі списку
[!a,b,c]	Будь який символ, крім перелічених в списку

Окрему групу операторів складають рядкові оператори. ***Оператор складання (конкатенація)*** двох рядків позначається символом `&`. Результатом буде змінна типу `String`, яка містить послідовно перший і другий рядок.

> **Якщо операція складання буде виконана до числових змінних, то мова VBA спочатку перетворить їх в текст, а потім здійснить операцію. В подальшому виконання арифметичних дій над отриманою змінною без додаткового перетворення в число буде неможливим!**

Для додавання рядкових змінних може використовуватися і знак «+», але рекомендується використовувати саме символ &, так як це дозволяє уникнути плутанини з типами даних.

Контрольні питання і завдання

1. Що таке макрос і яке його значення в мові VBA?
2. Назвіть основні елементи будь якого програмного коду.
3. Які основні типи даних в мові VBA?
4. Яким чином можна класифікувати основні оператори?
5. Назвіть характерні особливості числових типів даних.
6. Для чого використовуються користувацькі типи даних?

Практичні завдання

1. Напишіть програму перетворення одиниць виміру маси з кілограмів у фунти.
2. Напишіть програму конвертації валют.
3. Розробіть процедуру-функцію, яка обчислює площу квадрата по заданій довжині його сторони.
4. Розробіть процес-функцію, яка обчислює об’єм кубу по заданій довжині його ребра.

Приклади виконання завдання

Приклад 1. Перевірити результат присвоєння даних змінним різних типів.

```
Sub format_chisla()
Dim a As Byte, b As Integer, c As Long, d As Single
a = 11 * 9 'буде присвоєно значення 99
b = -11111 / 9 'буде присвоєно значення -1235 (без дробової частини)
c = 8884 / 3 'буде присвоєно значення 2961 (без дробової частини)
a = 111 / 9 'минуле присвоєне значення втрачається, на його місце буде присвоєно значення 12 (без дробової частини)
b = 11111 / 9 'минуле присвоєне значення втрачається, на його місце буде присвоєно значення 1235 (без дробової частини)
c = 11111 / 9 'минуле присвоєне значення втрачається, на його місце буде присвоєно значення 1235 (без дробової частини)
d = 11111 / 9 'буде присвоєно значення 1234,556
End Sub
```

Змінній *a* не може бути присвоєне значення, яке виходить за межі діапазону [0; 255], для змінної *b* діапазон допустимих значень [-32 768; 32 767]. Для цілих чисел дробова частина обрізається по правилам заокруглення.

Приклад роботи програми, що відображає результати присвоєння значень змінним, наведений на рис. 1.2.

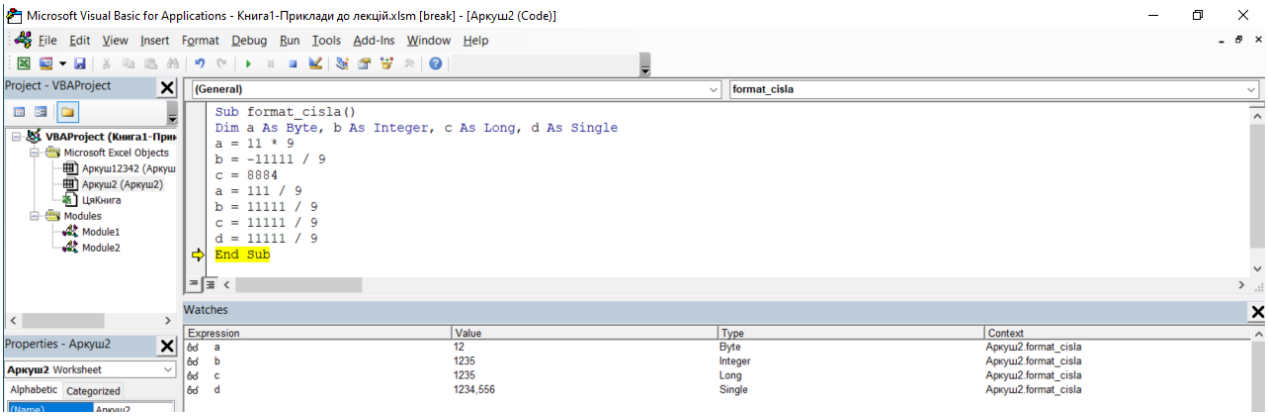


Рис. 1.2. Результат роботи програми

Примітка. Для перегляду поточних значень змінних необхідно використовувати вікно перегляду значення змінних (Watches). Для того що відкрити його, потрібно виконати через меню команду Debug->AddWatch і в рядку Expression ввести ідентифікатор потрібної змінної. При цьому програма повинна виконуватися покроково, що реалізується через команду Step Into, яка викликається в меню Debug або кнопкою F8.

Приклад 2. Розробити процедуру-функцію, що обчислює площу кола по заданому радіусу.

```
Function S_kola(d as Integer) as Single
Const Pi As Single = 3.14159 'задає значення константи π
S_kola = Pi * ((d / 2)^2) 'розраховує площу кола
End Function
```

Примітка. *Процедура-функція повертає значення по заданому ззовні параметру. Тому вона може бути запущена із іншої процедури, в якій це значення вводиться.*

```
Sub kolo()
Dim d As Integer
d = 12 'присвоює змінній значення діаметру кола
s = S_kola(d) 'присвоює змінній значення площі кола, що обчислюється в процедурі-функції
End Sub
```

Результат роботи програми наведений на рис. 1.3.

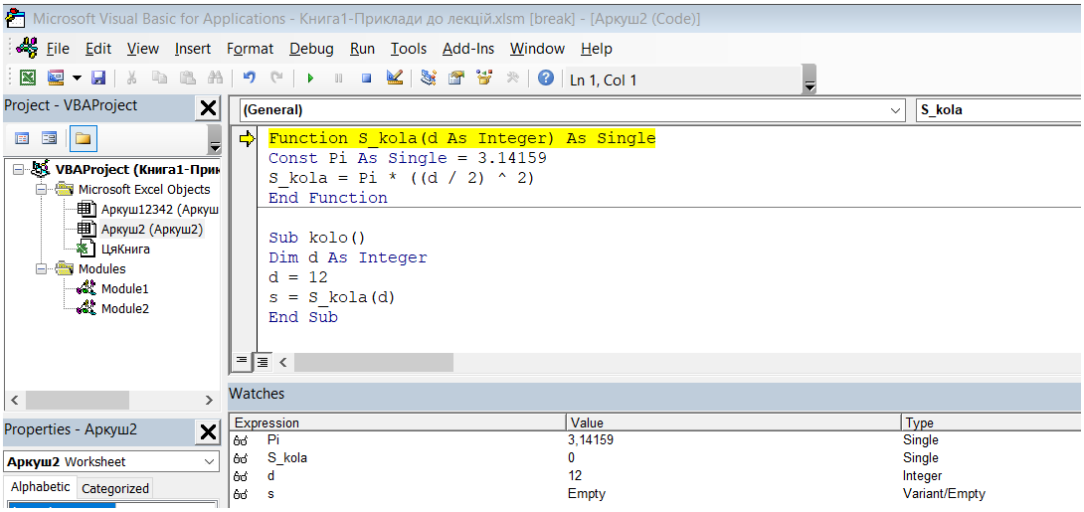


Рис. 1.3. Результат роботи програми

Примітка. *В процедурі-функції значення площі кола вираховується по заданому параметру* `Pi` *і діаметру, які отримані з процедури-підпрограми* `kolo()`*. На наступному кроці процедура-функція завершує своє виконання, значення всіх змінних в ній очищуються, а обчислення площі кола присвоюється змінній* `s` *в процедурі-підпрограмі.*

1.3. Створення процедур на мові VBA

Технології створення процедур

Самим простим способом створення макросів є їх протоколювання. Для запуску режиму протоколювання необхідно виконати послідовність у вкладці *Подання* → *Макроси* → *Записати макрос*. Також можна це зробити через вкладку *Розробник* → *Записати макрос*. В результаті відкриється діалогове вікно. Зовнішній вигляд даного вікна відрізняється для різних програмних продуктів пакету MS Office. Діалогове вікно *Запис макросу* в Microsoft Word 2016 представлено на рис. 1.4.

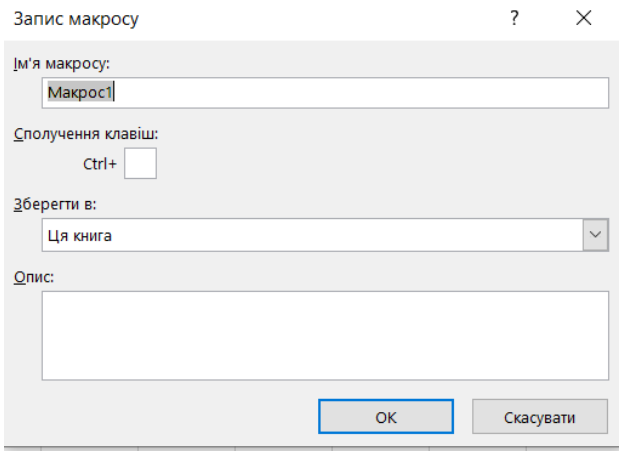


Рис. 1.4. Діалогове вікно *Запис макросу*

Макрос може бути призначений комбінації клавіш на клавіатурі. Після завершення дій, які потрібно записати та виконати в макросі, необхідно

натиснути кнопку *Зупинити запис*. В результаті буде створений макрос, який можна побачити у списку макросів і відредагувати за потреби в редакторі Visual Basic.

При необхідності можна не завершувати процес запису макросу, а тільки тимчасову його призупинити, натиснувши кнопку *Пауза* на *Панелі інструментів*.

Функція протоколювання необхідна для того, щоб не писати деякі елементи форматування з нуля в програмному модулі, а просто відредагувати в записаному програмному коді.

Другим способом створення процедур являється їх написання у вигляді програмного коду. Для цього необхідно відкрити вікно редактора мови VBA, обрати пункт меню *Insert* → *Module* або відповідну кнопку на панелі інструментів (див. рис. 1.5). Далі у вікні проєкту, що відкрилося, потрібно ввести послідовність операторів між ключовими словами, які відповідають початку і кінцю процедури.

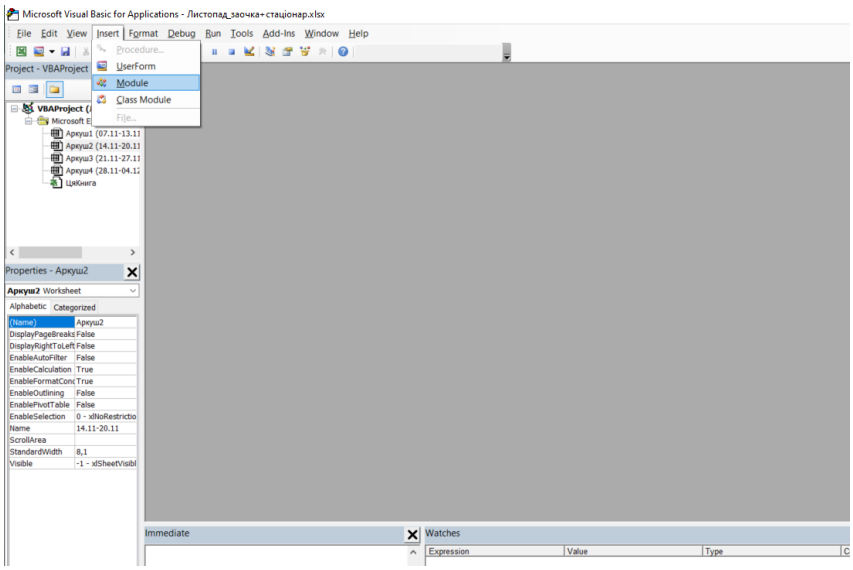


Рис. 1.5. Вставка модуля

Налагодження і редагування модулів

Створення будь-якого програмного продукту передбачає декілька етапів: проектування, розробку, тестування і налагодження, впровадження.

Процес тестування передбачає детальну перевірку працездатності програми в різних режимах. Налагодження полягає в усуненні помилок в програмі або внесенні необхідних змін.

Викликати список всіх доступних макросів можна за допомогою комбінації клавіш Alt + F8 або через послідовність *Подання* → *Макроси*. Вигляд відповідного діалогового вікна наведений на рис 1.6.

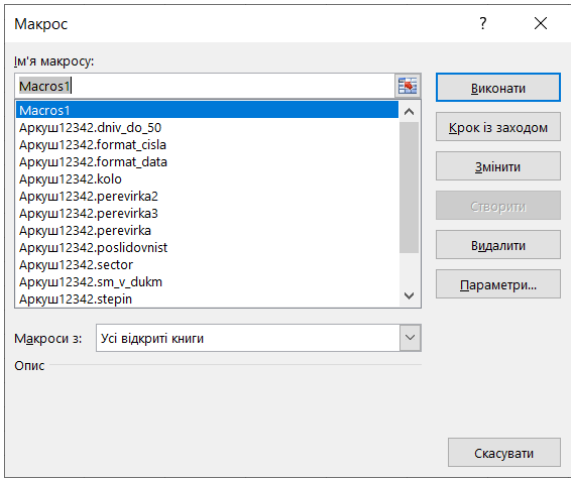


Рис. 1.6. Діалогове вікно «Макроси»

В даному списку відображаються макроси, що містяться у всіх відкритих книгах. В ньому можна обрати один макрос для запуску або редагування. Запуск макросу здійснюється подвійним кліком мишки по його назві або натисканням кнопки *Виконати*. Кнопка *Крок із заходом* також запускає макрос, але не у звичайному режимі, а в покроковому, який дозволяє відслідковувати процес обробки даних програмою. Кнопка *Змінити* відкриває відповідний модуль в редакторі VBA. Кнопка *Видалити* безповоротно видаляє макрос із документу.

Редагування макросів здійснюється в редакторі VBA шляхом запису або зміни команд і операторів.

Для налагодження роботи програми корисною буде функція покрокового запуску. Для цього необхідно обрати пункт меню *Debug* → *Step Into* або натиснути клавішу F8 на клавіатурі.

Екран вибору функції покрокового запуску наведений на рис. 1.7.

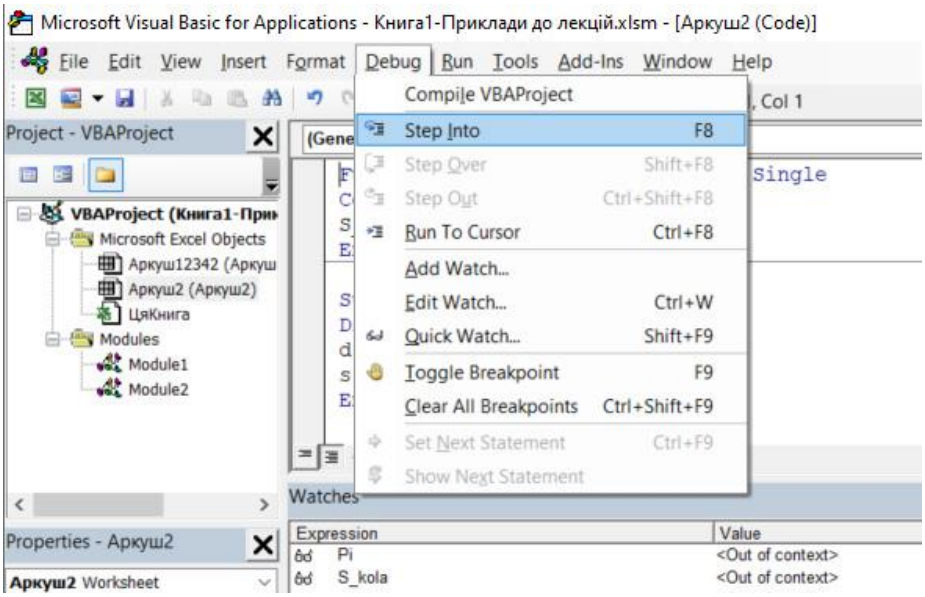


Рис. 1.7. Функція покрокового запуску

При цьому перший рядок процедури підсвічується жовтим кольором, а на полі зліва з'являється жовта стрілка. Для продовження руху по процедурі необхідно виконати ту саму команду або натиснути клавішу F8. В результаті підсвічування і стрілка перемістяться на наступний рядок і т. д., тобто кроком в VBA є рядок. Для керування процедурою можна використовувати наступні кнопки на панелі інструментів:  .

Якщо програмний модуль досить великий, то покроковий рух по процедурі стане незручним. Для зупинки в певному місці можна поставити відмітку зліва від потрібного рядка.

Натисканням лівої кнопки мишки поряд із рядком, на якому необхідно зупинити виконання процедури і перейти в режим покрокового руху, або натисканням клавіші F8 на клавіатурі ми можемо позначити цей рядок коричневим кружком (*контрольна точка*, англ. *breakpoints*). Прибрати контрольні точки можна аналогічними діями.

Скріншот екрану з встановленою відміткою наведений рис. 1.8.

Функцію переривання виконання програми також можна реалізувати через ключове слово `Stop`, що розміщене в коді програми перед потрібною нам командою.

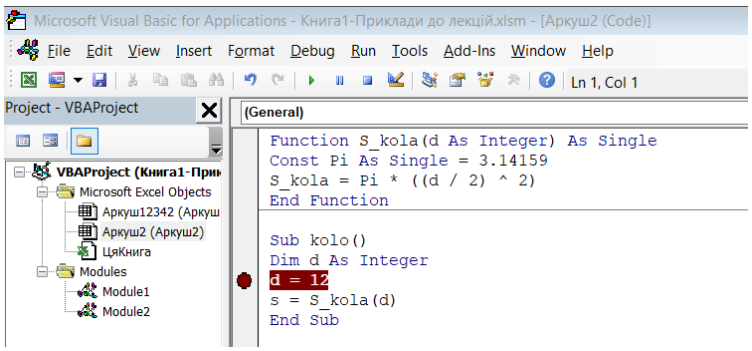


Рис. 1.8. Відмітка в програмі для переходу в режим покрокового руху

Для налагодження роботи програми необхідний не тільки покроковий рух по модулю, але і відслідковування значень змінних, що використовуються в програмі. Це можна реалізувати декількома способами. Якщо навести курсор на ім'я змінної, одночасно утримуючи клавішу *Ctrl* на клавіатурі, то у спливаючому полі буде вказано поточне значення цієї змінної. Крім цього відслідкувати значення одночасно декількох параметрів можна у вікні налагодження програми *(Watches)*. Викликати його на екран можна через пункт меню *View → Watch window*. При запуску вікно не буде містити ніяких змінних. Їх додавання відбувається через пункт меню *Debug → Add Watch*, після чого необхідно ввести ім'я потрібної змінної. Якщо в програмі помістити курсор на деяку змінну і виконати дану команду, то ця змінна буде запропонована у вікні як значення по замовчуванню (рис. 1.2).

Інколи виникає необхідність переглянути код в декількох місцях одночасно, особливо якщо він великий за розміром. Для цього можна

скористатися функцією розбиття вікна редагування коду на дві частини, перемістивши маркер розділення, який розташований над смугою пролистування, на потрібну позицію. Вигляд маркера наведений на рис. 1.9.

Рис. 1.9. Маркер розділення вікна редагування

Використання такого розділення дозволяє керувати переглядом коду у двох вікнах, наприклад можна скопіювати фрагмент коду з однієї частини і розмістити його в другій без додаткових переміщень по самому коду.

Мова VBA також володіє вбудованою системою перевірки. Якщо при написанні програми були допущені синтаксичні помилки, то редактор повідомить нам про це і запропонує виправити їх в момент запуску програми. Якщо певні помилки будуть виявлені в процесі виконання програми (наприклад, невідповідність типів даних), то помилка буде відображена при неможливості виконати відповідний рядок програми. При цьому мова VBA пропонує користувачу усунути помилку натиснувши кнопку *Debug (Налагодження)* і перемістивши його на рядок з помилкою. Приклад повідомлення про помилку наведений на рис. 1.10.

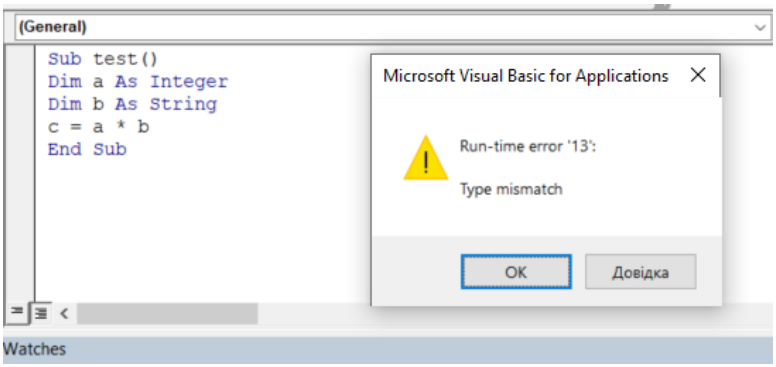


Рис. 1.10. Вікно помилки *Невідповідність типів даних*

Після усунення невідповідностей можна продовжувати виконання процедури.

Способи запуску програм

Як описувалось вище, запустити будь який макрос можна через вікно макросів або з вікна редактору. Для запуску процедури у вікні редактора можна скористатися кнопкою на панелі інструментів, обравши пункт меню *Run → Run Sub* або натиснути клавішу F5 на клавіатурі. Але перелічені способи більше підходять для запуску макросу самим розробником, а не користувачем. Для користувача бажано передбачити більш зручні способи запуску програми.

Для того, щоб створити кнопку на панелі інструментів і призначити їй макрос, необхідно виконати наступні дії.

1. Запустити вікно параметрів відповідної програми: *Файл → Параметри*.
2. Обрати пункт *Налаштування стрічки* або *Налаштування панелі швидкого доступу* у залежності від того, де планується розмістити кнопку.
3. В полі *Обрати команди* виділити пункт *Макроси*. В результаті в нижньому вікні буде відображений список всіх доступних макросів. Обрати з них потрібний і натиснувши кнопку *Додати*, помістити його в заплановане місце.

Приклад створення кнопки на панелі інструментів наведений на рис. 1.11.

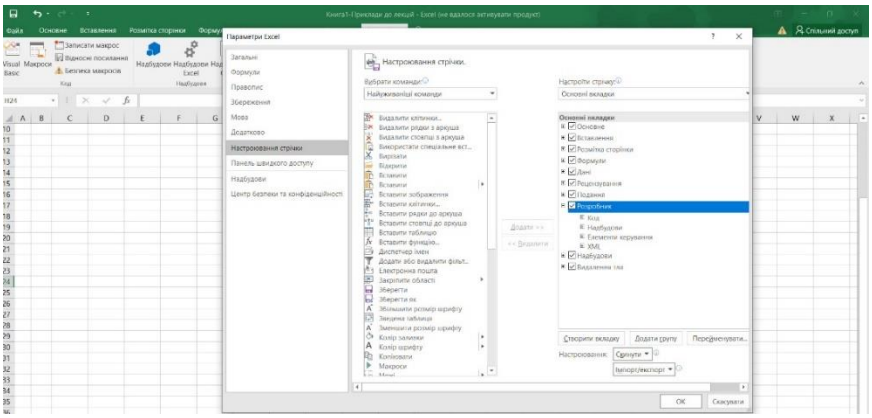


Рис. 1.11. Створення кнопки на панелі інструментів

4. Якщо при підсвіченому макросі у списку вкладок або панелі швидкого доступу натиснути кнопку *Змінити* або *Перейменувати*, то можна змінити зображення на кнопці макросу.
5. Після закриття вікна параметрів нова кнопка, що запускає макрос, буде відображена у вказаному місці.

В MS Word можна призначити комбінацію клавіш для запуску макросів. Для цього необхідно в пункті *Налаштування стрічки* меню *Параметри* натиснути на кнопку *Налаштування* після слів *Комбінація клавіш*. У вікні, що з’явиться знайти пункт *Макроси*, обрати потрібний макрос зі списку запропонованих і призначити йому комбінацію клавіш.

Контрольні питання і завдання

1. Назвіть основні способи створення макросів на мові VBA.
2. В чому переваги створення макросів у режимі протоколювання?
3. Яким чином можна перейти в режим редагування макросів?
4. Які прийоми для налагодження програми передбачені в мові VBA?
5. Перелічіть основні способи запуску програм на мові VBA.

Практичні завдання

1. За допомогою функції *Запис макросу* в MS Word створіть макрос, що дозволяє форматувати документ під задані вимоги.

2. За допомогою функції *Запис макросу* в MS Excel створіть макрос, який формує шаблон таблиці для заповнення розкладу.
3. Відкрийте в редакторі будь який зі створених макросів. Виконайте різні прийоми налагодження програми, прослідкуйте за значеннями оголошених змінних.

Індивідуальні практичні завдання

По заданим параметрам *a* і *b* обчисліть параметр *c*.

Варіант	a	b	c
1	12	8	$8a - b^2$
2	6	15	$5a + b/4$
3	4	7	$4(a + b^3)$
4	14	6	$12a^2 - 8b$
5	5	16	$A^3/3 - 3b$
6	3	18	$(a^3 - b)/2$
7	13	7	$(a - b)/(a + b)$
8	15	6	$A(b^2 + 16)/5$
9	8	15	$2(a^2/4 - b)$
10	9	17	$4(b + a) - a^2$

Тести

1. Процес розробки програм, що призначені для автоматизації офісної діяльності являють собою:
   A. Програмування;
   B. Офісне програмування;
   C. Програмування макросів.
2. Ціллю розробки в офісному програмуванні є:
   A. Програма;
   B. Документ;
   C. Операційна система.
3. Середовище розробки орієнтоване в першу чергу:
   A. На програмістів;
   B. Користувачів;
   C. Тестувальників.
4. Мова програмування, що вбудована в пакети прикладних програм Microsoft Office, називається:
   A. VBA;
   B. C#;
   C. Html.
5. Мова об’єктно-орієнтованого програмування – це:
   A. Метод програмування, при використанні якого головними елементами програм являються об’єкти;

B. Методологія розробки програмного забезпечення, що заснована на представленні програми у вигляді ієрархічної структури блоків;
C. Набір правил і обмежень, що направлені на побудову великих програмних систем з тривалим часом життя.

6. Об’єднання даних і властивих їх процедур обробки в одному об’єкті – це:
   A. Інкапсуляція;
   B. Спадкування;
   C. Поліморфізм.
7. Закінчена послідовність команд мови – це:
   A. Вихідний код;
   B. Програма;
   C. Процедура.
8. Самостійна програмна одиниця мови VBA, яка включає різні типи команд, починається із ключового слова *Sub* і унікального імені процедури – це:
   A. Процедура-функція;
   B. Процедура-підпрограма;
   C. Модуль.
9. Послідовність команд, які дозволяють автоматично виконувати визначену послідовність дій називається:
   A. Модуль;
   B. Макрос;
   C. Вихідний код.
10. Іменований елемент, який зберігає постійне незмінне значення в процесі виконання програми називається:
    A. Константи;
    B. Оператор;
    C. Змінні.
11. Комбінація ключових слів, операторів, змінних і констант, результатом яких є рядок, число або об’єкт називається:
    A. Операції мови;
    B. Вирази;
    C. Коментар.
12. Для позначення математичних операцій використовуються:
    A. Оператори порівняння;
    B. Оператори присвоєння;
    C. Арифметичні операції.

ТЕМА 2. ОСНОВИ МОВИ VBA

2.1. Вбудовані функції мови VBA

Перетворення типів даних

Функція являє собою особливий вид операцій, які виконують дії над виразами і генерують підсумкове значення, яке вставляється в місці виклику цієї функції.

У мові VBA передбачено декілька десятків вбудованих функцій, використання яких дозволяє ефективно працювати на мові VBA. Отримати повний алфавітний перелік вбудованих функцій можна через виклик довідки в середовищі розробки натисканням клавіші F1 або відповідної кнопки на панелі інструментів.

В даному навчальному посібнику функції розглядаються виходячи з їх функціональності. Першу групу представляють функції перетворення типів даних, які дозволяють змінити початково оголошений тип даних.

Функція `Val()` дозволяє перетворювати рядкові змінні в числові. При неможливості перетворення функція повертає значення 0. Ця функція використовується в мові VBA доволі часто, так як всі дані, які вводяться користувачами через форми, сприймаються саме як рядкові значення. Робота даної функції полягає у зчитуванні символів зліва направо. При досягненні першого нечислового символу (за винятком крапки, яка розділяє цілу частину від дробової) робота функції зупиняється і повертається отримане на цей момент значення, тобто якщо рядкова змінна містить декілька чисел, які розділені буквами, то буде повернене тільки перше число. Для зворотного перетворення (із числа в рядок) використовується функція `Str()`.

Функції для перетворення типів даних наведені в таблиці 2.1.

Таблиця 2.1. Функції перетворення типів даних

Функція	**Тип даних, що отримується**
Cbool	Boolean
CByte	Byte, а дробова частина заокруглюється
CChar	Char
CDate	Date
CDbl	Double
CDec	Decimal
CInt	Integer
CLng	Long
CSng	Single
CStr	String
Cvar	Variant

Для того, щоб уникнути помилок при перетворенні, можна спочатку перевірити значення на відповідність вихідного типу за допомогою функцій `IsNumeric()`, `IsDate()` і т.п. Ці функції повертають логічне значення `True`

або `False` в залежності від результату перевірки переданого їм значення. Перетворення в інші типи даних використовується не так часто.

Для приведення вихідних даних до необхідного формату використовується функція `Format`. Вона має два аргументи: 1) вираз, який потрібно форматувати, 2) задає шаблон виводу результату. Якщо шаблон не заданий, то результат буде аналогічний функції `Str`.

Приклад. Дано значення дати, яке необхідно вивести у форматі «YYYY:MM:DD».

```
Sub format_date()
Dim data As Date
Dim form_data As String
data = "01.07.2023"
form_data = Format(data, "YYYY:MM:DD")
End Sub
```

Скріншот екрану, де показано значення змінних наведено на рис. 2.1.

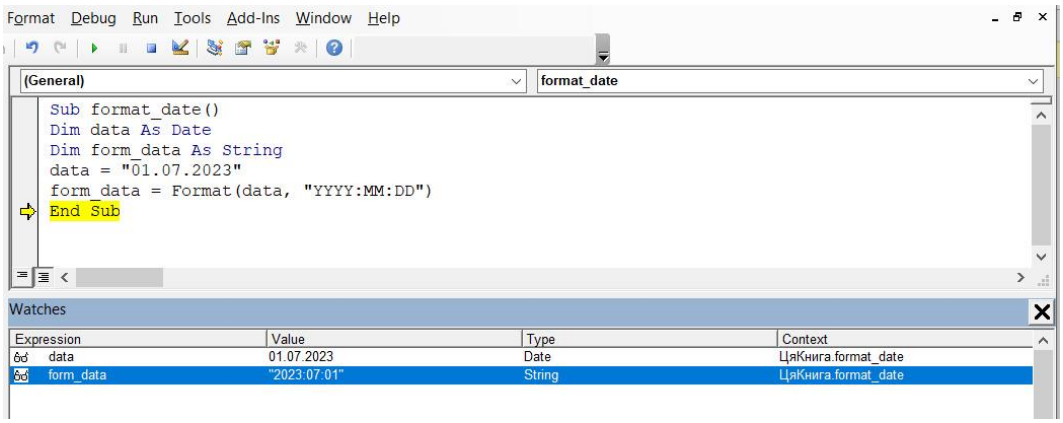


Рис. 2.1. Приклад програми для форматування дати

Математичні і тригонометричні функції

Синтаксис запису функцій складається з ключового слова, яке визначає функцію та аргументів, які записуються в дужках.

Ключові слова для деяких математичних функцій наведені в таблиці 2.2.

Таблиця 2.2. Основні математичні функції

Функція	**Математичний аналог**
Abs(x)	Модуль
Exp(x)	e^x
Fix(x) або Int(x)	Ціла частина числа (дробова частина відкидається, а не округлюється)
Log(x)	ln(x)
Sqr(x)	$\sqrt{x}$

Повний перелік можна переглянути в довідці. Крім основних ми розглянемо ще декілька математичних функцій, які можуть бути досить корисним при вивченні дисципліни.

Функція `Sgn` приймає значення –1 при x < 0, 1 при x > 0 і 0 при x = 0, тобто дана функція повертає знак аргументу.

Функція `Round (x, n)` заокруглює числа з дробовою частиною. Вона містить два аргументи: змінна та число знаків після коми. Якщо другий аргумент не заданий, то число заокруглюється до цілого.

Функція `Rnd()` повертає випадкове число із інтервалу від 0 до 1. Перед викликом функції `Rnd` ми рекомендуємо виконати команду `Randomize` для ініціалізації генератора випадкових чисел.

Основні тригонометричні функції наведені в таблиці 2.3.

Таблиця 2.3. Основні тригонометричні функції

Функція	**Тригонометричний аналог**
Atn(x)	Арктангенс
Cos(x)	Косинус
Sin(x)	Синус
Tan(x)	Тангенс

> **Аргументи тригонометричних функцій задаються в радіанах, а не в градусах!**

При програмуванні на мові VBA у програмі MS Excel доступні також всі вбудовані функції цього пакету.

Крім математичних і тригонометричних велике значення в мові VBA надається логічним операціям: `Not` (логічне заперечення), `And` (логічне І), `Or` (логічне АБО).

Для зміни послідовності логічних або арифметичних операцій використовуються тільки круглі дужки. Квадратні і фігурні дужки в мові VBA не використовуються.

Приклад. Обчислити площу сектору і відповідного сегменту кола з відомим діаметром і кутом.

```
Sub sector()
Const Pi = 3.1415926 'введемо константу - значення числа π
Dim a, r As Integer
Dim S_sect As Double, s_segm As Double
a = 120 'вихідні дані
r = 30
```

```
S_sect = (Pi * r * r * a) / 360
s_segm = S_sect - 0.5 * r * r * Sin(Pi * a / 180)
End Sub
```

Примітка. Змінна r містить значення радіуса кола, а змінна a – значення кута (в градусах). Змінюючи значення a і r, можна отримати відповідні значення площі сектору і сегменту.

Для обчислення площі сектору використовується формула

$$S_{sect} = \frac{\pi r^2 a}{360}$$

Площа сегмента обчислюється як різниця площі сектору і площі трикутника, який відсікається від сектору відповідною хордою, по формулі

$$S_{segm} = \frac{\pi r^2 a}{360} - \frac{1}{2} r^2 \sin(a).$$

Аргумент синусу в програмі вказаний в радіанах.

Скріншот програми, яка показує результати присвоєння значень змінним наведений на рис. 2.2.

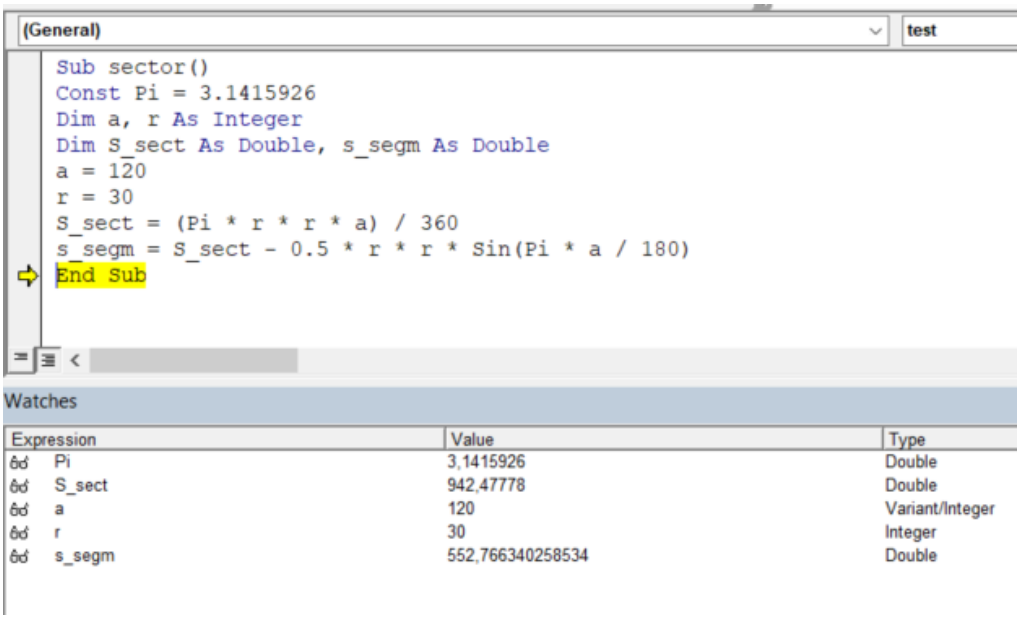


Рис. 2.2. Програма з результатами роботи

Функції дати і часу

Набором функцій, без якого обійтися практично неможливо у мові програмування VBA та і взагалі офісному програмуванні - це функції дати і часу, які після арифметичних та математичних операцій, виконуються найчастіше. Основні з них перераховані в таблиці 2.4.

Таблиця 2.4. Функції дати і часу

Функція	**Значення, що повертається**
Date	Повертає поточну системну дату
Time	Повертає поточний час по системному годиннику комп'ютера
Now	Повертає поточну дату і час по системному годиннику комп'ютера
Day() Month() Year() Hour() Minute() Second()	Повертає ціле число, яке позначає день, місяць, рік, години, хвилини і секунди відповідно
DateAdd()	Додає до дати вказану кількість років, кварталів, місяців і т.п.
DateDiff()	Повертає різницю між датами (в одиницях від кількості років до секунд)
DatePart()	Повертає вказану частину дати (наприклад, тільки рік, тільки місяць або тільки день тижня)
DateSerial()	Створює значення дати, задаючи місяць, рік і день числовими значеннями
DateValue()	Перетворює дату з текстового формату в формат дати
MonthName()	Повертає назву місяця по його номеру

Наприклад, синтаксис функції `Datediff` наступний:

```
Datediff (<одиниці виміру>, <дата1>, <дата2>)
```

Аргумент `Одиниці виміру` вказує, в яких одиницях часу необхідно повернути значення різниці між двома датами. Він може приймати наступні значення: «*yyyy*» – рік, «*q*» – квартал, «*m*» – місяць, «*y*» – день року, «*d*» – день, «*w*» – день тижня, «*ww*» – тиждень, «*h*» – година, «*n*» – хвилина, «*s*» – секунда. Всі ці позначення використовуються і в інших функціях дати та часу, в яких необхідно ввести одиницю виміру.

Ряд функцій в мові VBA не мають аргументів. Наприклад, функція `Time` повертає поточний час; функція `Now` повертає поточну дату і час.

Функція дати і часу `Timer` повертає кількість секунд, що пройшли з початку доби. Вона може використовуватися для відліку часу, що пройшов з початку певної події, якщо запам'ятати час початку події і порівняти поточну кількість секунд зі збереженням значення.

Додавання і віднімання від поточної дати певного часового інтервалу здійснюється за допомогою функції `DateAdd`, яка має наступний синтаксис:

```
DateAdd (<одиниці виміру> <період> <дата>)
```

Аргумент `Період` може приймати як додатні, так і від'ємні значення. В результаті буде повертатися дата більша або менша від поточної відповідно.

Приклад. Обчислити, скільки днів залишилось до 50-річчя.

```
Sub dniv_do_50()
Dim a As Integer
Dim d_int As Date, today As Date, birth As Date
Dim m As Variant
today = Now 'поточна дата
birth = "28.05.1973" 'вихідні дані (дата народження)
d_int = DateAdd ("yyyy", 50, birth) 'щоб замість 50 років ввести інше значення, необхідно змінити одиниці виміру і кількість періодів
a = DateDiff ("d", today, d_int) 'результат обчислення різниці між поточною датою і необхідною в днях
End Sub
```

Примітка. Змінна *today* містить значення поточної дати, *birth* – дату народження. Для обчислення дати 50-річчя використовується функція DateAdd. Результат обчислень міститься в змінній *a*.

Скріншот програми, що відображає результати присвоєння значень змінним, наведений на рис. 2.3.

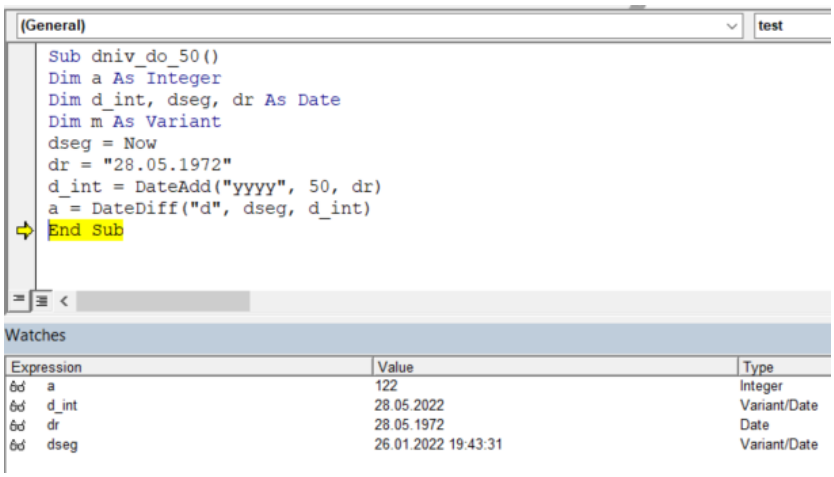


Рис. 2.3. Програма з результатом роботи

Контрольні питання і завдання

1. Для чого використовуються функції перетворення типів даних?
2. Вивчіть функцію `Format`. Яким чином задається формат виведення результатів?
3. У чому відмінність функцій `Int (x), Fix (x)` і `Round (x)`?
4. Які з функцій можна назвати взаємозамінними?

Практичні завдання

1. Напишіть програмний код, який повертає синус кута, заданого користувачем в градусах.
2. Напишіть програмний код, який повертає косинус кута, заданого користувачем в градусах.
3. Напишіть програмний код, який виводить поточний час в наступному форматі: «місяць: дата: рік: години: хвилини».
4. Напишіть програмний код, який виводить поточний час в наступному форматі: «години: хвилини: секунди рік: місяць: дата».
5. Напишіть програмний код, який видає фінальну комбінацію лотереї 5 з 36.
6. Напишіть програмний код, який видає фінальну комбінацію лотереї 6 з 45.
7. Напишіть програмний код, який розраховує кількість декад (прим.: період часу тривалістю десять днів) між двома заданими датами.
8. Напишіть програмний код, який розраховує кількість вівторків між двома заданими датами.

2.2. Керуючі конструкції мови

Умовний оператор

Більшість алгоритмів є нелінійними. Для розгалуження алгоритмів можуть використовуватися оператори умови (або умовного переходу) і циклу.

При розробці програм часто виникає необхідність вибору дій в залежності від певних умов. У мові VBA таку функцію виконує умовний оператор `If...Then...Else...End If.`

Даний оператор може бути записаний повною конструкцією або одним із варіантів скороченої інструкції. В самому простому вигляді даний оператор має однорядкову структуру:

`If <умова> Then <дія>.`

Умова - це логічний вираз, який може приймати значення `True` (істина) або `False` (брехня).

Оператор перевіряє умову на істинність і при отриманні результату `True` виконує дію, в протилежному випадку дія ігнорується. Умова, що перевіряється містить один оператор, а дія або блок дій повинен бути розміщений в один

рядок. Всі дії, розміщені на інших рядках, до умовного оператору відноситись не будуть.

Приклад. Визначити, чи являється значення змінної x додатнім числом.

```
Sub perevirka()
Dim x As Single
Dim y As String
x = 12
If x > 0 Then y = "додатне"
End Sub
```

Для того щоб розширити кількість дій, які потрібно виконати при істинності заданої умови, необхідно після перечислення всіх операторів дій позначити закінчення умовного оператора командою `End If`.

Приклад. Для додатних значень змінної x вирахувати корінь і логарифм.

```
Sub perevirka()
Dim x As Single, z As Single
Dim y As String
x = 16
If x > 0 Then
y = "додатне"
z_sqrt = Sqr(x)
z_ln = Log(x)
End If
End Sub
```

Скріншот програми, яка відображає результати присвоєння значень змінним наведений на рис. 2.4.

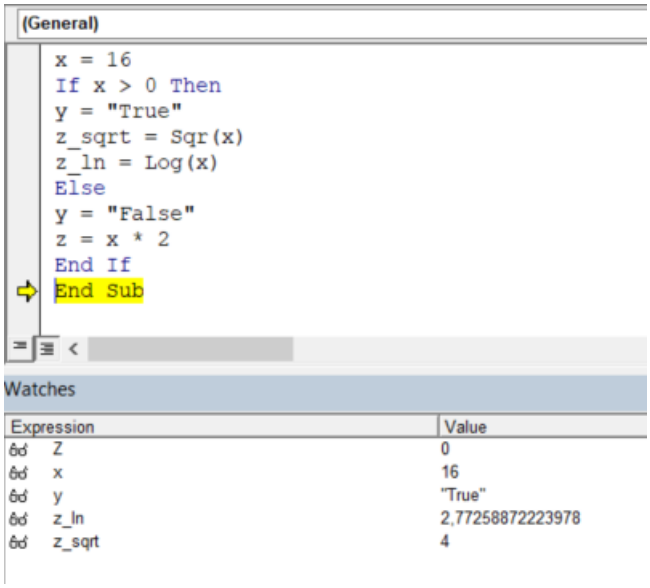


Рис. 2.4. Програма з результатами роботи

У випадку, коли при невиконанні умови також повинна виконуватися певна дія, використовується наступна форма оператора: `If <умова> Then <дія1> Else <Дія2>`. Вся команда повинна бути прописана одним рядком або повинен використовуватися оператор кінця умови `End If`. В останньому випадку команда `Else` повинна розташовуватися на окремому рядку. При такому синтаксисі всі команди від оператора `Then` до оператора `Else` будуть відноситись до дії, що виконується в разі істинності умови, а від оператора `Else` до оператора `End If` - в разі хибності умови.

Приклад. Визначити, чи являється значення змінної x додатнім або від'ємним.

```
Sub perevirka()
Dim x As Single
Dim y As String
x = -12
If x > 0 Then y = "додатне" Else y = "від'ємне"
End Sub
```

Скріншот програми, що відображає результат присвоєння значень змінним наведений на рис. 2.5.

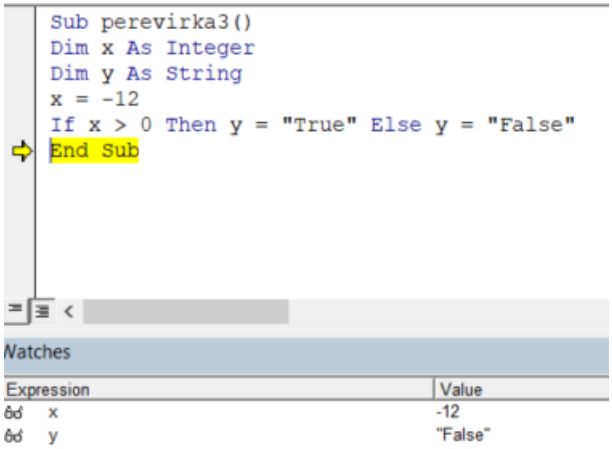


Рис. 2.5. Результат роботи програми

Приклад. Для додатних значень змінної x обчислити корінь і логарифм, для від'ємних – піднесення до степені.

```
Sub Perevirka2()
Dim x As Single
Dim y As String
Dim z_sqrt as Single, z_ln as Single, z as Single
x = -6
If x > 0 Then
y = "додатне"
z_sqrt = Sqr(x)
z_ln = Log(x)
Else
y = "від'ємне"
z = x^2
End If
End Sub
```

Умовний оператор може бути розміщений ієрархією, коли один оператор `If… Then…` розміщений всередині іншого оператора `If…Then….` Умови виконуються суворо послідовно в тому порядку, в якому вони написані.

Приклад. Створити програмний код, що дозволяє знаходити рішення рівняння другого степені.

Примітка. *Для розв'язку рівняння другої степені необхідно обчислити дискримінант, порівняти його значення з нулем, розрахувати значення коренів рівняння, якщо вони наявні. Вихідними даними є параметри a, b, c, які задають квадратне рівняння.*

```
Sub kv_r()
Dim a As Integer, b As Integer, c As Integer
Dim d As Integer
Dim k1 As Double, k2 As Double
Dim vidpovid As String
a = 1 'вихідні дані (параметри квадратного рівняння)
b = 1
c = -2
d = b ^ 2 - 4 * a * c 'обчислення дискримінанту
If d < 0 Then
otvet = "Рівняння не має коренів"
Else
If d = 0 Then
k1 = -b / (2 * a) 'обчислення коренів, якщо дискримінант
дорівнює нулю
vidpovid = "Рівняння має два однакових корені, рівних" &
k1
Else
k1 = (-b + Sqr(d)) / (2 * a) 'обчислення коренів, якщо
дискримінант більше нуля
k2 = (-b - Sqr(d)) / (2 * a)
vidpovid = "Рівняння має два корені, рівні " & k1 &
" i " & k2
End If
End If
MsgBox (vidpovid) 'виводить результати обчислень
End Sub
```

Скріншот результату роботи програми наведений на рис. 2.6. Для вирішення інших квадратних рівнянь з використанням даного програмного коду необхідно змінити параметри квадратного рівняння, які визначені в тілі програми після оголошення змінних.

Крім умовного оператора функції умовного вибору значень можуть бути реалізовані за допомогою вбудованої функції `Iif`: `Iif (<умова>, <вираз1>, <вираз2>)`. Вона повертає значення `<вираз1>` при істинності умови і значення `<вираз2>` – при хибності.

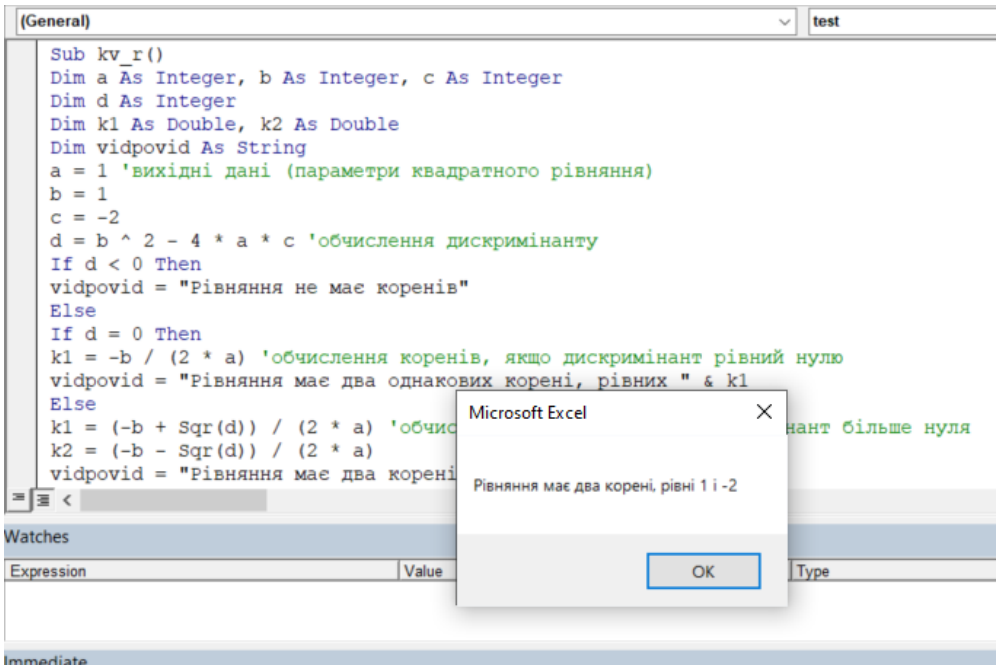


Рис. 2.6. Результат роботи програми

Оператор вибірки

При необхідності вибору дій порівняння однієї змінної з різними варіантами значень використовується оператор вибору. Синтаксис даного оператора:

```
Select Case <умовна змінна>
Case <вираз1>
<дія1>
Case <вираз2>
<дія2>
Case <вираз3>
<дія3>
…
Case <виразN>
```

```
<діяN>
Case Else
<дія0>
End Select
```

При виконанні оператора <умовна змінна> послідовно порівнюється з кожним із перелічених виразів. Якщо вона приймає значення <вираз1>, то виконується блок <дія1>, для <вираз2> виконається блок <дія2> і т. д. Якщо значення умовної змінної не відповідає жодному з перелічених виразів, то виконується блок <дія0>.

Якщо для декількох варіантів значень умовної змінної необхідно виконати один і той самий блок дій, то їх значення перелічуються після оператора Case через кому або інтервалом через оператор To.

Приклад. Кваліфікувати землетрус виходячи із шкали інтенсивності землетрусів в балах.

```
Sub Zemletrus()
Dim zeml as Integer
Dim m as String
Zeml = 6
Select Case zeml
Case 1
m = "Не відчувається"
Case 2, 3
m = "Слабкий"
Case 4
m = "Інтенсивний"
Case 5 To 7
m = "Сильний"
Case 8
m = "Руйнівний"
Case Else
m = "Катастрофа"
End Select
End Sub
```

Оператори циклу

Цикл – це такий оператор або послідовність (блок) операторів, які можуть багаторазово виконуватися в ході програми. Кількість повторів може бути задано на початку або визначатися додатковою умовою.

Інструкцією ***циклу з наперед заданим числом повторень*** є `For`.

Синтаксис оператора наступний:

```
For <лічильна змінна> = <початок циклу> To <кінець циклу> Step <крок>
<Оператори дій>
Next <лічильна змінна>
```

Кількість повторень циклу визначається числом вимірювань лічильної змінної від початкової позиції до кінцевої із заданим кроком. Крок є необов'язковим компонентом синтаксису (за замовчуванням передбачений крок, що дорівнює одиниці). Порядок виконання процедури наступний: рахунковій змінній присвоюється початкове значення, перевіряється чи досягнуто кінцеве значення циклу і виконується блок дій. В іншому випадку процедура завершується, при досягненні оператора `Next` лічильна змінна збільшується на величину кроку і цикл повторюється.

Приклад. Написати програму піднесення числа в задану степінь без використання відповідного оператора.

```
Sub Stepin ()
Dim i As Integer, k As Integer, n As Integer, st_n As Integer
k = 4 'степінь, в яку необхідно піднести число
n = 12 'число, яке необхідно піднести в ступінь
st_n = 1
For i = 1 To k
st_n = st_n * n 'результат обчислень
Next
End Sub
```

> **Не використовуйте всередині циклу в якості змінної лічильну змінну, так як це впливає на виконання циклу!**

Параметр кроку може задаватися будь-яким числом, в тому числі дробовим або від’ємним. В останньому випадку лічильна змінна буде

зменшуватись, а цикл триватиме до тих пір, поки лічильна змінна не виявиться меншою за кінцеве значення.

Цикли можуть ієрархічно вбудовуватися один в інший.

При необхідності достроково зупинити виконання циклу використовується команда `Exit For`. Якщо при виконанні процедури зустрічається цей оператор, то виконання циклу негайно зупиниться і програма перейде до виконання дій, що йдуть за циклом. Зазвичай дана команда розміщується всередині умовного оператора.

Приклад. Написати програмний код, який виводить послідовність чисел в межах першого десятка – кожне число, від 10 до 100 – десятками, від 100 до 1000 – сотнями.

```
Sub poslid ()
Dim i As Integer, j As Integer, k As Integer
Dim a As String, posl As String
For i = 1 To 3 'цикл, що задає, скільки десятків чисел буде представлено в послідовності
For j = 1 To 9 'цикл, що задає послідовність чисел від 1 до 9
Select Case i
Case 1
k = j 'для першого десятка дані будуть виводитися послідовно
Case 2
k = 10 * j 'для другого десятка дані будуть виводитися десятками
Case 3
k = 100 * j 'для третього десятка дані будуть виводитися сотнями
End Select
a = Str (k) 'переклад числового значення в рядкове
posl = posl & a 'послідовність з отриманих значень
Next j
Next i
MsgBox (posl) 'виводить результат
End Sub
```

Скріншот з результатом роботи програми, що відображає повідомлення з шуканою послідовністю (рис. 2.7).

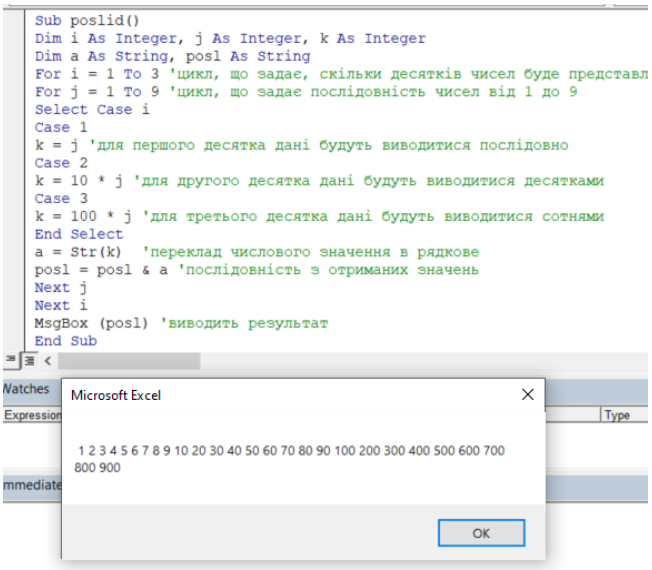


Рис. 2.7. Результат програми: шукана послідовність

Цикли з заздалегідь невідомою кількістю повторень бувають з передумовою і післяумовою. Дані цикли повторяються, поки логічна умова перевірки буде мати істинне значення. Перевірка цієї умови може здійснюватися до або після виконання блоку дій циклу.

Цикли з передумовою мають наступний синтаксис:

```
Do While <умова>
 <дія>
Loop
```

Виконання даного оператора здійснюється в наступному порядку: перевіряється істинність логічної умови, при значенні *True* виконується блок дій, по досягненню оператора `Loop` цикл повертається до оператора `Do While` і виконується знову.

Існує ще один варіант запису циклу з передумовою:

```
Do Until <умова>
<дія>
Loop
```

Відмінність даного варіанту від попереднього полягає у перевірці логічної умови не на істинність, а на хибність. Як тільки буде досягнуто істинне значення, повторення циклу зупиниться.

Цикли з післяумовою мають наступний синтаксис:

```
Do
```

```
<дія>
Loop While <умова>
```

або

```
Do
<дія>
Loop Until <умова>
```

Дані цикли базуються на перевірці істинності і хибності умови відповідно. На відміну від всіх попередніх ці цикли обов'язково виконаються хоча б один раз.

Для примусового переривання циклу використовується команда `Exit Do`.

> **Всередині умовних циклів необхідно прописувати дії, які впливають на стан логічної умови, інакше цикл може виконуватись нескінченно!**

Приклад. Розрахувати суму парних і непарних цифр заданого числа.

```
Sub suma_cifr()
Dim a As Integer, b As Integer, k As Integer, n As Integer
Dim cislo As Double
Dim vidpovid As String
cislo = 123456 'вихідне число
Do While cislo > 0 'цикл, що продовжується, поки не закінчяться цифри в числі
k = cislo Mod 10 остання цифра числа, яка отримана як залишок від ділення числа на 10
n = n + 1
If n Mod 2 = 0 Then a = a + k Else b = b + k
'сумує окремо парні і непарні цифри
cislo = cislo \ 10 'результат цілочисельного ділення
числа на 10 (зменшує число на один розряд, відкидаючи останню цифру)
Loop
If n Mod 2 = 0 Then
vidpovid = "Сума парних цифр рівна " & b & ", а
```

```
непарних цифр – "& a
Else vidpovid = "Сума парних цифр рівна " & a & ",
а непарних цифр " & b
'парні і непарні цифри числа рахуються зліва направо
MsgBox (vidpovid) 'виводить відповідь
End Sub
```

Примітка. *Вихідне число задано у змінній cislo. Спочатку сумуються цифри через одну з права наліво. Потім встановлюється, чи парна або ні загальна кількість цифр в числі. В залежності від цього виводиться загальний результат.*

Скріншот програми, яка відображає проміжкові результати присвоєння значень змінним наведений на рис. 2.8.

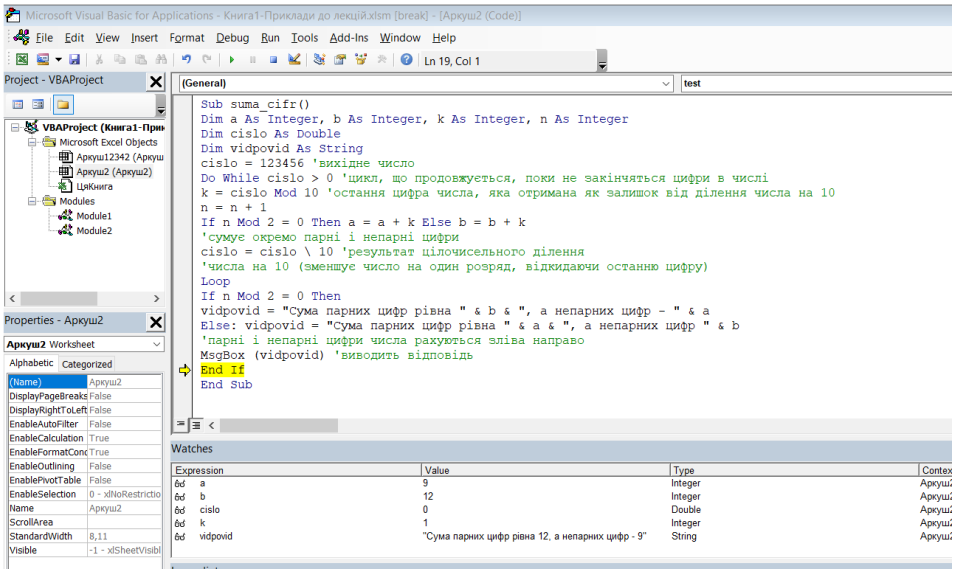


Рис. 2.8. Проміжковий результат роботи програми

Контрольні питання і завдання

1. Що таке умова?
2. В чому різниця одно- і багаторядкової форми запису умовного оператора?
3. В яких випадках можна замість оператора умови використовувати функцію `Iif`?
4. Наведіть структуру умовного оператора, яка може бути замінена оператором вибору.
5. Назвіть основні види циклів і охарактеризуйте їх.
6. Яку функцію виконує значення, що вказане в циклі `For` після оператора `Step`? В яких випадках воно може не вказуватися?
7. Як розрахувати кількість повторень циклу `For` виходячи зі значень початку, кінця і кроку циклу?
8. В яких випадках умовний цикл обов’язково буде виконаний хоча б один раз? А в яких не виконаний жодного разу?

Практичні завдання

1. Магазином встановлена знижка 10 % для всіх покупців по вихідним дням. Напишіть відповідний умовний оператор.
2. Керівництво театру прийняло рішення про надання 50%-ї знижки на білети для багатодітних сімей. Напишіть відповідний умовний оператор.
3. Мобільним оператором для роумінгу встановлена по хвилинна тарифікація першої хвилини розмови вартістю 3 грн. і по секундна кожної наступної хвилини, вартістю 5 грн./хв. Напишіть відповідний умовний оператор.
4. Процент доходу по вкладу в банку залежить від розміру вкладу і терміну депозиту. Для короткострокових вкладів (менше одного року) ставка рефінансування +2 %, для довгострокових (більше одного року) – ставка рефінансування +10 %. Для крупних вкладів (від 50000 грн. і вище) передбачений преміальний відсоток (+3 % до звичайної ставки). Напишіть відповідний умовний оператор.
5. Напишіть оператор вибору назви пори року в залежності від номеру місяця.
6. Напишіть оператор вибору для перетворення оцінок із дванадцяти бальної шкали в п’ятибальну.
7. Напишіть цикл, який обчислює факторіал заданого числа.
8. Напишіть цикл, який обчислює послідовність Фібоначі, яка складається із заданого числа елементів.
9. Напишіть цикл, який обчислює зведення відомого числа в задану степінь.
10. Замініть цикли завдань 7–9 умовними циклами з перед- або постумовою.

2.3. Робота з масивами на мові VBA

Поняття і типи масивів

Масив – набір компонентів (елементів масиву), які розташовані в пам'яті безпосередньо один за одним. В результаті обробка цих даних здійснюється як єдине ціле. Змінні, що утворюють масив, називають елементами. Звернення до певного елементу здійснюється через вказівку імені масиву та індексу конкретного елемента.

За кількістю використовуваних вимірів, необхідних для визначення елемента в масиві, виділяють одновимірні і багатовимірні масиви.

Одновимірний масив - це масив з одним виміром. В загальному випадку він є простим списком.

Багатовимірні масиви мають безліч вимірів. Прикладом двовимірного масиву є звичайна таблиця.

В залежності від задання розмірності при оголошенні масиву розрізняють статичні і динамічні масиви.

Якщо розмірність задана при оголошенні, то масив називається **статичним**, а його розмір залишається фіксованим протягом дії усієї програми.

Динамічні масиви мають змінну кількість елементів і можуть збільшуватися або зменшуватися в процесі роботи програми.

Способи ініціалізації

Оголошуються масиви так само, як і змінні, з тією лише різницею, що в дужках вказується розмірність. Синтаксис оголошення виглядає наступним чином:

```
Dim A (<розмір>) As <тип даних>
```

Розмірність може задаватися наступним чином:

- (6) – одновимірний масив з сімома елементами (так як в загальному випадку нумерація масивів в мові VBA починається з нуля);
- (1 To 10) – одновимірний масив з десятьма елементами;
- (1 To 4, 1 To 4) – двовимірний масив розміром 4*4;
- () – динамічний масив, розмірність якого буде задана при ініціалізації.

У мові VBA є можливість створювати і використовувати одновимірні і багатовимірні масиви. В багатовимірних масивах дані мають декілька вимірів, наприклад для визначення точки в тривимірному просторі необхідно ввести значення трьох вимірів: x, y і z.

За замовчуванням в мові VBA розмір кожного виміру починається з нуля. Для того, щоб масиви починались не з нуля, а з одиниці, необхідно на початку процедури помістити оператор `Option Base 1` або використовувати при оголошенні масивів конструкції з ключовим словом `To`.

Перед роботою з динамічними масивами повинна бути визначена їх розмірність за допомогою команди `ReDim`. При такому перевизначені масиву весь його вміст видаляється. Для того щоб зберегти дані в пам'яті масиву, необхідно використовувати ключове словосполучення `ReDim Preserve`.

> **В багатовимірних масивах можна змінювати тільки останній вимір! Неможна змінювати кількість вимірів!**

Приклад. Створити масив зберігання результатів сесії студентів групи.

```
Sub sesii

Dim Marks () as Integer

Option Base 1

…

ReDim Marks (25,5) 'створений масив для оцінок 25 студентів по 5 предметам

…
```

```
ReDim Preserve (25,10) 'для доповнення масиву результатами наступної сесії масив розширений до 10 стовпчиків

…

End Sub
```

Для очищення масиву використовується оператор `Erase`. У статичних масивах він очищає елементи масиву або присвоює їм нульові значення. У динамічних масивах даний оператор видаляє не тільки значення елементів, але і сам масив (тобто робить його нульової розмірності), щоб звільнити пам'ять.

Для заповнення масиву необхідно присвоїти значення кожному елементу. Для цього доцільно використовувати цикли.

Приклад**. **Заповнити масив, елементи якого розраховуються як сума номеру рядка і стовпчика, в якому даний елемент розташований.

```
Sub array_mn

Dim mn (1 To 4, 1 To 4) As Byte, i, j As Byte

For i = 1 To 4

For j = 1 To 4

mn(i, j) = i + j

Next j

Next i

End Sub
```

Крім того, є можливість присвоїти всі значення одного масиву іншому масиву даних. Для цього використовується оператор присвоєння, але в якості змінних в ньому використовуються імена масивів.

Приклад**. **Написати код для обчислення добутку двох матриць розміром 3х3.

***Примітка.** Результуюча матриця має стільки ж рядків, скільки перша, а стовпців – скільки друга. Елемент результуючої матриці є сумою добутків елементів відповідного рядка першої матриці і стовпчика другої матриці.*

```
Sub dobutok_matric()

Dim a(1 To 3, 1 To 3) As Integer, b(1 To 3, 1 To 3)

As Integer, c(1 To 3, 1 To 3) As Integer

Dim i, j, n As Integer

Dim cislo As Double

Dim vidpovid As String
```

```
    a(1, 1) = 1: a(1, 2) = 2: a(1, 3) = 3: a(2, 1) = 4:
    a(2, 2) = 5: a(2, 3) = 6
    a(3, 1) = 7: a(3, 2) = 8: a(3, 3) = 9: b(1, 1) = 1:
    b(1, 2) = 2: b(1, 3) = 3
    b(2, 1) = 1: b(2, 2) = 2: b(2, 3) = 3: b(3, 1) = 1: b(3,
    2) = 2: b(3, 3) = 3 'вихідні дані (матриці A і B)
    For i = 1 To 3 'цикл перебору по рядкам
    For j = 1 To 3 'цикл перебору по стовпчикам
    c(i, j) = a(i, 1) * b(1, j) + a(i, 2) * b(2, j) + a(i, 3)
* b(3, j) 'обчислення результуючої матриці
    vidpovid = vidpovid&Str(c(i, j)) & " " 'формування
виведення результуючої матриці
    Next j
    vidpovid = vidpovid + Chr(13) 'перехід на новий рядок для
виведення результуючої матриці
    Next i
    MsgBox (vidpovid) 'виведення результатів
    End Sub
```

Скіншот програми, яка відображає проміжкові результати присвоєння значень змінним наведений на рис. 2.9.

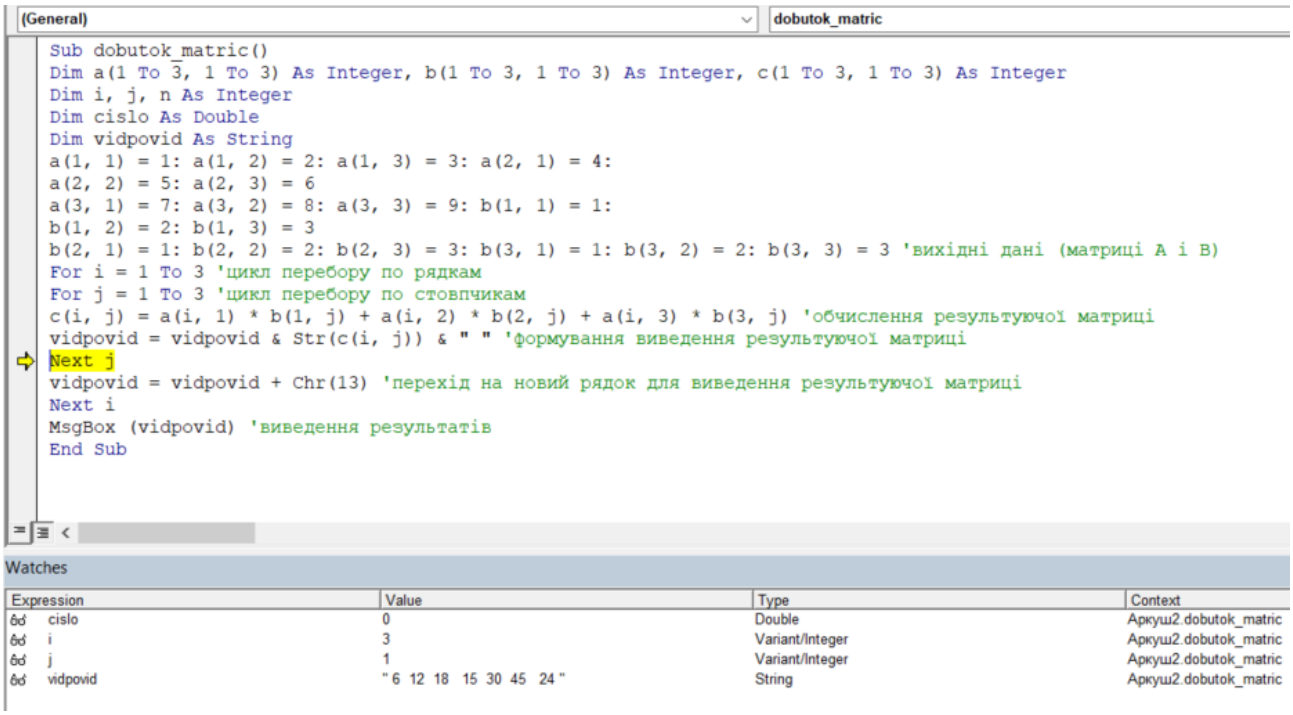


Рис. 2.9. Проміжкові результати роботи

Результат закінчення роботи програми являє собою суму парних і непарних цифр числа (рис. 2.10).

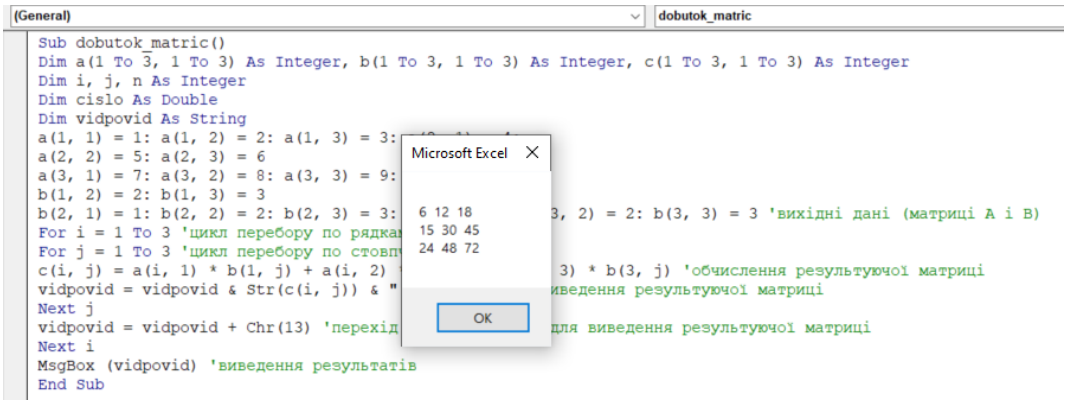


Рис. 2.10. Вікно з розрахунком результуючої матриці

Для обчислення результатів добутку інших показників необхідно замінити вихідні дані в тексті програмного коду. У процесі роботи з масивами для визначення нижньої і/або верхньої межі одновимірного масиву використовуються функції LBound і UBound відповідно.

Алгоритми сортування

Під сортуванням масиву розуміється процес перестановки його елементів з метою упорядкування їх відповідно до певних критеріїв. Для числових масивів найчастіше використовують сортування по зростанню і спаданню.

Завдання сортування часто використовується в інформаційних системах як попередній етап задачі пошуку, так як пошук в упорядкованому масиві можна виконати значно швидше, ніж в невпорядкованому.

Існують різні алгоритми сортування.

Сортування методом прямого вибору. Алгоритм сортування масиву по зростанню за допомогою даного методу може мати такий вигляд:

1. Переглядаючи масив від першого елементу, знайти мінімальний елемент і перемістити його на місце першого елементу, а перший – на місце мінімального (тобто поміняти елементи місцями);
2. Переглядаючи масив від другого елементу, знайти мінімальний елемент і помістити його на місце другого елементу, а другий – на місце мінімального;
3. Повторити описані дії для всіх елементів масиву, що залишилися (крім останнього).

Сортування бульбашковим методом. В основі алгоритму лежить ідея обміну сусідніх елементів масиву, якщо наступний елемент менше попереднього (у випадку сортування по зростанню). Кожен елемент масиву, починаючи з першого, порівнюється з наступним. Якщо він більший за наступний, то елементи міняються місцями. Таким чином, елементи з меншим значенням просуваються до початку масиву («спливають»), а елементи з більшим значенням «тонуть» (тому даний метод сортування часто називають

методом «бульбашки»). Для того щоб відсортувати масив, описаний вище процес обмінів треба повторити N - 1 раз, де N – кількість елементів масиву.

Алгоритми пошуку в масивах

При вирішенні багатьох завдань часто виникає необхідність встановити, чи містить масив певну інформацію або ні. Завдання такого типу називаються пошуком в масиві.

Для організації пошуку в масиві можуть бути використані різні алгоритми. Найбільш простий – це **алгоритм простого перебору**. Пошук здійснюється послідовним порівняння елементів масиву зі зразком до тих пір, поки не буде знайдений елемент, рівний зразку, або не будуть перевірені всі елементи. Алгоритм простого перебору застосовується, якщо елементи масиву невпорядковані.

Алгоритм перебору на практиці застосовується нечасто, так як його ефективність невисока: якщо масив невпорядкований, то потрібний елемент може бути як на початку, так і в кінці масиву. Очевидно, що чим більший масив, тим довше програма буде шукати потрібний елемент.

Бінарний пошук працює більш ефективно. якщо інформація в масиві впорядкована, то можна зробити припущення, в якій частині списку (ближче до початку або ближче до кінця) вона знаходиться, і шукати її саме там.

Суть методу бінарного пошуку полягає в наступному. Вибирається середній (за номером) елемент упорядкованого масиву (елемент з номером m), і зразок порівнюється з цим елементом. Якщо середній елемент дорівнює зразку, то задача вирішена. Якщо зразок менше середнього (передбачається, що масив впорядкований по зростанню), то шуканий елемент розташований до середнього елементу (між елементами з номерами t і $m - 1$). Якщо зразок більше середнього елемента, то шуканий елемент розташований нижче (після) середнього (між елементами з номерами $m + 1$ і b). Після визначення частини масиву, в якій може знаходитись шуканий елемент, пошук проводять в цій частині. Номер середнього елементу обчислюється за формулою

$$(b - t) / 2 + t.$$

Контрольні питання і завдання

1. Що таке масив? Коли використовуються масиви?
2. У чому відмінність статичних і динамічних масивів. Коли кожен з них використовується?
3. Яким чином можна внести дані в масив?
4. Наведіть способи сортування масивів. Чим вони відрізняються? Як можна оцінити їх ефективність?
5. Наведіть способи пошуку в масивах. Чим вони відрізняються? Як можна оцінити їх ефективність?

Практичні завдання

1. Напишіть код, який розраховує середнє арифметичне по рядках масиву і суму за стовпцями.
2. Напишіть код, який розраховує середнє геометричне за стовпцями масиву і суму по рядках.
3. Напишіть код пошуку мінімального елементу в одновимірному масиві.
4. Напишіть код пошуку максимального елементу в одновимірному масиві.
5. Напишіть код, що автоматизує процедуру сортування методом прямого вибору.
6. Напишіть код, що автоматизує процедуру сортування бульбашковим методом.
7. Напишіть код, що автоматизує процедуру бінарного пошуку.
8. Напишіть код, що автоматизує процедуру пошуку методом прямого перебору.

2.4. Робота з рядковими змінними

Синтаксис оголошення та ініціалізації

Рядкові змінні використовуються в мові VBA дуже часто. Це пов’язано, як з необхідністю роботи з текстовими даними, так і з особливістю, що введені дані користувачем через форми являються рядковими значеннями.

Для введення користувачем певної інформації зручно використовувати функцію `InputBox`. В результаті користувачу відобразиться діалогове вікно з полем для введення даних.

Для оголошення рядкової змінної використовується тип `String`. Розрізняють рядкову змінну постійну і змінної довжини. Змінні постійної довжини дозволяють економити оперативну пам'ять.

Синтаксис оголошення таких змінних наступний:

```
Dim <ім’я змінної> As String
Dim <ім’я змінної> As String * <довжина рядка>
```

У другому випадку рядкова змінна має вказану довжину. Якщо текст у рядку буде мати меншу довжину, то він буде заповнений пробілами, при більшій довжині – зайвий текст буде обрізаний.

Для отримання рядкової змінної із числового типу даних використовується функція перетворення типів даних (`Str`). Крім цього, в тексті програмного коду рядкове значення може бути присвоєно змінній шляхом розміщення його в лапках. Елементарна обробка (об’єднання) рядкових змінних може здійснюватися за допомогою рядкового оператора (&):

```
<ім’я змінної> = Str (<числове значення>)
<ім’я змінної > = "Текст"
<ім’я змінної > = "Текст1" & "Текст2"
```

Вбудовані функції обробки рядків

У мові VBA є великий набір функцій для обробки рядкових виразів.

Перша група функцій пов'язана з видаленням і додаванням пробілів в рядковій змінній. Для видалення пробілів використовуються функції `LTrim`, `RTrim` і `Trim`. Перша видаляє пробіли на початку рядка, друга – в кінці, а третя – і на початку, і в кінці змінної. Для додавання пробілів використовується функція `Space (<кількість пробілів>)`. Аргументом функції являється число, яке визначає кількість пробілів, що повертаються.

Друга група функцій пов'язана із можливістю виділення фрагменту тексту із рядкової змінної. Аргументами функцій `Left` і `Right` виступають рядкова змінна і кількість символів, які будуть обрізані, починаючи з крайнього лівого або крайнього правого відповідно.

Функція `Mid`, яка дозволяє обрати будь який підрядок та має наступний синтаксис:

```
Mid (<рядкова змінна>, <початкова позиція>,<довжина>)
```

Останній аргумент, який визначає довжину рядка, що повертається є необов'язковим і може бути не вказаним. В цьому випадку буде вирізана частина рядкової змінної від позначеної початкової позиції до кінця рядка. Корисними також будуть функції, які дозволяють визначити довжину рядка або положення певного набору символів у рядку. Для визначення довжини рядкової змінної використовується функція `Len`. Вона містить тільки один аргумент – рядкову змінну. Положення визначеного набору символів в рядку визначається за допомогою функцій `InStr` і `InStrRev`. Перша здійснює пошук з початку рядка (зліва направо), друга – з кінця (з права наліво).

Функції перетворення рядків включають перетворення рядкових символів у заголовні і навпаки: `UCase` і `LCase`. Особливістю цих функцій є те, що ними повертається тип даних `Variant`. Для повернення рядкової змінної необхідно використовувати функції `UCase$` і `LCase$` відповідно.

Функція `StrConv` дозволяє в текстовому виразі починати написання всіх слів із великої букви.

Функція `StrReverse` змінює порядок символів в текстовому рядку на протилежний.

Для заміни певної частини тексту корисно використовувати функцію `Replace`, яка має наступний синтаксис:

```
Replace (<рядкова змінна>, <текст заміщення>, <підстановочний текст>)
```

Ще одна група функцій пов'язана із поверненням текстових символів по ASCII-коду. Функція `ASC` повертає відповідний код першого символу рядкової змінної. Функція `Chr` навпаки перетворює ASCII-код у символ. Частіше всього

ця функція використовується, якщо необхідно ввести в текстову змінну символи, введення яких з клавіатури складно внести.

Для порівняння рядкових змінних можуть використовуватись оператори порівняння або функція StrComp. Остання повертає –1, 0 або 1 в залежності від результату порівняння рядків.

При цьому користувач може задати один із способів порівняння рядків: бінарне чи текстове порівняння (без врахування регістру).

Приклад. Знайти символ, який найчастіше зустрічається в рядковій змінній (крім пробілу).

Примітка. *Заданий рядок, в якому здійснюється пошук, міститься в змінній string.*

```
Sub symbol_as_string()

Dim symb As String, str As String, s As String,

symbmax As String

Dim k As Integer, i As Integer, n As Integer, j As

Integer, nmax As Integer

str = "Як умру, то поховайте Мене на могилі Серед степу широкого На Вкраїні милій, Щоб лани широкополі, І Дніпро, і кручі Було видно, було чути, Як реве ревучий. Як понесе з України У синєє море"

k = Len(str) 'довжина рядка

str = LCase(str) 'всі символи в рядку претворяются в рядкові

    Do While k > 0 'цикл перебору до тих пір, поки в рядку є символи

    symb = Left(str, 1) 'присвоює змінній перший лівий символ

For i = 1 To k 'цикл перебору по рядку

s = Mid(str, i, 1) 'послідовно обрізає символи в рядку

If s = symb Then n = n + 1 'підраховує кількість символів, рівних першому символу

Next i

If (n > nmax) And (symb <> " ") Then nmax = n:

symbmax = symb 'зберігає символ, якщо він зустрічається частіше, ніж попередній найбільш часто зустрічаючийся
```

```
string = Replace(str, symb, " ") 'видаляє всі
символи, які дорівнюють першому в рядку
k = Len(str) 'визначає нову довжину рядка
n = 0
Loop
MsgBox ("Символ "& symbmax & " зустрічається " &
nmax& " разів") 'вивід результату
End Sub
```

Примітка. *В рядку послідовно переглядаються всі символи, починаючи з лівого. Після перевірки символ із рядка видаляється. Це дозволяє послідовно зменшити кількість циклів перевірки.*

Скріншот додатку, що відображає проміжкові результати присвоєння значень змінним, наведений на рис. 2.11.

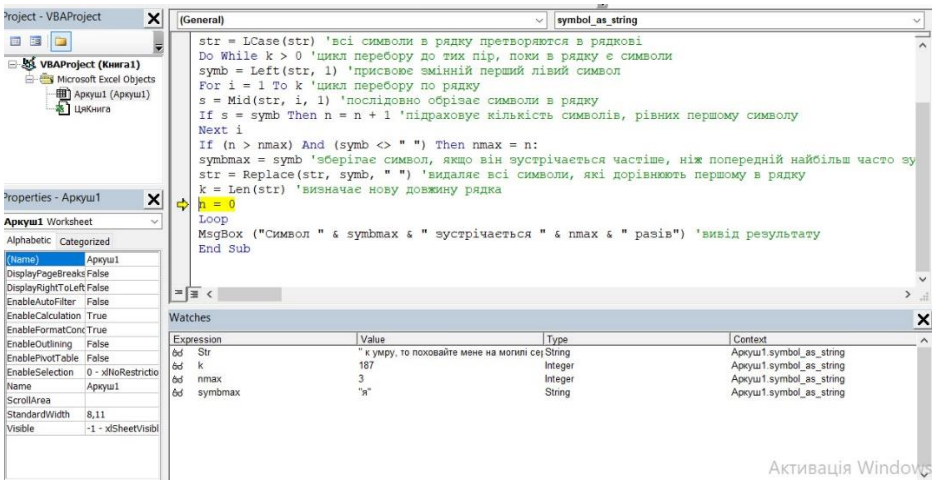


Рис. 2.11. Проміжковий результат програми

Контрольні питання і завдання

1. Назвіть особливості рядкових змінних у мові VBA.
2. Які групи функцій по обробці рядкових змінних можна виділити в мові VBA?
3. Наведіть приклади комбінації декількох функцій по роботі з рядковими змінними, які дозволяють замінити одну із інших функцій.
4. Для чого використовуються ASCII-коди в мові VBA?

Практичні завдання

1. Напишіть код, що перетворює введені ПІБ в прізвище та ініціали.
2. Напишіть код, що перетворює текст, який містить дату в форматі «ДД.ММ.РРРР», у формат ДД назва місяця РР.
3. У введеному тексті замініть всі букви Ж буквою Ч.

4. У введеному тексті замініть всі букви И буквою І.

Індивідуальні практичні завдання

1. Розрахуйте площу прямокутного трикутника, що заданий довжиною одного із катетів (a) і величиною протилежного кута (b) в градусах.

Варіант	***a***	***b***
1	3	36
2	4	50
3	5	68
4	3	17
5	4	42
6	7	25
7	6	72
8	5	56
9	6	24
10	4	62

2. Визначте дату наступного за заданою датою (date) дня тижня (weekday).

Варіант	***Date***	***Weekday***
1	12.06.2008	Понеділок
2	25.01.2017	Вівторок
3	20.12.2015	Середа
4	08.08.2020	Четвер
5	29.04.2012	П’ятниця
6	06.03.2011	Субота
7	11.07.2007	Неділя
8	21.09.2013	Понеділок
9	04.12.2016	Вівторок
10	18.05.2019	Середа

3. Обчисліть значення b на основі заданої умови і параметра a.

Варіант	**Умова**	**Значення істинне**	**Значення брехня**
1	a додатне	$b = \sqrt{a}$	$b = 4a^2 - 5a$
2	a від’ємне	$b = 120 - 3a$	$b = \ln(a)$
3	a парне	$b = \frac{a}{2} + 12$	b залишок від ділення (a/8)
4	a непарне	$b = (2a - 1)^2$	$b = \left\lvert \frac{3a}{2} - 16 \right\rvert$
5	a ділиться на 3	$b = (a + 9)/3$	$b = 1/(12^a)$
6	a не ділиться на 5	$b = \frac{\sqrt{4a}}{5}$	$b = \frac{\frac{a}{5} + 24}{2}$

7	*a* двозначне	*b* ціла частина від (*a*/7)	$b = (4a - 15)^3$
8	*a* трьохзначне	$b = (a/100)^3$	$b = \ln(a/3)$
9	складається із однакових цифр (наприклад, 22, 555 і т.д.)	$b = a^2 - 4a$	$b = e^a$
10	*a* менше 3.14	$b = \sin(a)$	$b = a/2 + 4$

4. Для інтервалу від *a* до *b* з кроком *s* обчисліть значення суми і множення елементів. Вирішіть задачу з використанням умовних і безумовних циклів.

Варіант	***a***	***b***	***s***
1	25	4	−3
2	10	40	6
3	100	35	-5
4	14	63	7
5	20	48	2
6	10	25	1.5
7	84	52	-4
8	7	42	5
9	66	40	-2
10	32	56	4

5. Виконайте необхідні перетворення рядкової змінної:

1) в заданому рядку перемістіть найдовше слово на початок;

2) в заданому рядку перемістіть найкоротше слово в кінець;

3) в заданому рядку після самого довго слова помістіть в дужках його довжину;

4) в заданому рядку після найкоротшого слова помістіть в дужках його довжину;

5) в заданому рядку знайдіть слово, в якому найчастіше зустрічається буква «а», і після нього помістіть в дужках його довжину (приклад рядка: по барабану забарабанив баран-барабанщик);

6) в заданому рядку знайдіть слова, в яких зустрічається комбінація «ар», і помістіть в кінці рядка в дужках їх кількість (наприклад рядки: Карл у Клари вкрав корали, Клара у Карла вкрала кларнет);

7) замініть в найдовшому слові рядка букви «а» буквами «и»;

8) виведіть слова рядка в зворотному порядку.

Тест

1. Особливий вид операції, яка виконує дію над виразами і генерує підсумкове значення, яке вставляється в місці появи, – це:

а) функція; б) оператор; в) константа.

2. Вкажіть ключове слово, яке дозволяє перетворити рядкові змінні в числові:

а) *Val*; б) *Dim*; в) *Str*.

3. Відмітьте, яка із функцій в мові VBA не має аргументу:
а) *Datediff*; б) *DateAdd*; в) *Timer*.
4. Відмітьте логічний вираз, який може приймати значення True (істина) або False (брехня):
а) умова; б) функція; в) оператор.
5. Вкажіть, як називається оператор, якщо для вибору дій необхідно порівняння однієї змінної з різними варіантами значень:
а) умовний оператор;
б) оператор вибірки;
в) оператор циклу.
6. Оператор або послідовність (блок) операторів, які можуть багаторазово виконуватися в ході виконання програми, – це:
а) умовний оператор;
б) оператор вибірки;
в) оператор циклу.
7. Назвіть послідовність елементів однакового типу, які мають загальне ім’я:
а) послідовність; б) масив; в) ряд.
8. Одновимірний масив являє собою:
а) рядок; б) список; в) таблицю.
9. Двовимірний масив являє собою:
а) рядок; б) список; в) таблицю.
10. Масив, розмірність якого задана при оголошенні, називається:
а) статичним; б) динамічним.
11. Вкажіть команду, за допомогою якої визначається розмірність динамічного масиву:
а) *ReDim*; б) *Option Base 1*; в) *LBound*.
12. Алгоритм сортування масиву по зростанню методом прямого вибору називається:
а) сортуванням методом прямого вибору;
б) сортуванням бульбашковим методом;
13. Алгоритм пошуку елементів в масиві методом послідовного порівняння елементів масиву зі зразком до тих пір, поки не буде знайдений елемент, що дорівнює взірцю, або не будуть перевірені всі елементи, – це:
а) бінарний пошук;
б) алгоритм простого перебору.
14. Змінні, що необхідні для роботи з текстовими даними, називаються:
а) цілочисельними; б) числовими; в) рядковими.

ТЕМА 3. КОРИСТУВАЦЬКІ ФОРМИ І ЕЛЕМЕНТИ КЕРУВАННЯ

3.1. Діалогові вікна в VBA

Функції MsgBox і InputBox

В число важливих елементів будь-якої програми входить її інтерфейс.

Інтерфейс користувача – це сукупність засобів і методів, що забезпечують взаємодію користувача з пристроєм в загальному і конкретному додатку вчасності.

Здійснити таку взаємодію можна за допомогою спеціальних діалогових вікон. У програмах на мові VBA передбачено два види вбудованих діалогових вікон – вікна повідомлень і вікна введення.

Функція `InputBox()` виводить на екран повідомлення і поле введення. Синтаксис даної функції наступний:

```
InputBox (<текст повідомлення>, <заголовок>, <текст за замовчуванням>, <ліва межа>, <верхня межа>, <довідка>, <розділ>)
```

Обов'язковим є тільки аргумент тексту повідомлення. Інші аргументи можуть бути пропущені. Якщо пропущений аргумент заголовок, то в діалоговому вікні буде відображатися ім'я програми. Текст за замовчуванням відображається в полі введення в момент відкриття діалогового вікна. Якщо користувач не введе інше значення, то саме воно буде повернуто функцією. У випадку, коли даний аргумент не заповнений, поле введення відображається пустим. Аргументи `ліва` і `верхня межа` задають відстань до лівої і верхньої межі екрану відповідно. Якщо аргумент не вказаний, то вікно вирівнюється по центру екрану. Аргументи `довідка` і `розділ` задають ім'я файлу довідки і номер розділу довідкової системи, що містить відомості про дане діалогове вікно (на даний момент не практикується і не рекомендується).

Дана функція повертає значення типу *String*, що містить текст, який ввів користувач в поле введення.

Функція `MsgBox()` виводить на екран повідомлення і декілька кнопок. Синтаксис даної функції наступний:

```
MsgBox (<текст повідомлення>, <кнопки>, <заголовок>, <довідка>, <розділ>)
```

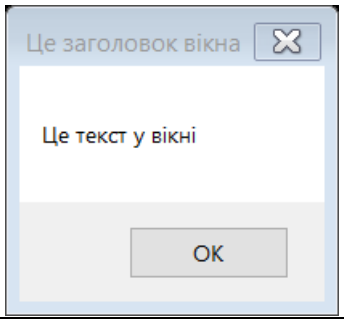
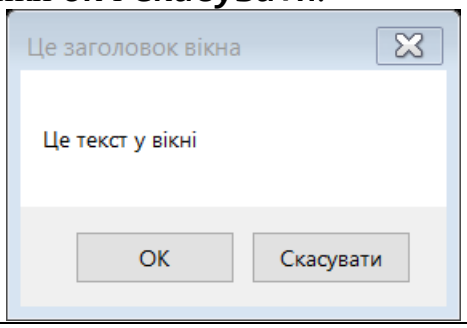
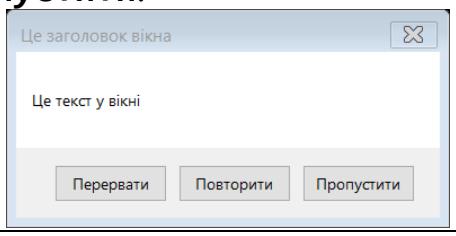
Обов'язковим є тільки аргумент тексту повідомлення. Інші аргументи можуть бути не вказані. Аргумент кнопки містить вказівку на число і тип кнопок для відображення, основну кнопку і модальність вікна повідомлення. Значення констант, які визначають число, тип кнопок і вигляд значка, наведені в таблиці 3.1. Інші аргументи аналогічні відповідним аргументам функції `InputBox`.

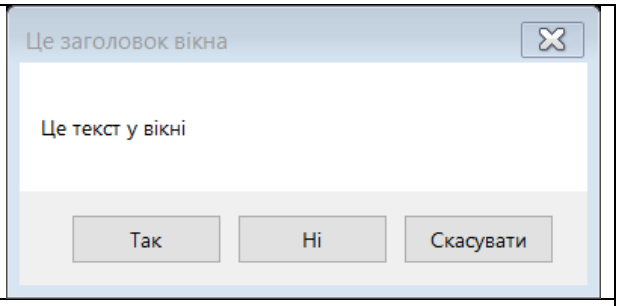
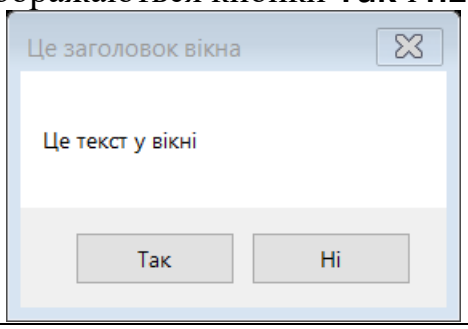
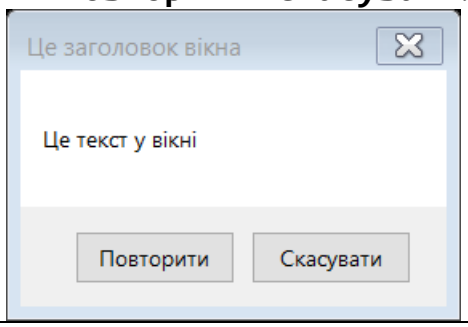
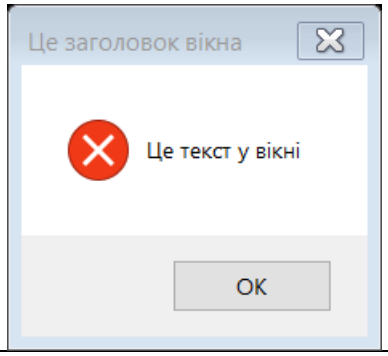
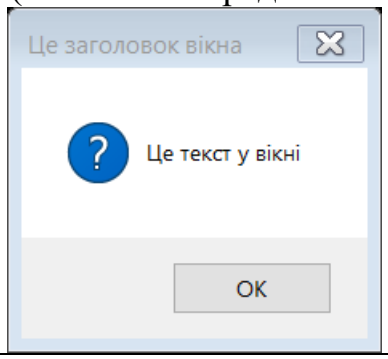
Якщо в діалоговому вікні є кнопка **Скасувати**, її натискання рівносильне натисканню клавіші ESC. Якщо в діалоговому вікні є кнопка **Довідка**, для нього доступна контекстна довідка. Однак до натискання будь-якої іншої кнопки жодне значення не повертається.

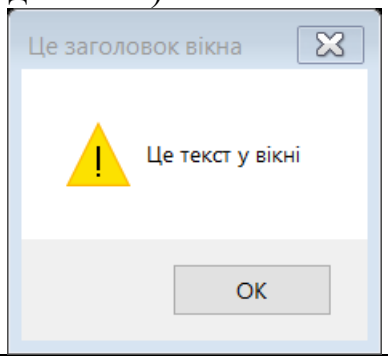
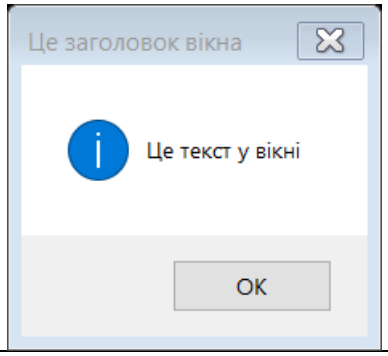
Примітка. *Для вказівки більше одного іменованого аргументу необхідно використовувати функцію* **`MsgBox`** *у виразі. Щоб пропустити деякі з позиційних аргументів, потрібно вставити відповідні коми.*

Дана функція повертає значення типу `Integer`, що містить номер натиснутої користувачем кнопки.

Таблиця 3.1. Значення аргументу Кнопки процедури `MsgBox`

Константа	Значення	Опис та код виклику
`vbOKOnly`	0	Відображається тільки кнопка **`OK`**. 
`vbOKCancel`	1	Відображаються кнопки **`OK`** і **`Скасувати`**. 
`vbAbortRetryIgnore`	2	Відображаються кнопки **`Перервати`**, **`Повторити`** і **`Пропустити`**. 
`vbYesNoCancel`	3	Відображаються кнопки **`Так`**, **`Ні`** і **`Скасувати`**.

		
vbYesNo	4	Відображаються кнопки **Так** і **Ні**. 
vbRetryCancel	5	Відображаються кнопки **Повторити** і **Скасувати**. 
vbCritical	16	Відображається значок **важливого повідомлення**. 
vbQuestion	32	Відображається значок **Warning Query** (Запит з попередженням). 
vbExclamation	48	Відображається значок **Warning**

		Message (Повідомлення з попередженням). 
vbInformation	64	Відображається значок **інформаційного повідомлення**. 
vbDefaultButton1	0	По замовчуванню активна перша кнопка.
vbDefaultButton2	256	По замовчуванню активна друга кнопка.
vbDefaultButton3	512	По замовчуванню активна третя кнопка.
vbDefaultButton4	768	По замовчуванню активна четверта кнопка.
vbApplicationModal	0	Модальність на рівні програми. Користувач повинен відповісти на повідомлення, щоб продовжити роботу в поточній програмі.
vbSystemModal	4096	Модальність на рівні системи. При очікуванні відповіді користувача на повідомлення призупиняється робота всіх програм.
vbMsgBoxHelpButton	16384	Додає кнопку **Довідка** в вікно повідомлення.
vbMsgBoxSetForeground	65536	Розташування вікна повідомлення на передньому плані.
vbMsgBoxRight	524288	Текст вирівнюється по правому краю.
vbMsgBoxRtlReading	1048576	Система читання з права наліво для івриту і арабської мови.

В значенні аргументу Кнопки може бути внесене тільки одне значення!

Перша група значень (0–5) відображає число і тип кнопок в діалоговому вікні. Друга група (16, 32, 48, 64) описує стиль значка повідомлення. Третя група (0, 256, 512) визначає активну по замовчуванню кнопку. І нарешті, четверта група (0, 4096) встановлює модальність повідомлення. При додаванні чисел в підсумкове значення аргументу *кнопки* необхідно використовувати тільки один аргумент з кожної групи. Наприклад, `Response = MsgBox(Msg, vbYesNo + vbCritical + vbDefaultButton2, Title, Help, Ctxt).`

При написанні програмного коду замість значень натиснутої кнопки зручніше використовувати константи мови VBA, що представлені в таблиці 3.2.

Таблиця 3.2. Значення констант, які визначають натиснуту кнопку процедури `MsgBox`

Константа	Значення	Натиснута кнопка
vbOK	1	OK
vbCancel	2	Відміна (Cancel)
vbAbort	3	Перервати (Abort)
vbRetry	4	Повторити (Retry)
vbIgnore	5	Пропустити (Ignore)
vbYes	6	Так (Yes)
vbNo	7	Ні (No)

Якщо в формі передбачено декілька кнопок, то при написанні програмного коду необхідно використовувати керуючі конструкції мови.

Приклад. Написати програмний код, який буде заокруглювати число із заданою точністю. Порівняйте результат з виконанням функції Round.

```
Sub zaokruglennya()
Dim a As Integer, i As Integer, pzap As Integer
Dim n As Double, m As Double
Dim c As String, d As String, s As String
vidpovid As String
c = InputBox ("Введіть число, яка необхідно заокруглити")
d = InputBox ("Введіть точність заокруглення (число
знаків після коми)", , 2)
'введення вихідних даних
n = Val(d) 'перетворення рядкової змінної в число
Do 'цикл перебору символів в числі, що заокруглюється
i = i + 1
s = Mid(c, i, 1)
If s = ","
Then pzap = i 'визначає позицію коми в числі
Loop Until (pzap <> 0) Or (i = Len(c))
```

```
    If pzap > 0
    Then 'умовний оператор, що перевіряє наявність коми в
числі
    s = Right(c, Len(c) - pzap) 'вирізає дробову частину
введеного числа
    a = CInt(n)
    d = Mid(s, a + 1, 1) 'вирізає символ, що йде за точністю
заокруглення
    m = Val(d)
    i = 0
    If m >= 5 Then
    i = 1 'якщо цифра, що йде за заокругленою, більше 5, то
остання цифра заокруглення повинна бути збільшена на 1
    End If
    d = Left(s, a)
    m = Val(d) + i
    d = LTrim(Str(m)) 'видаляє пробіл перед отриманою дробовою
частиною
    i = 0
    s = Right(d, 1) 'вирізає останній символ в отриманій
дробовій частині
    Do While (s = "0") And (Len(d) > 0) 'цикл, що видаляє
останні нулі в отриманій дробовій частині
    d = Left(d, Len(d) - 1)
    s = Right(d, 1)
    Loop
    If d = "" Then
    vidpovid = Left(c, pzap - 1) 'якщо при заокругленні
дробова частина рівна 0, то виводити тільки цілу частину
    Else If (s = "1") And (Len(LTrim(Str(m))) > a) Then
    vidpovid = Val(Left(c, pzap - 1)) + 1 'якщо ціла частина
повинна бути збільшена на 1
    Else
    vidpovid = Left(c, pzap) &d 'ціла і дробова частина з
врахування заокруглення
    End If
    Else
    vidpovid = c 'якщо коми у вихідному числі немає, то саме
число буде результатом заокруглення
    End If
    If MsgBox ("Результат заокруглення: " & vidpovid & ".
    Порівняти із заокругленням функцією Round?", vbYesNo) =
    vbYes Then 'при виведенні відповіді запитуємо необхідність
    порівняння з результатами функції заокруглення
    d = Round(c, n)
```

```
    MsgBox ("Результат заокруглення програмою рівний" &
vidpovid & "." & Chr13 & " Результат заокруглення функцією
Round " & d)
    'вивід результату
    End If
    End Sub
```

Діалогове вікно макросу з прикладу представлено на рисунку 3.1.

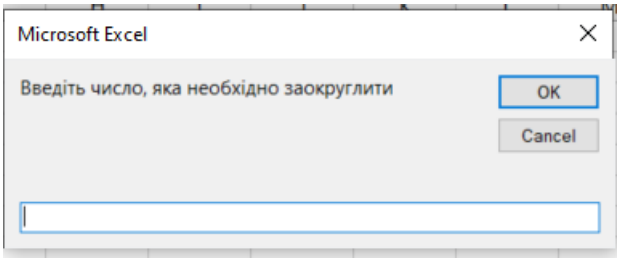


Рис. 3.1. Діалогове вікно для введення вихідних даних

Примітка. Ідея алгоритму в тому, щоб знайти кількість знаків після коми, що відповідають точності заокруглення, і по результатам порівняння наступної за ними цифри після коми залишити незмінною або збільшити на одиницю останню заокруглену цифру. Винятком можуть бути випадки, коли після коми стоять всі дев'ятки і коли число, що заокруглюється не містить знаків коми.

Використання користувацьких форм в проекті

1. **Поняття і функції користувацької форми.** У мові VBA передбачена також можливість замість вбудованих діалогових вікон використовувати форми, які створені користувачем. Їх функціональність значно вища, ніж у вбудованих функцій, так як вони можуть містити декілька полів, а також інші елементи, такі як випадаючі списки, прапорці, полоси прокрутки і т. д.

В результаті може бути створений інтерфейс, який дозволяє користувачу працювати з операціями вводу і виводу більше ефективно.

Для створення користувацьких діалогових форм використовуються об'єкти класу `Userform`. Вони складаються з вікна форми і розміщених в ній відповідних елементів. Всі елементи керування, які розташовані в формі, представляють собою об'єкти класу `Controls`. Кожна із форм або елементів керування має свій набір подій (дій) і властивостей.

Для створення нової форми необхідно виконати операцію в меню **Insert → UserForm** у вікні редактору **Visual Basic**. В результаті даної дії буде створено нову пусту користувацьку форму. По замовчуванню поряд зі створеною формою відображається вікно **Toolbox**, яке містить елементи керування, які можуть бути розташовані у формі (рис. 3.2).

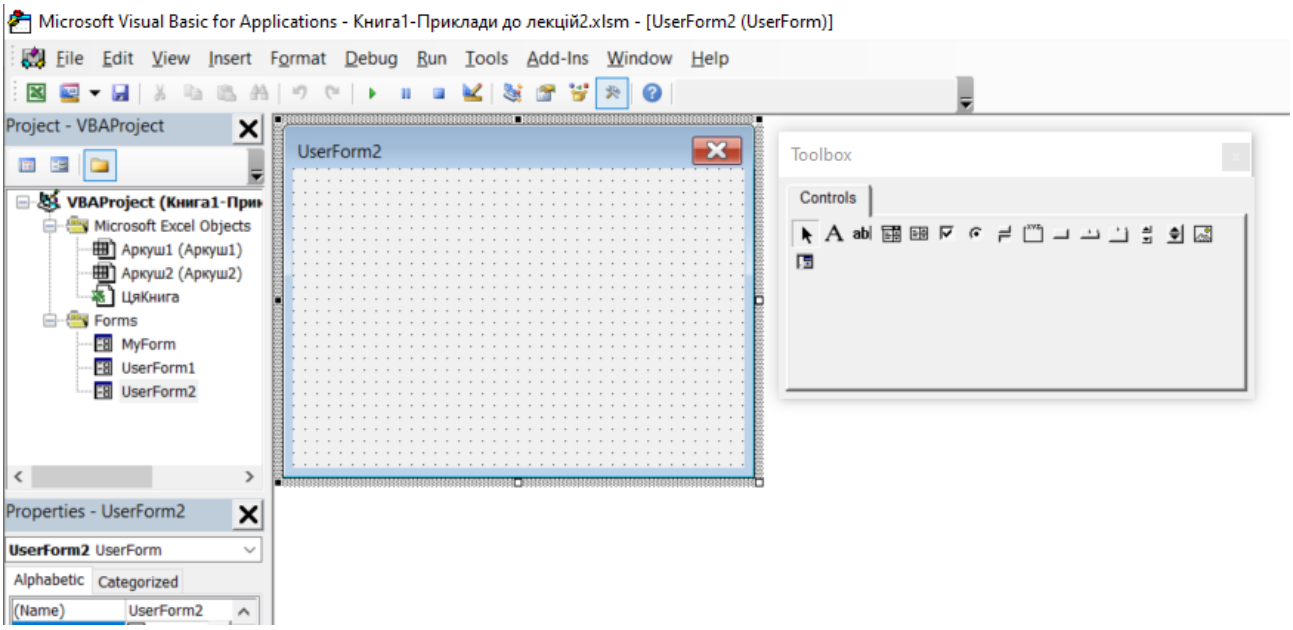


Рис. 3.2. Скріншот створення користувацької форми

Для виконання дій з користувацькою формою можуть використовуватися вбудовані методи, а також створюватися користувацькі програмні коди. Для переходу між вікном програмного коду і вікном візуального редактору активної форми використовуються кнопки **View Code** і **View Object** на панелі інструментів.

Для виклику користувацької форми використовується метод `Show`, який має наступний синтаксис:

```
<Ім’я форми>.Show
```

> **Розміщуватись дана команда повинна не у вікні програмного коду форми, а в будь-якому іншому модулі!**

Форма залишається видимою на екрані до моменту її закриття. Закрити форму можна через спеціальну кнопку в правому верхньому куті або за допомогою операторів `Unload` (видаляє форму з пам’яті) або `Hide` (приховує форму, але не видаляє її з пам’яті), які можна присвоїти будь-якій дії над формою. Закриття шляхом натискання кнопки **Закрити** призводить до некоректного закриття: виконання всіх процедур даної форми при цьому обривається, а дані, що були введені користувачем, не зберігаються. Для коректного закриття краще використовувати спеціально створені кнопки, які передбачають виконання ряду процедур, що необхідні для коректного завершення роботи форми, або ж прописати всі оператори в спеціальній процедурі `Userform_Terminate`.

2. Властивості користувацької форми. Дані властивості відображаються у вікні відповідному вікні «**Properties**». Якщо таке вікно в редакторі не відображається, то його можна викликати натисканням кнопки F4.

Всі властивості представлені на двох вкладках: в алфавітному порядку (**Alphabetic**) і по категоріям (**Categorized**). Зліва перераховані імена властивостей, а з права – поточні значення даної властивості.

Категорії властивостей включають наступні групи:

• **Appearance** – зовнішній вигляд форми (властивості кольору фону, вигляду рамки, назви і т. д.);

• **Behavior** – поведінка форми (активність, модальність і т. д.);

• **Font** – шрифт виводу повідомлення;

• **Misc** – інші властивості, які не входять у інші категорії;

• **Picture** – наявність малюнку для фону форми;

• **Position** – вихідне положення і розміри форми;

• **Scrolling** – наявність і тип полос прокрутки.

Першою властивістю будь-якої користувацької форми є її ім’я. По замовчуванню при створенні формам присвоюється імена UserForm1, UserForm2 і т. д. Змінити їх можна у вікні властивостей, записавши в поле Name нове ім’я. Дане ім’я буде використовуватися для керування формою в програмному коді. В самій формі воно не відображається. За відображення назви форми відповідає властивість Caption.

> **Важливо! Змінювати назви форм рекомендовано на самому початку роботи з нею, до написання програмного коду. В іншому випадку при зміні назви форми, вам доведеться вручну змінювати всі згадування про неї у коді.**

Контрольні запитання і завдання

1. Що таке інтерфейс і яким чином він забезпечується?
2. Які вбудовані форми мови VBA ви знаєте?
3. Яким чином можна змінювати функціональність вбудованих форм?
4. Для чого потрібні користувацькі форми у мові VBA?
5. Які основні властивості користувацьких форм?
6. Які методи використовуються при роботі з користувацькими формами?
7. Яким чином можна викликати і закрити користувацьку форму?

Практичні завдання

1. Напишіть програмний код для перемноження двох чисел, які введені користувачем.
2. Напишіть програмний код для додавання двох чисел, які введені користувачем.
3. Напишіть програмний код для розрахунку точного віку на основі введеної користувачем дати народження. Після виведення результату запропонуйте користувачу визначити в який день тижня буде день народження в наступному році.
4. Напишіть програмний код, який повертає день тижня на основі введеної користувачем дати. Після виведення результату запропонуйте

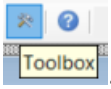
користувачу визначити скільки днів пройшло (залишилось) до введеної дати.

5. Створіть користувацьку форму і напишіть програмний код для перемноження чотирьох чисел.
6. Створіть користувацьку форму і напишіть програмний код для виконання 4 арифметичних операцій (одна операція – одна кнопка) над двома числами.

3.2. Поняття і призначення елементів керування

Поняття і принципи роботи з елементами керування

Користувацька форма містить різні елементи керування. До основних елементів керування, які використовуються практично в кожній програмі, можна віднести наступні: `Label` (надпис), `TextBox` (поле), `CommandButton` (кнопка), `CheckBox` (прапорець), `OptionButton` (перемикач), `ListBox` (список), `ComboBox` (поле з випадаючим списком), `Image` (рисунок).

Всі можливі елементи керування наведені на панелі інструментів **ToolBox**. За замовчуванням вона автоматично відображається при активізації або створенні користувацької форми, але може бути викликана вручну через меню **View – ToolBox** або натисканням кнопки на панелі інструментів 

.

Для додавання в форму елемента керування необхідно вибрати його на панелі інструментів **ToolBox** і натиснути кнопкою мишки по користувацькій формі. Потрібний елемент стандартного розміру буде розміщений таким чином, що лівий верхній кут елементу керування буде розташовуватися там, де знаходився курсор мишки. Зміна розмірів і положення елементів керування, їх виділення, копіювання, видалення та інші операції здійснюються на основі стандартних підходів офісного програмного забезпечення.

Крім форматування за допомогою маніпулювання мишкою зручно використовувати команди меню **Format**. Більшість з них застосовуються для одночасного форматування декількох елементів керування і дозволяє, наприклад, зробити їх одного розміру (функція **Make Same Size**) або вирівняти по домінуючому елементу (функція **Align**).

За допомогою команд **Horizontal Spacing** (Відстань по горизонталі) і **Vertical Spacing** (Відстань по вертикалі) із меню **Format** можна змінювати відстань між виділеними елементами керування одним із чотирьох способів: збільшити (зменшити) відстань на один крок сітки, видалити (розміщує елементи керування впритул один до одного) або вирівняти відстань (крайні елементи залишаються на місці, а середні розподіляються рівномірно між ними).

Елементи керування, як і користувацькі форми, володіють певними властивостями. Властивості можуть бути як загальними для всіх елементів

керування, так і специфічними, які характерні тільки для конкретного виду елементів керування. Змінювати властивості можна як у вікні властивостей, так і програмно в процедурі.

Загальними властивостями, які характерні для всіх (більшості) елементів керування, є **Width** (ширина), **Height** (висота), **Font** (шрифт), **Top** і **Left** (визначають положення елементу керування в формі), **Caption** (надпис елементу керування), **AutoSize** (дозволяє автоматично встановлювати розміри області елемента керування в залежності від розмірів введеного у властивості Caption тексту).

Серед загальних властивостей наявні також **Enable** і **Locked**, які визначають, чи будуть елементи керування доступні користувачу.

Загальними методами елементів керування є `Add` (додає елемент керування в процесі виконання програми), `Move` (переміщує елемент керування), `SetFocus` (встановлює фокус на елементі керування).

Загальними подіями елементів керування є `Click` (користувач вибирає елемент керування за допомогою одинарного кліку кнопкою мишки), `DblClick` (користувач вибирає елемент керування за допомогою подвійного кліка кнопкою мишки), `KeyPress` (користувач натискає будь-яку клавішу на клавіатурі, крім функціональних і клавіш керування курсором), `Change` (зміна значення елемента керування), `GotFocus` і `LostFocus` (елемент керування отримує або втрачає фокус), `Error` (повідомлення про помилку).

Для того щоб форма і її елементи реагували на події, які відбуваються, для кожного із елементів повинна бути прописана відповідна процедура. Якщо виконати бажану дію в режимі проектування форми, то вона створить відповідну процедуру в коді до форми.

Для звернення до конкретного елемента керування можна використовувати його повну назву або короткий префікс із заданням імені. Значення основних префіксів наведені в таблиці 3.3.

Таблиця 3.3. Значення префіксів, які відповідають елементам керування мови VBA

Елемент керування	**Префікс**
`TextBox`	`Txt`
`Label`	`Lbl`
`CommandButton`	`Cmd`
`ListBox`	`Lst`
`ComboBox`	`Cbo`
`ScrollBar`	`Scr`
`SpinButton`	`Spn`
`OptionButton`	`Opt`
`CheckBox`	`Chk`

Нижче будуть розглянуті основні елементи керування, які використовуються в користувацьких формах. Розміщення у формі тільки одного елемента часто недоцільно. Тому далі в прикладах, не дивлячись на використання декількох видів елементів керування, найбільшу увагу будемо приділяти елементу, що буде розглядатись.

Елемент керування Label (Надпис)

Елемент керування Label призначений для відображення деякої текстової інформації. Він розміщується поверх фонового рисунка. З точки зору користувача програми, надпис – це не елемент керування, так як даний текст не може змінюватися користувачем. Частіше за все він містить пояснювальну інформацію або повідомлення для користувача. Текст, який розміщується відображається у полі властивості Caption.

Серед специфічних властивостей даного елементу керування є WordWrap, яка визначає можливість переносу тексту на наступний рядок.

Розміщувати в елементі керування Label можна тільки рядкові дані. Тому числові дані до запису у властивість Caption треба перетворити в текстовий формат (за допомогою функції Str або Format).

Для розміщення тексту з нового рядка можна використовувати ASCII-код для символу Новий рядок (функція Chr з кодом 13) або іменовану константу vbCr.

Приклад. В надписі Label відобразити інформацію про розмір нарахованої заробітної плати, в тому числі суму, яка була виплачена працівнику, і суму податку на прибуток.

```
Label.Caption = "Нараховано: " & Str(salaryfull) & "грн."
& Chr(13) & "Виплачена сума: " & Str(salary) & "грн." &
vbCr & "Податок на прибуток: " & Str(podatok) & "грн."
```

Елемент керування CommandButton (Кнопка)

Елемент **Кнопка** використовується для можливості користувача ініціювати певні дії: наприклад вибір деякого варіанту, підтвердження введення даних, запуск програми і т. д. Елемент **Кнопка** являє собою прямокутник з пояснюючим текстом на ньому. Можна також налаштувати текст або зображення, які будуть відображатися при натисканні на елемент керування CommandButton. Основною подією яка задана за замовчуванням для даного елемента керування є **Click**, який запускає відповідну процедуру. За замовчуванням для елементу керування CommandButton задано властивість Value.

Серед специфічних властивостей **Кнопки** як елемента керування є Default, значення True для якої може бути присвоєно тільки одній кнопці на

формі. Це означає, що дана кнопка буде знаходитися у фокусі в момент відкриття форми і натискання клавіші **Enter** запустить виконання події `Click` для цієї кнопки.

Властивість `Cancel` призначає кнопці функцію відміни (аналогічну натисканню клавіші Esc).

Властивість `Accelerator` дозволяє призначити клавішу на клавіатурі, натискання якої одночасно з клавішею **Alt** запускає процедуру кнопки.

Елемент керування `TextBox` (Поле)

Елемент `TextBox` використовується для введення даних користувачем. Він також може відображати набір даних, таких як таблиця, запит, лист або результат підрахунку. Якщо `TextBox` прив'язаний до джерела даних, то зміна вмісту `TextBox` також змінює значення пов'язаного джерела даних. В певній мірі даний елемент керування аналогічний функції `InputBox`. Створювати користувацьку форму з даним елементом керування є сенс тоді, коли від користувача необхідно отримати різноманітну інформацію (потрібні додаткові елементи керування), необхідно перевірити введені користувачем дані і т. д.

Зручним є розташування в текстовому полі значення за замовчуванням, яке буде відображатись в момент відкриття користувацької форми. Якщо користувачу підходить запропонований текст, то йому не доведеться вводити дані. Для цього необхідно заповнити властивість `Value` (Значення) у вікні властивостей або ввести текст в поле форми в режимі введення тексту.

Для збереження інформації, яку ввів користувач, необхідно призначити деякій змінній значення, яке міститься в текстовому полі. Не завжди доцільно зберігати значення в процедурі `TextBox1_Change()`, так як кожна зміна символу буде викликати дану процедуру. Можна створити кнопку, натискання якої буде зберігати дані в змінну.

Приклад. Створити користувацьку форму, яка виводить кількість днів у введеному місяці.

Вид користувацької форми наведений на рисунку 3.3. За замовчуванням в полі **Рік** введено 2022. Представлений не весь програмний код, а тільки основні елементи. Для поля, в якому вводиться рік, встановлена перевірка на введення числового значення.

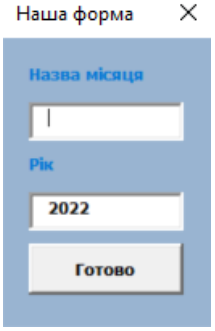


Рис. 3.3. Зовнішній вигляд користувацької форми

Для поля **Місяць** передбачено введення назви місяця, так і його номер. В даному випадку посимвольна перевірка не відбувається. Оператор вибору наведений в неповному вигляді (тільки до квітня). Для лютого використовується умовний оператор, який дозволяє врахувати високосний або не високосний рік.

```
Private Sub CommandButton1_Click()
Dim k2 As Integer
Dim k1, vyvid As String
k1 = TextBox1.Value
k2 = Val(TextBox2.Value)
Select Case k1
Case "січень", "01", "1"
vyvid = "В січні " & k2 & " року 31 день"
Case "березень", "03", "3"
vyvid = "В березні " & k2 & " року 31 день"
Case "квітень", "04", "4"
vyvid = "В квітні " & k2 & " року 30 днів"
Case Else
If (k2 Mod 4 = 0) Then vyvid = "В лютому " & k2 & "року 29
днів" Else vyvid = "В лютому " & k2 & "року 28 днів"
End Select
MsgBox (vyvid)
End Sub
Private Sub TextBox2_Change()
k = TextBox2.Value
If Val(k) = 0 Then MsgBox ("Введіть числове значення
номеру року")
End Sub
```

Специфічними властивостями даного елементу керування є `MaxLength` (визначає максимальну довжину введеного тексту), `MultiLine` (можливість багаторядкового вводу тексту), `ScrollBars` (визначає режим відображення в полі полос прокрутки), `PasswordChar` (символ, який відображається при вводі тексту без відображення символів, наприклад при вводі паролю).

Елементи керування `CheckBox` (Прапорець) і `OptionButton` (Перемикач)

Елементи керування **Прапорець** і **Перемикач** надають можливість користувачу обрати один або декілька варіантів із запропонованих. **Прапорець** являє собою квадрат, в якому галочкою відмічається активний елемент. За керування активністю прапорця відповідає властивість `TripleState`.

Коли користувач обирає `CheckBox`, він відображає спеціальну мітку (наприклад, X) і його поточний параметр стає Так, `True` або `On`; якщо

користувач не обирає CheckBox, він пустий, а його параметр – Ні, False або Off. В залежності від значення властивості TripleState checkBox також може мати значення null.

Якщо елемент CheckBox прив'язаний до джерела даних, то при зміні налаштування змінюється значення цього джерела. Вимкнений елемент CheckBox показує поточне значення, але недоступний і не дозволяє змінювати значення із користувацького інтерфейсу.

Можна також використовувати прапорці всередині групи, щоб обрати один або декілька пов'язаних елементів групи. Наприклад, можна створити форму замовлення, яка містить список доступних товарів з елементом CheckBox, розташованим перед кожним товаром. Користувач може обрати конкретний товар або товари, встановлюючи відповідні прапорці CheckBox.

Перемикач, на відміну від **Прапорця**, дозволяє користувачу обрати тільки один елемент із запропонованих. **Перемикач** представляє собою групу кружечків. VBA сприймає групу перемикачів, які розташовані в одному місці форми, як такі, що належать до однієї групи. Тому активним може бути тільки один з них. Якщо у формі нема окремих частин, відокремлених рамкою (Frame), сторінкою (MultiPage) або вкладкою (Tabstrip), то всі перемикачі даної форми будуть відноситися до однієї групи.

Елементи керування ListBox (Список) і ComboBox (Поле зі списком)

Елемент керування ListBox використовується для зберігання даних, з яких користувач може обрати один варіант.

Елемент керування ComboBox використовується для зберігання і відображення списку значень, з яких користувач здійснює вибір. На відміну від елементу ListBox, в полі зі списком є можливість ввести власне значення в пустий рядок. Перевагою поля зі списком є також його компактний вигляд. Дане поле розташовується в один рядок, і для того щоб відкрити всі можливі варіанти, потрібно натиснути кнопку з трикутником поряд з цим рядком.

Серед спеціальних властивостей активно використовуються ColumnWidth (ширина стовпчика списку), ColumnCount (кількість колонок в списку), ColumnHeads (виведення заголовків списків), ListCount (кількість елементів у списку), ListRows (кількість елементів у розгорнутому списку), ListStyle (спосіб виділення обраного елементу зі списку), MatchEntry (можливість вибору потрібного елементу при наборі імені), MultiSelect (можливість вибору декількох елементів в списку) і т. д.

Для вводу значень в списки використовується метод AddItem або список звязується з джерелом даних в MS Excel або MS Access через властивість RowSource.

Отримати дані про вибір користувача можна за допомогою властивості Value.

Інші елементи керування

ScrollBar (Полоса прокрутки) – це окремо стоячий елемент керування, який можна помістити на форму. ScrollBar являє собою полосу з повзунком, яка дозволяє переміщувати в області видимості частину форми. Це візуально схоже на панелі прокрутки, які ми бачимо у певних об'єктів, таких як ListBox або у розгорнутому списку ComboBox. Однак, на відміну від полос прокрутки у цих прикладах, елемент ScrollBar не являється частиною іншого об'єкта.

Щоб за допомогою Полоси прокрутки встановити або прочитати значення іншого елементу, необхідно написати код для подій і методів ScrollBar. Наприклад, щоб за допомогою полоси прокрутки оновити значення TextBox, можна написати код, який зчитує властивість Value із scrollBar, а потім задає властивість Value textBox.

***Примітка**. Щоб створити горизонтальний або вертикальний об'єкт ScrollBar, змініть його властивість Orientation.*

Серед специфічних властивостей цього елементу – Max і Min (максимальне і мінімальне значення положення полоси прокрутки), SmallChange (крок зміни значення).

Елемент керування SpinButton (Лічильник) дозволяє плавно прокручувати вміст елементу керування і таким чином встановити необхідне значення. Даний елемент керування по властивостям аналогічний полосі прокрутки, але він «прокручує» не всю форму, а тільки дані одного елементу і не містить «бігунець», а тільки регулює рух вверх і вниз.

Клік на SpinButton (Лічильник) змінить тільки його значення. Ми можемо написати код, який використовує SpinButton для оновлення значення іншого елементу керування. Наприклад, ми можемо скористатися SpinButton для зміни місяця або року.

Також за допомогою SpinButton можна переглянути ряд значень або елементів списку, змінити значення в текстовому полі.

Щоб відобразити значення, оновлене SpinButton, необхідно назначити значення SpinButton елементу, наприклад властивості Надпису або властивості TextBox.

Щоб створити горизонтальний чи вертикальний об'єкт SpinButton, змініть його властивість Orientation.

Елемент керування `Image` (Рисунок) використовується для відображення малюнків у формах, на кнопках і т. д. Наприклад, `Image` можна використовувати для відображення фотографій працівників у формі персоналу.

`Image` дозволяє обрізати малюнок, змінити його розмір або масштаб, але не дозволяє редагувати сам малюнок. Наприклад, `Image` не можна використовувати для зміни кольорів малюнка, щоб добитися прозорості зображення або покращити його якість. Для цих цілей необхідно використовувати програми редагування зображень.

`Image` підтримує наступні формати файлів:

- *.bmp
- *.cur
- *.gif
- *.ico
- *.jpg
- *.wmf

Примітка. *Ми також можемо відобразити зображення на* ***Підписі****. Однак* `Label` *не дозволяє обрізати малюнок, змінювати його розмір або масштаб.*

Подією за замовчуванням для елемента `Image` є подія кліку кнопкою мишки.

Приклад. Створити користувацьку форму і написати програмний код обчислення відсотків по депозиту в залежності від введених користувачем умов.

Примітка. *Користувач задає варіант нарахування відсотків (тільки на суму депозиту або з врахуванням нарахованих відсотків (капіталізації)), суму депозиту, розмір відсотку, частоту нарахувань відсотків, термін депозиту.*

Зовнішній вигляд користувацької форми представлений на рисунку 3.4.

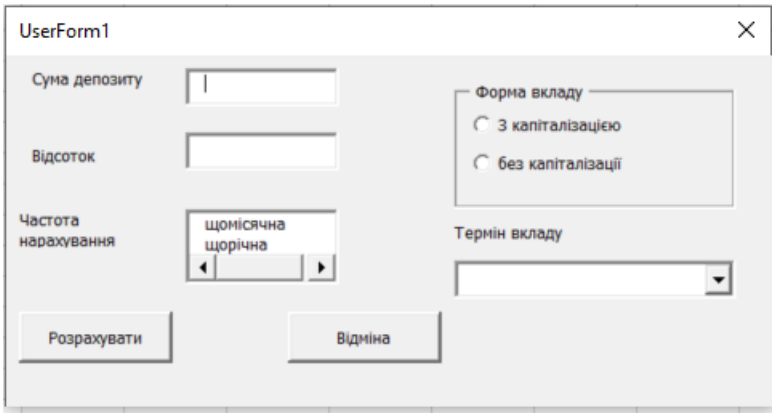


Рис. 3.4. Зовнішній вигляд форми для обчислення відсотків по депозиту

Програмний код для створеної користувацької форми наступний:

```
Private Sub CommandButton1_Click()
Dim vidpovid As String
If (TextBox1.Value = " ") Or (TextBox2.Value = " ") Or
(ListBox1.Value = " ") Or (ComboBox1.Value = " ") Or
(OptionButton1.Value = False And OptionButton2.Value =
False) Then
MsgBox ("Заповніть всі необхідні поля форми")
Else
suma = Val(TextBox1.Value)
stavka = Val(TextBox2.Value)
If ComboBox1.Value = "3 місяці" Then kil_periodiv = 3
If ComboBox1.Value = "6 місяців" Then kil_periodiv = 6
If ComboBox1.Value = "1 рік" Then kil_periodiv = 12
If ComboBox1.Value = "2 роки" Then kil_periodiv = 24
If ComboBox1.Value = "3 роки" Then kil_periodiv = 36
If ComboBox1.Value = "5 років" Then kil_periodiv = 60
If ListBox1.Value = "щомісячна" Then Period = 12 Else
Period = 1: kil_periodiv = kil_periodiv / 12
If OptionButton2.Value = True Then vidsotok = suma *
(1 + ((stavka * kil_periodiv) / (Period * 100))) Else
viddsotok = suma * ((1 + (stavka / (Period * 100))) ^
(kil_periodiv))
vidpovid = vidsotok
MsgBox (vidpovid)
UserForm1.Hide
End If
End Sub
Private Sub userform_Initialize()
ComboBox1.List = Array("3 місяці", "6 місяців", "1 рік",
"2 роки", "3 роки", "5 років")
ListBox1.List = Array("щомісячна", "щорічна")
End Sub
Private Sub CommandButton2 _ Click()
Unload Me
End Sub
```

Контрольні питання і завдання

1. Для чого використовуються елементи управління?
2. Яким чином можна змінити текст в елементі Label?
3. Назвіть ефективні прийоми роботи з елементом TextBox.
4. В чому полягають основні відмінності Списку і Поля зі списком?
5. В яких випадках використовуються прапорці і перемикачі в формах?

Практичні завдання

1. Створіть користувацьку форму, яка розраховує розмір щомісячного платежу по кредиту виходячи з відсоткової ставки, терміну і розміру кредиту і способу оплати відсотків.
2. Створіть користувацьку форму, яка розраховує розмір щомісячного відсотку, який отримується по депозиту виходячи з відсоткової ставки, терміну і розміру депозиту і способу капіталізації.
3. Створіть користувацьку форму, яка дозволяє розрахувати вартість замовлення у кафе (вибір з обмеженого набору страв, задання кількості, розрахунок загальної вартості).
4. Створіть користувацьку форму, яка дозволяє визначити вартість замовлення в швейному ательє (вибір виду одягу, наявність додаткових аксесуарів, їх кількість, розрахунок загальної вартості).

Індивідуальні практичні завдання

1. Вивести різницю між мінімальним і максимальним числом із введених користувачем. Кількість чисел для вводу визначається користувачем.
2. Серед введених користувачем слів знайти максимальне по довжині. Кількість слів для вводу визначається користувачем.
3. Визначити середнє значення серед введених користувачем п’яти чисел і вивести ті з них, які більші за середнє.
4. Користувач вводить три числа, які визначають сторони трикутника. Визначити, чи можна побудувати цей трикутник і чи буде він прямокутним.
5. Для введених користувачем двох чисел вивести їх найменше загальне кратне і найбільший загальний дільник.
6. Впорядкувати цифри введеного користувачем числа по зростанню.

Тест

1. Сукупність засобів і методів, які забезпечують взаємодію користувача з пристроєм в загальному і конкретній програмі в тому числі, називається:

а) інтерфейсом користувача;
б) інтерфейсом;
в) мовою програмування.

2. Що використовується замість вбудованих діалогових вікон у мові VBA:

а) форми;
б) користувацькі форми;
в) запити.

3. До основних елементів керування користувацької форми відносяться:

а) CommandButton;
б) Datediff;
в) Label;
г) ListBox.
д) LBound;

4. До загальних властивостей елементів керування відносяться:

а) Width;
б) Enable;

в) Font;
г) Horizontal Spacing.

5. Загальними методами елементів керування є:

а) Add;
б) SetFocus;
в) AutoSize.

6. Загальними подіями елементів керування є:

а) KeyPress;
б) Move;
в) Change;
г) Locked.

7. Елемент керування Label призначений:

а) для відображення деякої текстової інформації;
б) збереження інформації, введеної користувачем;
в) розміщення тексту з нового рядка.

8. Елемент керування ListBox використовується:

а) для переміщення частини форми;
б) зберігання даних, з яких користувач може обрати один варіант;
в) можливості користувача ініціювати певні дії.

9. Елемент керування SpinButton дозволяє:

а) плавно прокручувати вміст елементу керування і таким чином встановити необхідне значення;
б) розміщувати текст з нового рядка;
в) зберігати інформацію, введену користувачем.

ТЕМА 4. ІЄРАРХІЯ ОБЄКТІВ ПАКЕТУ OFFICE

4.1. MS Excel і його об’єкти

Класи, які задають властивості книги

У мові VBA для кожної програми визначено множину об’єктів, організованих в ієрархію. Верхнім фрагментом ієрархії є об’єкт **Application** (Програма). Даний об’єкт керує налаштуваннями рівня програми, наприклад параметрами MS Excel. Повне посилання на об’єкт складається з ряду імен вкладених послідовно один в одного об’єктів. Розділювачами імен об’єктів в цьому ряді є крапки, ряд починається з об’єкту **Application** і закінчується іменем самого об’єкта. Однак наводити кожний раз повне посилання на об’єкт не обов’язково. Достатньо вказати тільки нові об’єкти, а об’єкти, які активні в даний момент, можна пропустити.

Наступний рівень ієрархії представлений колекцією об’єктів **Workbooks** (Книги). Цій колекції належить множина об’єктів **Workbook**, які представляють собою файли MS Excel. Для звернення до книги потрібно вказати її назву. Якщо книга активна, то для звернення до неї достатньо вказати **ActiveWorkbook**.

Приклад. Способи звернення до книги Excel

```
Excel.Application.Workbooks.Open ("Приклад місяці.xlsm")
'шлях вказувати не обов’язково, якщо потрібний файл знаходиться
в тій самій папці, що і файл з програмним модулем
Application.Workbooks("Приклад").Close
ActiveWorkbook 'звертаємося до активної книги
ThisWorkbook 'звертаємося до книги, в якій створений програмний модуль
```

Як відмічалося вище, будь-який об’єкт у мові VBA характеризується методами, властивостями і подіями. Основні методи об’єкта **Workbook** наведені в таблиці 4.1.

Таблиця 4.1. Основні методи об’єкта Workbook

Метод	Дія, що виконується
Activate	Активізує робочу книгу таким чином, що її перший робочий лист стає активним
Add	Створює новий об’єкт для сімейства Workbooks
Protect	Захищає робочу книгу від внесення до неї змін
Unprotect	Знімає захист з робочої книги
Close	Закриває робочу книгу
Open	Відкриває існуючу робочу книгу
Save	Зберігає робочу книгу
SaveAs	Зберігає робочу книгу в іншому файлі
PrintPreview	Запускає попередній перегляд
Printout	Друкує вміст робочої книги

> Якщо програма Excel відкрита програмно (наприклад, в результаті виконання процедури з іншої програми), то нова книга автоматично **НЕ СТВОРЮЄТЬСЯ!**

Кількість листів в новій робочій книзі можна задати командою `Application.SheetsInNewWorkbook = <кількість листів>`, яка повинна розташовуватися до команди створення нової книги.

Основні властивості об'єкта **`Workbook`** наведені в таблиці 4.2.

Таблиця 4.2. Основні властивості об'єкта `Workbook`

Властивості	Значення, що повертаються
`ActiveSheet`	Повертає активний лист книги
`ActiveDialog`	Повертає активне діалогове вікно
`ActiveChart`	Повертає активну діаграму
`Sheets`	Повертає сімейство всіх листів книги
`Worksheets`	Повертає сімейство всіх робочих листів книги
`Charts`	Повертає сімейство всіх діаграм книги
`Count`	Повертає число об'єктів сімейства `Workbooks`
`HasPassword`	Визначає наявність паролю захисту у документа. Приймає значення `True` і `False`
`FullName`	Містить повне ім'я файлу, включаючи шлях до нього
`Path`	Повертає шлях до файлу

Основні події об'єкта `Workbook` наведені в таблиці 4.3.

Таблиця 4.3. Основні події об'єкта `Workbook`

Подія	Момент виникнення події
`BeforeClose`	Закриття робочої книги
`BeforePrint`	Друк робочої книги
`BeforeSave`	Збереження робочої книги
`Deactivate`	Робоча книга втрачає фокус вводу
`NewSheet`	Додавання нового листа
`Open`	Відкриття робочої книги
`Sheet Activate`	Активізація будь-якого робочого листа
`Sheet Deactivate`	Робочий лист втрачає фокус вводу

Корисним є створення об'єктних змінних. Це дозволяє в майбутньому при зверненні до даного об'єкта не прописувати ряд вкладених один в одного об'єктів, а вказувати ім'я змінної. Синтаксис оголошення об'єктної змінної наступний:

```
Dim some_book As Excel.Workbook
```

```
Set some_book = ActiveWorkbook
some_book.Close
```

Класи листів

Наступний рівень ієрархії об'єктів MS Excel представлений робочими листами – об'єктами **Worksheet** із колекції **Worksheets**. Для звернення до листа використовується його ім'я або номер по порядку (нумерація листів здійснюється зліва направо). Другий варіант дозволяє знаходити потрібний лист незалежно від мовної версії MS Excel, в якій буде виконуватися програма. Крім того, для звернення до поточного листа може використовуватися конструкція **ActiveSheet**.

Основними властивостями листів у мові VBA є **Name** (повертає ім'я робочого листа), Cells, Column і Rows (повертає сімейство комірок, стовпчиків або рядків листа). Корисною є властивість Count, яка застосовується до всієї колекції листів Worksheets і повертає кількість листів в книзі.

Основні методи об'єкта Worksheet наведені в таблиці 4.4.

Таблиця 4.4. Основні методи об'єкта Worksheet

Метод	Дія, що виконується
Activate	Активує робочий лист
Add	Створює новий робочий лист
Delete	Видаляє робочий лист
Protect	Захищає робочий лист від внесення в нього змін
Unprotect	Знімає захист з робочого листа
Copy	Копіює робочий лист в інше місце робочої книги
Move	Переміщує робочий лист в інше місце робочої книги

Перелічені методи можуть мати ряд атрибутів. Повний синтаксис методу Add наступний:

```
Worksheets.Add (<перед яким листом>, <після якого листа>,
<кількість листів>, <шаблон листа>)
```

Повний опис всіх атрибутів можна переглянути в довідці або у спливаючій підказці при написанні програмного коду.

Основні події об'єкта Worksheet представлені в таблиці 4.5.

Таблиця 4.5. Основні події об'єкта Worksheet

Подія	Коли виникає подія
BeforeClose	При закритті робочої книги
BeforePrint	Перед друком робочої книги

BeforeSave	Перед збереженням робочої книги
Deactivate	Коли робоча книга втрачає фокус
NewSheet	При додаванні нового листа
Open	При відкритті робочої книги
Sheet Activate	При активізації будь-якого робочого листа
Sheet Deactivate	Коли робочий лист втрачає фокус

Листи MS Excel можуть бути не тільки джерелом даних, але і інтерфейсом взаємодії з користувачем. Найпростішим способом створення інтерфейсу за допомогою листів є визначення окремих комірок в якості джерел вводу/виводу інформації. Крім того, на листі можна розташувати елементи керування, аналогічні елементам керування користувацьких форм.

Класи комірок і діапазонів

Центральним об'єктом ієрархії MS Excel є діапазон комірок (Range). Він дозволяє звертатися до окремої комірки, стовпчика, рядка або довільного діапазону комірок. Для звернення до комірок в атрибутах об'єкта Range, які є властивостями об'єкта Worksheets, вказується адреса першої і останньої комірок діапазону. У VBA застосовують два способи посилання на комірки робочого листа:

1) через задання імені комірки, наведеного в лапках;

2) шляхом задання числових координат комірки через кому.

В останньому випадку перше число відноситься до номеру рядка, а друге – до номеру стовпчика. Одиницею відліку при цьому буде адреса комірки, що вказана в об'єкті Range, а координати розміщуються у властивість Cells. За замовчуванням початком координат буде комірка **A1**:

```
ActiveSheet.Range("B5").Activate 'активує комірку B5

ActiveSheet.Range("B5:D6").Activate 'активує діапазон B5:D6

ActiveSheet.Cells(2, 5).Activate 'активує комірку E2

ActiveSheet.Range(Cells(4, 6), Cells(8, 7)).Activate

'активує діапазон F4:G8

ActiveSheet.Range("B1").Cells(1, 2).Activate 'активує комірку B1.
```

Крім того, можливе звернення до комірок через задані імена діапазонів і за допомогою ключового слова ActiveCell, а також через об'єкт Selection, який повертає діапазон, вручну виділений користувачем або програмно з використанням методу Select.

Основні властивості об’єкта Range наведені в таблиці 4.6.

Таблиця 4.6. Основні властивості об’єкта Range

Властивість	Значення, що повертається
Value	Повертає значення комірки
Name	Повертає ім’я діапазону
Count	Повертає число об’єктів в наборі
CurrentRegion	Повертає число рядків поточного діапазону. Поточний діапазон обмежений пустим рядком і стовпчиком і містить даний елемент
EntireColumn, EntireRow	Повертає рядок і стовпчик, які містять активну комірку
ColumnWidth, RowHeight	Повертає ширину стовпчиків і висоту рядків діапазону
Formula	Повертає формулу в комірці
FormulaArray	Повертає формулу діапазону
Font	Повертає об’єкт Шрифт, який має властивості Name (ім’я шрифту), FontStyle (стиль: Regular (звичайний), Bold (напівжирний), Italic (курсив)), size (розмір) та інші.

Вказані властивості використовується не тільки для того, щоб отримати значення, що міститься в комірці, але і для того, щоб присвоїти їм деякі значення.

Приклад. На листі MS Excel наведена таблиця. Необхідно визначити кількість заповнених рядків.

```
row1 = ActiveCell.CurrentRegion.Rows.Count 'активною повинна бути комірка в середині діапазону
Range("I6").Value = "У вказаному діапазоні " & row1 & "рядків"
```

Формули, які повертаються властивістю Formula, мають англомовні терміни і формат A1. Для кириличних формул і/або формату R1C1 використовуються ключові слова FormulaLocal і FormulaRlCl.

Основні методи об’єкта Range наведені в таблиці 4.7.

Таблиця 4.7. Основні методи об’єкта Range

Метод	Дія, що виконується
Address	Повертає адресу комірки
Clear	Очищує діапазон

AutoFit	Налаштовує ширину стовпчика і висоту рядка
Copy, Cut, Delete	Копіює, вирізає та видаляє діапазон відповідно
Offset	Повертає діапазон, зміщений відносно даного на величини вказані в аргументах
Select	Виділяє діапазон

При роботі з об'єктами Range зручно використовувати цикл For Each … Next. Його робота аналогічна циклу For … Next (послідовно перебирати всі об'єкти виділеного діапазону). Синтаксис даного циклу наступний:

```
For Each <змінна> in <діапазон>
<оператори>
Next
```

Змінна в даному випадку повинна бути визначена як відповідний об'єкт, наприклад Range.

Приклад. Написати програмний код обчислення визначника матриці n-го порядку. Матриця записана на активному листі MS Excel, починаючи з комірки A1.

Примітка. *Визначник може бути розрахований тільки для квадратної матриці, тому перевіряти розмірність матриці можна тільки в одному напрямку.*

Для обчислення використовується алгоритм Гауса, при якому шляхом елементарних перетворень всі числа нижче головної діагоналі стають рівними нулю. Визначник обчислюється як добуток елементів головної діагоналі, розділений на добуток коефіцієнтів, які використовуються для елементарних перетвореннях.

```
Sub viznachnik()
Dim i As Integer, sum As Integer, j As Integer
m As Integer
Dim v As Double, a As Double, k As Double
Worksheets("Лист1").Activate
Cells.Select
Selection.Copy
Sheets.Add After:=Sheets(Sheets.Count)
ActiveSheet.Paste 'копіює дані з листа 1 (що містить матрицю) на новий лист
```

```
sum = ActiveSheet.Cells(65536, 1).End(xlUp).Row 'визначає кількість рядків в матриці
v = 1
For i = 1 To sum 'цикл перебору рядків
For m = 1 To sum - i
k = -ActiveSheet.Range("A1").Cells(i, i).Value /
ActiveSheet.Range("A1").Cells(i + m, i).Value 'коефіцієнт
для елементарних перетворень
For j = i To sum 'цикл перебору стовпчиків
ActiveSheet.Range("A1").Cells(i + m, j).Value =
ActiveSheet.Range("A1").Cells(i + m, j).Value * k +
ActiveSheet.Range("A1").Cells(i,  j).Value  'елементарні перетворення рядків нижче аналізованого, при яких елементи нижче головної діагоналі рівні нулю
Next j
v = v / k 'обчислює визначник матриці
Next m
Next i
For i = 1 To sum
v  =  v  *  ActiveSheet.Range("A1").Cells(i,  i).Value 'обчислює визначник матриці
Next i
MsgBox (v) 'виводить результати
End Sub
```

Контрольні питання і завдання

1. Яка ієрархія об’єктів MS Excel для мови VBA?
2. Наведіть основні характеристики робочої книги як об’єкта мови VBA.
3. Наведіть основні характеристики робочого листа як об’єкта мови VBA.
4. Які відмінності різних способів звернення до комірок і діапазонів? Який з них і в яких випадках більш кращий для мови VBA?
5. Які основні методи об’єкта Range?

Практичні завдання

1. Розробіть програмний продукт, який дозволяє будувати графік функції двох змінних.
2. Розробіть програмний продукт, який дозволяє замальовувати комірки виділеного діапазону в шахматному порядку.
3. На листі 1 в стовбці А розміщені прізвища, в стовпці В - імена, а в стовпці С – по-батькові працівників організації. Напишіть процедуру, яка на листі 2 в стовпці А виведе список прізвищ і ініціалів працівників.
4. Напишіть процедуру, яка перевіряє число, що розміщене в комірці А1, на наявність дільників. Якщо число просте, то потрібно вивести повідомлення, що число просте, в іншому випадку – розкласти його на множники в стовпці В.

4.2. MS Word і його об’єкти

Класи, які задають властивості документа

Найголовнішим об’єктом MS Word, як і в інших програмних продуктах є об’єкт `Application`. Він знаходиться на початку ієрархії і містить в собі всі інші об’єкти. Але центром всіх процедур є інший об’єкт – `Document`, який входить в колекцію об’єктів `Documents`.

Для того щоб процедура могла працювати з документом, необхідно вказати його в коді. Для роботи з активним в даний момент документом зручно використовувати ключове слово `ActiveDocument`. До інших документів можна звертатися по імені файлу або через об’єктні змінні, яким призначений даний файл. Активувати вже відкритий файл можна через команду `Activate`.

Для створення нового документу використовується метод Add. Його атрибутами є шаблон, на основі якого буде створений документ, а також те, чи буде даний документ сам являти собою шаблон:

```
Dim NewDocAs Word.Document
Set NewDoc = Application.Documents.Add()
```

Для того, щоб відкрити певний документ можна використати метод `Open`, f для закриття – метод `Close`.

Для збереження документів використовуються методи `Save` та `SaveAs` об’єкту `Document`. Метод `Save` також може бути використаний для колекції `Documents`, що дозволяє зберегти одразу всі відкриті документи MS Word, проте це не завжди зручно.

Основні методи документів в MS Word аналогічні методам робочих книг в MS Excel.

Основні властивості об’єкту `Document` наведені в таблиці 4.8.

Таблиця 4.8. Основні властивості об'єкту Document

Властивості	Опис
Background	Повертає об'єкт Shape (фонове зображення)
Characters	Повертає колекцію об'єктів Range, кожний з яких являє собою один символ
Content	Повертає текст документу
Fields	Повертає посилання на колекцію Fields (робота з полями)
Footnotes	Повертає колекцію виносок
Name і FullName	Повертають ім'я документу, без шляху і з шляхом відповідно
Paragraphs	Повертає колекцію абзаців
Type	Повертає тип документу (звичайний, шаблон або web-сторінка зі фреймами)

Класи, що задають структуризацію тексту

Дочірнім об'єктом, який направлений на роботу з текстом, є наприклад, Selection (виділена частина тексту). Цей об'єкт належить до об'єкту Window (вікно) або Pane (частина вікна), а не до всього документу.

Ще одним об'єктом, який містить неперервну область (діапазон) в MS Word, є об'єкт Range. Відмінність Range від Selection полягає в тому, що перший являє собою програмне виділення тільки текстової інформації. А об'єкт Selection виділяє будь-які об'єкти в області вікна та дозволяє користувачу бачити виділену область, але при цьому він займає більший об'єм пам'яті. Ці об'єкти можуть створюватися один з одного. Для виділення об'єкту Selection використовується метод Select, для зворотного перетворення – властивість Range. Більшість методів і властивостей для цих об'єктів співпадає.

Для збільшення виділеної області зручно використовувати метод Expand. Він додає виділений текст у кінець виділеного діапазону. В якості тексту можуть виступати символ (wdCharacter), слово (wdWord), речення (wdSentence), абзац (wdParagraph) і т. д. Дія методу додає тільки один із обраних об'єктів до відповідного виділеного діапазону. Синтаксис оператора виглядає наступним чином:

```
Selection.Expand (<потрібний об'єкт>)
```

> Додавати виділення на початок діапазону цим методом неможливо

Для перевизначення обраного діапазону використовуються методи MoveStart та MoveEnd, які переміщують початок і кінець виділеного діапазону в конкретну точку. Для того, щоб визначити, на скільки одиниць повинне бути

виконане зміщення, можна використовувати метод Move, який повертає числове значення, що відповідає кількості заданих об’єктів (символ, слово і т. д.), на які повинно відбутися зміщення. Розглянуті методи дозволяють розширювати або ж звужувати діапазон для якого вони застосовуються. Синтаксис конструкції виглядає наступним чином:

```
Selection.MoveStart(<потрібний об’єкт>, <кількість одиниць>)
```

Метод Collapse дозволяє стиснути виділення таким чином, щоб воно не містило тексту або об’єктів. Даний метод використовується коли необхідно вставити певний об’єкт в вказану точку. По замовчуванню виділення стискується до його початкової точки.

Для роботи з текстом використовуються стандартні методи: Copy (копіювання), Cut (вирізання), Paste (вставка), Delete (видалення), які застосовуються до вказаного об’єкта або виділеного діапазону. Дані методи також працюють з буфером обміну. У мові програмування VBA рекомендується використовувати рядкові змінні, яким присвоюється текст (властивість Text) або текст з врахуванням форматування (властивість FormattedText), який передбачений для копіювання або вставки, так як це більш ефективно з точки зору використання пам’яті.

Основні властивості об’єктів діапазону наведені в таблиці 4.9.

Таблиця 4.9. Основні властивості об’єктів Range і Selection

Властивості	Опис
Borders	Колекція об’єктів Border, які визначають рамку навколо тексту
Font	Параметри форматування тексту (Name, Size, Bold)
ParagraphFormat	Параметри форматування абзацу (LeftIndent, LineSpacing і т. д.)
TabStops	Тип і розташування точок табуляції (доступ до властивості здійснюється через додатковий об’єкт Paragraph)
Start та End	Визначення номеру першого і останнього символу у виділенні відповідно
Text	Введення тексту у місці виділення

Важливою властивістю об’єктів діапазону є Find, який повертає об’єкт Find за допомогою методу Execute. Синтаксис, який дозволяє знайти необхідний текст в документі наведено нижче:

```
With Selection.Find
.ClearFormatting 'не враховує форматування
.Execute FindText:="library"
End With
```

Контрольні питання та завдання

1. Назвіть основні об’єкти MS Word для мови VBA. Яка відмінність із ієрархією об’єктів MS Excel?
2. Наведіть основні методи і властивості об’єкту `Document`.
3. Які є відмінності і схожості об’єктів `Range` і `Selection`?
4. Які основні методи роботи з текстом ви знаєте?

Практичні завдання

1. В документі знайдіть слова, які починаються з великої літери та запишіть їх в інший документ.
2. По всьому документу змініть словосполучення «завідувач кафедри» на словосполучення «в. о. завідувача кафедри».
3. Створіть шаблон довідки про місце навчання. Використовуючи програмний код заповніть дану довідку для студентів зі списку вашої групи.
4. Створіть шаблон запрошення на конференцію. Використовуючи програмний код заповніть запрошення для гостей із наперед створеного Вами списку.

Індивідуальні практичні завдання

На листі MS Excel у Вас повинні бути наступні дані: ПІБ студента, номер групи, дисципліна, дата здачі іспиту, оцінка.

Підготуйте користувацьку форму, яка дозволяє додавати в дану таблицю нові дані. Сформуйте на другому листі таблицю у відповідності з наступними умовами.

Варіант	Умова
1	Виведіть дані про склад групи і розрахуйте середній бал для кожної групи
2	Виведіть дані про складені дисципліни та розрахуйте середній бал по кожній дисципліні
3	Виведіть дані про студентів та розрахуйте середній бал кожного
4	Виведіть дані про студентів з середнім балом більше 74 і розрахуйте середній бал кожного з них

5	Виведіть дані про студентів з середнім балом нижче 60 і розрахуйте середній бал кожного з них
6	Виведіть дані про групу з найбільшим середнім балом і розрахуйте середній бал кожного з них
7	Виведіть дані про групу з найменшим середнім балом і розрахуйте середній бал кожного з них
8	Виведіть дані про дисципліну з найбільшим середнім балом і розрахуйте середній бал кожного студента, який здавав дану дисципліну
9	Виведіть дані про дисципліну з найменшим середнім балом і розрахуйте середній бал кожного студента, який здавав дану дисципліну
10	Виведіть дані про здачу іспитів по датам і розрахуйте середній бал по кожній дисципліні за кожну дату

Тести

1. Для роботи з активним на даний момент документом зручно використовувати ключове слово:
   а) Application; б) ActiveDocument; в) Activate.
2. Який метод використовується для відкриття будь якого документу:
   а) Open; б) SaveAs; в) Close; г) Save?
3. Дочірнім об’єктом, який направлений на роботу з текстом це:
   а) Selection; б) Collapse; в) Pane; г) Cut.
4. Для збільшення виділеної області найкраще використовувати метод:
   а) Selection; б) Expand; в) Pane; г) Cut.
5. Який метод дозволяє стиснути виділення таким чином, щоб воно не містило тексту та об’єктів?
   а) Collapse; б) Selection; в) Pane; г) Cut.
6. Для роботи з текстом використовуються стандартні методи:
   а) Copy; б) Selection; в) Pane; г) Paste.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Language reference for Visual Basic for Applications (VBA) [Електронний ресурс] - Режим доступу: https://learn.microsoft.com/uk-ua/office/vba/api/overview/language-reference
2. Чаповська Р.Б. Основи алгоритмізації та програмування: середовище VB / Р. Б. Чаповська, М. В. Делявсткий, А. Є. Жмуркевич, М. В. Одрехівський - Чернівці: Книги – XXI, 2015. – 430с.
3. Bradley P. Excel VBA: Intermediate Lessons in Excel VBA Programming for Professional Advancement / P. Bradley. Amazon Kindle, 2018. – 92p.
4. Van Niekerk M. VBA Automation for Excel 2019 Cookbook: Solutions to automate routine tasks and increase productivity with Excel and other MS Office applications /Mike Van Niekerk. Packt Publishing, 2020. – 362p.
5. Дудзяний І.М. Програмування мовою Visual Basic/VBA / І.М. Дудзяний. – Львів: Видавничий центр ЛНУ імені Івана Франка, 2004. – 240 с.
6. Бондаренко С. Г. Інформаційні технології. Процедури та форми VBA [Електронний ресурс]: навч. посіб. для студ. спеціальності 161 «Хімічні технології та інженерія» / КПІ ім. Ігоря Сікорського; уклад.: С. Г. Бондаренко, А. О. Абрамова., С. І. Заєць – Київ : КПІ ім. Ігоря Сікорського, 2019. – 158 с.
7. Шварич Г.Г. Інформатика та комп’ютерна техніка. Навчальний посібник / Г.Г. Швачич, О.В. Овсянніков, В.В. Кузьменко, Н.І. Нечаєва, Л.М. Петричук.– Дніпропетровськ: НМетАУ, 2007. – 52 с.

Навчально-методичне видання

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОФІСНИХ СИСТЕМ

Комп’ютерний набір і дизайн
О.І. Шпак, М.І. Роль
Коректура і технічна редакція
М.І. Роль

Навчально-методичний посібник створено
на кафедрі програмного забезпечення систем ДВНЗ «УжНУ»
88015, м. Ужгород, вул. Заньковецької, 89
E-mail: kaf-software@uzhnu.edu.ua

Редакційно-видавничий відділ ДВНЗ «УжНУ»
88015, м. Ужгород, вул. Заньковецької, 89
E-mail: dep-editors@uzhnu.edu.ua