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

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

ПРОЦЕДУРИ ТА ФОРМИ VBA

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для студентів,
які навчаються за спеціальністю 161 «Хімічні технології та інженерія»

Київ
КПІ ім. Ігоря Сікорського
2019

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 – Електронні текстові данні (1 файл: 3,7 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2019. – 158 с.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 10 від 20.06.2019 р.) за поданням Вченої ради хіміко-технологічного факультету (протокол № 5 від 26.05.2019 р.)

Електронне мережне навчальне видання

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

ПРОЦЕДУРИ ТА ФОРМИ VBA

Укладачі: Бондаренко Сергій Григорович, канд. техн. наук, доцент

Абрамова Алла Олександрівна, канд. техн. наук, доцент

Заєць Сергій Іванович, завідувач навчальної лабораторією обчислювальної техніки

Відповідальний редактор Бондаренко С. Г., канд. техн. наук, доцент

Рецензенти: Толстопалова Наталія Михайлівна, канд. техн. наук, доцент, в.о. зав. кафедри ТНР, В та ЗХТ ХТФ КПІ ім. Ігоря Сікорського

Навчальний посібник розроблено відповідно до програми підготовки бакалаврів за спеціальністю 161 «Хімічні технології та інженерія». Дисципліна «Інформаційні технології» закладає основу професійних знань щодо програмного забезпечення сучасних персональних комп'ютерів. Дана дисципліна відноситься до циклу природничо-наукової підготовки і є базовою у підготовці бакалаврів вказаного напряму. Представлені матеріали мають за мету закріплення знань та набуття вміння застосовувати програми модульної структур із використанням функцій і підпрограм та застосовувати форми при роботі з VBA в MS Excel при роботі у хімічній галузі промисловості. У даному навчальному посібнику наведений перелік тем, які студент повинен вивчити, матеріали для самостійної роботи та матеріали на основі яких виконуються лабораторні роботи.,

Зміст

Передмова

Навчальний посібник розроблено відповідно до програми підготовки бакалаврів за спеціальністю 161 «Хімічні технології та інженерія». Дана дисципліна закладає основу професійних знань щодо програмного забезпечення сучасних персональних комп'ютерів. Дисципліна «Інформаційні технології» відноситься до циклу природничо-наукової підготовки і є базовою у підготовці бакалаврів вказаного напряму.

Робота студента над учбовим матеріалом з дисципліни: «Інформаційні технології» складається з наступних видів робіт: вивчення матеріалу по навчальних посібниках і підручниках; відвідування лекцій; виконання лабораторних робіт; написання модульної контрольної роботи; індивідуальних консультацій; здачі заліку. При цьому значна частина часу відводиться на самостійну роботу.

Представлені матеріали мають за мету закріплення знань та набуття вміння застосовувати навички з автоматизації обчислень та програмування за допомогою VBA MS Excel при роботі у хімічній галузі промисловості.

Надані теоретичні відомості сприятимуть засвоєнню матеріалу курсу та можуть бути використані під час виконання курсових проектів та робіт, запланованих у дисциплінах професійного циклу, та у дипломному проектуванні. У даному виданні наведений перелік тем, які студент повинен вивчити, за матеріалами яких виконуються лабораторні роботи, надані методичні вказівки з виконання цих робіт, наведені основні теоретичні положення з ілюстрацією на конкретних прикладах. Начальний посібник містить завдання для лабораторних робіт, вимоги до оформлення звіту і контрольні питання для самопідготовки студентів, а також наведені заходи безпеки, яких треба дотримуватись при виконанні лабораторних робіт (додаток А).

ЛАБОРАТОРНА РОБОТА № 1.
ПРОГРАМИ МОДУЛЬНОЇ СТРУКТУРИ. РОБОТА З ПРОЦЕДУРАМИ-ФУНКЦІЯМИ У VBA.

Мета та основні завдання: дослідити можливості VBA MS Excel при роботі з процедурами-функціями (функціями користувача). Набути вмінь роботи з функціями користувача. Вивчити модульний принцип побудови програм і організацію обміну інформацією між програмними модулями.

Завдання. Вивчити наведені нижче теоретичні питання та стисло описати у протоколі питання:

– правила запису функцій користувача та їх організацію;

– особливості виклику функцій користувача;

– оголошення процедур-функцій в VBA.

– способи передачі параметрів процедурами-функціями.

1.1. Короткі теоретичні відомості

У VBA підтримується наступна структура програми. На вищому рівні ієрархії стоїть додаток, далі йдуть проекти, пов'язані з фактичними документами цього додатку, на третьому рівні знаходяться модулі (модулі додатку, модулі користувача, модулі класу, модулі форм і модулі посилань). А на останньому рівні знаходяться процедури і функції цих модулів.

Дана структуризація програм повністю, задовольняє принципам структурного і модульного програмування.

Використання принципів модульного програмування Windows-додатку в середовищі VBA виражене в двох аспектах.

1. Розбиття програмного коду на окремі компактні модулі.

2. Розбиття програмного коду вже всередині модуля на окремі частини – процедури.

Програма може складатися (і зазвичай таки складається) з багатьох процедур і функцій, які можуть розташовуватися в одному або декількох

модулях. Модулі групуються у **проекти**, при цьому в одному проекті можуть співіснувати декілька різних програм, що використовують загальні модулі або процедури.

Така технологія дозволяє розбити задачу на підзадачі і розробити процедури (підпрограми) для розв'язання цих підзадач. Кожна процедура є окремою частиною програми, яка незалежна від інших. А загальна програма повинна здійснювати взаємодію між всіма процедурами, з яких складається програма. При цьому важливо правильно організувати обмін інформацією між процедурами. Тут найчастіше використовується механізм формальних та фактичних параметрів.

Процедури VBA діляться на процедури-підпрограмами (далі підпрограми) і процедури-функції (далі функції). Це фрагменти програмного коду, який знаходиться між операторами Sub і End Sub (підпрограми) або між операторами Function і End Function (функції) відповідно.

Процедури-функції VBA

Вбудовані (стандартні) функції VBA

У мові програмування VBA передбачено декілька десятків ***вбудованих функцій***. Вони доступні в будь-якій програмі на мові VBA. За стандартними функціями закріплені стандартні імена, які не можна використовувати в якості ідентифікаторів (імен змінних). У довідці по VBA вбудовані функції згруповані за абеткою. Більшість математичних функцій, які необхідні в практичній роботі (типу косинус, тангенс та інші) були розглянуті раніше.

Стандартна функція записується наступним чином: спочатку записується ім'я функції, а потім в круглих дужках її аргумент.

Наприклад, SIN(x+1).

Викликається стандартна функція за допомогою операторів VBA, що виконуються. Наприклад, в операторі присвоєння можна записати: Y = SIN(x+1).

Функція обчислює результат із застосуванням спеціальної вбудованої програми і повертає результат в місце виклику.

Нестандартні функції VBA

У VBA можна за допомогою оператора **Function** написати власні процедури-функції. Ці функції називають нестандартними функціями або користувацькими функціями. Нестандартні функції можна як і вбудовані використовувати для отримання значень: у виразах; в операторах присвоєння; в якості аргументів для інших функцій і підпрограм тощо. Частину програми, що багато разів повторюється, доцільно оформити процедурою-функцією. Але при цьому слід пам'ятати, що результатом роботи функції є тільки одне значення, і навіть тоді, коли цей результат отриманий з великої послідовності обчислень.

Користувацька функція має наступний формат:

Function <ім'я функції> [(список формальних параметрів)] [As <Тип >]

…
…
[Exit Function]
…
< ім'я функції > = <результат>

} «Тіло функції» – оператори мови VBA

End Function

Заголовок функції формується за допомогою ключового слова Function і записується в одному програмному рядку. Ім'я функції повинно відповідати правилу оформлення імен у VBA (імена повинні починатися з букви і не містити символів проміжку, арифметичних і логічних операцій, й не повинні співпадати з ключовими словами VBA). Після імені функції слідує у круглих дужках список формальних параметрів (інша назва – список аргументів), які розділяються комами. Формальні параметри відображають форму оброблення даних за допомогою операторів функції. Через них відбувається передача вхідних (фактичних) даних при виклику функції. Аргументами функції можуть бути може бути як прості змінні, так і масиви. В списку

формальних параметрів після імені масиву потрібно записати пусті круглі дужки (наприклад, A()). У заголовку функції може бути описаний тип значення результату, що повертається – [As <Тип >]. Якщо тип даних результату не вказано, то він буде мати тип Variant.

Навіть якщо функція не має аргументів в рядку заголовку необхідно записувати круглі дужки після її імені.

«Тіло функції» складається з операторів мови VBA. Тут повинен знаходитися хоча би один оператор, який присвоює імені функції якесь значення. Для дострокового виходу з функції може бути використаний спеціальний оператор **Exit Function**. При необхідності таких операторів в «тілі функції» може бути декілька.

Для повернення результату роботи функції в «тілі функції» записується оператор присвоєння, **де змінній, яка має таке ж саме ім'я, як і ім'я функції**, присвоюється значення результату.

< ім'я функції > = <результат>

Завершують «тіло функції» ключові слова **End Function**, які записується в окремому програмному рядку.

Виклик функції

У VBA процедура **Function** викликається точно так, як і будь-яка вбудована функція. Оператор виклику функції складається з імені функції і списку фактичних параметрів. Список фактичних параметрів записується в круглих дужках, і змінні в ньому розділяється комами.

Зручно використовувати виклик за допомогою оператора присвоєння:

<**ім'я Змінної**> = <**ім'я Функції**>(<список фактичних параметрів>)

Ім'я змінної – це ім'я змінної, в який планується збереження результату роботи функції. **Ім'я функції** – це ім'я тієї функції, що викликається.

Фактичні параметри – це змінні, значення яких передаються в процедуру-функцію, що викликається, і зі значеннями котрих будуть виконані розрахунки у функції.

Між формальними і фактичними параметрами має бути повна відповідність, тобто формальних і фактичних параметрів має бути однакова кількість; порядок запису в списках фактичних і формальних параметрів має бути один і той же; тип кожного фактичного параметра повинен збігатися з типом відповідного йому формального параметра. До моменту виклику функції всі змінні, які визначають фактичні параметри, повинні бути оголошені й одержати певні значення. Якщо формальним параметром є масив, то після імені масиву потрібно записати пусті круглі дужки (наприклад, B()).

Програму, що викликає функцію називають головною програмою (або такою, що викликає), а процедуру-функцію – функцією (або тією, що викликають).

Приклад 1: Знайти площі двох трикутників: перший зі сторонами a, b , c, а другий – k, l, m. Розрахунок площ здійснити у процедурі-функції за формулою Герона $S = \sqrt{p(p-a)(p-b)(p-c)}$, де S – площа трикутника; a, b, c – сторони трикутника, $p = \frac{a+b+c}{2}$ – напівпериметр.

```
Sub prim()
Dim a As Single, b As Single, c As Single
Dim k As Single, l As Single, m As Single
Dim SA As Single, SB As Single
 ' Введення даних
a = 4: b = 5: c = 6
k = 7: l = 8: m = 9
' Виклик функції
SA = ST(a, b, c)
Debug.Print "Площа трикутника зі сторонами:"; "a="; a, "b ="; b, "c ="; c
 Debug.Print "SA ="; SA
 SB = ST(k, l, m)
Debug.Print "Площа трикутника зі сторонами:"; "k="; k, " l ="; l, "m ="; m
 Debug.Print "SB ="; SB
End Sub
```

```
Function ST(a, b, c)
Dim p As Single, S As Single
p = (a + b + c) / 2
S = Sqr(p * (p - a) * (p - b) * (p - c))
ST = S
End Function
```

Рис. 1.1. Програма з використанням процедури-функції

```
mmediate
Площа трикутника зі сторонами:a= 4        b = 5        c = 6
SA = 9,921567
Площа трикутника зі сторонами:k= 7        l = 8        m = 9
SB = 26,83282
```

Рис. 1.2. Результати роботи програми

Вставлення функцій в модуль VBA

1. Вставити процедуру-функцію можна вручну.

Для цього після введення програмного коду головної програми слід після рядка з ключовими словами End Sub ввести рядок:

Function ім'я функції або Function ім'я функції ().

Наприклад , Function ST або Function ST().

Потім натиснути клавішу **Enter**.

Редактор коду автоматично додасть рядок End Function і роздільник між процедурами.

```
End Sub
Function ST()

End Function
```

В круглих дужках слід ввести список формальних параметрів розділених комою, і між рядками Function ST() та End Function записати програмні рядки функції.

При необхідності перед ключовим словом Function можна вказати **область визначення** функції: **Private –** закрита функція (можливий її виклик з модуля, де вона знаходиться); **Public –** відкрита функція (можливий її виклик з будь-якого модуля). Перед ключовим словом Function можна записати також і ключове слово **Static**. Наявність або відсутність ключового слова **Static** говорить про статус локальних змінних, тобто змінних, оголошених усередині цієї процедури. За наявності цього слова локальні змінні зберігатимуть свої значення між послідовними викликами цієї процедури. За відсутності слова **Static** – змінні не будуть зберігати свої значення.

З урахуванням області визначення функції рядок заголовка можна записати:

[Private|Public][Static]Function <ім'я функції> **(**<список формальних параметрів >**) [As** <Тип>**]**

2. Вставити процедуру-функцію можна скориставшись меню VBA:

Insert (Вставка) – **Procedure**.

У вікні, що з'явилось (рис. 1.1) слід ввести ім'я процедури в поле **Имя** (**Name**).

А потім за допомогою перемикачів:

- задати тип процедури, що додається: **підпрограма (Sub)**, **функція (Function)** або **властивість (Property)**;
- задати загальну – **загальна** (**Public**) або особисту – **особиста** (**Private**) область визначення процедури.

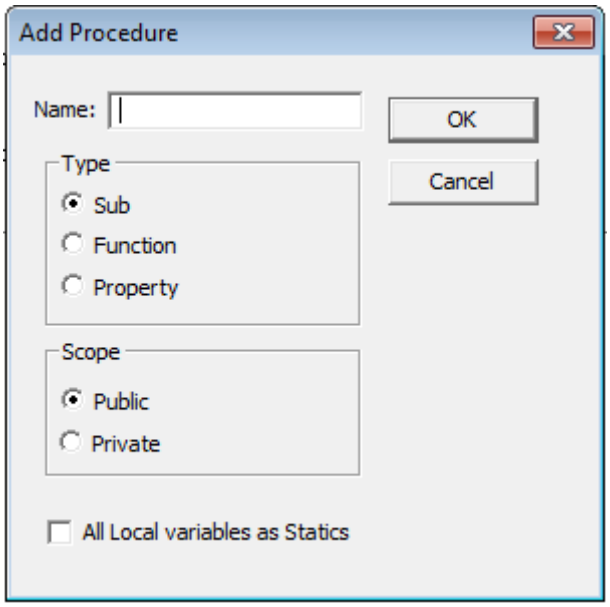


Рис. 1.3. Вікно додавання процедур

Якщо локальні змінні є статичними, слід встановити прапорець – **Всі локальні змінні вважати за статичні** (**All Local Variables As Statics**).

Натиснути кнопку ОК. Редактор коду автоматично додасть шаблон порожньої процедури із заданим ім'ям і роздільник між процедурами.

Наприклад:

```
End Sub
Public Function ST()

End Function
```

Запуск програми на виконання здійснюється стандартним способом за допомогою кнопки  або меню Run.

Приклад 2: Обчислити суму елементів масиву А з шістьох елементів – А(6) і В з восьми елементів – В(8). Розрахунок сум елементів здійснити у процедурі-функції. Нехай елементи масиву А: 3, 5, 4, 2, 1, 10. Нехай елементи масиву В: 1, 2, 3 , 4, 5, 6, 7, 8.

```
Public Sub ppp()
Dim A() As Integer, N As Integer
Dim B() As Integer, M As Integer
Dim i As Integer, SA As Integer, SB As Integer
N = Val(InputBox("Введите розмірність масива N=", "Вікно введення", "6"))
M = Val(InputBox("Введите розмірність масива M=", "Вікно введення", "8"))
ReDim A(N) As Integer
ReDim B(M) As Integer
 ' Введення і друк початкових даних
For i = 1 To N
          A(i) = Val(InputBox("Введіть число"))
 Next i
       Debug.Print "Початковий масив А"
For i = 1 To N
        Debug.Print A(i);
Next i
For i = 1 To M
          B(i) = Val(InputBox("Введіть число"))
 Next i
Debug.Print
 Debug.Print "Початковий масив В"
For i = 1 To M
         Debug.Print B(i);
Next i
 ' Пошук суми
 ' Виклик функції
SA = Smas(A(), N)
Debug.Print
Debug.Print "SA ="; SA
 SB = Smas(B(), M)
Debug.Print
Debug.Print "SB ="; SB
End Sub
```

```
Public Function Smas(X() As Integer, K As Integer) As Integer
Dim S As Integer
S = 0
For i = 1 To K
          S = S + X(i)
Next i
Smas = S
End Function
```

Рис. 1.4. Програма з використанням процедури-функції

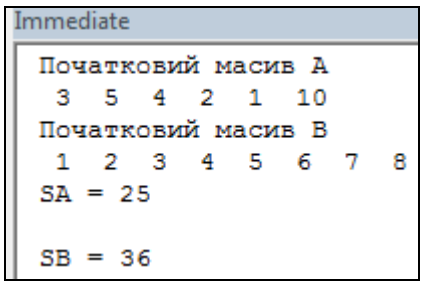


Рис. 1.5. Результати роботи програми

Способи передачі змінних процедурі

Допускається два різні способи передачі змінних процедурі (підпрограмі або функції): **за посиланням і за значенням**.

Якщо змінна передається **за посиланням**, то це означає, що процедурі буде передана адреса цієї змінної в пам'яті. При цьому відбувається ототожнення формального аргументу процедури і переданого їй фактичного параметра. Процедура, яка викликається, може змінити значення фактичного параметра (якщо буде змінений формальний аргумент процедури). Тоді це призведе до зміни переданого при виклику фактичного параметра.

Якщо ж фактичний параметр передається **за значенням**, то формальний аргумент процедури або функції, що викликається, набуває тільки значення фактичного параметра. Тим самим всі зміни значення формального аргументу не позначаються на значенні змінної, що є фактичним параметром.

Спосіб передачі параметрів процедурі або функції вказується при описі її аргументів: імені аргументу може передувати явний описувач способу передачі.

Описувач **ByRef** задає передачу **за посиланням**, а **ByVal** – **за значенням**. Якщо ж явна вказівка способу передачі параметра відсутня, то **за замовчуванням** мається на увазі передача **за посиланням**.

Приклад:

Public Function Example1(x, ByVal y, ByRef z)

Перший аргумент в списку функції прикладу передається за посиланням (діє умовчання), другий – за значенням, а третій – знову за посиланням.

Передача параметрів за значенням

При передачі параметра за значенням (**by value**) процедурі передається копія змінної, виступаючої як параметр процедури, для подальшої обробки. Якщо процедура змінює значення параметра, це зачіпає тільки копію змінної, а не саму змінну. Значення змінної оригінала у процедурі, яка є викликаючою, зберігається колишнім.

Фактичні параметри, задані константами, завжди передаються за значенням.

Передача параметрів за посиланням

Передача процедурі параметрів за посиланням (**by reference**) відкриває їй доступ до області пам'яті, де зберігається вміст змінної. В результаті процедура може змінювати значення змінної, що є її параметром. В деяких випадках це може привести до виникнення помилок.

За замовчуванням в VBA всі параметри передаються по посиланню. Тобто передача даних **за посиланням** дозволяє змінити передані початкові дані, а передача даних **за значенням** – не дозволяє.

1.2.Опис лабораторних засобів та обладнання

Лабораторна робота виконується на персональному комп’ютері стандарту IBM PC під керуванням операційної системи MS Windows зі стандартним пакетом MS Office.

1.3.Заходи безпеки під час виконання лабораторної роботи

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

1.4.Послідовність виконання роботи

1. Відповідно до свого варіанту (див.додаток Б) розробити програму, що реалізує поставлене завдання з використанням процедури-функції (або функцій).
2. Продемонструвати роботу програми викладачу.
3. Зберегти робочу книгу на своєму носії інформації та зберігати до кінця семестру.
4. Оформити протокол лабораторної роботи.

1.5.Обробка та аналіз результатів. Оформлення звіту

При оформленні звіту з лабораторної роботи до заздалегідь підготовленого протоколу (звіт повинен містити: тему та мету лабораторної роботи; короткі теоретичні відомості за зазначеними питаннями; індивідуальне завдання; пояснення до виконання завдання (при необхідності); результати роботи (та необхідні висновки) додаються роздруковані аркуші з результатами виконаної роботи: лістинг програмного коду та результатів розрахунків

1.6. Контрольні запитання

1. Який формат має процедура-функція?
2. Що таке формальні і фактичні параметри? Які вимоги до списків формальних і фактичних параметрів?
3. Навіщо застосовується оператор Exit Function в процедурі-функції?
4. Як повертається результат роботи процедури-функції в головну програму?
5. Як здійснити виклик функції з головної програми?
6. В яких місцях програми не можна помістити виклик функції?
7. Для чого записується у списку формальних параметрів ключове слово **ByVal**?

8. Для чого записується у списку формальних параметрів ключове слово **ByRef**?

9. Для чого записується у рядку заголовка функції ключове слово **Public**?

10. Для чого записується у рядку заголовка функції ключове слово **Private**?

ЛАБОРАТОРНА РОБОТА № 2.
ПРОГРАМИ МОДУЛЬНОЇ СТРУКТУРИ. РОБОТА З ПРОЦЕДУРАМИ-ПІДПРОГРАМАМИ У VBA.

Мета та основні завдання: дослідити можливості VBA MS Excel при роботі з процедурами-підпрограмами. Набути вмінь роботи з процедурами-підпрограмами. Вивчити модульний принцип побудови програм і організацію обміну інформацією між програмними модулями.

Завдання. Вивчити наведені нижче теоретичні питання та стисло описати у протоколі питання:

– правила запису процедур-підпрограм та їх організацію;

– особливості виклику процедур-підпрограм;

– оголошення підпрограм-підпрограм в VBA.

– способи передачі параметрів підпрограмі.

2.1. Короткі теоретичні відомості

Підпрограми поділяють на загальні і процедури обробки подій. Процедури обробки подій починають працювати у відповідь на яку-небудь подію, наприклад, натиснення, кнопки в екранній формі. Загальні процедури починають працювати після явного їх виклику з якого-небудь місця іншої програми. Після виконання такої процедури-підпрограми відбувається автоматичне повернення в те місце головної програми, звідки процедура була викликана.

Загальна процедура може входити до складу модуля екранної форми (у файл frm) або до складу стандартного модуля (у файл bas). Зазвичай ці процедури реалізують якісь загальні дії.

Загальна процедура-підпрограма (підпрограма) на відміну від процедури-функції може повернути у головну програму декілька результатів. Тому, якщо за алгоритмом задачі потрібне отримання більш ніж одного

результату обчислень, то, як правило, використовуються процедури-підпрограми.

Відмінність процедур-підпрограм від процедур-функцій наступна:

1. У заголовку підпрограми відсутній опис типу її імені, бо ім'я підпрограми ніяк не пов'язане з результатом (одним або декількома), що вона повертає.
2. Спосіб виклику процедури відрізняється від способу виклику функції (виклик підпрограми здійснюється за допомогою спеціального оператора виклику).
3. Для передачі результатів роботи підпрограми в списки формальних і фактичних параметрів повинні бути включені параметри, за допомогою яких результати роботи підпрограми передаються в викликаючу програму.

Примітка. *Спосіб передачі підпрограмі інформації за допомогою списку аргументів зменшує кількість змінних, що мають область видимості* ***Public****. Таким чином зникає потенціальне джерело виникнення помилок. Таким чином для передачі інформації підпрограмі слід використовувати список аргументів, а не змінні рівня модуля.*

Процедура-підпрограма має наступний формат:

Формат:

Sub <ім'я підпрограми> [(список формальних параметрів)]

…

[Exit Sub]

…

«Тіло підпрограми» – оператори мови VBA

End Sub

Заголовок підпрограми формується за допомогою ключового слова **Sub** і записується в одному програмному рядку. Ім'я підпрограми повинно відповідати правилу оформлення імен у VBA (імена повинні починатися з

букви і не містити символів проміжку, арифметичних і логічних операцій, й не повинні співпадати з ключовими словами VBA).

Після імені підпрограми записують у круглих дужках список формальних параметрів (список аргументів), що розділяються комами, з явним або неявним їх оголошенням. Механізм передачі інформації через списки формальних і фактичних параметрів для функцій і підпрограм однаковий.

Оголошення кожного аргументу має наступний синтаксис:

<ім'я Аргументу1> [As <тип Даних>], <ім'я Аргументу2> [As <тип Даних>], …

де <ім'я Аргументу> – ідентифікатор, складений згідно з правилам створення імен у VBA; <тип Даних> – тип даних, що визначений користувачем. Якщо тип даних аргументу не вказано, то він буде мати тип Variant. Аргументами процедури-підпрограми можуть бути як прості змінні, так і масиви. В списку формальних параметрів після імені масиву потрібно записати пусті круглі дужки (наприклад, A()). Навіть якщо підпрограма не має аргументів в рядку заголовку необхідно записувати круглі дужки після її імені.

«Тіло підпрограми» складається операторів мови VBA. Для дострокового виходу з підпрограми може бути використаний спеціальний оператор **Exit Sub**. При необхідності таких операторів в «тілі підпрограми» може бути декілька.

Завершують «тіло підпрограми» ключові слова **End Function**, які записується в окремому програмному рядку.

Виклик підпрограми

Виклик підпрограми з непустим списком параметрів здійснюється за допомогою оператора Call.

Формат:

Call ім'я підпрограми (список фактичних параметрів)

Список фактичних параметрів записується в круглих дужках, і змінні в ньому розділяється комами. Якщо формальним параметром є масив, то після

імені масиву потрібно записати пусті круглі дужки (наприклад, B()). Між формальними і фактичними параметрами має бути повна відповідність.

У випадку підпрограми без параметрів оператор виклику підпрограми також не містить параметрів.

Примітка. *Можна викликати підпрограму і без використання ключового слова Call:*

Ім'я підпрограми *[список фактичних параметрів]*

Тут список фактичних параметрів розділяється комами і не береться в круглі дужки.

Для повернення результату роботи підпрограми в списки формальних і фактичних параметрів повинні бути включені параметри, за допомогою яких результати роботи підпрограми передаються в програму, що викликається.

До моменту звертання до підпрограми всі змінні, які визначають фактичні параметри, повинні одержати будь-які значення.

Примітка.

Якщо в проекті використовується декілька підпрограм з однаковими назвами (це можливо, якщо вони в різних модулях), то при їх виклику перед ім'ям підпрограми треба вказувати (через точку) ім'я модуля, в якому процедура розташована:

<ім'я модуля>.<ім'я процедури> <список фактичних параметрів>

Наприклад:

MyModule.MySub Arg1, Arg2 ' виклик процедури з модуля MyModule

Для виклику загальних (Public) підпрограм з іншого проекту додатково до імен модуля і підпрограми указується ім'я проекту:

<ім'я проекту>.<ім'я модуля>.<ім'я процедури> <список фактичних параметрів>

Способи передачі змінних процедурі

Підпрограми, як і функції, допускають два різні способи передачі змінних: **за посиланням** (описувач **ByRef** задає передачу **за посиланням**) **і за**

значенням (описувач **ByVal** – **за значенням**), які розглянуті в п. 1.1 (див. теоретичні відомості до лабораторної роботи №1).

Якщо ж явна вказівка способу передачі параметра відсутня, то за замовчуванням мається на увазі передача за посиланням.

Вставлення функцій в модуль VBA

1. Вставити процедуру-підпрограму можна вручну.

Для цього після введення програмного коду головної програми слід після рядка з ключовими словами End Sub ввести рядок:

Sub ім'я підпрограми або Sub ім'я підпрограми ().

Наприклад , Sub ABC або Sub ABC().

Потім натиснути клавішу **Enter**.

Редактор кода автоматично додасть рядок End Sub і роздільник між процедурами.

```
End Sub
Sub ABC()

End Sub
```

В круглих дужках слід ввести список формальних параметрів розділених комою, і між рядками Sub ABC() та End Sub записати програмні рядки функції.

При необхідності перед ключовим словом Sub можна вказати **область визначення** підпрограми: **Private –** закрита підпрограма (можливий її виклик з модуля, де вона знаходиться); **Public –** відкрита підпрограма (можливий її виклик з будь-якого модуля). Перед ключовим словом Sub можна записати також і ключове слово **Static**. Наявність або відсутність ключового слова **Static** говорить про статус локальних змінних, тобто змінних, оголошених усередині цієї процедури. За наявності цього слова локальні змінні зберігатимуть свої значення між послідовними викликами цієї процедури. За відсутності слова **Static** – змінні не будуть зберігати свої значення.

З урахуванням області визначення функції рядок заголовка можна записати:

[Private|Public][Static] Sub <ім'я підпрограми> [(список формальних параметрів)]

2. Вставити процедуру-підпрограму можна скориставшись меню VBA:

Insert (Вставка) – **Procedure**.

Ця можливість розглянута в п. 1.1 (див. теоретичні відомості до лабораторної роботи №1).

Запуск програми на виконання здійснюється стандартним способом за допомогою кнопки  або меню Run.

Приклад 1:

Розглянемо приклад з лабораторної роботи №1 але з використанням процедури-підпрограми, щоб побачити відмінності в роботі функцій та підпрограм.

Знайти площі двох трикутників: перший зі сторонами a, b , c, а другий – k, l, m. Розрахунок площ здійснити у процедурі-підпрограмі за формулою Герона $S = \sqrt{p(p-a)(p-b)(p-c)}$, де S – площа трикутника; a, b, c – сторони трикутника, $p = \frac{a+b+c}{2}$ – напівпериметр.

(General) ST

```
Public Sub prim1()
Dim a As Single, b As Single, c As Single
Dim k As Single, l As Single, m As Single
Dim SA As Single, SB As Single
 ' Введення даних
a = 4: b = 5: c = 6
k = 7: l = 8: m = 9
' Виклик підпрограми
Call ST(a, b, c, SA)
Debug.Print "Площа трикутника зі сторонами:"; "a="; a, "b ="; b, "c ="; c
 Debug.Print "SA ="; SA
Call ST(k, l, m, SB)
Debug.Print "Площа трикутника зі сторонами:"; "k="; k, " l ="; l, "m ="; m
 Debug.Print "SB ="; SB
End Sub
Public Sub ST(x As Single, y As Single, z As Single, S As Single)
Dim p As Single
p = (x + y + z) / 2
S = Sqr(p * (p - x) * (p - y) * (p - z))
End Sub
```

Рис.2.1. Програма з використанням процедури-підпрограми

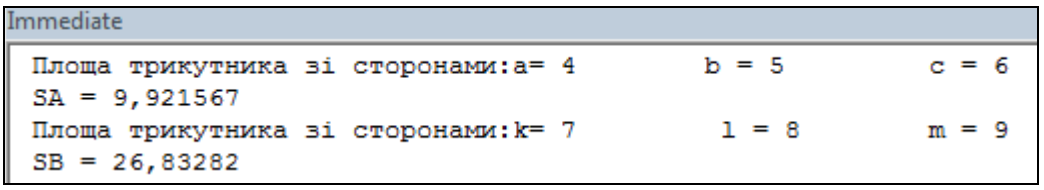


Рис. 2.2. Результати роботи програми

Як видно з наведеного прикладу, що для виклику підпрограми застосовано спеціальний оператор Call, де в списку виклику додана зміна через яку повертається результат роботи підпрограми (для першого трикутника це фактична змінна SA, а для другого – SB; формальна змінна через яку повертається значення площі – S). Результат повертається в місце виклику. В «тілі підпрограми» відсутній оператор присвоєння, через який повертається результат.

Приклад 2:

Обчислити суми і добутки елементів масиву А з шістьох елементів – А(6) і масиву В з восьми елементів – В(8). Розрахунок сум елементів здійснити у процедурі-підпрограмі. Нехай елементи масиву А: 3, 5, 4, 2, 1, 10. Нехай елементи масиву В: 1, 2, 3 , 4, 5, 6, 7, 8

Call SumDob(A(), N, SA, DA), Call SumDob(B(), M, SB, DB).

При виклику підпрограми в список фактичних параметрів включений масив і його розмірність (для першого масиву – А(), N; для другого масиву – В(), М), а також додаткові змінні через які буде повернутий результат роботи підпрограми (для першого масиву – SA – сума елементів масиву А, DA – добуток елементів масиву А; для другого масиву – SB – сума елементів масиву В, DB – добуток елементів масиву В):

В заголовку підпрограми:

Public Sub SumDob(X() As Integer, K As Integer, S As Integer, D As Long)

X() – формальний масив, а K – його розмірність. Через додаткові змінні: S – сума елементів формального масиву X і D – добуток елементів формального масиву X підпрограма поверне результати роботи в головну програму. Фактичні масиви були оголошені (описані допомогою оператора DIM) в

головній програмі: Dim A() As Integer, Dim B() As Integer. Формальний масив X() описувати в підпрограмі ще раз (наприклад, x() As Integer) непотрібно. Такий опис буде повторним і викличе помилку. Але при введенні якогось допоміжного масиву або допоміжних змінних в підпрограмі необхідно виконати їх опис в підпрограмі.

General) PrimSD

```
Public Sub PrimSD()
Dim A() As Integer, N As Integer
Dim B() As Integer, M As Integer
Dim i As Integer, SA As Integer, SB As Integer
Dim DA As Long, DB As Long
N = Val(InputBox("Введите розмірність масива N=", "Вікно введення", "6"))
M = Val(InputBox("Введите розмірність масива M=", "Вікно введення", "8"))
ReDim A(N) As Integer
ReDim B(M) As Integer
' Введення та друк масивів
     For i = 1 To N
          A(i) = Val(InputBox("Введіть число"))
    Next i
 Debug.Print "Початковий масив A"
For i = 1 To N
     Debug.Print A(i);
Next i
For i = 1 To M
          B(i) = Val(InputBox("Введіть число"))
    Next i
Debug.Print
 Debug.Print "Початковий масив B"
For i = 1 To M
     Debug.Print B(i);
Next i
  ' Пошук суми і добутку
' Виклик підпрограми
Call SumDob(A(), N, SA, DA)
Debug.Print
 Debug.Print "SA ="; SA, "DA ="; DA
' Виклик підпрограми
Call SumDob(B(), M, SB, DB)
Debug.Print
 Debug.Print "SB ="; SB, "DB ="; DB
End Sub
```

```
Public Sub SumDob(X() As Integer, K As Integer, S As Integer, D As Long)
S = 0
D = 1
For i = 1 To K
          S = S + X(i)
      D = D * X(i)
Next i
End Sub
```

Рис.2.3. Програма з використанням процедури-підпрограми

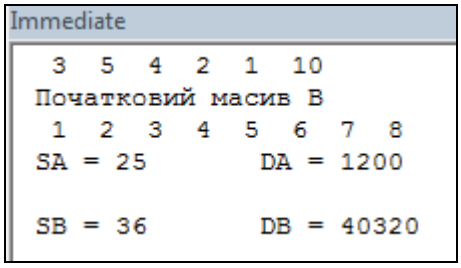


Рис. 2.4. Результати роботи програми

2.2.Опис лабораторних засобів та обладнання

Лабораторна робота виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows зі стандартним пакетом MS Office.

2.3.Заходи безпеки під час виконання лабораторної роботи

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

2.4.Послідовність виконання роботи

1. Відповідно до свого варіанту (див. додаток В) розробити програму, що реалізує поставлене завдання з використанням процедури-підпрограми (або підпрограм).
2. Продемонструвати роботу програми викладачу.
3. Зберегти робочу книгу на своєму носії інформації та зберігати до кінця семестру.
4. Оформити протокол лабораторної роботи.

2.5.Обробка та аналіз результатів. Оформлення звіту

При оформленні звіту з лабораторної роботи до заздалегідь підготовленого протоколу (звіт повинен містити: тему та мету лабораторної

роботи; короткі теоретичні відомості за зазначеними питаннями; індивідуальне завдання; пояснення до виконання завдання (при необхідності); результати роботи (та необхідні висновки) додаються роздруковані аркуші з результатами виконаної роботи: лістинг програмного коду та результатів розрахунків.

2.6. Контрольні запитання

1. Який формат має процедура-підпрограма?
2. Що таке формальні і фактичні параметри? Які вимоги до списків формальних і фактичних параметрів?
3. Навіщо застосовується оператор Exit Sub в процедурі-підпрограми?
4. Як повертається результат роботи процедури-підпрограми в головну програму?
5. Як здійснити виклик підпрограми з головної програми?
6. В чому полягає відмінність підпрограми і функції?
7. Для чого записується у списку формальних параметрів ключове слово **ByVal**?
8. Для чого записується у списку формальних параметрів ключове слово **ByRef**?
9. Для чого записується у рядку заголовка функції ключове слово **Public**?
10. Для чого записується у рядку заголовка функції ключове слово **Private**?
11. Чому у заголовку підпрограми відсутній опис типу її імені?

ЛАБОРАТОРНА РОБОТА № 3.
СТВОРЕННЯ КОРИСТУВАЦЬКОГО ІНТЕРФЕЙСУ. ФОРМИ КОРИСТУВАЧА У VBA.

Мета та основні завдання: дослідити можливості VBA MS Excel при роботі з формами користувача. Набути вмінь роботи з формами користувача. Вивчити використання властивостей і методів об'єкта UserForm і створення процедур обробки подій для конкретної форми і її елементів управління.

Завдання. Вивчити наведені нижче теоретичні питання та стисло описати у протоколі питання:

– поняття властивостей, методів і подій об’єктів VBA;

– етапи створення форми користувача;

– використання властивостей і методів об'єкта UserForm;

– створення процедур обробки подій для конкретної форми і її елементів управління.

3.1. Короткі теоретичні відомості

Практично у всіх додатках MS Office використовуються призначені для користувача діалогові вікна. Діалогові вікна в VBA називаються формами (об'єкт UserForms). Кожному об'єкту UserForm притаманні певні властивості, методи і події, які він успадковує від класу об'єктів UserForms. Загальні відомості про об'єктну модель Excel і інших додатків Office наведені в **Додатку Д**. Діалогові вікна (форми) і елементи управління складають основу сучасного візуального інтерфейсу. Всі елементи управління і технологія роботи з ними в основному стандартизовані і схожі для різних платформ і програмних середовищ. Ці об'єкти поміщені в спеціальну бібліотеку MSForms.

Форма – це головний об'єкт, що створює візуальну основу додатку. За своєю суттю форма є об'єктом, котрий являє собою нестандартне вікно, в якому можна розміщувати різні керуючі елементи. Для створення форми

необхідно виконати команду **Insert (Вставка)** → **UserForm** (або кнопка *Insert UserForm* на панелі інструментів редактора *Visual Basic* –). Відкриється вікно конструктора – дизайнера форм (**Form designer**), в якому буде представлено порожнє сіре вікно форми (за замовчуванням воно має назву UserForm1) і поряд з ним панель з набором елементів управління Toolbox (рис. 3.1).

Далі можна при необхідності змінювати розміри форми, розміщувати на ній елементи управління і т.п.

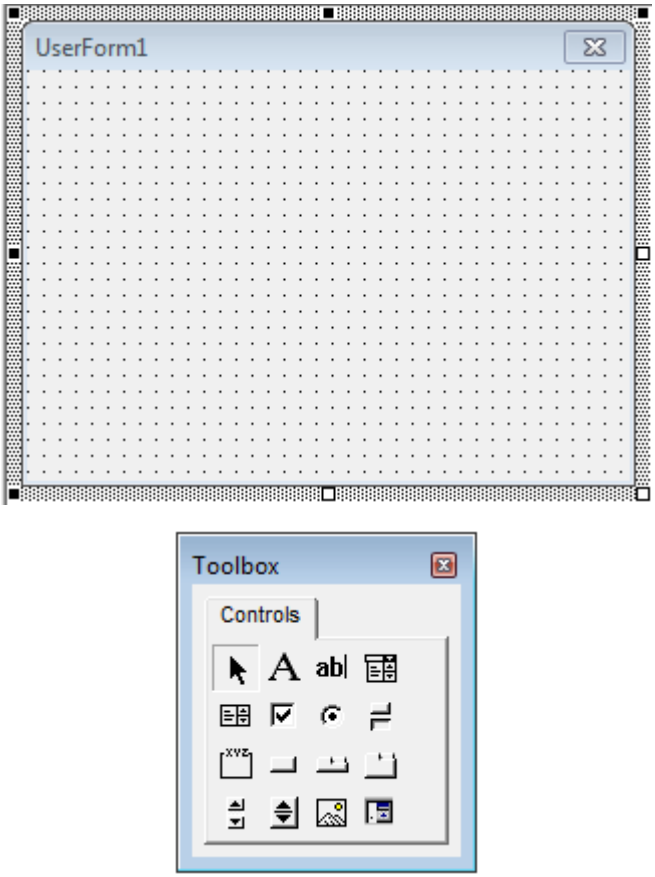
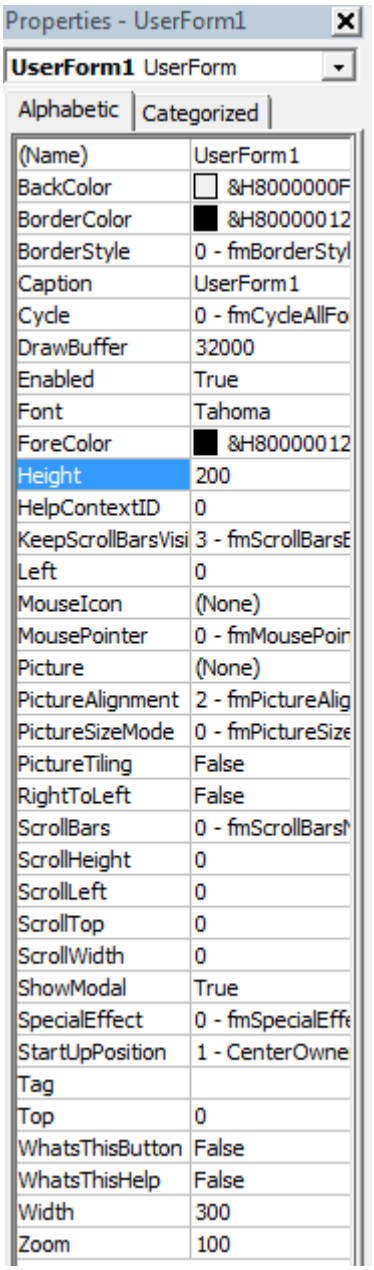


Рис. 3.1. Вікно конструктора – дизайнера форм (**Form designer**) і панель з набором елементів управління Toolbox.

Змінити розмір форми можна:

- за допомогою миші, натиснувши на одному з маркерів і перетягнувши його;
- за допомогою введення відповідних значень у вікні властивостей (Properties_User Form), визначаючи значення властивостей Width (Ширина) і Height (Висота).



У вікні властивостей (рис. 3.2) представлені всі властивості форми.

Дуже зручно, що для форм і елементів управління можна налаштовувати властивості за допомогою графічного інтерфейсу вікна властивостей. Ця обставина різко зменшує кількість програмного коду, яки потрібно написати та ввести вручну.

Управління формою здійснюється за допомогою використання властивостей і методів об'єкта UserForm, а також створення процедур обробки подій для конкретної форми і її елементів управління.

Властивості об'єкта форма можна визначити чи змінити у вікні **Properties (Властивості)** редактора VB або програмними засобами. Не всі властивості форми доступні для змін обома способами: деякі з них можуть бути встановлені тільки у вікні **Properties (Властивості).** У таблиці 3.1 наведені основні властивості об'єкта форма.

Рис. 3.2. Вікно властивостей форми.

Перехід у вікно програмного коду для цієї форми можна виконати за допомогою функціональної клавіші **F7** (або подвійним клацанням мишею по формі), а повернення назад у вікно дизайнера форм за допомогою комбінації клавіш **Shift + F7**.

Примітка. Якщо вікно **Properties (Властивості)** не відкрито, то то його можна відкрити: за допомогою команди **View (Вид)** ⟶ **Properties Window**; функціональної клавіші **F4**; кнопки на панелі інструментів редактора Visual Basic – .

Таблиця 3.1.

Найчастіше вживані властивості об'єктів UserForm

Властивість	Опис
1	2
ActiveControl	Повертає посилання на елемент управління у формі, який в поточний момент знаходиться у фокусі. Має атрибут тільки для читання.
BackColor	Ціле значення типу Long, що представляє конкретний колір фону форми. Найпростіший спосіб завдання цієї властивості – використання вікна Properties для вибору потрібного кольору (при необхідності номер кольору можна скопіювати з вікна Properties в текст програми).
ForeColor	Аналогічно властивості BackColor, але встановлює колір зображення – зазвичай тексту об'єкта форми.
Caption	Текст, який відображається як заголовок форми. Має атрибут читання/запису.
Cycle	Визначає, чи викликає натискання клавіші <Tab> циклічний перехід фокусу по всіх елементах управління у всіх групових рамках і у всіх сторінках багатосторінкових елементів управління або тільки в поточній рамці або сторінці. Може містити одну з двох вбудованих констант: fmCycleAllForms або fmCycleCurrentForm. Має атрибут читання/запису.
Enabled	Визначає доступність елемента. Містить значення типу Boolean (можливі значення True/False). Якщо значення дорівнює False, жоден з елементів управління форми не доступний. Має атрибут читання/запису.

Закінчення табл. 3.1.

1	2
Font	Повертає посилання на об'єкт Font, за допомогою якого можна вибрати характеристики шрифту форми або елемента управління.
Scrollbars	Визначає, які смуги прокрутки присутні на екрані. Значення даного поля може дорівнювати значенню однієї з наступних констант: fmScrollbarsNone - смуги прокрутки відсутні; fmScrollbarsHorizontal - присутній тільки горизонтальна смуга прокрутки; fmScrollbarsVertical - присутній тільки вертикальна смуга прокрутки; fmScrollbarsBoth - присутні обидві смуги прокрутки.

Сітка з точок (рис. 3.1), що знаходиться на поверхні форми, допомагає вирівнювати і визначати розміри елементів управління у формі. Вона відображається тільки в режимі конструктора. У VBA можна вмикати і вимикати прив'язку до сітки, налаштовувати її розмір і приховувати її.

Для цього необхідно виконати команду **Tools (Сервис)** → **Options (Параметри)** редактора Visual Basic і на вкладці **General** (рис. 3.3) встановити чи видалити потрібні опції (в розділі Параметри сітки в формі – Form Grid Settings).

За допомогою вкладок діалогу вікна Option (Параметри) можна виконати налаштування вікна програмного коду для підвищення ефективності його роботи.

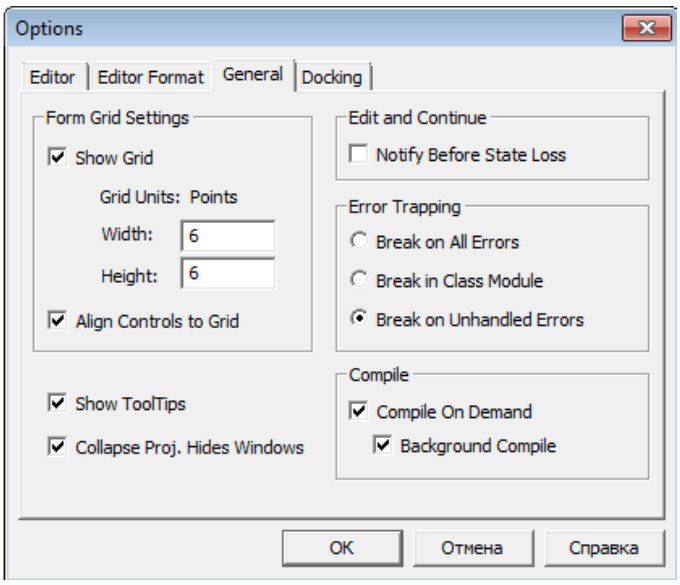


Рис. 3.3 Вікно Options, вкладка General

Методи об'єкта UserForm, що найчастіше використовуються, наведені у таблиці 3.2.

Таблиця 3.2.

Методи об'єктів UserForm, що зазвичай використовуються

Метод	Застосування
1	2
Copy	Копіює вибраний в елементі управління текст в буфер обміну Windows.
Cut	Вирізає вибраний в елементі управління текст и поміщає його в буфер обміну Windows.
Paste	Вставляє вміст буфера обміну Windows в поточний елемент управління.
Hide	Приховує об'єкт UserForm, не вивантажуючи його з пам'яті і зберігаючи значення елементів управління і будь-яких змінних, оголошених в модулі класу форми.
Show	Робить форму видимою на екрані. Якщо форма ще не завантажена в пам'ять, цей метод спочатку завантажує її.

Закінчення табл. 3.2.

1	2
PrintForm	Виводить зображення форми, включаючи будь-які дані, введені до елементів керування форми, на принтер за замовчуванням.
Repaint	Викликає перемальовування зображення форми на екрані. Цей метод можна використовувати, якщо потрібно перемалювати форму, не чекаючи закінчення звичайного періоду оновлення.
Load	Завантажує форму, але не відображає її на екрані.
Unload	Видаляє форму з пам'яті.
Move	Переміщує і змінює розміри форми.
PrintForm	Друкує зображення форми.

Найважливіша концепція VBA це – події. **Подія** (event) це – те, що відбувається з програмою і може бути розпізнано їй. Наприклад, до подій відносяться клацання мишею, натиснення на клавіші, відкриття і закриття форм, переміщення форми по екрану і тому подібне.

VBA побудований таким чином, щоб створювати програми, що керуються подіями (event-driven). Такі програми протиставляються застарілому процедурному програмуванню.

Найважливіші події форм наведені в табл. 3.3.

Таблиця 3.3.

Події об'єктів UserForm, що зазвичай використовуються

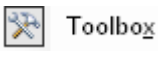
Подія	Опис
Activate	Викликається у всіх випадках, коли форма стає активним вікном. Ця подія використовується для оновлення вмісту елементів управління діалогового вікна, щоб вони відображали будь-які зміни, що відбулися, поки вікно форми було неактивним.
Click	Викликається при натисканні лівою кнопкою миші на формі (її частини, що не зайнята елементом управління).
DblClick	Викликається при подвійному клацанні лівою кнопкою миші на формі (її частини, що не зайнята елементом управління).
Deactivate	Викликається, коли форма перестає бути активним вікном.
Initialize	Викликається, коли форма завантажується в пам'ять в результаті виконання інструкції Load або методу Show. Ця подія призначена для ініціалізації елементів управління або подання форми.
Terminate	Викликається, коли форма вивантажується з пам'яті. Ця подія слугує для виконання будь-яких спеціальних завдань "прибирання", які можуть знадобитися перш, ніж змінні форми будуть відкинуті.
Resize	Викликається при зміні розміру форми

Елементи управління VBA.

Елементи управління – це спеціалізовані об'єкти, які можна розміщувати на формах VBA (і безпосередньо в документах), і які використовуються для організації взаємодії з користувачем. У VBA можна використовувати як стандартні елементи управління (CommandButton,

CheckBox, OptionButton), так і нестандартні (будь-які інші, які є на комп'ютері, наприклад, Internet Explorer, Calendar і тому подібне). Елементи управління реагують на події, які генерує користувач (натиснення на кнопку, введення значення, переміщення повзунка і тому подібне).

Додавання елементів управління на форму

Додавання елементів на форму найчастіше проводиться з дизайнера (конструктора) форм Toolbox (рис. 3.1). Якщо вікно конструктора не виведене, то його можна вивести на екран за допомогою команди **View** (**Вид**) →  або кнопки панелі інструментів редактора Visual Basic –  .

Для додавання елементів управління необхідно вибрати елемент управління в Toolbox і перетягнути його на форму. Або (що значно зручніше):

- виділити елемент управління в Toolbox (клацнувши по ньому мишею);
- потім підвести покажчик миші до області форми, де він набере вигляду хрестика із значком вибраного елементу управління;
- встановити покажчик миші в потрібну область форми, де повинен бути розташований лівий верхній кут нового елемента управління;
- клацнути лівою кнопкою миші і, утримуючи її, перетягнути покажчик миші на потрібну відстань, тим самим, задаючи потрібний розмір елементу управління;
- при відпусканні лівої кнопки миші цей елемент управління буде вставлений в форму.

Для **видалення** елементу управління слід виділити потрібний елемент управління на формі та натиснуту клавішу **Delete**.

Примітка. *Коли елемент на формі виділений (рамка об'єкту містить маленькі прямокутники) можна змінювати його розміри і переміщати за допомогою миші, а також переглядати і змінювати його властивості у вікні властивостей.*

Додавання елементів управління можна проводити і програмним способом (за допомогою методу Add() колекції Controls), проте при цьому доведеться указувати в коді програми велику кількість властивостей створюваного елементу управління , що не дуже зручно і займає багато часу.

Елементи управління в Toolbox

Елементи управління додаються на форму за допомогою панелі інструментів **Toolbox** (рис. 3.4). Ці доступні елементи управління відображаються у вікні ToolBox (їх певна кількість, та їх не можна видалити). Ці елементи іноді називають внутрішніми. Крім доступних елементів управління можна додати і інші елементи за допомогою меню

Tools ⟶ Additional Controls.

Детальна інформація про призначення елементів управління наведена в Таблиці Е1 (див. Додаток Е).

Рис.3.4. Елементи управління Toolbox

Елементи управління є об'єктами. Тому, як будь-які об'єкти, вони мають властивості, методи і події. Детальна інформація про властивості, методи і події елементів управління наведена в Таблицях Е2-Е4 (див. Додаток Е).

Для елемента управління, що намальований на формі, можна: змінювати його розмір; переміщати його; копіювати; видаляти; змінювати форматування (шрифт, стиль) чи властивості; редагувати і форматувати їх заголовки. Для цих дій застосовуються стандартні засоби і команди Windows і вікно **Властивості** редактора VBA.

Створення форми користувача

Для створення форми користувача потрібно виконати наступні дії:

1. Відкрити файл Excel для роботи (новий або створений раніше і записаний на диск).

2. Відкрити вікно редактора Visual Basic (наприклад, застосувати комбінацію клавіш **Alt + F11**. Відкриється вікно редактора Visual Basic).

3. Створити форму користувача. (**Insert** (**Вставка**) ⟶ **UserForm** (або кнопка Insert UserForm на панелі інструментів редактора Visual Basic – ). З'явиться нова екранна форма з панеллю елементів управління (рис.3.1). Розміри форми можна змінити стандартним способом (наприклад, розтягування вікна).

4. Застосувати для створеної форми необхідні властивості.

Цей і наступні етапи створення форми будемо розглядати на прикладі.

Приклад:

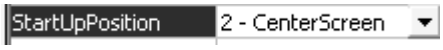
Створити екранну форму, в текстове поле якої вводиться число, розраховується корінь квадратний з цього числа і результат виводиться в нове текстове поле. На формі передбачити дві командні кнопки: кнопку запуску на розрахунок і кнопку приховування форми після отримання результату.

У вікні **Властивості** (**Properties**) встановимо тільки значення наступних властивостей:

Caption (заголовок) – Розрахунок кореня;

SpecialEffect (контур форми) – 2 (можна обирати 0 – 6);

StartUpPosition (місце розташування на екрані при виводі) – 2 (це означає, що форма буде розташована в центрі екрана Center Screen).

Примітка. Деякі властивості можна обирати з наведених списків (наприклад: 

).

Форма отримає вигляд.

5. Створити на формі елементи управління. (Додавання елементів на форму проводиться з вікна конструктора форм Toolbox рис. 3.4)

Для нашого приклада додаємо на форму такі інструменти:

5. 1. Елемент **Label (надпис).** Використовується для вставки тексту в форму (наприклад, заголовки, підказки і т.і.).

Перший надпис – **Значення** (це значення числа, з якого потрібно витягувати корінь).

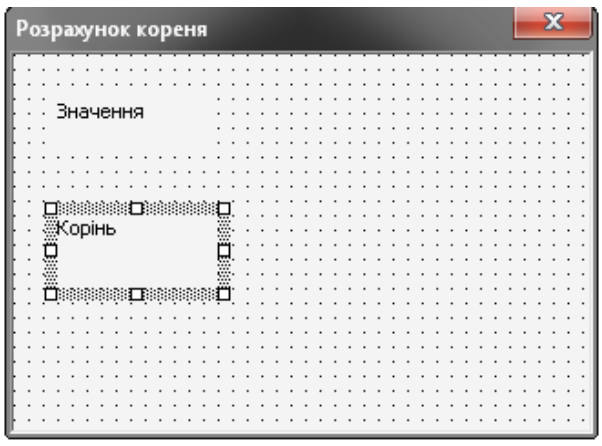
Другий надпис – **Корінь** (це результат обчислень).

У вікні **Властивості** (**Properties**) встановимо тільки значення наступних властивостей:

Caption (заголовок) – Значення (для Label1) і

Caption (заголовок) – Корінь (для Label2).

В результаті форма набула наступний вигляд:

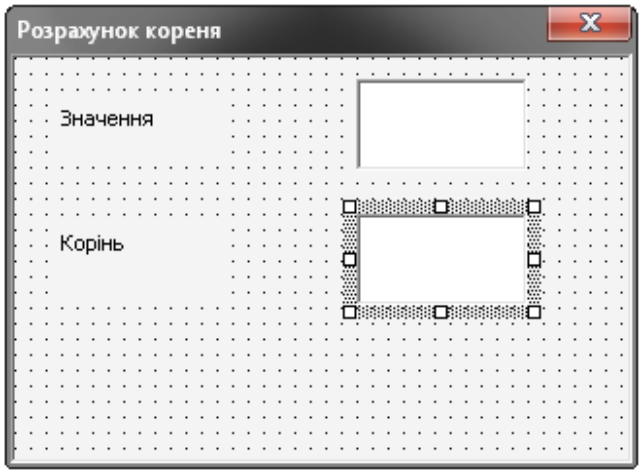


5.2. Елемент **TextBox** (**поле**). Текстове вікно яке застосовується для отримання і відображення даних у вигляді тексту. Створимо:

Перше поле – для введення значення числа, з якого потрібно витягувати корінь;

Друге поле – для виведення значення результату обчислень.

В результаті форма набула наступний вигляд:



5.3. Елемент **CommandButton** (**кнопка управління**). Вставляє у форму користувача командну кнопку. При натисненні на командну кнопку виконуються запрограмовані дії. Створимо:

Перша кнопка **Розрахунок** – (при натисканні на кнопку **Розрахунок** буде запускатися програма для розрахунку квадратного кореня).

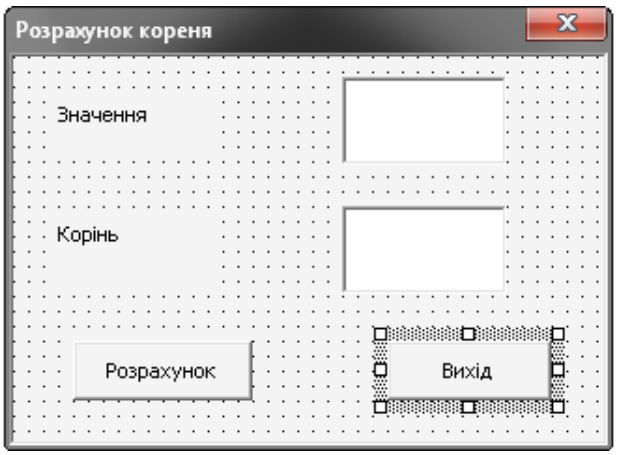
Друга кнопка – **Вихід** (при натисканні на кнопку **Вихід** форма буде приховуватись з екрана).

У вікні **Властивості** (**Properties**) встановимо тільки значення наступних властивостей:

Caption (заголовок) – Розрахунок (для CommandButton1) і

Caption (заголовок) – Вихід (для CommandButton2)

В результаті форма набула наступний вигляд:



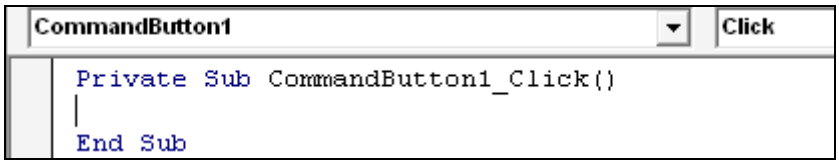
Перед написанням програми (процедури обробки подій) перевіримо працездатність форми. Для виведення форми на екран виконаємо команду **Run → Run Sub/UserForm** або натиснемо клавішу **F5**. На екрані з'явиться створена нами форма, але при натисненні на кнопку **Розрахунок** або **Вихід** нічого відбуватися не буде, оскільки не був запрограмований відгук на подію (натиснення кнопки).

6. Написати процедури обробки подій.

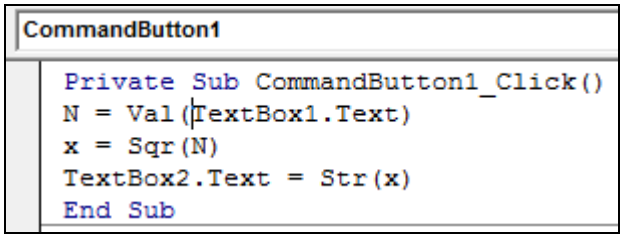
Треба запрограмувати дві події, які будуть відбуватися при натисканні на кнопки **Розрахунок** і **Вихід**. Для цього:

6.1. У формі потрібно двічі клацнути на кнопці **Розрахунок**, щоб вивести заготовку програми, що пов'язана з цією командною кнопкою.

На екрані з'явиться шаблон програми:



Між рядками Private Sub …End Sub запишемо текст програми.



6.2. Тепер запрограмуємо кнопку **Вихід**. Перейдемо знову в редактор VBA до елементу UserForm1 (через вікно **Проект або** за допомогою

комбінації клавіш **Shift + F7**) і потім двічі клацнемо по кнопці Вихід у формі. З'явиться шаблон програми.

```
Private Sub CommandButton2_Click()

End Sub
```

Між рядками Private Sub … End Sub вставимо один рядок програми UserForm1.Hide.

У цьому рядку: UserForm1 – об'єкт; Hide – це метод (метод приховує об'єкт і не вивантажує його з пам'яті і зберігає в ньому значення змінних).

CommandButton2

```
Private Sub CommandButton1_Click()
N = Val(TextBox1.Text)
x = Sqr(N)
TextBox2.Text = Str(x)
End Sub

Private Sub CommandButton2_Click()
UserForm1.Hide
End Sub
```

```
Private Sub CommandButton2_Click()
UserForm1.Hide
End Sub
```

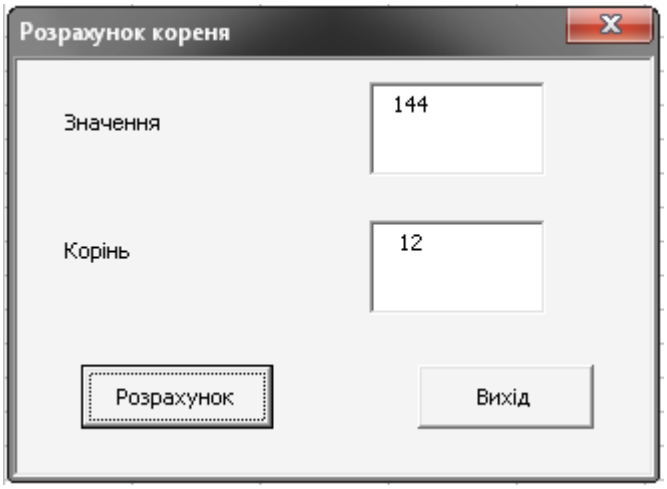
Примітка.

Перехід від екрана з формою до екрана з кодом для цієї форми виконується за допомогою клавіші **F7**, а повернення назад у вікно дизайнера форм – за допомогою комбінації клавіш **Shift + F7**.

Форма **готова** для застосування. **Запуск** здійснюється за командою **Run → Run Sub/UserForm**, натисканням клавіші **F5** або зо допомогою кнопки  на панелі інструментів редактора.

Приклад розрахунку:

Після введення значення в потрібне поле (144) і натискання на кнопку **Розрахунок** в полі Корінь отримаємо результат.



Після отримання потрібного результату і натисканні на кнопку **Вихід** форма буде прихована з екрана.

Примітка. При необхідності можна створити процедуру, яка відображає форму:

```
Public Sub aa()
UserForm1.Show
End Sub
```

Show – це метод, який дозволяє знову побачити форму на екрані.

Примітка. Крім методів і подій VBA має дві інструкції (два оператори), які корисні при роботі з формами: Load і Unload.

Формат:

Load Object

або Unload Object .

де Object – любе допустиме посилання на об'єкт UserForm. Інструкція Load завантажує об'єкт в пам'ять, а Unload прибирає вказаний об'єкт з пам'яті (форму і весь її програмний код).

Для того, щоб швидко викликати створену програму чи форму можна створити або намалювати для цього в додатку Excel спеціальну кнопку. За допомогою цієї кнопки запуск можна здійснювати безпосередньо з додатку Excel.

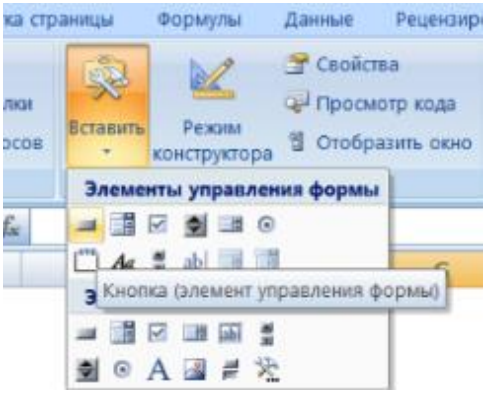
У версії 2007 є дві принципово схожі можливості створити кнопки за допомогою елементів управління і елементів ActiveX. Для кнопки ActiveX

можна задати колір, фон і т.п.

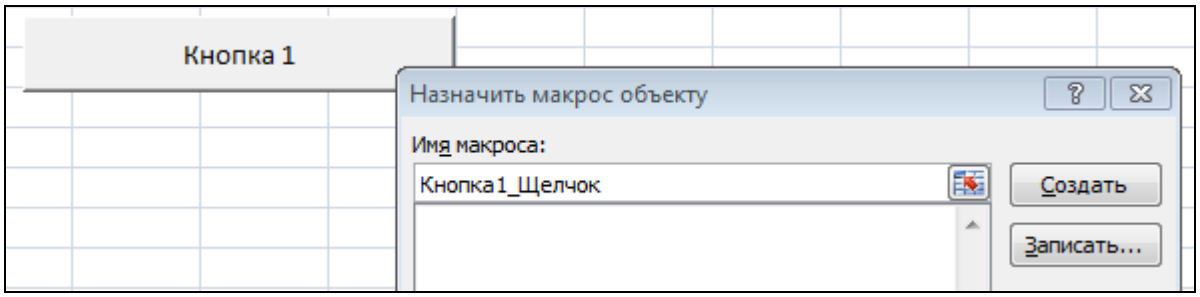
Найпростіший спосіб для створення кнопки:

1. Потрібно перейти в додаток Excel на будь-який лист.

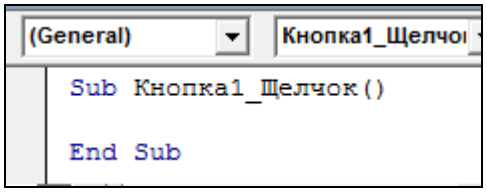
2. На панелі елементів управління (**Разработчик** → в групі **Элементы Управления** → кнопка **Вставить**) вибрати інструмент CommandButton – і намалювати кнопку на листі Excel.



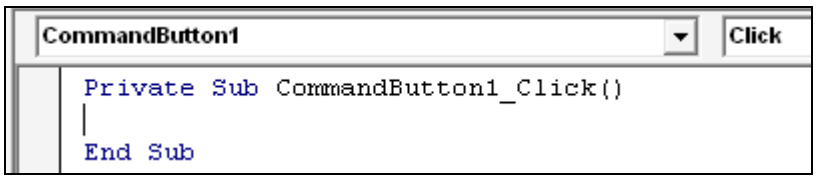
З'явиться інструмент малювання (курсор перетворився на хрестик). Кнопку можна намалювати в потрібному місці і розтягнути до потрібних розмірів.



Відразу відкриється віконце макросів, де до кнопки можна прив’язати існуючий макрос чи створити новий.

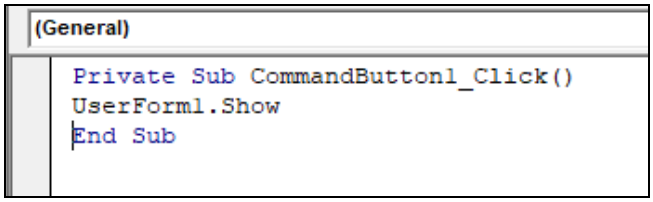


або



Між рядками Sub … End Sub (SubPrivate … End Sub) вставимо один рядок

програми наприклад, UserForm1.Show, що означає показати форму UserForm1 на екрані.



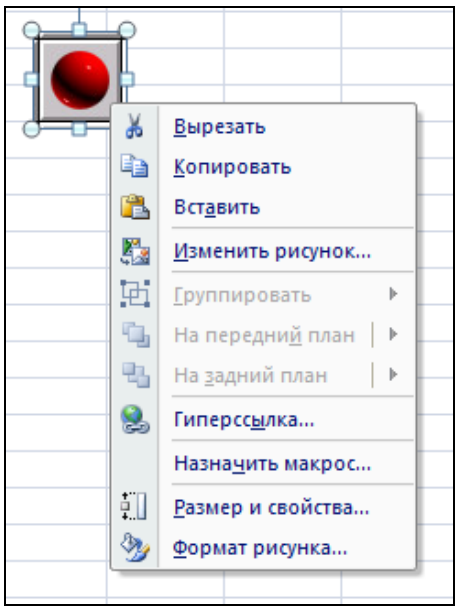
У цьому рядку: UserForm1 – об'єкт; Show – це метод (метод, який виводить форму на екран).

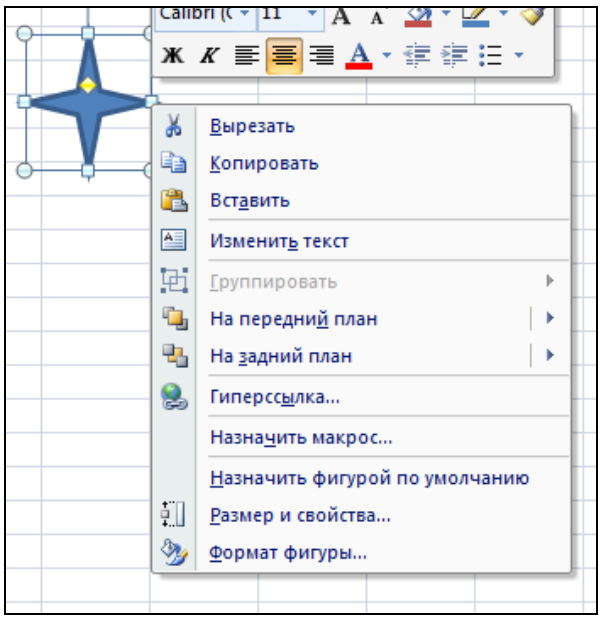
З використанням правої кнопки миші можна змінити назву кнопки, шрифт і інші властивості.

Кожне натискання на кнопку буде викликати форму, в якій знаходяться попередні дані. Для нових розрахунків у відповідні поля треба вводити нові значення.

Для користувачів Excel 2007 і вище в якості кнопки можна використати автофігуру (вставка якої в лист Excel здійснюється: вкладка **Вставка** (Insert) – **Фігури** (Shapes)) чи рисунок (вставка якого в лист Excel здійснюється: вкладка **Вставка** (Insert) – **Рисунок** (Picture)). Можна обрати необхідну автофігуру або рисунок з наявних на комп'ютері.

Для **автофігури** і **рисунка** процедура призначення макросу абсолютно однакова: слід натиснути правою кнопкою миші на елементі управління фігурі або зображенні і обрати пункт **Назначить макрос**, як це показано нижче (на першому рисунку кнопка – рисунок, а на другому – автофігура).





Слід зауважити, що можна призначити макрос не тільки зазначеним елементам, а й діаграмі, елементу Напис, об’єкту WordArt, рисунка SmartArt.

Приклад 2: Створити форму для отримання добутку від’ємних елементів масиву. Кількість елементів масиву вводиться через форму. Результат роботи програми також виводиться в форму. Форма має дві командні кнопки: Запуск і Вихід.

1. Створимо форму користувача.

2. У вікні **Властивості** (**Properties**) встановимо тільки значення наступної властивості:

Caption (заголовок) – Розрахунок добутку.

3. Створимо на формі елементи управління.

3.1. Для нашого приклада додаємо на форму такі інструменти:

Елемент **Label (напис).**

Перший напис – **Кількість елементів.**

Другий напис – **Добуток** (це результат обчислень).

У вікні **Властивості** (**Properties**) встановимо тільки значення наступних властивостей:

Caption (заголовок) – ількість елементів (для Label1) і

Caption (заголовок) – Добуток (для Label2).

3.2. Елемент **TextBox** (**поле**).

Перше поле – для введення кількості елементів;

Друге поле – для виведення значення добутку.

3.3. Елемент **CommandButton** (**кнопка управління**).

Перша кнопка – **Запуск** (при натисканні на кнопку Запуск буде запускатися програма для розрахунку добутку).

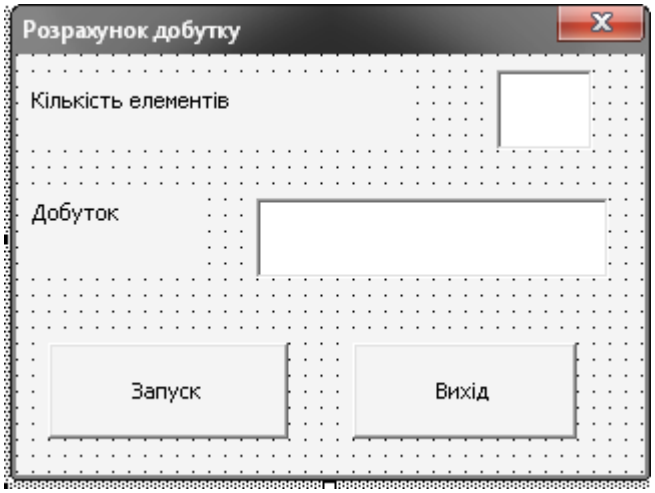
Друга кнопка – **Вихід** (при натисканні на кнопку Вихід форма буде приховуватись з екрана).

У вікні **Властивості** (**Properties**) встановимо тільки значення наступних властивостей:

Caption (заголовок) – Запуск (для CommandButton1) і

Caption (заголовок) – Вихід (для CommandButton2).

В результаті форма набула наступний вигляд:

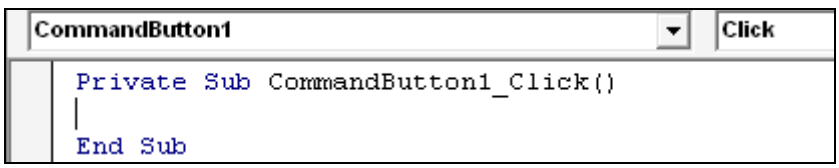


4. Написати процедури обробки подій.

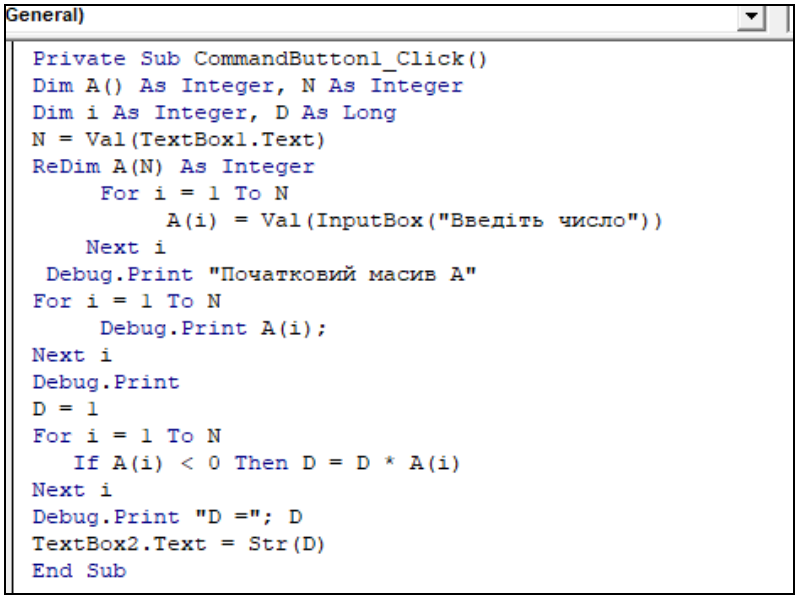
Треба запрограмувати дві події, які будуть відбуватися при натисканні на кнопки **Запуск** і **Вихід.** Для цього:

4.1. У формі потрібно двічі клацнути на кнопці **Запуск**, щоб вивести шаблон програми, що пов'язана з цією командною кнопкою.

На екрані з'явиться шаблон програми:



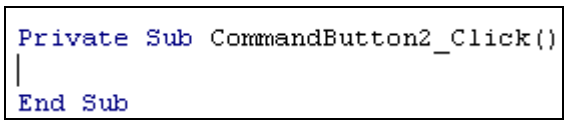
Між рядками Private Sub …End Sub запишемо текст програми.



```
General)
Private Sub CommandButton1_Click()
Dim A() As Integer, N As Integer
Dim i As Integer, D As Long
N = Val(TextBox1.Text)
ReDim A(N) As Integer
     For i = 1 To N
          A(i) = Val(InputBox("Введіть число"))
    Next i
 Debug.Print "Початковий масив A"
For i = 1 To N
     Debug.Print A(i);
Next i
Debug.Print
D = 1
For i = 1 To N
   If A(i) < 0 Then D = D * A(i)
Next i
Debug.Print "D ="; D
TextBox2.Text = Str(D)
End Sub
```

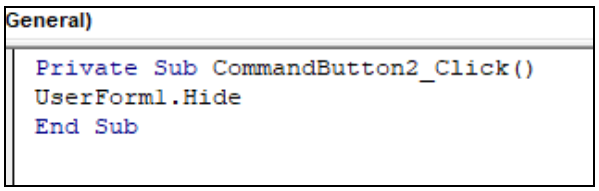
Як видно з тексту програми введений масив буде виводитись у вікно Immediate (за допомогою оператора Debug.Print), а значення добутку від'ємних елементів масиву одночасно у вікно Immediate та на форму.

4.2. Тепер запрограмуємо кнопку **Вихід**. Перейдемо знову в редактор VBA до елементу UserForm1 (через вікно Проект або за допомогою комбінації клавіш **Shift + F7**) і потім двічі клацнемо по кнопці **Вихід** у формі. З'явиться шаблон програми.



```
Private Sub CommandButton2_Click()

End Sub
```

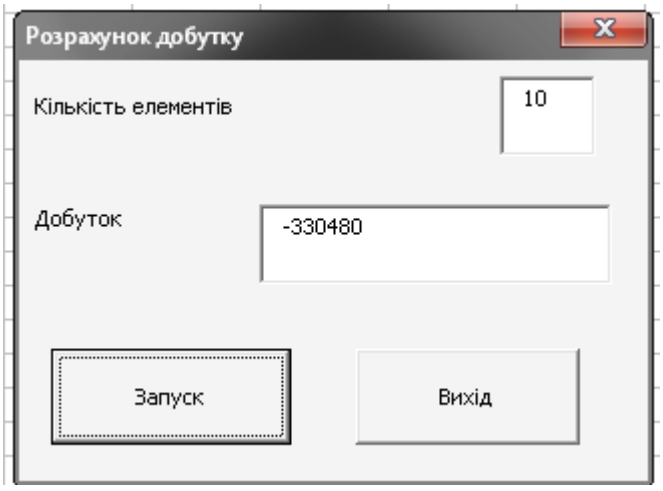
Між рядками Private Sub …End Sub вставимо один рядок програми:



```
General)
 Private Sub CommandButton2_Click()
 UserForm1.Hide
 End Sub
```

Після введення кількість елементів масиву (у прикладі 10), значень цих елементів (у прикладі - 6 -12 4 8 -45 -6 9 -2 90 -51) і натискання кнопки **Запуск** отримаємо наступні результати.

Результати роботи програми:



Натискання на кнопку **Вихід** прибере форму з екрана.

При бажанні можна запрограмувати і кнопку запуску форми з Excel.

Примітка.

Рекомендується відразу після створення елементів управління на формах привласнювати їм значущі імена. Інакше об'єкти отримують імена за замовчуванням типу TextBox1, CommandButton1 і т.д.

Інформацію на форму можна поміщати не тільки в текстове поле за допомогою елемента управління TextBox, а і просто на форму в заздалегідь підготовлене місце за допомогою елемента управління Label.

Приклад 3:

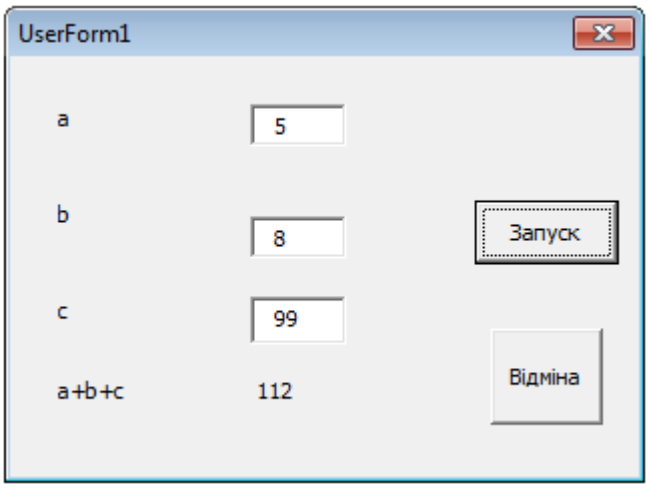
Потрібно створити форму, за допомогою якої підраховуються сума значень трьох змінних a, b і c.

Для введення значень цих змінних підготуємо на формі текстові поля TextBox1 (для введення значення a), TextBox2 (для введення значення b) і TextBox3 (для введення значення c) за допомогою елемента управління TextBox.

За допомогою елемента управління Label підготуємо наступні надписи: Label1 – для позначення текстового поля для змінної a, Label2 – для позначення текстового поля для змінної b, Label3 – для позначення текстового поля для змінної c, Label4 – для позначення суми змінних a + b + c і Label5 – для позначення місця, де буде записуватись значення суми цих елементів (під текстовими полям для введення значень).

Також зробимо дві командні кнопки: **Запуск** і **Відміна**.

Розроблена форма має вигляд:



Тести процедур подій для командних кнопок мають вигляд:

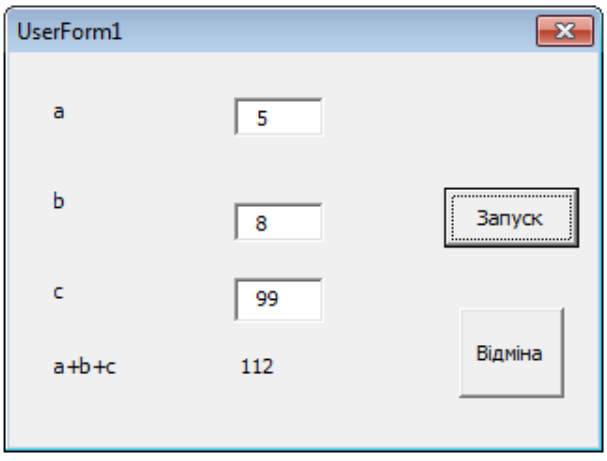
Для кнопки **Запуск**.

```
(General)                                          CommandB
Private Sub CommandButton1_Click()
Dim a As Integer, b As Integer, c As Integer, t As Integer
a = Val(TextBox1.Text)
b = Val(TextBox2.Text)
c = Val(TextBox3.Text)
t = a + b + c
Label5.Caption = Str(t)
End Sub
```

Для кнопки **Відміна.**

```
(General)
Private Sub CommandButton2_Click()
UserForm1.Hide
End Sub
```

Результати роботи:



Приклади роботи з формами наведено у додатку Ж.

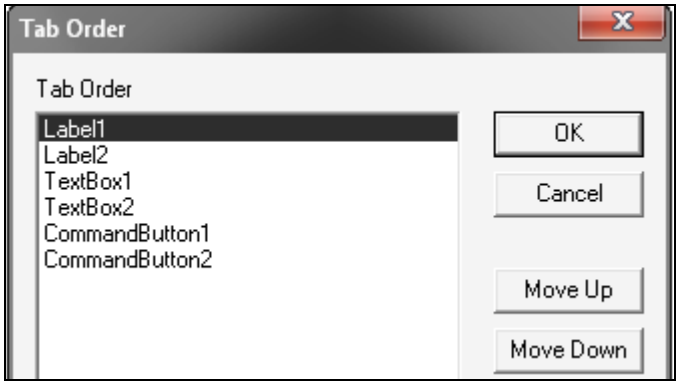
Керування послідовністю переходу

Коли форма активна, то вона поводить себе аналогічно іншим діалоговим вікнам Windows. Користувач діалогового вікна може здійснювати перехід від одного елемента управління до іншого за допомогою клавіші **Tab**. Для переходу між елементами управління в зворотному напрямку можна використати комбінацію клавіш **Shift + Tab**.

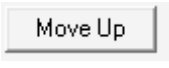
Порядок, в якому елементи управління стають активними при натисканні клавіші **Tab** або комбінації клавіш **Shift + Tab** називається послідовністю переходу. Зазвичай потрібно щоб перехід між елементами управління виконувався зліва направо і зверху вниз. Редактор Visual Basic назначає послідовність переходу між елементами управління по мірі додавання їх у форму. Але буває, що необхідно переміщувати елементи для придання формі більш привабливого вигляду. Тоді за замовчуванням не забезпечується потрібний логічний перехід між елементами вікна.

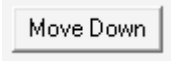
Для зміни послідовності переходу необхідно:

1. Виконати команду **View (Вид)** → **Tab Order.** Редактор Visual Basic покаже вікно, де відображені всі елементи управління в тій послідовності, яка діє на даний момент.



2. В наведеному списку слід обрати елемент, послідовність якого потребує змін.

3. За допомогою кнопок 

і 

елемент ставиться на потрібне місце. Також можна перемістити і інші елементи списку.

4. Для підтвердження нової послідовності слід клацнути по клавіші ОК.

3.2.Опис лабораторних засобів та обладнання

Лабораторна робота виконується на персональному комп'ютері стандарту IBM PC під керуванням операційної системи MS Windows зі стандартним пакетом MS Office.

3.3.Заходи безпеки під час виконання лабораторної роботи

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

3.4.Послідовність виконання роботи

1. Відповідно до свого варіанту (див. додаток К) створити і провести тестування форми користувача, що реалізує поставлене завдання.
2. Продемонструвати роботу форми користувача викладачу.
3. Зберегти робочу книгу на своєму носії інформації та зберігати до кінця семестру.
4. Оформити протокол лабораторної роботи.

3.5.Обробка та аналіз результатів. Оформлення звіту

При оформленні звіту з лабораторної роботи до заздалегідь підготовленого протоколу (звіт повинен містити: тему та мету лабораторної роботи; короткі теоретичні відомості за зазначеними питаннями; індивідуальне завдання; пояснення до виконання завдання (при необхідності); результати роботи (та необхідні висновки) додаються роздруковані аркуші з результатами виконаної роботи: лістинги.

3.6. Контрольні запитання

1. Як чином вивести на екран вікно форми?
2. Навіщо використовується панель з набором елементів управління Toolbox?
3. Як змінюють розміри форми і де їх можна визначити?
4. Поясніть призначення методів форми: Hide і Show?
5. Які події для форм події Ви знаєте?
6. Яке призначення елемента управління Label?
7. Яке призначення елемента управління CommandButton?
8. Які основні події елемента управління CommandButton Ви знаєте?
9. Яке призначення елемента управління TextBox?
10. Яке призначення елемента управління CheckBox?
11. Яке призначення елемента управління Frame?
12. Яке призначення елемента управління ToggleButton?
13. Яке призначення елемента управління OptionButton?
14. Яке призначення елемента управління ListBox?
15. Яке призначення елемента управління ScrollBar?
16. Яке призначення елемента управління MultiPage?
17. Яке призначення елемента управління Image?

Список рекомендованої літератури

1. Харрис М. Освой самостоятельно программирование MS Excel 2000 для за 21 день. – М.: Вильямс (SAMS), 2000. – 880 с.
2. Орвис В. Excel для ученых, инженеров и студентов. – К.: Юниор, 1999. – 528 с.
3. Гарнаев А. Самоучитель VBA. Технология создания пользовательских приложений. – СПб.: BHV, 1999. – 512 с.
4. Microsoft Visual Basic 6.0 для профессионалов. Шаг за шагом: Практ. пособие./Пер. с англ. – М.: Издательство ЭКОМ, 2001. – 720 с.
5. Уокенбах, Джон Профессиональное программирование на VBA в Excel 2002.: Пер.сангл. – М.:Издательский дом «Вильямс» , 2003. –784с.
6. Уокенбах, Джон Профессиональное программирование на VBA в Excel 2003.: Пер.сангл. – М.:Издательский дом «Вильямс» , 2005. –800с.
7. Культин Н. Б. Visual Basic. Освой на примерах. – СПб.: БХВ-Петербург, 2004. – 288 с.
8. VisualBasic.NET: учебный курс / В.Долженков, М.Мозговой. – СПб.: Питер, 2003. – 464 с.
9. John Walkenbach, Excel 2010 Power Programming with VBA.–Published by Wiley Publishing, Inc., Indianapolis, Indiana Published simultaneously in Canada, 2010. – p. 1007.
10. Дудзяний І.М. Програмування мовою Visual Basic/VBA. Навчальний посібник. – Львів: Видавничий центр ЛНУ імені Івана Франка, 2004. – 240 с.
11. Демидова Л.А., Пылькин А.Н. Программирование в среде Visual Basic for Applications: Практикум. – М.: Горячая линия – Телеком, 2004. – 175 с.

Додаток А. Заходи безпеки під час виконання лабораторних робіт

Цикл лабораторних робіт з дисципліни «Інформаційні технології» виконуються в комп'ютерному класі кафедри кібернетики хіміко-технологічних процесів хіміко-технологічного факультету, де розміщені персональні комп'ютери. Обладнання живиться електричним струмом напругою 220 В. Тому при виконанні лабораторних робіт слід дотримуватися заходів безпеки наступних інструкцій.

ІНСТРУКЦІЯ

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

1. Знання і суворе дотримання цих правил є обов'язковим для всіх осіб, допущених до роботи на ПЕОМ. Доведення їх до кожного зі студентів підтверджується особистим підписом кожного з них у контрольному листі з техніки безпеки. Особи, які не одержали такого інструктажу та не поставили підпис у контрольному листі з техніки безпеки, до роботи на ПЕОМ не допускаються.
2. Всі роботи в учбових лабораторіях кафедри кібернетики ХТП проводяться лише з дозволу викладача або співробітника кафедри.
3. Під час проведення занять в учбовій лабораторії не повинні знаходитися сторонні особи, в тому числі студенти інших груп. Студенти не повинні самовільно залишати учбову лабораторію під час занять.
4. При роботі на ПЕОМ треба пам'ятати, що в них використовується напруга, небезпечна для життя.
5. Всі особи, працюючі в учбових лабораторіях кафедри КХТП повинні бути ознайомлені з правилами надання першої медичної допомоги при ураженні електричним струмом.

6. Перед вмиканням ПЕОМ кожен з працюючих повинен отримати дозвіл викладача або співробітника кафедри.
7. У випадках виникнення короткого замикання, горіння, диму, вогню в апаратурі, пристрій необхідно негайно вимкнути з мережі та доповісти викладачеві або співробітникові кафедри. Самостійні дії по усуненню пошкодження забороняються.
8. У випадку виходу з ладу обладнання або програмного забезпечення, що зумовлені іншими причинами, доповісти викладачеві або співробітникові кафедри. Вимикати апаратуру при цьому не дозволяється. Самостійні дії по усуненню пошкодження забороняються.
9. Працюючі в учбових лабораторіях кафедри кібернетики ХТП несуть майнову та адміністративну відповідальність за збереження та використання обладнання, наданого для їх праці.
10. Категорично забороняється:

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

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

ІНСТРУКЦІЯ

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

1. Всі студенти повинні знати та ретельно виконувати «Загальні правила пожежної безпеки в КПІ ім. Ігоря Сікорського».
2. Завідуючий кафедрою та завідуючий лабораторією відповідають за забезпечення пожежної безпеки всіх приміщень кафедри та за справність протипожежного обладнання та сигналізації.
3. Все електричне обладнання, яке знаходиться в лабораторіях та приміщеннях кафедри, повинно мати заземлення.
4. В усіх приміщеннях повинно дотримуватись чистоти, не займати приміщення непотрібними меблями, обладнанням та матеріалами.
5. Всі двері основних та додаткових виходів утримувати у стані швидкого відкривання.
6. Зберігання та використання горючих та легкоспалахуючих рідин у приміщеннях кафедри забороняється.
7. Ремонт електричного обладнання проводити у строгій відповідності з правилами пожежної безпеки.
8. Всі електрозахисти повинні знаходитися у закритому положенні, не займаними сторонніми предметами.
9. Коридори, проходи, тамбури, евакуаційні виходи та підходи до першочергових засобів пожежогасіння, а також комунікаційні ніші повинні бути постійно вільними, чистими та нічим не зайнятими.
10. Відповідальні особи перед закриттям приміщень повинні ретельно оглянути їх, забезпечити прибирання виробничих відходів, перевірити якість перекриття води, газу, відключити напругу електромережі, перевірити стан пожежної сигналізації та засобів пожежогасіння.
11. Від усіх приміщень мати два комплекти ключів. Один комплект

здавати черговому, а інший – зберігати в певному місці, яке відомо обслуговуючому персоналу.

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

Додаток Б. Індивідуальні завдання лабораторної роботи №1

*Тема: «**Користувацькі функції**»*

Набір індивідуальних завдань 1

Розробити програму, що реалізовує поставлене завдання, використовуючи процедуру функцію (функції).

1. В довільному масиві ***E*** з ***n*** елементів визначити добуток непарних елементів за допомогою процедури-функції.
2. В довільному масиві ***A*** з ***n*** елементів визначити кількість від'ємних елементів, кратних чотирьом за допомогою процедури-функції.
3. В довільному масиві ***F*** з ***n*** елементів визначити суму елементів, значення яких не перевищують заданої константи: $F_i \le \alpha$ за допомогою процедури-функції.
4. В довільному масиві ***Q*** з ***n*** елементів визначити добуток елементів, значення яких знаходяться в межах заданого інтервалу: $\alpha \le Q_i \le \beta$ за допомогою процедури-функції.
5. В довільному масиві ***A*** з ***n*** елементів визначити найменший елемент з парних елементів масиву за допомогою процедури-функції.
6. В довільному масиві ***Q*** з ***n*** елементів визначити суму парних елементів, значення яких знаходяться в межах заданого інтервалу: $\alpha \le Q_i \le \beta$ за допомогою процедури-функції.
7. В довільному масиві ***E*** з ***n*** елементів визначити кількість непарних елементів, значення яких не менше заданої константи ($E_i \ge \alpha$) за допомогою процедури-функції.
8. В довільному масиві ***F*** з ***n*** елементів визначити добуток парних елементів за допомогою процедури-функції.
9. В довільному масиві ***A*** з ***n*** елементів визначити найменший елемент з додатних елементів масиву за допомогою процедури-функції.

10. В довільному масиві ***A*** з ***n*** елементів визначити найбільший елемент з елементів, значення яких знаходяться в межах заданого інтервалу: $\alpha \le Q_i \le \beta$ за допомогою процедури-функції.
11. В довільних масивах ***Q*** з ***n*** елементів і ***F*** з ***n*** елементів визначити найменші елементи за допомогою процедури-функції.
12. В довільних масивах ***Q*** з ***n*** елементів і ***F*** з ***n*** елементів визначити добутки їх елементів за допомогою процедури-функції.
13. В довільних масивах ***Q*** з ***n*** елементів і ***F*** з ***n*** елементів визначити суми елементів кратним трьом за допомогою процедури-функції.
14. В довільних масивах ***A*** з ***n*** елементів і ***E*** з ***n*** елементів визначити добутки елементів кратним чотирьом за допомогою процедури-функції.
15. В довільних масивах ***Q*** з ***n*** елементів і ***F*** з ***n*** елементів визначити суми елементів, значення яких знаходяться в межах заданого інтервалу: $\alpha \le Q_i \le \beta$ за допомогою процедури-функції.
16. В довільному масиві ***A*** з ***n*** елементів визначити найменший елемент з непарних елементів масиву за допомогою процедури-функції.
17. В довільному масиві ***A*** з ***n*** елементів визначити суму від'ємних елементів, кратних чотирьом за допомогою процедури-функції.
18. В довільних масивах ***Q*** з ***n*** елементів і ***F*** з ***n*** елементів визначити добутки їх додатних елементів за допомогою процедури-функції.
19. В довільному масиві ***A*** з ***n*** елементів визначити найменший елемент з елементів масиву кратних 5 за допомогою процедури-функції.
20. В довільному масиві ***B*** з ***n*** елементів визначити найбільший парний елемент за допомогою процедури-функції.
21. В довільному масиві ***Q*** з ***n*** елементів визначити суму парних від'ємних елементів за допомогою процедури-функції.
22. В довільному масиві ***B*** з ***n*** елементів визначити найбільший непарний елемент за допомогою процедури-функції.

23. В довільному масиві ***A*** з ***n*** елементів визначити кількість нулів за допомогою процедури-функції.
24. В довільному масиві ***Q*** з ***n*** елементів визначити суму непарних елементів, значення яких не менше заданої константи ($E_i \geq \alpha$) за допомогою процедури-функції.
25. В довільному масиві ***F*** з ***n*** елементів визначити суму квадратів елементів за допомогою процедури-функції.

Набір індивідуальних завдань 2

1. Заданий довільний масив A(n). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає кількість елементів, значення яких по модулю не менше 2.
2. Заданий довільний масив A(m, n). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найбільший елемент.
3. Заданий довільний масив A(m, n). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає суму и кількість від'ємних елементів, кратних трьом.
4. Заданий довільний масив A(n). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає суму елементів, значення яких по модулю не перевищують 5.
5. Заданий довільний масив ***A(m, n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає суму від'ємних елементів, кратних чотирьом.
6. Заданий довільний масив A(n). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найбільший елемент, кратний трьом.

7. Заданий довільний масив A(m, n). Розробити програму, яка визначає в заданому масиві середнє значення додатних елементів, кратних п’яти, за допомогою процедури-функції.

8. Заданий довільний масив A(n). Розробити програму, яка визначає суму додатних елементів заданого масиву, кратних двом, за допомогою процедури-функції.

9. Заданий довільний масив A(m, n). Розробити програму, яка визначає найменший по модулю елемент в заданому масиві за допомогою процедури-функції.

10. Заданий довільний масив ***A(n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найменший елемент, кратний п’яти.

11. Заданий довільний масив ***A(m, n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найбільший по модулю елемент.

12. Заданий довільний масив ***A(n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає середнє значення з додатних елементів, кратних чотирьом.

13. Заданий довільний масив ***A(m, n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає добуток елементів, значення яких знаходяться в межах заданого інтервалу: $\alpha \le a_{ij} \le \beta$.

14. Заданий довільний масив ***A(n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найбільший парний елемент.

15. Заданий довільний масив ***A(m, n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає добуток елементів, кратних трьом.

16. Заданий довільний масив ***A(n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає кількість елементів, які рівні нулю.

17. Заданий довільний масив ***A(m, n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найбільший по модулю від'ємний елемент.

18. Заданий довільний масив ***A(n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає середнє значення.

19. Заданий довільний масив ***A(m, n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найбільший елемент.

20. Заданий довільний масив ***A(n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає кількість елементів, значення яких по модулю не менше **2.**

21. Заданий довільний масив ***A(m, n)***. Розробити програму з використанням процедури-функції, яка в заданому масиві визначає суму и кількість від'ємних елементів, кратних трьом.

22. Заданий довільний масив A(n). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає суму елементів, значення яких по модулю не перевищують 5.

23. Заданий довільний масив A(n). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає суму елементів, значення яких по модулю не перевищують 5.

24. Заданий довільний масив ***A***(***n***). Розробити програму з використанням процедури-функції, яка в заданому масиві визначає найбільший елемент, кратний трьом.

25. Заданий довільний масив ***A(m, n)***. Розробити програму, яка визначає в заданому масиві середнє значення додатних елементів, кратних п'яти, за допомогою процедури-функції.

26. Заданий довільний масив ***A(n)***. Розробити програму, яка визначає суму додатних елементів заданого масиву, кратних двом, за допомогою процедури-функції.

27. Заданий довільний масив ***A(m, n)***. Розробити програму, яка визначає найменший по модулю елемент в заданому масиві за допомогою процедури-функції.

Набір індивідуальних завдань 3

1. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 5; $x_{i+1} = x_i + \delta x$ ($x_0 = 2.2$; $\delta x = 1.8$); $y_{i+1} = y_i + \delta y$ ($y_0 = 1.8$; $\delta y = 0.7$);

$$F(x_i, y_i) = \frac{\sqrt{x_i^2 + y_i^2}}{x_i + y_i}.$$

2. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 6; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.7$; $\delta x = 0.5$); $y_i = x_i^2$;

$$F(x_i, y_i) = (x_i * y_i) + \frac{1}{x_i + y_i}.$$

3. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 7; $x_{i+1} = x_i + \delta x$ ($x_0 = 4.5$; $\delta x = 1.5$); $y_{i+1} = y_i + \delta y$ ($y_0 = 0.2$; $\delta y = 0.3$);

$$F(x_i, y_i) = tg^2(x_i) * (x_i^2 + 0{,}5 * y_i).$$

4. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 8; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.2$; $\delta x = 0.25$); $y_i = x_i + 0.1$;

$$F(x_i,y_i) = \sin^2(x_i^2 * y_i) + 0.5 \times \cos^3(x_i * y_i^3);$$

5. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 5; $x_{i+1} = x_i + \delta x$ ($x_0 = 0.2$; $\delta x = 1.4$); $y_{i+1} = y_i + \delta y$ ($y_0 = 1.2$; $\delta y = 0.8$);

$$F(x_i,y_i) = tg^3(x * y_i) * (x_i^2 + 0{,}8 * y_i).$$

6. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 8; $x_{i+1} = x_i + \delta x$ ($x_0 = 5.1$; $\delta x = 0.9$); $y_i = 0.45 * x_i$;

$$F(x_i,y_i) = tg^2(x_i^2 + 0{,}6 * y_i).$$

7. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 10; $x_{i+1} = x_i + \delta x$ ($x_0 = 3.25$; $\delta x = 2.85$); $y_{i+1} = y_i + \delta y$ ($y_0 = 4.81$; $\delta y = 2.33$);

$$F(x_i, y_i) = e^{-1/x} + x^{0.5y} - y^{0.5x}$$

8. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 6; $x_{i+1} = x_i + \delta x$ ($x_0 = 7.25$; $\delta x = 0.25$); $y_i = 1.7/x_i$;

$$F(x_i,y_i) = (x_i^2 * y_i) + 0.5(x_i * y_i^3);$$

9. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 8; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.7$; $\delta x = 0.8$); $y_{i+1} = y_i + \delta y$ ($y_0 = 0.2$; $\delta y = 0.2$);

$$F(x_i, y_i) = 1 + e^{-1/x} + x^y$$

10. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 14; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.9$; $\delta x = 0.55$); $y_i = x_i^2 + 0{,}7*(x-1)$;

$$F(x_i, y_i) = \frac{\sqrt{1+y_i}}{x_i + \operatorname{tg} x_i} * (1 + e^{-x_i}).$$

11. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 7; $x_{i+1} = x_i + \delta x$ ($x_0 = 4.3$; $\delta x = 0.8$); $y_{i+1} = y_i + \delta y$ ($y_0 = 1.8$; $\delta y = 0.2$);

$$F(x_i, y_i) = e^{-2/x} + 2{,}8 \cdot x^{0.4y} - 1{,}3 \cdot y^{0.5x}.$$

12. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 5; $x_{i+1} = x_i + \delta x$ ($x_0 = 3.2$; $\delta x = 0.45$); $y_i = 16.7 - x_i^2$;

$$F(x_i, y_i) = \frac{\sqrt{1+2y_i}}{x_i + 0.5\operatorname{tg} x_i}.$$

13. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 9; $x_{i+1} = x_i + \delta x$ ($x_0 = 9.2$; $\delta x = 2.8$); $y_{i+1} = y_i + \delta y$ ($y_0 = 3.8$; $\delta y = 2.2$);

$$F(x_i, y_i) = (x * y)^2 + 0{,}8 \cdot x^2 - 1{,}2y^3.$$

14. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 5; $x_{i+1} = x_i + \delta x$ ($x_0 = 6.2$; $\delta x = 0.8$); $y_i = x_i^{0.5}$;

$$F(x_i, y_i) = e^{-2/x} + 0{,}5 \cdot x^{0.5} - 1{,}7 \cdot y^2.$$

15. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних

точках (x_i, y_i), де ***n*** = 7; $x_{i+1} = x_i + \delta x$ ($x_0 = 0.22$; $\delta x = 0.78$); $y_{i+1} = y_i + \delta y$ ($y_0 = 3.12$; $\delta y = 0.33$);

$$F(x_i, y_i) = 12 \cdot x_i^2 - (1{,}9 \cdot x_i^3 - 0{,}85 \cdot y_i^{0.7}) \cdot tg^2(x_i / y_i).$$

16. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 8; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.2$; $\delta x = 0.25$); $y_i = 4*x + x_i^2$;

$$F(x_i, y_i) = \sin^2(x^3 + y^4) + x^2 y^3.$$

17. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 6; $x_{i+1} = x_i + \delta x$ ($x_0 = 5.8$; $\delta x = 1.2$); $y_{i+1} = y_i + \delta y$ ($y_0 = 4.5$; $\delta y = 0.3$);

$$F(x_i, y_i) = e^{1{,}8/x_i} + 3{,}2y_i * \cos^2 x_i.$$

18. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 10; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.2$; $\delta x = 0.2$); $y_i = 2*x + x_i^2$;

$$F(x_i, y_i) = 15 + \sin^2(x^3 + y^4) + x^2 y^3.$$

19. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 5; $x_{i+1} = x_i + \delta x$ ($x_0 = 2.2$; $\delta x = 1.8$); $y_{i+1} = y_i + \delta y$ ($y_0 = 1.8$; $\delta y = 0.7$);

$$F(x_i, y_i) = \frac{\sqrt{x_i^2 + y_i^2}}{x_i + y_i}.$$

20. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 6; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.7$; $\delta x = 0.5$); $y_i = x_i^2$;

$$F(x_i, y_i) = (x_i * y_i) + \frac{1}{x_i + y_i}.$$

21. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 7; $x_{i+1} = x_i + \delta x$ ($x_0 = 4.5$; $\delta x = 1.5$); $y_{i+1} = y_i + \delta y$ ($y_0 = 0.2$; $\delta y = 0.3$);

$$F(x_i, y_i) = tg^2(x_i) * (x_i^2 + 0{,}5 * y_i).$$

22. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 8; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.2$; $\delta x = 0.25$); $y_i = x_i + 0.1$;

$$F(x_i, y_i) = \sin^2(x_i^2 * y_i) + 0.5 \times \cos^3(x_i * y_i^3);$$

23. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 5; $x_{i+1} = x_i + \delta x$ ($x_0 = 0.2$; $\delta x = 1.4$); $y_{i+1} = y_i + \delta y$ ($y_0 = 1.2$; $\delta y = 0.8$);

$$F(x_i, y_i) = tg^3(x * y_i) * (x_i^2 + 0{,}8 * y_i).$$

24. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 8; $x_{i+1} = x_i + \delta x$ ($x_0 = 5.1$; $\delta x = 0.9$); $y_i = 0.45 * x_i$;

$$F(x_i, y_i) = tg^2(x_i^2 + 0{,}6 * y_i).$$

25. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 10; $x_{i+1} = x_i + \delta x$ ($x_0 = 3.25$; $\delta x = 2.85$); $y_{i+1} = y_i + \delta y$ ($y_0 = 4.81$; $\delta y = 2.33$);

$$F(x_i, y_i) = e^{-1/x} + x^{0.5y} - y^{0.5x}$$

26. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 6; $x_{i+1} = x_i + \delta x$ ($x_0 = 7.25$; $\delta x = 0.25$); $y_i = 1.7/x_i$;

$$F(x_i, y_i) = (x_i^2 * y_i) + 0.5(x_i * y_i^3);$$

27. Розробити програму, яка реалізує поставлене завдання у вигляді процедури-функції. Обчислити і запам'ятати значення $z_i = F(x_i, y_i)$ в ***n*** різних точках (x_i, y_i), де ***n*** = 8; $x_{i+1} = x_i + \delta x$ ($x_0 = 1.7$; $\delta x = 0.8$); $y_{i+1} = y_i + \delta y$ ($y_0 = 0.2$; $\delta y = 0.2$);

$$F(x_i, y_i) = 1 + e^{-1/x} + x^y$$

Додаток В. Індивідуальні завдання лабораторної роботи №2

Розробити програму, що реалізовує поставлене завдання з використанням підпрограм-процедур.

Набір індивідуальних завдань 1

1. Розробити програму вирішення біквадратного рівняння, де визначення коренів квадратного рівняння проводиться у підпрограмі.
2. Розробити програму, що порівнює середні значення двох числових масивів різної розмірності.
3. Розробити програму, що визначає мінімальні значення двох матриць різної розмірності.
4. Розробити програму, що записує в окремий масив мінімальні елементи стовпців введеної матриці. Визначення і запис мінімальних елементів в масив проводиться в підпрограмі.
5. Розробити програму, що порівнює добутки парних елементів двох числових масивів різної розмірності.
6. Розробити програму, що визначає мінімальні елементи з елементів, що розташовані на головних діагоналях двох квадратних матриць різної розмірності.
7. Розробити програму, що визначає суми непарних елементів двох матриць різної розмірності.
8. Розробити програму, що визначає суми елементів кратних п’яти двох числових масивів різної розмірності.
9. Розробити програму, що визначає добутки елементів, розташованих на побічних діагоналях двох квадратних матриць різної розмірності.
10. Розробити програму, що визначає суми і добутки елементів, значення яких перевищують 3.5, в двох числових масивах різної розмірності.
11. Розробити програму, що визначає максимальні парні елементи в двох матрицях різної розмірності.

12. Розробити програму, що визначає і порівнює в числовому масиві з **30** елементів суми перших і останніх **10** елементів. Визначення суми **10** елементів масиву проводиться в підпрограмі.

13. Розробити програму, що визначає найбільший дійсний корінь двох заданих квадратних рівнянь. Визначення коренів квадратного рівняння проводиться в підпрограмі.

14. Розробити програму, що визначає суми елементів, що розташовані в непарних рядках двох матриць різної розмірності.

15. Розробити програму, що визначає у введеному масиві максимальний і мінімальний елементи, і що міняє їх місцями. Визначення максимального і мінімального елементів в масиві проводиться в підпрограмі.

16. Розробити програму, що визначає добутки елементів, що розташовані в парних стовпцях двох матриць різної розмірності.

17. Розробити програму, що визначає в двох введених масивах суму парних і непарних елементів. Визначення сум проводиться в підпрограмі.

18. Розробити програму, що визначає суми тих елементів, що розташовані на головних діагоналях двох квадратних матриць різної розмірності.

19. Розробити програму, що визначає у введеному масиві максимальний елемент, і, що міняє його місцями з першим елементом. Визначення максимального елементу в масиві проводиться в підпрограмі.

20. Розробити програму, що визначає максимальні від'ємні елементи в двох матрицях різної розмірності.

21. Розробити програму, що визначає у введеному масиві мінімальний елемент, і, що міняє його місцями з останнім елементом. Визначення мінімального елементу в масиві проводиться в підпрограмі.

22. Розробити програму, що порівнює середні значення двох числових матриць різної розмірності.

23. Розробити програму, що визначає максимальні від'ємні елементи в трьох масивах різної розмірності.

24. Розробити програму, що записує в окремий масив мінімальні елементи рядків введеної матриці. Визначення і запис мінімальних елементів в масив проводиться в підпрограмі.

25. Розробити програму, що порівнює добутки парних і непарних елементів числового масиву. Добутки парних і непарних елементів масиву визначаються в окремих підпрограмах.

26. Розробити програму, що визначає суми непарних і непарних елементів числового масиву. Суми парних і непарних елементів масиву визначаються в окремих підпрограмах.

27. Розробити програму, що визначає суми і добутки елементів, значення яких не перевищують 5, в двох числових масивах різної розмірності.

28. Розробити програму, що записує в окремий масив максимальні елементи рядків введеної матриці. Визначення і запис максимальних елементів в масив проводиться в підпрограмі.

29. Розробити програму, що визначає максимальний і мінімальний від'ємні елементи в заданій матриці.

30. Порівняти площі двох трикутників, заданих своїми сторонами. Визначення площі трикутника проводиться в підпрограмі по формулі:

$s = \sqrt{p \cdot (p-a) \cdot (p-b) \cdot (p-c)}$, де $p = \frac{a+b+c}{2}$.

Набір індивідуальних завдань 2

1. Розробити програму, що записує в зворотному порядку введений масив. Запис масиву в зворотному порядку проводиться в підпрограмі.

2. Розробити програму, що визначає суми елементів кратних трьом у двох числових масивах різної розмірності.

3. Розробити програму, що записує в окремий масив максимальні елементи рядків введеної матриці. Визначення і запис максимальних елементів в масив проводиться в підпрограмі.

4. В довільному масиві A(n) визначити суму і кількість додатних елементів.

5. Заданий довільний масив D(m, n). Знайти суми елементів першого і останнього рядків.

6. В довільному масиві Z(n) знайти найменший і найбільший елементи та їх порядкові номери.

7. Заданий довільний масив D(m, n). Знайти добуток елементів першого рядка та визначити середнє значення елементів останнього рядка.

8. В довільному масиві T(n) знайти суму елементів кратних трьом і добуток всіх інших.

9. Заданий довільний масив V(n, n). Визначити максимальні елементи головної та побічної діагоналей.

10. В довільному масиві G(n) знайти добуток и кількість додатних елементів, кратних п’яти.

11. Заданий довільний масив B(n, n). Визначити максимальні елементи, кратні п’яти, в кожному стовбці матриці B.

12. В довільному масиві E(n) знайти найменший додатний елемент і найбільший від’ємний елемент.

13. Заданий довільний масив B(n, n). Визначити суму та кількість додатних елементів, які знаходяться на головній діагоналі матриці B.

14. В довільному масиві V(n) знайти суму и кількість елементів, значення яких не менше заданої константи (vi ≥ α).

15. Заданий довільний масив U(n, n). Найти мінімальні елементи головної та побічної діагоналей.

16. В довільному масиві Z(n) знайти найбільший парний елемент серед елементів другої половини масиву (т. е. серед елементів від n/2-го до n-го).

17. Заданий довільний масив D(m, n). Визначити суму и добуток від'ємних елементів кратних 4.

18. В довільному масиві Q(n) знайти суму и добуток елементів, значення яких знаходяться в межах заданого інтервалу: α ≤ qi ≤ β.

19. Заданий довільний масив B(m, n). Найти максимальні елементи та їх порядкові номери в кожному рядку матриці.

20. В довільному масиві F(n) знайти суми парних і непарних елементів.

21. Заданий довільний масив G(m, n). Визначити суму и добуток елементів, які знаходяться на перетині непарних строк и парних стовбців.

22. В довільному масиві S(n) знайти суму и кількість від'ємних елементів, кратних чотирьом.

23. Заданий довільний масив T(n, n). Знайти добуток елементів першого та останнього стовбців.

24. В довільному масиві B(n) знайти суму и середнє значення елементів Bср.

25. Заданий довільний масив L(m, n). Найти добуток и кількість від'ємних елементів в непарних рядках матриці L.

26. В довільному масиві A(n) визначити мінімальний по модулю від'ємний елемент.

27. Заданий довільний масив H(m, n). В кожному стовбці матриці визначити суми додатних елементів, кратних п'яти.

Набір індивідуальних завдань 3

1. В довільному масиві Q(n) знайти найбільший непарний елемент.
2. Заданий довільний масив ***R*(*m, n*)**. Найти суми від'ємних елементів кожного рядка.
3. В довільному масиві ***F*(*n*)** знайти середнє значення елементів, значення яких не перевищують заданої константи: $F_i \le \alpha$.
4. Заданий довільний масив ***A*(*m, n*)**. Найти суму елементів, значення яких знаходяться в межах заданого інтервалу: $\alpha \le a_{ij} \le \beta$, та добуток інших елементів.
5. В довільному масиві ***U*(*n*)** знайти найменший та найбільший елементи кратні трьом.
6. Заданий довільний масив ***E*(*n, n*)**. Визначити суму парних додатних елементів, які знаходяться на головній та побічній діагоналях матриці ***E***.
7. В довільному масиві ***A*(*n*)** знайти добуток и кількість додатних елементів, кратних двом.
8. Заданий довільний масив ***L*(*m, n*)**. В кожному парному стовбці матриці ***L*** визначити мінімальні елементи.
9. В довільному масиві ***V*(*n*)** знайти кількість елементів, які рівні нулю та добуток інших.
10. Заданий довільний масив ***Q*(*n, n*)**. Визначити окремо суми всіх додатних и парних додатних елементів матриці ***Q***.
11. В довільному масиві ***G*(*n*)** знайти окремо кількість додатних елементів, від'ємних елементів і елементів, які рівні нулю.
12. Заданий довільний масив ***Z*(*m, n*)**. В кожному парному рядку матриці ***Z*** знайти окремо добутки парних і непарних елементів.
13. В довільному масиві ***A*(*n*)** визначити суму і кількість додатних елементів.
14. Заданий довільний масив ***D(m, n)***. Знайти суми елементів першого і останнього рядків.
15. В довільному масиві ***Z*(*n*)** знайти найменший і найбільший елементи та їх порядкові номери.

16. Заданий довільний масив $D(m, n)$. Знайти добуток елементів першого рядка та визначити середнє значення елементів останнього рядка.
17. В довільному масиві $T(n)$ знайти суму елементів кратних трьом і добуток всіх інших.
18. Заданий довільний масив $V(n, n)$. Визначити максимальні елементи головної та побічної діагоналей.
19. В довільному масиві $G(n)$ знайти добуток и кількість додатних елементів, кратних п’яти.
20. Заданий довільний масив $B(n, n)$. Визначити максимальні елементи, кратні п’яти, в кожному стовбці матриці B.
21. В довільному масиві $E(n)$ знайти найменший додатний елемент і найбільший від’ємний елемент.
22. Заданий довільний масив $B(n, n)$. Визначити суму та кількість додатних елементів, які знаходяться на головній діагоналі матриці B.
23. В довільному масиві $V(n)$ знайти суму и кількість елементів, значення яких не менше заданої константи ($v_i \geq \alpha$).
24. Заданий довільний масив $U(n, n)$. Найти мінімальні елементи головної та побічної діагоналей.
25. В довільному масиві $Z(n)$ знайти найбільший парний елемент серед елементів другої половини масиву (т. е. серед елементів від $n/2$-го до n-го).
26. Заданий довільний масив $D(m, n)$. Визначити суму и добуток від’ємних елементів кратних 4.
27. В довільному масиві $Q(n)$ знайти суму и добуток елементів, значення яких знаходь*ться в межах заданого інтервалу: $\alpha \leq q_i \leq \beta$.
28. Заданий довільний масив $B(m, n)$. Найти максимальні елементи та їх порядкові номери в кожному рядку матриці.
29. В довільному масиві $A(n)$ визначити мінімальний по модулю від’ємний елемент.

30. Заданий довільний масив ***H*(*m, n***). В кожному стовбці матриці визначити суми додатних елементів, кратних п’яти.

Додаток Д. Загальні відомості про об'єктну модель Excel і інших додатків Office

VBA заснований на принципах об'єктно -орієнтованого програмування. Ідея цієї концепції полягає в наступному: програмний додаток, як і оточуючій нас навколишній світ, складається з окремих об'єктів, кожен з яких має свої властивості (*properties*) і відрізняється своєю поведінкою.

У VBA кожен об'єкт складається з програми (програмний код) і даних, що зв’язані з нею, і якими можна яким-небудь чином маніпулювати. Об'єктами є сам додаток Excel і всі його компоненти: робочі книги (**Workbook**), листи (**Worksheet**), діапазони даних (**Range**), комірки (**Cell**), діаграми (**Chart**), таблиці, графічні об'єкти, діалогові вікна тощо. У VBA міститься більше 100 вбудованих об’єктів.

Програмні об'єкти групуються в ієрархічні **класи або колекції** об'єктів, як об'єкти реального миру навколо нас. Всі об'єкти в одному і тому ж класі мають одні і ті ж (або подібні) властивості і методи. Кожен клас об'єктів може містити один або декілька підкласів. Об'єкти, що належать до певного класу об'єктів, називаються членами (*members*) цього класу.

Сімейство (об’єкт **Collection**) являє собою об’єкт, що містить декілька інших об’єктів, як правило одного типу. Наприклад, об’єкт **Workbooks (Робочі книги)** містить в собі всі відкриті об’єкти – відкриті книги Workbook. Кожний елемент сімейства нумерується і може бути ідентифікований або за ім’ям, або за номером. Наприклад, **Worksheets (1)** означає перший робочий аркуш активної книги, а **Worksheets (Лист 1)** – робочий аркуш з ім’ям **Лист 1**.

Більшість об'єктів належить до групи подібних об'єктів. Ці групи називаються **класами**. Об’єкти і класи – є базовими поняттями об'єктно-орієнтованого програмування. Клас є узагальненням поняття і задає властивості та поведінку об’єктів (екземплярів класу). Кожен об’єкт належить деякому класу. Клас визначає ім’я об’єкту, його властивості і дії,

що над ним виконуються. Наприклад, Excel належить до загального класу об'єктів – класу додатків, що підтримують мову VBA.

Для любого додатку VBA особливості ієрархічних відношень між об'єктами називають об'єктною моделлю додатку.

Об’єктна бібліотека VBA містить різноманітні об’єкти, що розміщені на різних рівнях ієрархії. Ієрархія визначає зв’язок між об’єктами і показує шляхи доступу до кожного з них.

Повне посилання на об’єкт складається з рядка імен послідовно вкладених один в одного об’єктів. Роздільниками імен об’єктів є точки й ряд починається з об’єкта **Application** і закінчується власне іменем об’єкта. Наприклад, повне посилання на комірку **A1** робочого аркуша **Лист 1** робочої книги з ім’ям **Завдання** має вигляд:

Application.Workbooks(”Завдання”).Workssheets(”Лист 1”).Range(”A1”)

Кожного разу наводити повне посилання на об’єкт не обов’язково. Зазвичай достатньо використовувати неявне посилання на об’єкт. У неявному посиланні, на відміну від повного, об’єкти, що активні в даний момент, як правило, можна опускати. В розглянутому випадку, якщо посилання на комірку **A1** надане в програмі, що виконується в середовищі Excel, то посилання на об’єкт **Application** може бути опущено, тобто достатньо привести наступне посилання:

Workbooks(”Завдання ”).Workssheets(”Лист 1 ”).Range(”A1 ”)

Якщо робоча книга **Кафедра** активна, то посилання можна записати ще коротше:

Workssheets(”Лист 1 ”)».Range(”A1 ”)

Якщо в робочій книзі **Лист 1** активний, то у відносному посиланні цілком достатньо скористатися лише згадкою діапазону **A1:**

Range(”A1 ”)

Як і об'єкти реального миру, об'єкти VBA мають різні притаманні ним **властивості** (лист в Excel можна бачити і він може бути прихованим, рядки в робочому листі мають висоту, стовпці – ширину і так далі).

Множина значень властивостей визначає стан і поведінку об'єкта. Властивість є атрибутом об'єкту, який визначає його характеристики (наприклад, такі, як розмір, колір, положення на екрані і стан об'єкту, доступність або видимість. Щоб змінити характеристики об'єкта потрібно змінити властивості об'єкта. Деякі властивості можна змінювати, а деякі ні (наприклад, можна змінити ім'я автора робочої книги, але не можна змінити її ім'я, бо воно записано на диск). Синтаксис установки значення властивості:

Об'єкт. Властивість = ЗначенняВластивості

Наприклад змінити заголовок вікна Excel можна за допомогою завдання властивості **Caption** об'єкту **Application**:

Application.Caption = "База даних"

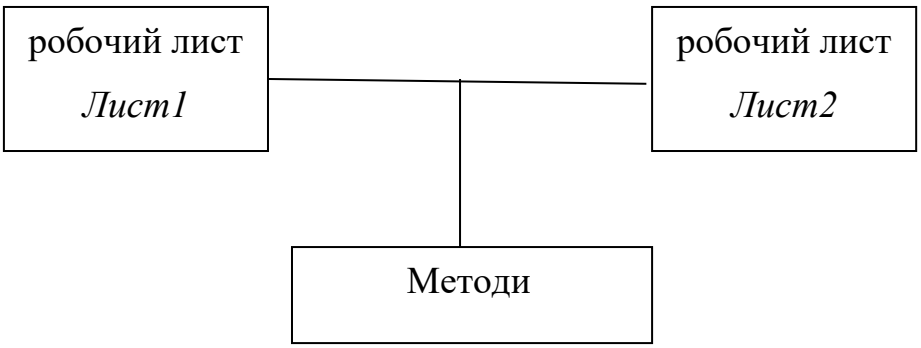
Властивість можна змінити відразу в усіх об'єктів сімейства.

Для того, щоб визначити поточний стан об'єкта або його поведінку слід звернутися до властивостей об'єкта.

Об'єкти реального миру майже завжди мають той тип властивої ним поведінки або дії, який вони можуть виконати. Об'єкти VBA також мають притаманну їм поведінку або можливості, що називають **методами** (*methods*). Метод – це іменований набір дій, який може виконувати даний об'єкт (іншими словами це те, що об'єкт "уміє" робити). Він може виконувати певні операції, приймати й повертати значення. Таким чином, **метод** являє собою дію, що виконується над об'єктом.

Наприклад, об'єкт робочої книги Excel має вбудовану можливість додавання нового робочого листа. Тобто володіє методом додавання нового робочого листа Add.

Однотипні об'єкти VBA спільно використовують методи.



До методів об’єкту можна звернутися тільки через об’єкт. Для виклику методу об’єкту (функції обробки даних об’єкту) указується не тільки найменування методу, але і об’єкт (ім’я), якому належить метод.

Синтаксис застосування методу:

Об’єкт. Метод

Наприклад, за допомогою методу **Quit (Закрити)** закривається додаток(об’єкт **Application**):

Application. Quit

Метод можна застосовувати до усіх об’єктів сімейства. Наприклад, до сімейства **Chartobjects (Діаграми)** робочого листа **Лист1** застосований метод **Delete (Видалити)**, який призводить до видалення усіх діаграм з робочого аркуша **Лист1:**

Worksheets (“Лист1”). ChartObjects. Delete

Подія – це початок або завершення будь-якої дії у програмі, яка ініціюється самою програмою або середовищем (операційною системою). Подія є дією, що розпізнається об’єктом (наприклад, клацання мишею або натиснення клавіші), і для якої можна запрограмувати відгук. Події виникають в результаті дій користувача або програми, або ж вони можуть бути викликані системою.

Суть програмування на VBA якраз і полягає в цих двох поняттях: подія і відгук на неї. Якщо користувач робить якусь дію, наприклад натискає кнопку, тоді в якості відгуку виконується код створеної користувачем процедури. Якщо такий відгук не створений, тобто не написана відповідна процедура, то система ніяк не реагує на цю подію і вона залишається без відповіді. Таким чином, дії, що відбуваються в системі, є подіями, а відгуки на них – процедурами. Для конкретного об’єкта можна написати код (процедуру опрацювання події), який буде виконуватися у випадку виникнення певної події. Цей спеціальний вид процедур, що генерують відгук на події, називається процедурами обробки подій. В цілому

програмування на VBA полягає в створенні коду програм, які генерують прямо або побічно відгуки на події.

Використання об’єктів

Інструкції VBA, що мають справи з об’єктами, можуть виконувати з ними наступні дії:

- перевіряти стан (або статус) об’єкта (для цього потрібно звернутись до значення, що записано в його властивостях);
- змінювати стан (або статус) об’єкта (для цього потрібно змінити значення, що записано в його властивостях);
- виконати якесь завдання, що притаманне об’єкту (для цього потрібно звернутись до метода).

У операторах VBA використовується наступний загальний синтаксис для визначення властивості або методу об’єкту:

Формат:

Object.Identifier

де **Object** – любе допустиме посилання на об’єкт; **Identifier** – любе допустиме посилання на властивість або метод. Посилання відокремлюється від імені властивості або методу точкою (.).

VBA відображає повідомлення про *runtime-помилку* при спробі використовувати властивості або методи, які не є частиною вказаного об’єкту.

Щоб скористатися методами, властивостями чи подіями об’єкта потрібно знати способи їх виклику:

1. **Способи виклику методу** (три способи).

Як ми вже казали, метод – це іменований набір дій, який може виконувати даний об’єкт. Він може виконувати певні операції, приймати й повертати значення.

– **<u>перший</u>** (найлегший) спосіб виглядає так:

Об’єкт.Метод

Приклад:

aBook.Activate *(Робоча книга Excel має метод Activate, котрий активізує робочу книгу і перший лист цієї книги. Змінна aBook в прикладі є посиланням на об’єкт робочої книги. Наведена інструкція активує цей робочий лист.)*

При цьому не повертаються і не приймаються ніякі дані, параметри.

– **другий** спосіб:

Об’єкт.Метод параметр1 [, параметр2, … , параметрN]

Параметри в списку розділяють комами.

Приклад:

ActiveWorkbook.SaveAs Filename:= «C:\VBA2011\NewFile.xls»

Параметри метода можуть бути обов’язковими і необов’язковими.

Приклад ілюструє метод SaveAs об’єкта робочої книги. В прикладі використовується один обов’язковий аргумент для методв SaveAs.

SaveAs – зберігання робочої книги в іншому файлі.

Синтаксис: ***SaveAs (Filename)***

де Filename — рядок, що вказує ім’я файлу, в якому буде збережена робоча книга.

В цьому випадку в програмі ігнорується те, що повертає метод і тому список параметрів в дужки не заключають.

– **третій** спосіб:

Багато об’єктів мають методи, що повертають значення, як це роблять функції. Для того, щоб використати значення, що повертає метод, потрібно заключити в дужки список аргументів і помістити виклик метода в оператор присвоювання, де повертаєме значення буде присвоєне деякій змінній.

Формат:

Змінна = Об’єкт.Метод(параметр1 [, параметр2, … , параметрN])

Приклад:

```
Public Sub ccc()
adr = Cells(1, 1).Address()
MsgBox adr
End Sub
```

Address – метод об’єкта Range. Повертає адресу комірки.

В цьому випадку значення, які повертає метод, присвоюються змінній adr. Потім її значення виводиться у вікно за допомогою функції MsgBox.

Результат:

При цьому використовуються круглі дужки для параметрів, що передаються. У випадку, коли ніякі параметри не передаються, круглі дужки все одно обов’язкові: Змінна = Об’єкт.Метод(). *(Як показано в прикладі.)*

2. **Способи виклику властивостей**.

Через властивості об’єкта можна отримати доступ до інформації, яка зберігається в цьому об’єкті чи змінити її.

1. Знайти інформацію можна за допомогою синтаксису:

Змінна = Об’єкт.Властивість

де Змінна – люба змінна, тип якої співпадає з типом властивостей об’єкту; Об’єкт – любе допустиме посилання на об’єкт; Властивість – любе допустиме ім’я властивостей об’єкта, до котрого звертаються.

З формату видно, що значення властивостей можна присвоїти змінній. *Також можна їх використовувати у виразах в якості параметрів функцій і підпрограм або в якості параметрів для методів об’єктів.*

Приклад:

```
Public Sub ccc()
```

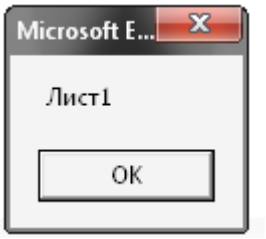
adr = ActiveSheet.Name

MsgBox adr

End Sub

В цьому випадку ім’я активного листа присвоюються змінній adr. Потім її значення виводиться у вікно за допомогою функції MsgBox.

Результат:



2. Змінити інформацію в об’єкті можна за допомогою синтаксису:

Об’єкт.Властивість = Значення

де Значення – любий вираз VBA, тип якого співпадає з типом властивостей об’єкту;

Значення може бути:

– звичайною константою;

– простим виразом;

– властивістю іншого об’єкта:

Об’єкт1.Властивість = Об’єкт2.Властивість

– значенням будь-якого методу:

Об’єкт1.Властивість = Об’єкт2.Метод()

Звичайно, значення не всіх властивостей можна змінювати. Деякі властивості доступні лише для зчитування інформації, інші – для зчитування і запису, треті (дуже рідко) – лише для запису.

В прикладі властивості присвоюється нове Значення.

Приклад:

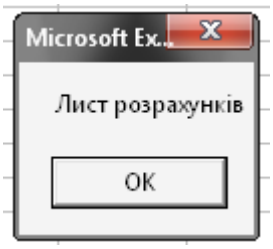
Public Sub ccc()

ActiveSheet.Name = «Лист розрахунків»

MsgBox ActiveSheet.Name

End Sub

В цьому випадку змінюється ім’я активного листа на нове – «Лист розрахунків». Потім це нове значення виводиться у вікно за допомогою функції MsgBox.



В книзі Excel ім’я активного листа «Лист1» було замінено новим – «Лист розрахунків».

Оголошення об’єктних змінних в VBA

Для цього в VBA існує тип даних Object. Змінні чи вирази типу Object посилаються на об’єкт VBA або на об’єкт, що належить додатку (Наприклад, на об’єкти додатку Excel – **Workbook, Worksheets і Range**). Оголошення можна виконати за допомогою оператора **Dim**, як це виконується для даних інших типів.

Приклад:

Dim myObject **As** Object

Змінна myObject вміщує посилання на любий об’єкт VBA або додатку.

Для вказівки визначеного типу об’єктів оголошення можна виконати наступним чином:

Dim myBook **As** Workbook

Тоді змінна myBook вміщує посилання на об’єкти Workbook. При спробі встановити посилання змінній myBook на об’єкти Worksheets або Range буде виведене повідомлення про помилку.

Основні об’єкти VBA

Об’єкт Application

Об’єкт Application – (додаток) є головним в ієрархії об’єктів Excel і представляє сам додаток Excel. Він має більше 120 властивостей і 40 методів. Ці властивості і методи призначені для установки загальних параметрів додатку Excel. Крім того, об’єкт Application дозволяє викликати більше 400 вбудованих функцій робочого листа за допомогою конструкції виду:

Application.ФункціяРабочегоЛиста(Аргументи)

Приклад1: Пошук суми елементів записаних в лист Excel

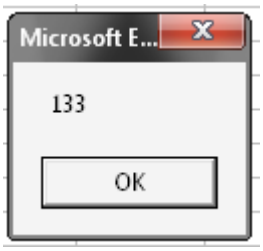
Програма:

```
Public Sub ccc()
s = Application.Sum(Range(«A2:G2»))
MsgBox s
End Sub
```

Дані, що записані в лист Excel:

	A	B	C	D	E	F	G
1							
2	2	4	7	11	40	59	10
3							

Результат:



Приклад 2: Пошук максимального з елементів записаних в лист Excel

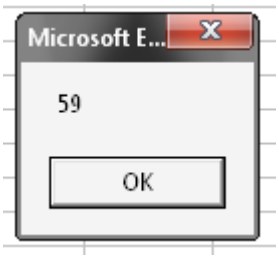
Програма:

```
Public Sub ccc()
s = Application.Max(Range(«A2:G2»))
MsgBox s
End Sub
```

Дані, що записані в лист Excel:

	A	B	C	D	E	F	G
1							
2	2	4	7	11	40	59	10
3							

Результат:



Приклад 3: Пошук мінімального з елементів записаних в лист Excel

Програма:

```
Public Sub ccc()
s = Application.Min(Range(«A2:G2»))
MsgBox s
End Sub
```

Дані, що записані в лист Excel:

	A	B	C	D	E	F	G
1							
2	2	4	7	11	40	59	10
3							

Результат:

Об’єкт Workbook

У ієрархії Excel об’єкт Workbook (робоча книга) йде відразу після об’єкту Application і представляє файл робочої книги. Робоча книга зберігається або у файлах формату XLS, XLSX (стандартна робоча книга) або

XLA (додаток, що повністю відкомпільований). Властивості і методи робочої книги дозволяють працювати з файлами.

Об’єкт Worksheet

У ієрархії Excel об’єкт Worksheet йде відразу після об’єкту Workbook і представляє робочий лист.

Об’єкти Range і Selection

У ієрархії Excel об’єкт Range (діапазон) йде відразу після об’єкту Worksheet. Об’єкт Range є одним з ключових об’єктів VBA. Об’єкт **Range** являє собою посилання на комірку, рядок, стовпець або діапазон комірок.

Об’єкт Selection (вибір) виникає в VBA двояко – або як результат роботи методу Select, або при виклику властивості Selection. Тип отримуваного об’єкту залежить від типу виділеного об’єкту. Найчастіше об’єкт Selection належить класу Range і при роботі з ним можна використовувати властивості і методи об’єкту Range.

Об’єкт **Range** використовується для роботи з комірками, рядками, стовпцями, а також їх групами. Для доступу до об’єкту найчастіше використовуються властивості Range і Cells (хоча є і інші можливості).

Якщо використовується властивість Range, то в якості аргументу указується будь-яке допустиме в MS Excel посилання у форматі A1.

Наприклад:

Worksheets(«Лист1»).Range(«A5») = 15

Тут в комірку A5 листа Лист1 записується значення 15.(тобто комірці присвоюється значення 15).

Worksheets(«Лист1»).Range(«A2:C3») = 15

Тут в діапазон комірок «A2:C3» листа Лист1 записується значення 15.

Temp = Worksheets(«Лист1»).Range(«A2»)

Тут змінній Temp присвоюється вміст комірки A2 листа Лист1.

Код програми:

```
Public Sub ccc()
Temp = Worksheets(«Лист1»).Range(«A2»)
MsgBox Temp
End Sub
```

Властивість **Cells** використовується для доступу до окремої комірки. Як аргументи указуються номер рядка і стовпця.

Наприклад:

```
Worksheets(«Лист1»).Cells(6, 1) = 25
```

В комірку А6 листа Лист1 записується значення 25.

```
Temp = Worksheets(«Лист1»).Cells(2, 3)
```

Змінній Temp присвоюється вміст комірки С2 листа Лист1.

*Можна також використовувати властивість **Cells** для альтернативної вказівки діапазону. Наприклад, Range(«A2:C3») і Range(Cells(2,1), Cells(3,3)) визначають один і той же діапазон.*

Посилання на об’єкти за допомогою **With…End With**

Процедури можуть посилатися на один і той же об’єкт багатократно, коли декілька інструкцій використовують один і той же об’єкт чи властивості цього об’єкта. VBA надає особливу структуру – структуру With…End With, що дозволяє посилатися на властивості або методи, які належать одному і тому ж об’єкту, без завдання всього об’єктного посилання кожного разу.

Структура With…End With має наступний формат:

Формат:

With Об’єкт

оператори, що використовують властивості і методи об’єкта

End With

де Object – це будь-яке допустиме посилання на об’єкт.

В таблиці Д1 наведені деякі загальні і корисні властивості об’єктів Excel, а в таблиці Д2. – деякі основні методи об’єктів Excel.

Таблиця Д1.

Загальні і корисні властивості об’єктів

Властивість	**Тип/значення**	**Використовується в об’єктах**
1	2	3
ActiveCell	Object: активна комірка робочого листа	Application, Window, Workbook
ActiveChart	Object: активний графік (діаграма)	Application, Window, Workbook
ActiveCheet	Object: активний робочий лист	Application, Window, Workbook
Count	Integer: кількість об’єктів в колекції	Всі колекції об’єктів
Formula	String: формула для комірки робочого листа	Range
Index	Integer: кількість об’єктів в колекції	Worksheet
Name	String: ім’я об’єктів	Application, Workbook та інші
Path	String: ім’я диска і каталог(папка), де зберігається об’єкт	AddIn, Application, Workbook
Saved	Boolean: чи відбувалося збереження робочої книги з моменту останньої зміни	Workbook
Selection	Object: поточне виділення	Application, Window
StatusBar	String: повідомлення в вікні стану	Application

Закінчення табл. Д1

1	2	3
ThisWorkBook	Object: робоча книга, з якої виконується поточна процедура	Application
Type	Integer: число, яке відповідає типу об’єкта	Window, Worksheet, Chart
Visible	Boolean: вказує на те, чи є об’єкт Excel видимим	Application, Worksheet, Range та інші
Value	Може бути різним; реальне значення, що показує комірка	Range

Таблиця Д2.

Основні методи об’єктів Excel

Метод	**Мета застосування**	**Використовується в об’єктах**
1	2	3
Activate	Активує об’єкт	Window, Workbook, Worksheet, Range та інші
Address	Повертає координати комірки певного об’єкта	Range
Calculate	Робить обчислення в відкритій робочій книзі, робочому листі чи в деякому діапазоні	Application, Range, Worksheet
Calls	Повертає об’єкт Range	Application, Range, Worksheet
Charts	Повертає колекцію діаграм робочого листа	Application, Workbook

Закінчення табл. Д2

1	2	3
Clear	Видаляє дані з певного об’єкта	Range
Close	Закриває певний об’єкт	Window, Workbook, Workbooks
Justify	Вирівнює текст в певному об’єкті	Range
Run	Виконує певну процедуру чи функцію	Application, Range
Save	Зберігає файл робочого листа	Application, Workbook
SaveAs	Зберігає певний об’єкт в іншому файлі	Workbook, Worksheet
Select	Виділяє певний об’єкт	Range, Worksheets, Sheets
SendKeys	Відсилає в активне діалогове вікно чи додаток коди комбінацій клавіш, імітуючи натискання клавіш користувачем	Application
Sheets	Повертає колекцію всіх листів робочої книги	Application, Workbook
Volative	Реєструє функцію як тип Volative	Application
Workbooks	Повертає колекцію робочих книг	Application
Worksheets	Повертає колекцію робочих листів	Application, Workbook

Додаток Е. Елементи управління в Toolbox

Таблиця Е1.

Елементи управління в Toolbox

Елемент керування	Назва	Призначення
1	2	3
	Label (Надпис)	Створює елемент управління Label. Написи використовуються для вставки тексту в форму, наприклад заголовків елементів управління, які не мають власних інструкцій, підказок по використанню форми і т.п. Написи застосовуються також для відображення такої інформації, як поточне значення лічильника або смуги прокрутки. Текст в елементі керування Label можна змінити за допомогою вихідного тексту VBA, але користувач форми не може редагувати текст напису. Ці елементи управління слід використовувати для відображення тексту, який не буде мінятися, або зміна якого користувачем небажана.
	CommandButton (Командна кнопка або Кнопка)	Створює елемент управління CommandButton. Командні кнопки використовуються для виконання таких дій, як пов'язані з клацанням на кнопці. Діями можуть бути закриття або приховування форми, оновлення даних, підтвердження даних в елементах управління форми і т.п.

Продовження табл. Е1

1	2	3
ab|	Textbox (Текстове поле або Поле)	Створює елемент управління TextBox. Поля використовуються для отримання або відображення даних, які можуть бути представлені у вигляді тексту, таких як імена або адреси, грошові суми, дати і т.п. Поля слугують також для відображення такої інформації, як поточне значення лічильника або смуги прокрутки, і для забезпечення альтернативних способів встановлення значення лічильника або смуги прокрутки. У полях можуть відображатися один нам кілька рядків тексту. При необхідності поля будуть забезпечуватися вертикальною смугою прокрутки для прокручування кількох рядків тексту.
☑	CheckBox (Прапорець)	Створює елемент управління CheckBox. Прапорці в якості складової частини елемента управління містять напис. Прапорці використовують для вибору значень типу істинно / хибно, включено / вимкнено, так / ні, котрі не є взаємовиключними (наприклад, якщо потрібно виконати пошук тексту з урахуванням регістра і тільки цілих слів, то для вибору цих параметрів пошуку потрібно було б використовувати відповідні прапорці).

Продовження табл. Е1

1	2	3
	Frame (Рамка)	Створює елемент управління Frame. Насправді ці елементи управління самі нічого не роблять, але вони містять інші елементи управління, зазвичай перемикачі, прапорці або вимикачі. Рамка використовується для вказівки користувачу, які елементи управління в діалоговому вікні пов'язані один з одним, або для відділення особливої групи елементів управління від інших елементів управління діалогового вікна. Можна заважати користувачеві входити або виходити з групи елементів управління, що поміщені в рамку, за допомогою натискання клавіші <Tab>, встановивши значення властивості Cyc1e рівним fmCycleCurrentForm. Для розміщення інших елементів управління в рамці спочатку необхідно помістити у форму елемент управління Frame, а потім помістити в неї необхідні елементи управління.
	ToggleButton (Вимикач)	Створює елемент управління ToggleButton. Вимикачі використовуються для вибору значень типу істинно / хибно, включено / виключено, так / ні. Вони слугують для тієї ж мети, що і прапорці, але відображають налаштування у вигляді кнопки, що "натиснута" або "віджата". Їх слід використовувати для вибору значень, які не є взаємовиключними.

Продовження табл. Е1

1	2	3
	OptionButton (Перемикач)	Створює елемент управління OptionButton. Перемикачі використовуються для вибору взаємовиключних значень типу істинно/хибно, включено / вимкнено, так / ні (наприклад, одночасно не можна виконувати пошук і в прямому, і в зворотному напрямку, тому для вибору напрямку пошуку потрібно використовувати перемикач). Як правило, і прапорці, і перемикачі поміщаються всередину рамки.
	ListBox (Список)	Створює елемент управління ListBox. Списки використовуються при необхідності відобразити список елементів і надати користувачу можливість вибору в ньому одного або кількох елементів, без можливості додавати в список нові елементи. Для визначення того, чи може користувач вибирати в списку більше одного елемента, слід використовувати властивість списку MultiSelect. Списки використовуються для відображення таких елементів, як імена робочих книг і листів, діапазони імен, місяця року і т.п.
	ScrollBar (Смуга прокрутки)	Створює елемент управління ScrollBar. Смуги прокрутки використовуються для створення вручну елементів управління, що прокручуються у формі або для введення послідовних числових значень.

Продовження табл. Е1

1	2	3
	ComboBox (Поле зі списком)	Створює елемент управління ComboBox, який об'є днує в собі властивості поля та списку. Поле зі списком використовується для створення списків, що розкриваються, і в тих випадках, коли потрібно запропонувати користувачу значення, з яких він може вибирати. Поля зі списком можуть бути установлені для надання користувачу можливості вводити значення, що відрізняються від представлених у списку, або для обмеження його тільки цими значеннями (це виконується установкою властивості Style). Поля зі списком можуть використовуватися для вибору тільки одного елемента одночасно.
	MultiPage (Набір сторінок)	Створює елемент управління MultiPage. Цей елемент управління має кілька клієнтських областей, в які можна включати інші елементи керування. При виборі нової вкладки вся клієнтська область змінюється для відображення різних груп елементів управління. Елементи управління MultiPage використовуються для створення діалогових вікон з вкладками, в яких інформація на кожній сторінці відображається в інших елементах керування. Діалогове вікно Options (Параметри) додатка Excel є прикладом елемента управління MultiPage.

Продовження табл. Е1

1	2	3
	TabStrip (Набір закладок)	Створює елемент управління TabStrip. Набір вкладок має одну клієнтську область, в якій можна поміщати інші елементи управління, подібно елементу управління Рамка. При виборі нової вкладки клієнтська область не змінюється. Замість цього, подія Change використовується для оновлення елементів управління в клієнтській області набору вкладок у відповідності з конкретною програмою. Набори вкладок застосовуються для діалогових вікон з вкладками, в яких інформація на кожній сторінці ділового вікна відображається в одних і тих же елементах управління, але й в різних категоріях. (Наприклад, якщо потрібно створити форму для відображення підсумкової інформації за результатами квартального продажу, можна було б використовувати набір з п'яти вкладок: по одній для кожного кварталу і одна для річного підсумку. Кожна вкладка відображала б одну і ту ж інформацію: зростання продажів, прибуток і т.п. для різних кварталів.)

Продовження табл. Е1

1	2	3
	SpinButton (Лічильник)	Створює елемент управління SpinButton. Лічильники використовуються для введення в діалоговому вікні послідовних чисельних значень. Зазвичай лічильники об'єднуються з написами для відображення значення лічильника. Лічильник можна також об'єднати з полем для забезпечення альтернативного методу введення значення. При об'єднанні лічильника з написом або полем необхідно створити процедуру, що синхронізує обидва елементи управління.
	Image (Зображення або Малюнок)	Створює елемент управління Image. Ці елементи керування використовуються для відображення в формах таких графічних об'єктів як логотипи, фотографії тощо. Елементи управління зображень становлять події аналогічно іншим елементам управління, тому зображення можна використовувати для відповіді на клацання мишею. Елементи управління зображень не можуть редагуватися, тому вони не можуть отримувати фокус.

Закінчення табл. Е1

1	2	3
	RefEdit (Поле з приєднаною справа кнопкою)	Створює елемент управління RefEdit. Цей елемент керування відображає поле з приєднаною до нього справа кнопкою. Клацання на приєднаній кнопці елемента управління RefEdit призводить до згортання всієї форми до об'єкта, що не перевищує за розмірами поле RefEdit. При цьому користувач може вибирати листи в робочій книзі і діапазони в листах. Повторне клацання на кнопці елемента управління RefEdit відновлює форму в її первинних розмірах. Елементи управління RefEdit використовуються, коли потрібно надати користувачу можливість відкривати максимальну поверхню екрану, що зазвичай приховується формою, і спростити користувачеві введення запитуваних діапазонів робочих листів.

Властивості елементів управління

Всі властивості елементів управління можна розбити на кілька категорій:

Категорія *Appearance (Вигляд).* Властивості цієї категорії визначають зовнішній вигляд елементів управління або інших об'єктів.

Категорія *Behavior (Поведінка).* Ці властивості визначають поведінку елементів управління. До даної категорії відносяться властивості, що визначають доступність елементів управління, можливість вибору в списку декількох елементів, спосіб вирівнювання тексту і т.д.

Категорія *Data (Дані).* Властивості категорії *Data* визначають дані, пов'язані з елементами управління.

Категорія *Font (Шрифт).* До даної категорії відноситься всього одна властивість *Font.*

Категорія *Miscellaneous (Різне).* Властивості цієї категорії визначають такі характеристики елемента управління, як порядок переходу за елементами управління у формі, вид покажчика миші і т.д.

Категорія *Position (Розміщення).* Її властивості визначають розмір і місце розташування елемента керування на формі.

Категорія *Scroll (Прокручування).* Містить властивості, що визначають деякі характеристики смуг прокрутки.

Категорія *Tabs (Табуляція).* Містить спеціальні властивості об'єктів *MultiPage, TabStrip, Page і Tab*.

Категорія *Image (Малюнок).* Включає в себе властивості, керуючі картинкою.

Таблиця Е2.

Основні властивості стандартних елементів управління

Властивість	Елементи управління, до якого відноситься властивість	Опис
1	2	3
Accelerator	CheckBox,Tab, CommandButton, Label,Page, OptionButton, ToggleButton	Містить символ, використовуваний як клавіші швидкого доступу до елементу управління. Натискання комбінації клавіш <Alt + клавіша швидкого доступу>викликає вибір елемента управління.

Продовження табл. Е2

1	2	3
BackColor	Всі елементи управління	Число, яке представляє колір фону елемента управління.
Caption	CheckBox, CommandButton, Frame, Label, OptionButton, Page, Tab, ToggleButton, UserForm	Для напису (це текст, який відображається елементом управління); для інших елементів управління (це напис, що з'являється на поверхні кнопки або вкладки, або відображається поруч з рамкою, прапорцем або перемикачем).
Cancel	CommandButton	Визначає кнопку, еквівалентну кнопці Відміна (клацання на цій кнопці або натискання клавіші <Esc> скасовує діалог). Тільки одна кнопка у формі може мати цю властивість.
ControlTipText	Всі елементи управління	Визначає текст, який відображається в якості підказки елемента управління (підказка інструменту), коли покажчик миші затримується над елементом управління.
Default	CommandButton	Визначає кнопку, що обрана за замовчуванням. Коли користувач натискає клавішу <Enter> в діалоговому вікні, кнопка поводить себе таким чином, як би на ній було виконано клацання мишею.

Продовження табл. Е2

1	2	3
Enabled	Всі елементи управління	Зберігає значення Boolean, що визначає чи активізований елемент управління. Якщо значенням властивості Enabled є False, елемент управління з'являється в діалоговому вікні, але не може бути обраний.
ForeColor	Всі елементи управління	Аналогічно властивості BackColor, але визначає колір зображення елементу управління – зазвичай його тексту.
List	ComboBox, ListBox	Масив типу Variant (одновимірний або багатомірний), який представляє список, що міститься в елементі управління. Значення індексу в змінній Value використовується в якості індексу набору List для отримання тексту вибраного елементу списку. Методи AddItem і RemoveItem елемента управління слугують для додавання або видалення елементів списку.
Max	ScrollBar, SpinButton	Значення типу Long, що визначає максимальне значення лічильника або значення в крайній нижній (для вертикальної смуги прокрутки) або крайньої правої (для горизонтальної смуги прокрутки) позиції смуги прокручування.

Продовження табл. Е2

1	2	3
Min	ScrollBar, SpinButton	Значення типу Long, що визначає мінімальне значення лічильника або значення в крайній верхній (для вертикальної смуги прокрутки) або крайньої лівої (для горизонтальної смуги прокрутки) позиції смуги прокручування.
Name	Всі елементи управління	Містить ім'я елемента управління. Цю властивість можна встановлювати тільки в діалоговому вікні Properties.
RowSourse	ComboBox, ListBox	Вказує джерело списку об'єкта (в VBA Excel значенням властивості RowSourse зазвичай є діапазон робочого листа).
Selected	ListBox	Повертає масив значень типу Boolean для списків, які допускають вибір декількох значень (кожен елемент масиву містить один елемент, який відповідає кожному елементу в списку). Якщо елемент в масиві Selected має значення True, то відповідний елемент списку вибраний.
TabIndex	Всі елементи управління	Число, яке вказує позицію елемента управління при переході до нього при натисканні клавіші <Tab> (може приймати значення від 0 до числа, рівного кількості елементів управління у формі).

Закінчення табл. Е2

1	2	3
TabStop	Всі елементи управління	Значення типу Boolean, що показує, чи може елемент управління бути вибраний натисканням клавіші <Tab> (якщо значення властивості TabStop рівне False, елемент управління все одно може бути обраний клацанням миші на ньому).
Value	Всі елементи управління	Значення поточної настройки елемента управління: текст у текстовому полі; вибрано прапорець або перемикач; індекс обраного елемента у списку або число, яке вказує поточну позицію смуги прокрутки або лічильника.
Visible	Всі елементи управління	Значення типу Boolean, що показує, чи вибраний елемент управління.

Розглянемо деякі з властивостей елементів управління:

- *Об'єкт.* ***Accelerator*** [= String] – це властивість дозволяє отримати або призначити *Об'єкту* комбінацію клавіш Alt + <клавіша>.
- *Об'єкт.* ***ActiveControl*** – показує що елемент управління *Об'єкт* стає активним.
- *Об'єкт.* ***Alignment*** [= *fmAlignment]* – вирівнює заголовок елемента управління, зазначеного у властивості *Caption.* Параметр *fmAlignment* приймає його з наступних значень:*fmAlignmentLeft* – заголовок знаходиться лівіше елемента управління, *fmAlignmentRight* – заголовок знаходиться правіше елемента управління.
- *Об'єкт.* ***AutoSize*** [= **Boolean]** – автоматичне масштабування розміру напису в залежності від висоти об'єктів. Якщо властивість *AutoSize* має значення **True,** то розмір шрифту підбирається автоматично.

- *Об'єкт.* ***AutoTab*** [= **Boolean]** – властивість, що дозволяє вказати, чи повинен здійснюватися перехід фокусу до наступного по табульованому порядку елементу управління при максимальному заповненні текстового поля.

- *Об'єкт.* ***AutoWordSelect*** [= **Boolean]** – ця властивість задає базову одиницю при поширенні виділення. Якщо властивість має значення **True,** то базисної одиницею є повне слово. Якщо **False,** то один символ.

- *Об'єкт.* ***BackColor*** [= **Long]** – задає колір фону об'єкта.

- *Об'єкт.* ***BackStyle*** [= *FmBackStyle]* – задає стиль відображення фону об'єкта. Параметр *fmBackStyle* має наступні значення: *fmBackStyleTransparent* - прозора основа, *fmBackStyleOpaque* – непрозора.

- *Об'єкт.* ***BorderColor*** [= **Long]** – задає колір рамки, що обмежує об'єкт.

- *Об'єкт.* ***BorderStyle*** [= *fmBorderStyle]* – властивість, що задає обмежуючу рамку. Параметр fmBorderStyle має наступні значення: *fmBorderStyleNone* - рамка відсутня,*fmBorderStyleSingle* - рамка товщиною в один піксель.

- *Об'єкт.* ***BoundColumn*** [= **Variant]** – задає стовпець, значення якого буде відображатися у властивості **Value.** Ця властивість є у наступних елементів управління: **ComboBox, ListBox** вони можуть мати декілька стовпців.

- *Об'єкт.* ***BoundValue*** [= **Variant]** – властивість вказує на стан об'єкта, коли він отримує фокус. Параметр **Variant** для різних елементів управління має різні значення. Для елементів управління **Checkbox, OptionButton, ToggleButton** – **Null,** коли елемент активний, але він перебуває за межами - 1 **(True)** - елемент обраний; 0 **(False)** – елемент не обраний. Для елементів **ScrollBar, SpinButton** – цілочисельне значення, що знаходиться в інтервалі між *Max* і *Min* (вказано в цих властивостях) і відображає поточний стан. Для елементів **ComboBox, ListBox** – значення властивості *BoundColumn* для вибраного рядка. Для **CommandButton** - значення завжди **False.** Для *MultiPage* - цілочисельне значення, що показує індекс поточної активної

вкладки. Перша вкладка має індекс 0, друга - 2 і т.д. Для **TextBox** – рядковий вираз, набраний в текстовому полі.

- *Об'єкт.* ***Cancel*** [= **Boolean]** – для командної кнопки *Об'єкт* показує, що вона є кнопкою Cancel для форми. Це властивість має значення **True** тільки для одного об'єкта форми.
- *Об'єкт.* ***CanPaste*** – дозволяє дізнатися, може *Об'єкт* отримати дані з буфера обміну. У тому випадку якщо значення **True,** то вставка з буфера можлива.
- *Об'єкт.* ***Caption*** [= **String]** – містить текст, що знаходиться в полі напису.
- *Об'єкт.* ***CanRedo*** – це властивість перевіряє чи можна для об'єкта *Об'єкт* виконати повернення до останньої скасованим дії. Якщо ця властивість має значення **True,** то можна повернутися.
- *Об'єкт.* ***CanUndo*** – визначає можливість скасування останньої дії. Якщо ця властивість має значення **True,** то скасування можливе.
- *Об'єкт.* ***ColumnCount*** [= **Long]** – властивості містить або дозволяє встановити число стовпців в елементах управління **ComboBox, ListBox.**
- *Об'єкт.* ***ControlTipText*** [= **String]** – містить текст випливаючої підказки, яка з'являється, коли курсор миші потрапляє у фокус об'єкта.
- *Об'єкт.* ***Count*** – вказує число об'єктів, що знаходяться в сімействі об'єкта *Об'єкт.*
- *Об'єкт.* ***CurLine*** [= **Long]** – містить значення поточного рядка, де в даний момент знаходиться курсор. Перший рядок має індекс 0, другий – 1, третій – 2 і т.д.
- *Об'єкт.* ***Default*** [= **Boolean]** – показує на командну кнопку, яка при ініціалізації форми за замовчуванням буде мати фокус.
- *Об'єкт.* ***Delay*** [= **Long]** – задає час затримки перед генерацією подій: *SpinUp, SpinDown, Change* (в мілісекундах), для елементів управління **SpinButton** і **ScrollBar.**

- *Об'єкт.* ***DragBehavior*** [= *fmDragBehavior*] – дає можливість використовувати «перетягнути – і – залишити» в полях редагування елементів управління **TextBox** і **ComboBox.** Значення параметра *fmDragBehavior:* якщо *fmDragBehaviorDisabled,* то перетягування неможливе, якщо *fmDragBehaviorEnabled,* то виділені фрагменти можна перетягувати в межах поля редагування.
- *Об'єкт.* ***Enabled*** [= **Boolean]** – дозволяє робити *Об'єкт* недоступним, тобто *Об'єкт* набуває сірого кольору і не реагує на будь-які дії над ним.
- *Об'єкт.* ***ForeColor*** [= **Long]** – визначає колір шрифту елемента управління.
- *Об'єкт.* ***GroupName*** [= **String]** – дозволяє задати ім'я групи, яка об'єднує кілька елементів управління **OptionButton.**
- *Об'єкт.* ***Index*** [= **Integer]** – показує значення, що характеризує стан вкладок (об'єктів **Tab** і **Page)** один щодо одного в родинах **Tabs** і **Pages.** Перша вкладка має індекс – 0, друга – 1 і т.д.
- *Об'єкт.* ***LayoutEffect*** – дозволяє дізнатися, чи був переміщений елемент управління при зміні положення. Властивість може приймати два значення: *fmLayoutEffectNone* – елемент керування не був переміщений, *fmLayoutEffectInitiate* – був переміщений.
- *Об'єкт.* ***LineCount*** – повертає число текстових рядків, визначених для елементів управління **TextBox** і **ComboBox.**
- *Об'єкт.* ***ListCount*** – повертає число рядків, наявних в списку.
- *Об'єкт.* ***ListIndex*** [= **Variant]** – вказує на номер рядка, де знаходиться курсор в елементах управління **ListBox, ComboBox.** Має значення 0 – обраний перший рядок, 1 – обраний другий рядок і т.д.
- *Об'єкт.* ***Locked*** [= **Boolean]** – показує, може чи ні елемент управління бути відредагований. Якщо властивість має значення **True** – ніякі зміни внести неможливо. Якщо **False** – можливо відредагувати.
- *Об'єкт.* ***MatchEntry*** [= *fmMatchEntry]* – встановлює порядок пошуку в списках елементів управління **ListBox** і **ComboBox.** Параметр *fmMatchEntry*

може набувати таких значень: *fmMatchEntryFirstLetter* – проводиться пошук по першому введеному символу. Після введення першого символу в списку виконується автоматичний пошук і виділення слів, що починаються з цього символу. Для переходу до наступного слова, що починається з цього символу, введіть його знову. Пошук відбувається циклічно. Якщо *fmMatchEntryComplete* – пошук проводиться за випадковим збігом послідовно введених символів. Якщо *fmMatchEntryNone* – пошук не проводиться.

- *Об'єкт.* ***MatchFound*** – це властивість встановлює, чи відповідає текст, який Ви самі ввели, одному з елементів, наявних в списку. Якщо **True** – відповідність знайдено .Якщо **False** – відповідність не виявлено.
- *Об'єкт.* ***MatchRequired*** [= **Boolean]** – показує, чи повинен текст, введений в область редагування, обов'язково збігатися з однією з наявних рядків списку. Якщо **True** – до тих пір, поки не буде досягнуто відповідності між введеним і наявним текстом, передача фокуса іншому елементу керування неможлива. Якщо **False** – дозволяє вводити в область редагування текст, який не збігається зі списком.
- *Об'єкт.* ***MultiLine*** [= **Boolean]** – показує чи може елемент керування працювати з кількома рядками **(TextBox).**
- *Об'єкт*.***PasswordChar*** [=**String**] – дозволяє заховати текст, що вводиться в поле, символами-замінювачами. Параметр **String** містить той символ, на який буде замінятись введений текст.
- *Об'єкт*.***RowSource*** [=**String**] – містить посилання на джерело даних для списків елементів управління **ListBox** и **ComboBox**.
- *Об'єкт*.***Selected***(*index*) [=**Boolean**] – визначає стан елемента списку, виділений він чи ні. Якщо властивість має значення **True** – елемент з індексом *index* виділено.
- *Об'єкт*.***SelectedItem*** – повертає посилання на поточний виділений об'єкт **Tab** або **Page**.

- *Об'єкт*.***SelLength*** [=**Long**] – містить число символів, виділених в текстовому полі **TextBox** або поле редагування елемента управління **ComboBox**.
- *Об'єкт*.***SelText*** [=**String**] – містить текст, який в даний момент є виділеним.
- *Об'єкт*.***TabIndex*** [=**Integer**] – містить індивідуальний індекс об'єкта, який визначає порядок обходу об'єктів у формі.
- *Об'єкт*.***Tag*** [=**String**] – вільна властивість, яке може бути використано користувачем на власний розсуд.
- *Об'єкт*.***Text*** [=**String**] – містить строкове вираження, що відображається в полі елемента управління **TextBox** або в вибраному рядку елементів управління **ComboBox** або **ListBox**.
- *Об'єкт*.***Value*** [=**Variant**] – містить значення, яке вказує на поточний стан елемента управління. Параметр **Variant** для різних елементів управління має різні значення. Для елементів управління **Checkbox**, **OptionButton**, **ToggleButton** – **Null**, коли елемент активний, але він перебуває за межами; –1 (**True**) – елемент обраний; 0 (**False**) – елемент не обраний. Для елементів **ScrollBar**, **SpinButton** – цілочисельне значення, що знаходиться в інтервалі між *Max* и *Min* (вказано в цих властивостях) і відображає поточний стан. Для елементів **ComboBox**, **ListBox** – значення властивості *BoundColumn* для вибраного рядка. Для **CommandButton** – значення завжди **False**. Для **MultiPage** – цілочисельне значення, що показує індекс поточної активної вкладки. Перша вкладка має індекс 0, друга - 2 і т.д. Для **TextBox** – рядковий вираз, набране в текстовому полі.
- *Об'єкт*.***Visible*** [=**Boolean**] – визначає, чи є елемент управління видимим чи ні. Якщо **True** – елемент управління видимий, якщо **False** – не видимий.

Таблиця Е3.

Загальні методи елементів управління

Метод	Опис
Add	Дозволяє додати елемент керування під час виконання програми.
Move	Переміщує елемент керування.
Zorder	Поміщає об'єкт до або після всіх пересічних з ним об'єктів.
SetFocus	Встановлює фокус на елементі керування, що викликав цей метод. Часто застосовується в програмах обробки помилок.
Enter	Викликається завжди, коли елемент управління отримує фокус.
Show	Відображає форму на екрані.
Hide	Приховує форму

Тут представлені деякі з методів:

– *Об'єкт*.***Add*** – метод **Add** має два основних синтаксису:

– **Set** *Object* = *Об'єкт*.**Add**([**Name**[, **Caption**[, **index**]]]) – необхідно для додавання об'єктів Tab і Page в необхідно для додавання об'єктів **TabStrip** і **MultiPage**.

– **Set** *Control* = *Об'єкт*. **Add**(**ProgID**[, **Name**[, **Visible**]]) – для всіх інших елементів управління.

– **Variant** = *Об'єкт*.**AddItem**[**Item**[, **varIndex**]] – додає в списки, що складаються з одного стовпчика, новий елемент, а в списки, що складаються з декількох стовпців, новий рядок. Параметр **Item** – ідентифікатор елемента або рядка, **varIndex** – вказує на ідентифікатор елемента, за яким буде розміщений новий елемент або рядок.

– *Об'єкт* ***Clear*** – видаляє всі об'єкти з об'єкта або сімейства *Об'єкт.*

– *Об'єкт*.***Copy*** – копіює вміст об'єкта *Об'єкт* в буфер обміну.

– *Об'єкт*.***Cut*** – копіює виділений вміст об'єкта *Об'єкт* в буфер обміну. Після цього дані видаляються.

– **Set** *Object* = *Об'єкт*.**Item**(*collectionindex*) – цей метод дозволяє отримати об'єкт сімейства по його індексу в сімействі. Параметр *collectionindex* – включає в себе ім'я або індекс об'єкта, що міститься в сімействі *Об'єкт*.

Об'єкт.***Paste*** – слугує для вставки в об'єкт *Об'єкт* даних з буфера обміну.

– **Boolean** = *Объект*.***RedoAction*** – відновлює останню скасовану дію. Скасування останньої виконаної дії здійснюється за допомогою методу **UndoAction**.

– *Об'єкт*.***Remove***(*collectionindex*) – видаляє об'єкт *Об'єкт* із сімейства, форми, рамки або вкладки. Параметр *collectionindex* – строкове або цілочисельне значення, що визначає ім'я або індекс об'єкта. Індекс треба вказувати, якщо об'єкт видаляється з сімейства. Цей метод дозволяє видаляти тільки ті об'єкти, які були створені динамічно під час виконання програми.

– **Boolean** = *Об'єкт*.***RemoveItem***(*index*) – видаляє зі списку рядок з індексом *index*.

– *Об'єкт*.***SetFocus*** – цей метод дозволяє передати фокус об'єкту *Об'єкт.*

Робота програми полягає в реакції об'єкта на будь-яку подію. Кожній події об'єкта можна призначити процедуру обробки даної події. В таблиці Е4 представлені основні події елементів управління.

Таблиця Е4.

Основні події елементів управління

Подія	Опис
1	2
Activate	Дана подія генерується, коли форма стає активною
AddControl	Подія генерується, коли в форму, рамку або на вкладку додається новий елемент управління
AfterUpdate	Викликається, коли міняємо дані в елементі управління
BeforeUpdate	Викликається після зміни значення даного елемента управління, (але перед тим, як елемент управління буде оновлено)
BeforeDragOver	Генерується перед ініціалізацією процесу переміщення (drag - and - drop)
BeforeDropOrPaste	Генерується, коли користувач переносить дані або вставляє дані в об'єкт
Deactivate	Генерується, коли форма стає неактивною
DropButtonClick	Генерується при кожному розкритті або згортанні списку
Change	Викликається при всіх змінах значення елемента управління
Click	Викликається при кожному клацанні кнопкою миші на елементі управління. Ця подія використовується для зв'язування кнопки з будь-якою дією
DblClick	Викликається при кожному подвійному клацанні кнопкою миші на елементі управління. Ця подія використовується для відображення додаткових форм (наприклад, текстового поля великих розмірів)
Enter	Викликається завжди, коли елемент управління отримує фокус

Закінчення табл. Е4

1	2
Exit	Викликається завжди, коли елемент управління втрачає фокус
Error	Генерується при виникненні помилки
Initialize	Генерується відразу після завантаження об'єкта, перед його візуалізацією
KeyDown	Генерується при натисненні на клавішу
KeyPress	Генерується при натисканні будь-якої клавіші, поєднання клавіш Ctrl + <будь-яка клавіша>, клавіші Backspace, клавіші Esc
KeyUp	Генерується при відпусканні клавіші
Layout	Генерується при зміні елемента управління поточної форми
MouseDown	Генерується при натисненні на кнопку миші
MouseUp	Генерується, коли клавіша миші відпускається
MouseMove	Генерується при русі покажчика миші над об'єктом, коли об'єкт активний, тобто знаходиться у фокусі
RemoveControl	Генерується при видаленні об'єкта з материнського об'єкта-контейнера
Scroll	Дана подія генерується при переміщенні бігунка смуги прокрутки форми
SpinDown	Подія генерується при натисканні нижньої або лівої кнопки елемента управління SpinButton (лічильник)
SpinUp	Подія генерується при натисненні верхньої або правої кнопки елемента управління SpinButton (лічильник)
Terminate	Генерується при вивантаженні об'єкта з пам'яті або при видаленні зв'язку між об'єктної змінної і областю пам'яті, де розташований об'єкт
Zoom	Генерується при зміні масштабу

Відповідно до загальноприйнятих угод про імена об'єктів перші три символи імені повинні відображати вид елементу, а решта символів - призначення. У таблиці Е5 представлені поєднання перших трьох символів для найчастіше використовуваних елементів.

Таблиця Е5.

Рекомендації щодо поєднання перших трьох символів імен

Об'єкт	**Перші 3 символи в імені**	**Приклад імені**
Форма	frm	frmMyForm
Надпис	lbl	lblInfo
Текстове поле	txt	txtInput
Командна кнопка	cmd	cmdExit
Прапорець	chk	chkSound
Перемикач	opt	optLevel
Список	lsb	lsbTypes
Рамка	fra	fraChoices
Смуга прокрутки	vcb	vcbSpeed
Рисунок	pic	picChema

Кольори, що встановлюються властивостями BackColor, ForeColor і BorderColor, задаються шістнадцятирічними числами. Якщо ці властивості встановлювати не за допомогою вікна Властивості (Properties), а програмно, замість цих шістнадцятирічних чисел зручніше використовувати відповідні постійні, що задають кольори (табл. Е6).

Таблиця Е6.

Константи, що задають

Константа	**Значення**	**Колір**
vbBlack	0x0	Чорний
vbRed	0xFF	Червоний
vbGreen	0xFF00	Зелений
vbYellow	0xFFFF	Жовтий
vbBlue	0xFF0000	Синій
vbMagenta	0xFF00FF	Рожевий
vbCyan	0xFFFF00	Блакитний
vbWhite	0xFFFFFF	Білий

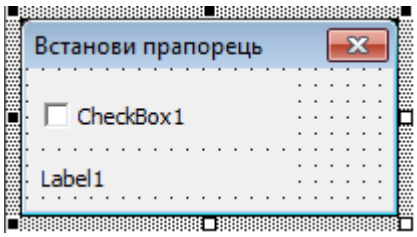
Додаток Ж. Приклади роботи з формами

Приклад 1 (робота з елементом управління CheckBox – Прапорець):

Створити екранну форму, де на форму виводиться повідомлення про встановлений Прапорець.

Додаємо на форму елементи **Label** та **CheckBox**.

У вікні Властивості (**Properties**) форми встановлюємо значення наступної властивості: Caption (заголовок) – **Встанови прапорець** (для UserForm1)



У вікні Коду (**Code**) для події Click елементу CheckBox1 записуємо код:

```
If CheckBox1.Value = True Then
  Label1.Caption = " Включений"
 Else
  Label1.Caption = " Виключений"
End If
```

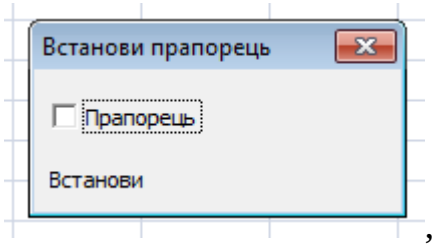
Для події Activate елементу UserForm1 записуємо код

```
CheckBox1.Caption = "Прапорець"
 CheckBox1.Value = False
 Label1.Caption = "Встанови"
```

Лістинг програми:

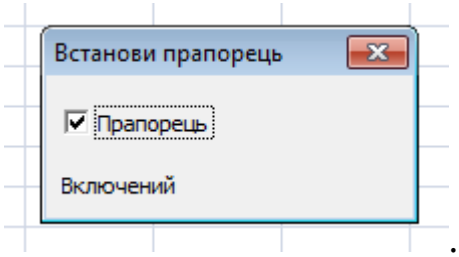
```
CheckBox1
Private Sub CheckBox1_Click()
 If CheckBox1.Value = True Then
   Label1.Caption = "Включений"
  Else
   Label1.Caption = "Виключений"
 End If
 End Sub
Private Sub UserForm_Activate()
 CheckBox1.Caption = "Прапорець"
 CheckBox1.Value = False
 Label1.Caption = "Встанови"
End Sub
```

Після запуску програми на екран буде виведена форма:



,

а після встановлення прапорця отримаємо:



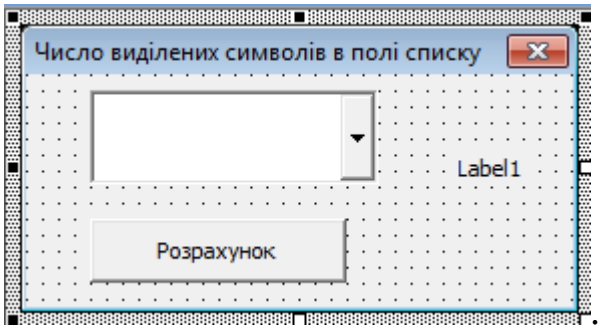
.

Приклад 2 (робота з елементом управління ComboBox – Поле зі списком):

Створити екранну форму, де на форму виводиться повідомлення про кількість символів, що була виділена в полі списку. Текст для виділення може бути введений в поле або обраний з попередньо створеного списку.

Додаємо на форму елементи **ComboBox**, **Label** та **CommandButton**.

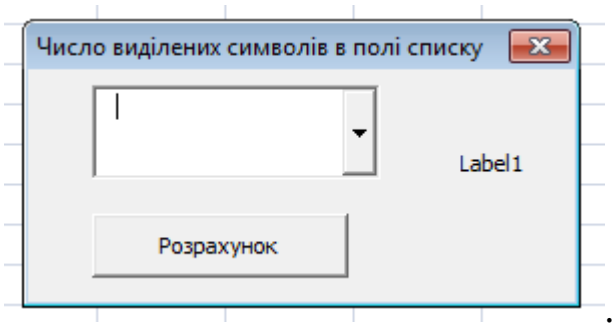
У вікні Властивості (**Properties**) форми встановлюємо значення властивості Caption (заголовок) – **Число виділених символів в полі списку** (для UserForm1), а у вікні Властивості Кнопки (CommandButton1) – Caption (заголовок) – **Розрахунок**. Отримаємо:



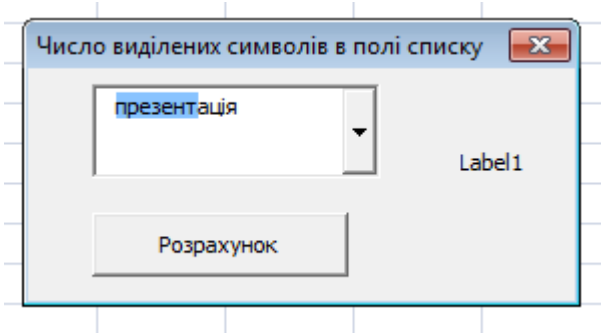
У вікні Коду (**Code**) для подій записуємо код:

```
UserForm
Private Sub CommandButton1_Click()
 Label1.Caption = ComboBox1.SelLength
End Sub
Private Sub UserForm_Activate()
 ComboBox1.ColumnCount = 2
 ComboBox1.AddItem "Виділіть дещо"
 ComboBox1.AddItem "Кількість слів"
End Sub
```

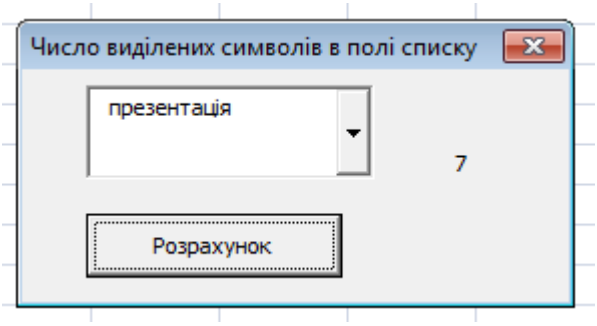
Після запуску програми на екран буде виведена форма:



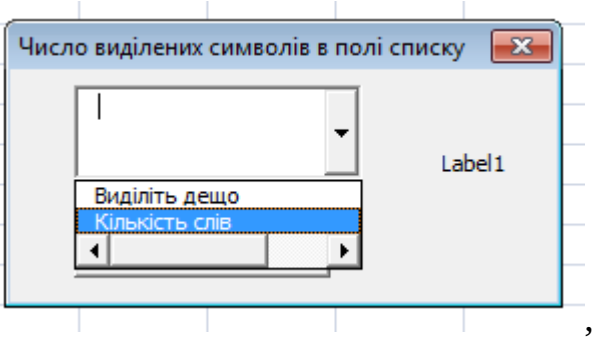
В полі списку можна ввести свій текст і виділити потрібну кількість символів:



та натиснути кнопку Розрахунок. На форму в обраному місці Label (Надпис) буде виведена кількість символів, що була виділена.

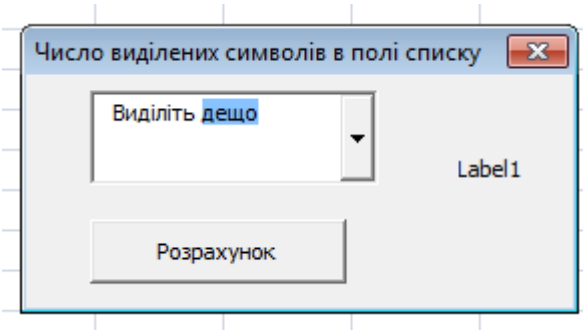


Можна також натиснути  та зробити вибір із запропонованого списку



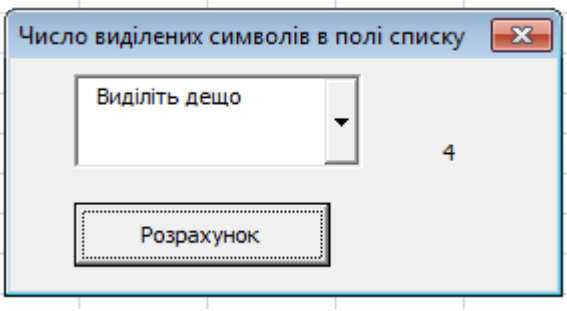
,

а потім у вікні виділити потрібну кількість символів:



,

та натиснути кнопку Розрахунок. На форму в обраному місці Label (Надпис) буде виведена кількість символів, що була виділена.

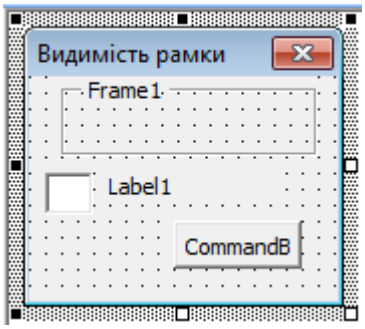


Приклад 3 (робота з елементом управління Frame – Рамка):

Створити екранну форму, де в залежності від введеного значення 1 або 0 на форму виводиться або не виводиться Рамка.

Додаємо на форму елементи **Frame**, T**extBox** та **Label**.

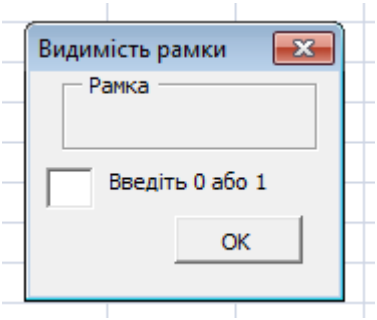
У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Видимість рамки** (для UserForm1). Отримаємо:



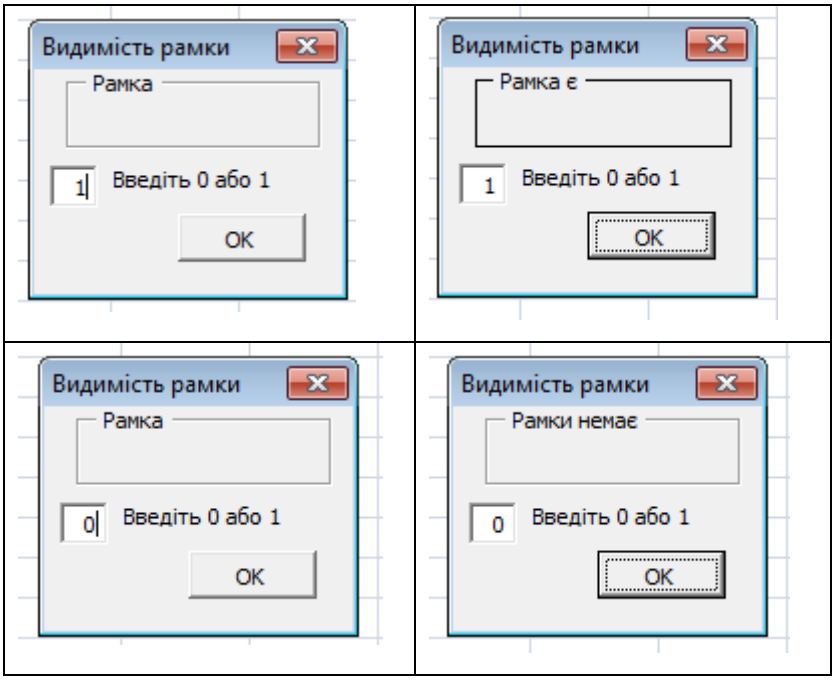
У вікні Коду (**Code**) для подій записуємо код:

```
UserForm                                                    Activa
Private Sub CommandButton1_Click()
 If TextBox1.Text = 0 Then Frame1.Caption = "Рамки немає"
 If TextBox1.Text = 1 Then Frame1.Caption = "Рамка є"
 If TextBox1.Text = 0 Then Frame1.BorderStyle = fmBorderStyleNone
 If TextBox1.Text = 1 Then Frame1.BorderStyle = fmBorderStyleSingle
 If TextBox1.Text > 1 Then MsgBox "Тільки 0 або 1"
 If TextBox1.Text < 0 Then MsgBox "Тільки 0 або 1"
End Sub
Private Sub UserForm_Activate()
 Frame1.Caption = "Рамка"
 CommandButton1.Caption = "OK"
 Label1.Caption = "Введіть 0 або 1"
End Sub
```

Після запуску програми на екран буде виведена форма:



Залежно від введеного значення: 1 рамка з’являться; 0 рамка зникає після натискання кнопки ОК.

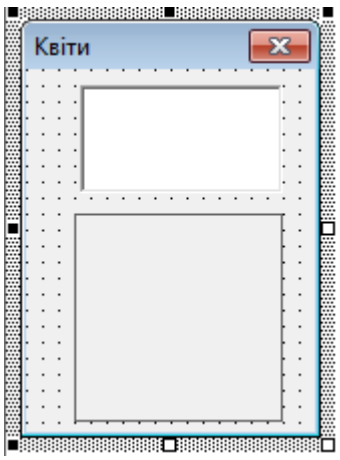


Приклад 4 (робота з елементом управління Image – Зображення або Малюнок):

Створити екранну форму, де на форму виводиться Малюнок в залежності від елементу списку, що обраний в полі списку.

Додаємо на форму елементи **ListBox** та **Image**.

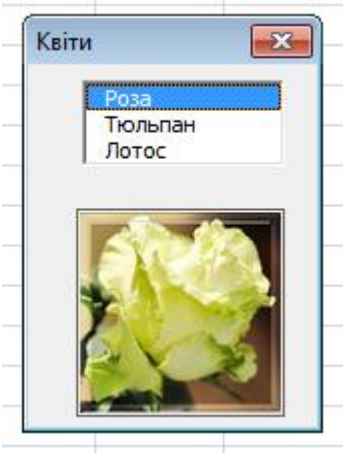
У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Квіти** (для UserForm1). Отримаємо:



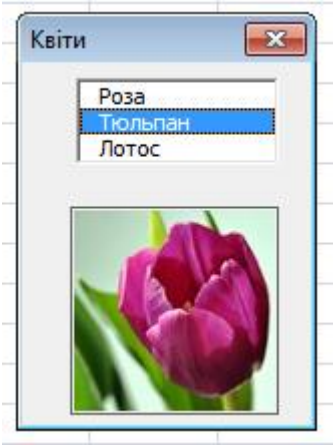
У вікні Коду (**Code**) для подій записуємо код:

```
UserForm
Private Sub ListBox1_Click()
 Dim bs As Variant
 bs = Array("d:\Роза.jpg", "d:\Тюльпан.jpg ", "d:\Лотос.jpg ")
 Image1.Picture = LoadPicture(bs(ListBox1.ListIndex))
End Sub
Private Sub UserForm_Initialize()
 ListBox1.AddItem "Роза"
 ListBox1.AddItem "Тюльпан"
 ListBox1.AddItem "Лотос"
 ListBox1.ListIndex = 0
End Sub
```

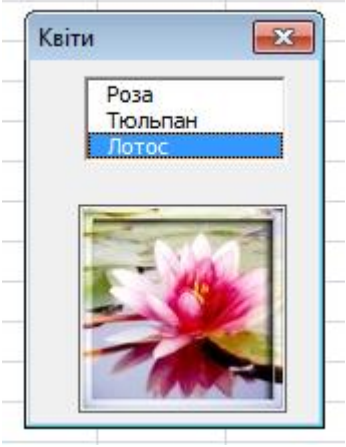
Після запуску програми на екран буде виведена форма



Залежно від вибору елементів списку можемо отримати:



або



Примітка. В програмному коді вказаний шлях до файлів Роза, Тюльпан, Лотос. де зберігаються рисунки: d:\Роза.jpg", "d:\Тюльпан.jpg ", "d:\Лотос.jpg. Для коректної роботи програми в кореневому каталозі диску d: повинні знаходитись ці файли.

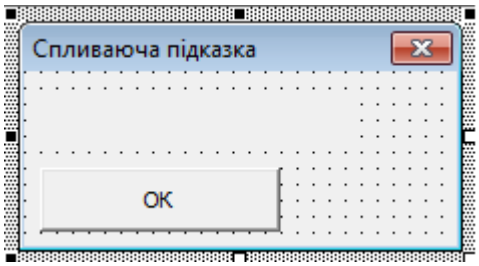
Приклад 5 (робота з спливаючою підказкою та елементом управління Label – Надпис):

Створити екранну форму, де на форму виводиться спливаюча підказка для елементу управління (Label), що обраний для цього.

Додаємо на форму елементи **Label** та **CommandButton**.

У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Спливаюча підказка** (для UserForm1), а у вікні Властивості Кнопки (CommandButton1) – Caption (заголовок) – **ОК**, Caption (заголовок) – для Label1 заголовок відсутній.

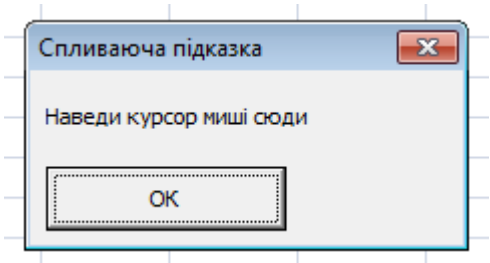
Отримаємо:



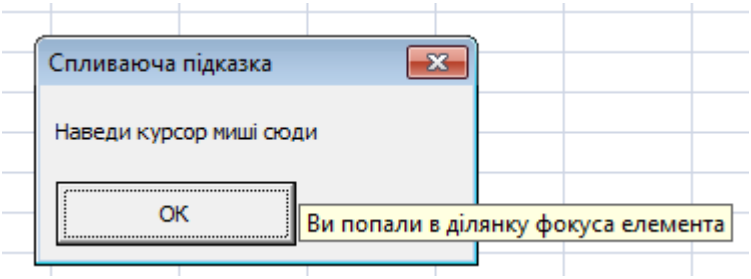
У вікні Коду (**Code**) для подій записуємо код:

```
UserForm

Private Sub CommandButton1_Click()
 UserForm1.Hide
End Sub
Private Sub Label1_Click()
 Label1.ControlTipText = "Ви попали в ділянку фокуса елемента"
End Sub
Private Sub UserForm_Initialize()
 Label1.Caption = "Наведи курсор миші сюди"
End Sub
```

Після запуску програми на екран буде виведена форма:



Після клацання мишею на Надпису спливаюча підказка буде з’являтися, коли курсор миші попадає у фокус об’єкта (Надпис).



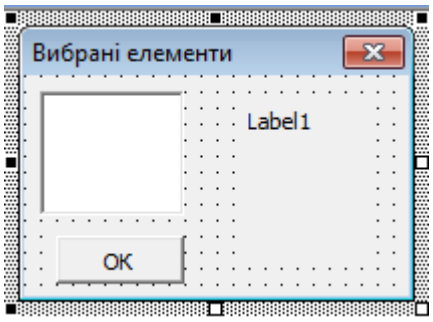
Приклад 6 (робота з елементом управління ListBox – Список):

Створити екранну форму, де на форму виводиться елементи списку, що виділені у ньому.

Додаємо на форму елементи **Label**, **ListBox** та елемент **CommandButton**.

У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Вибрані елементи** (для UserForm1), а у вікні Властивості Кнопки (CommandButton1) – Caption (заголовок) – **ОК**.

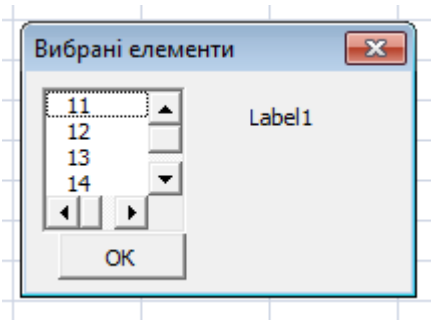
Отримаємо:



У вікні Коду (**Code**) для подій записуємо код:

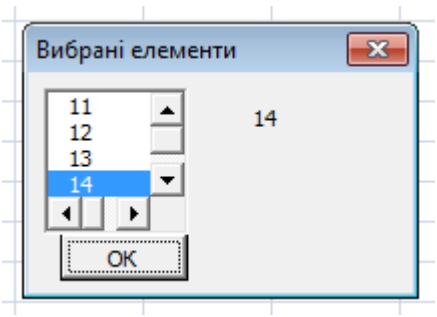
```
UserForm
Private Sub CommandButton1_Click()
Dim i As Integer
Dim s As String
For i = 0 To ListBox1.ListCount - 1
 If ListBox1.Selected(i) Then
  s = s + CStr(ListBox1.List(i)) & vbCr
 End If
Next
If s = Empty Then
 Label1.Caption = "Вибери"
Else
 Label1.Caption = s
End If
End Sub
Private Sub UserForm_Activate()
 With ListBox1
  .List = Array(11, 12, 13, 14, 15)
  .ListIndex = 0
  .MultiSelect = fmMultiSelectMulti
 End With
End Sub
```

Після запуску програми на екран буде виведена форма:

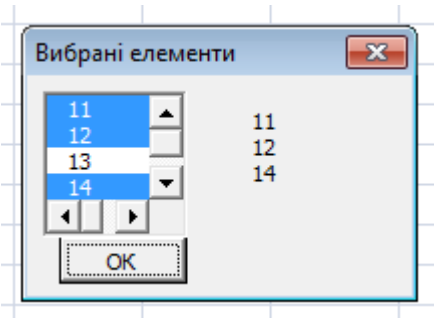


На форму в обраному місці Label (Надпис) після натискання кнопки ОК будуть виведені елементи, що були виділені (один елемент або декілька).

Наприклад.



або

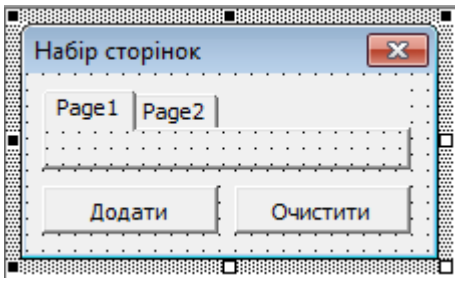


Приклад 7 (робота з елементом управління MultiPage – Набір сторінок):

Створити екранну форму, де на форму додаються сторінки за допомогою кнопки Додати, а що виділені у ньому за допомогою кнопки Очистити всі сторінки видаляються.

Додаємо на форму елемент **MultiPage**, та два елементи **CommandButton**.

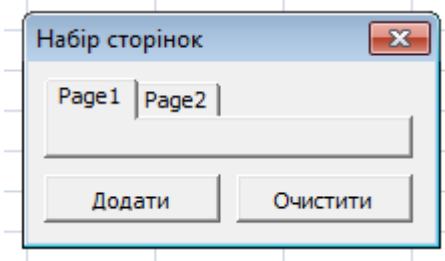
У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Набір сторінок** (для UserForm1), а у вікні Властивості Кнопки (CommandButton1) – Caption (заголовок): для першої кнопки **Додати**, а для другої – **Очистити**. Отримаємо:



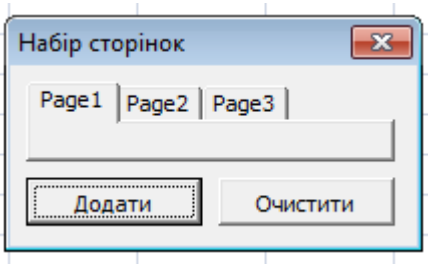
У вікні Коду (**Code**) для подій записуємо код:

```
UserForm
Private Sub CommandButton1_Click()
 MultiPage1.Pages.Add
End Sub
Private Sub CommandButton2_Click()
 MultiPage1.Pages.Clear
End Sub
Private Sub UserForm_Click()

End Sub
```

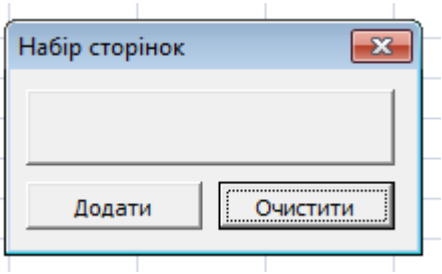
Після запуску програми на екран буде виведена форма:



При натисканні на кнопку Додати на форму додаються сторінки.



При натисканні на кнопку Очистити сторінки видаляються з форми.



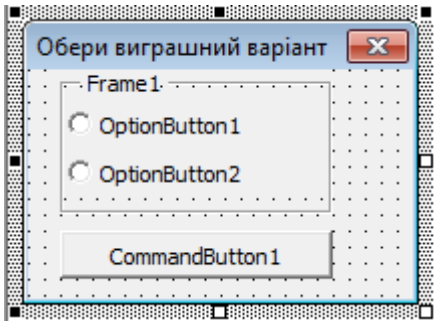
Приклад 8 (робота з елементом управління **OptionButton** – Перемикач):

Створити екранну форму, де на форму додається Рамка, два Перемикачі та Кнопка. Форма надає можливість визначення виграшного варіанту при киданні монети.

Додаємо на форму елемент**Frame**, два елементи **OptionButton** та елемент **CommandButton**.

У вікні Властивості (**Properties**) встановлюємо значення наступні властивості: Caption (заголовок) – Обери виграшний варіант (для UserForm1); Caption (заголовок) – Варіант (для Frame1).

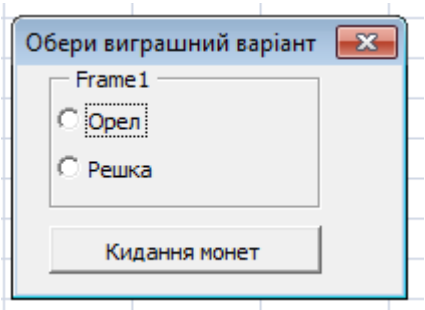
Отримаємо:



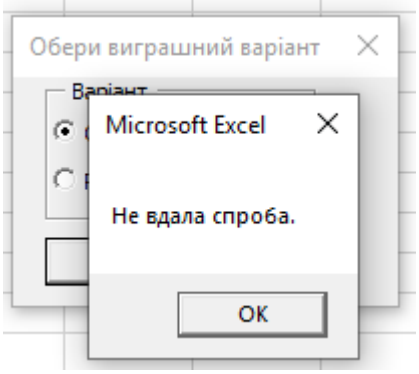
У вікні Коду (**Code**) для подій записуємо код:

```
CommandButton1
Private Sub CommandButton1_Click()
Dim M As Single
Randomize
M = Int(2 * Rnd)
If OptionButton1.Value = True Then
 If M = 0 Then MsgBox "Не вдала спроба."
 If M = 1 Then MsgBox "Вдала спроба."
End If
If OptionButton2.Value = True Then
 If M = 1 Then MsgBox "Не вдала спроба."
 If M = 0 Then MsgBox "Вдала спроба."
End If
End Sub
Private Sub UserForm_Activate()
 OptionButton1.Caption = "Орел"
 OptionButton2.Caption = "Решка"
 CommandButton1.Caption = "Кидання монет"
End Sub
```

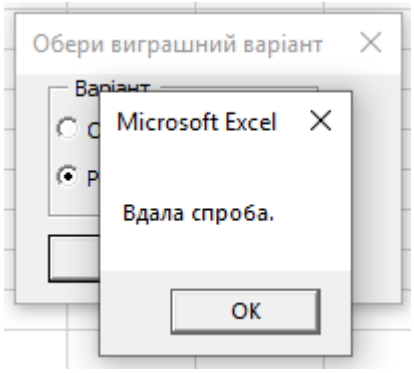
Після запуску програми на екран буде виведена форма:



Далі встановлюється необхідний перемикач **Орел** або **Решка** і натискається кнопка **Кидання монет**. Результат виводиться на екран.



або

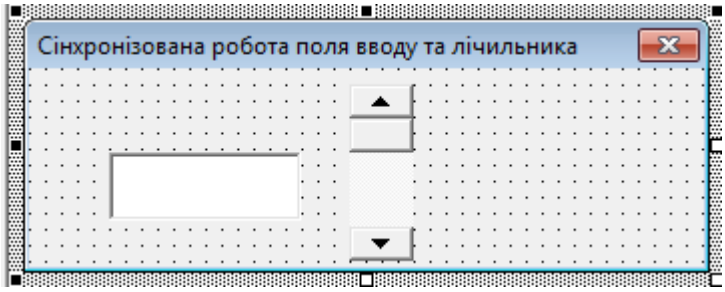


Приклад 9 (робота з елементом управління ScrollBar – Смуга прокрутки):

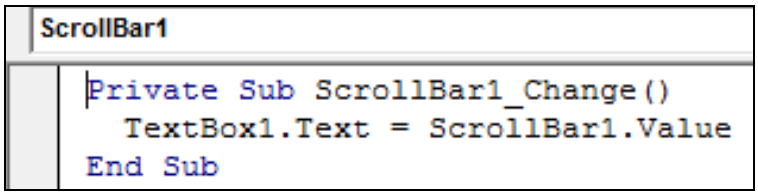
Створити екранну форму, де за допомогою Смуги прокрутки на формі у текстовому полі змінюються значення лічильника.

Додаємо на форму елементи **TextBox** та **ScrollBar**.

У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Синхронізована робота поля вводу та лічильника**. Отримаємо:



У вікні Коду (**Code**) для події Change елементу ScrollBar1 записуємо код:



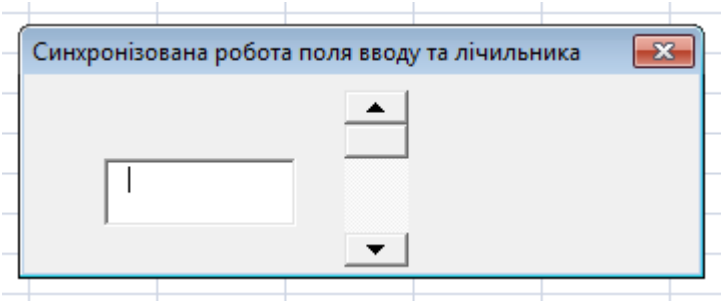
Для події **Change** елементу TextBox1 записуємо код:

```
Dim r AsVariant
r = TextBox1.Text
IfIsNumeric(r) Then
If ScrollBar1.Min <= r And r <= ScrollBar1.Max Then
  ScrollBar1.Value = r
EndIf
EndIf
```

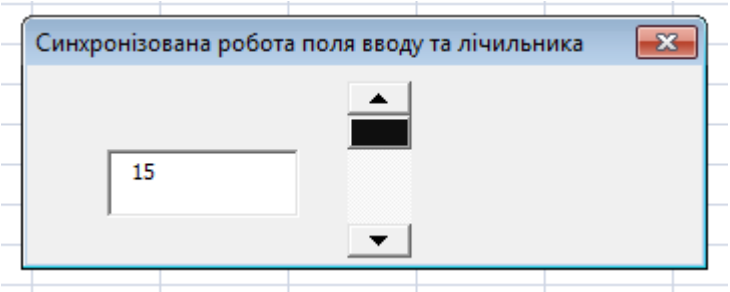
Для події Initialize елементу UserForm1 записуємо код:

```
ScrollBar1.Min = 0
ScrollBar1.Max = 16
```

Після запуску програми на екран буде виведена форма:



При натисканні на кнопки смуги прокрутки в текстове вікно виводяться значення.

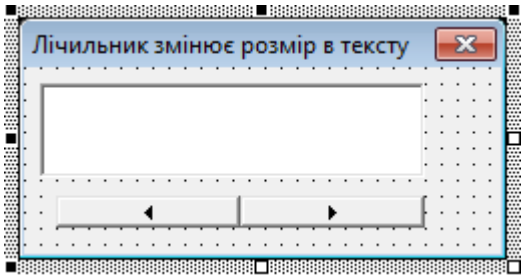


Приклад 10 (робота з елементом управління SpinButton – Лічильник):

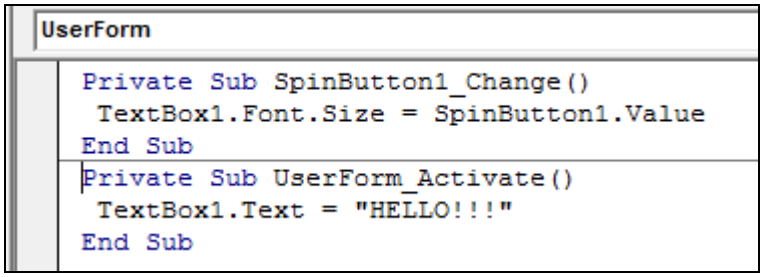
Створити екранну форму, де за допомогою Лічильника на формі у текстовому полі змінюються розмір тексту.

Додаємо на форму елемент **TextBox** та елемент **SpinButton**.

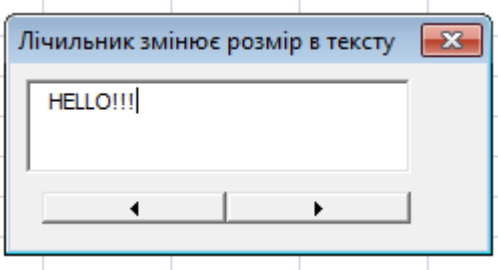
У вікні Властивості (**Properties**) встановлюємо значення наступну властивість Caption (заголовок) – Лічильник змінює розмір тексту (для UserForm1)



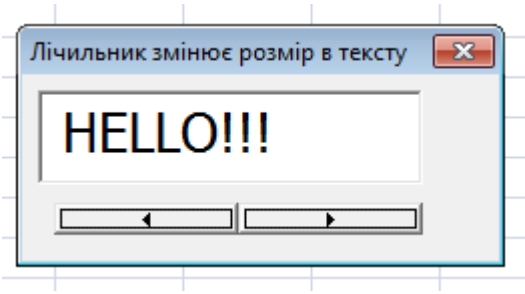
У вікні Коду (**Code**) для подій записуємо код:



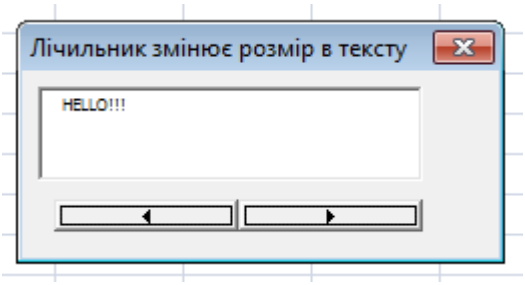
Після запуску програми на екран буде виведена форма:



Змінюючи значення лічильника змінюється розмір тексту у вікні в більшу або меншу сторону.



або



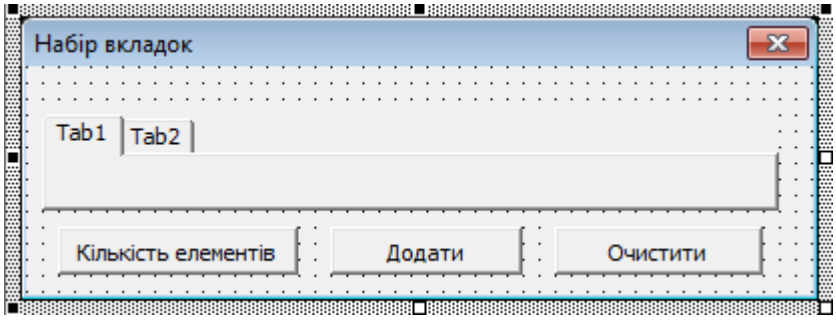
Приклад 11 (робота з елементом управління TabStrip – Набір закладок (вкладок)):

Створити екранну форму, де за допомогою кнопок можна додати на форму вкладки чи очистити її від вкладок.

Додаємо на форму елемент **TabStrip**, та три елементи **CommandButton**.

У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Набір вкладок** (для UserForm1), а у вікні Властивості Кнопки (CommandButton1) – Caption (заголовок): для першої кнопки (для CommandButton1) – **Кількість елементів,** для другої (для CommandButton2) – **Додати,** а для третьої (для CommandButton3) – **Очистити**.

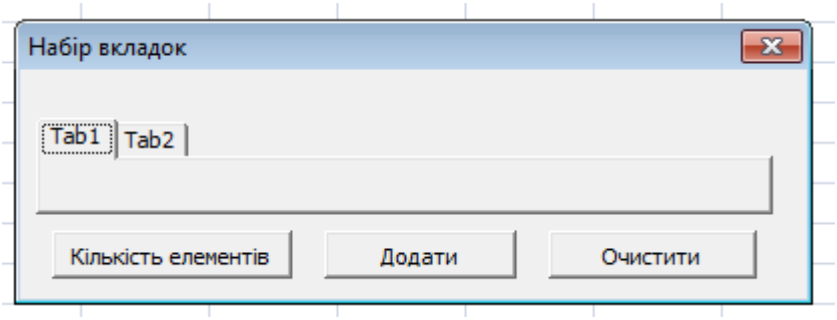
Отримаємо:



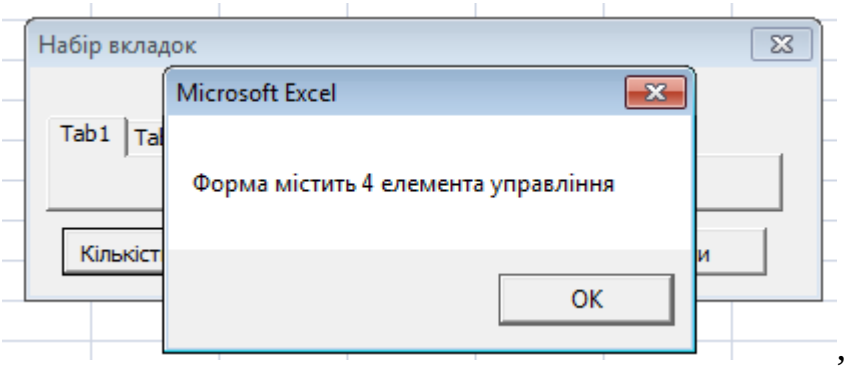
У вікні Коду (**Code**) для подій записуємо код:

```
UserForm                                                    Click
Private Sub CommandButton1_Click()
 Dim i As Control
 MsgBox "Форма містить " & Controls.Count & " елемента управління"
 For Each i In Controls
  If (i.Name Like "TabStrip*") Then _
  MsgBox i.Name & "містить вкладок =" & i.Tabs.Count
 Next
End Sub
Private Sub CommandButton2_Click()
 TabStrip1.Tabs.Add
End Sub
Private Sub CommandButton3_Click()
  TabStrip1.Tabs.Clear
End Sub
Private Sub UserForm_Click()

End Sub
```

Після запуску програми на екран буде виведена форма:

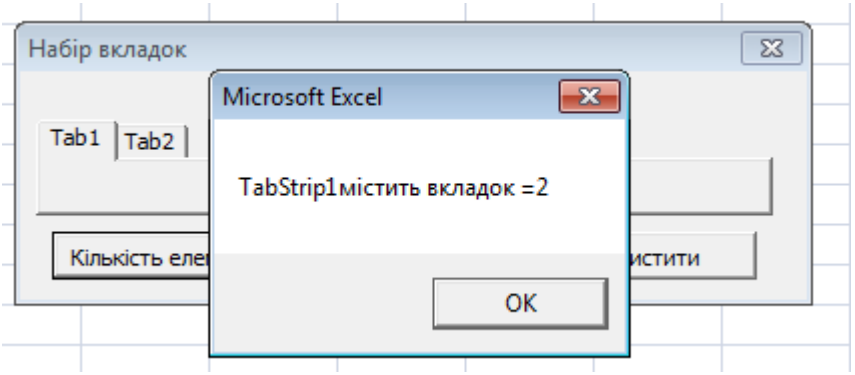


Натискання на кнопку **Кількість елементів** дозволить визначити кількість елементів управління

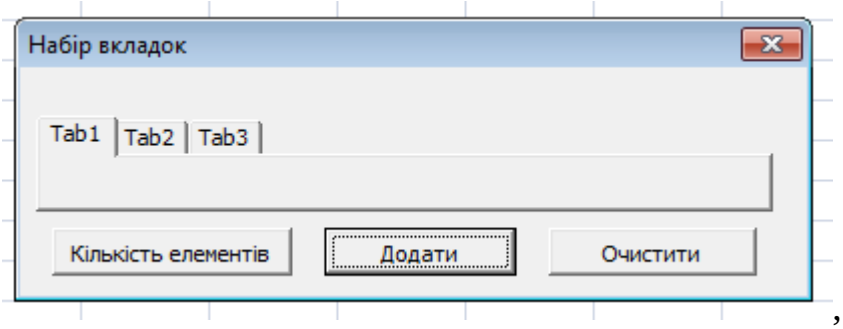


,

а натискання на кнопку ОК дозволить визначити кількість вкладок.

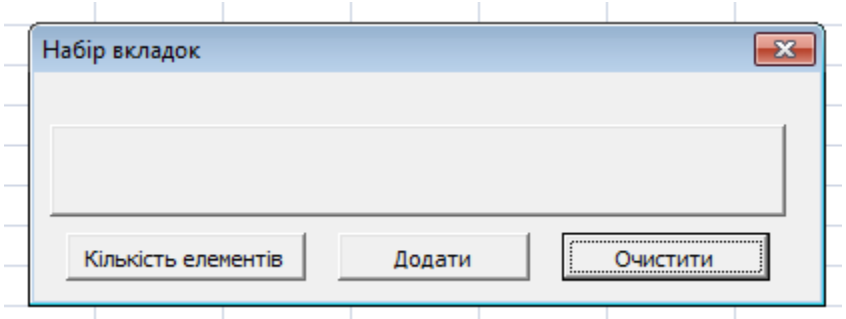


Натискання на кнопку **Додати** дозволить додати вкладки на форму:



,

а натискання на кнопку **Очистити** дозволяє видалити вкладки з форми.

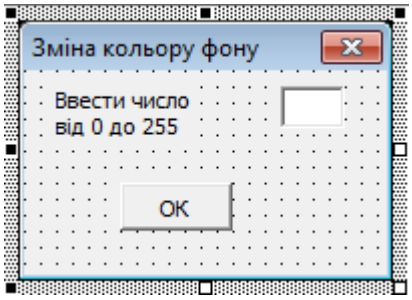


Приклад 12 (робота з елементом управління Textbox – Текстове поле або Поле):

Створити екранну форму, де за допомогою значення, що введено в Текстове поле можна змінити колір її фону.

Додаємо на форму елементи **Label**, **TextBox** та **CommandButton**.

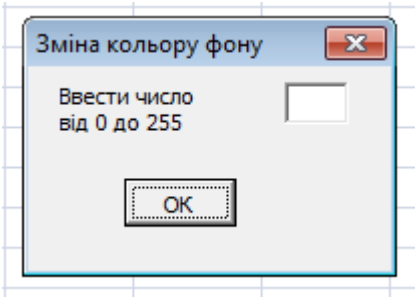
У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Зміна кольору фону** (для UserForm1); у вікні Властивості Кнопки (CommandButton1) – Caption (заголовок) – **ОК**; у вікні Властивості Мітки (Label1) – Caption (заголовок) – **Ввести число від 0 до 255**. Отримаємо:



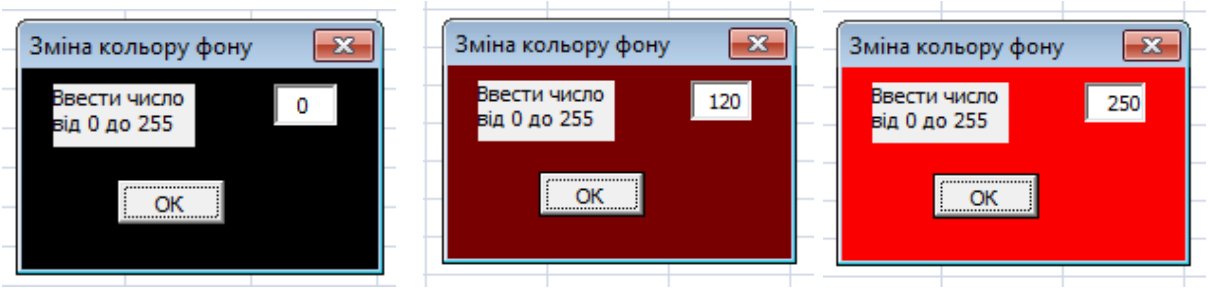
У вікні Коду (**Code**) для події Click елементу CommandButton1 записуємо код:

```
CommandButton1
Private Sub CommandButton1_Click()
 UserForm1.BackColor = TextBox1.Value
End Sub
```

Після запуску програми на екран буде виведена форма:



Вводячи число (від 0-чорний колір до 255-червоний) та натискаючи кнопку ОК зміниться колір фону.



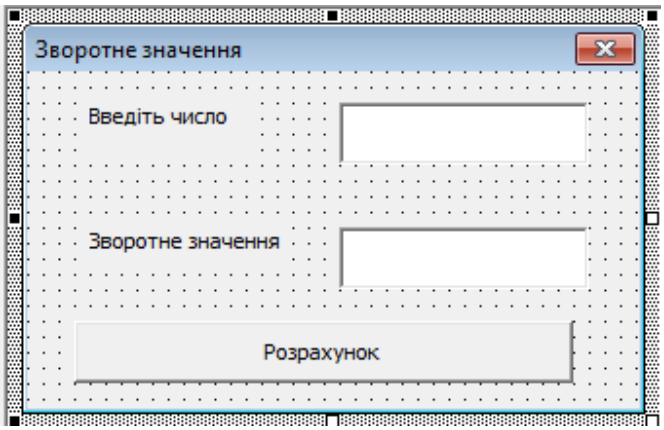
Приклад 13 (робота з екранною формою та зміною кольору фону)

Створити екранну форму, де знаходиться зворотне значення, до того що введено та перевіряється правильність введеного значення.

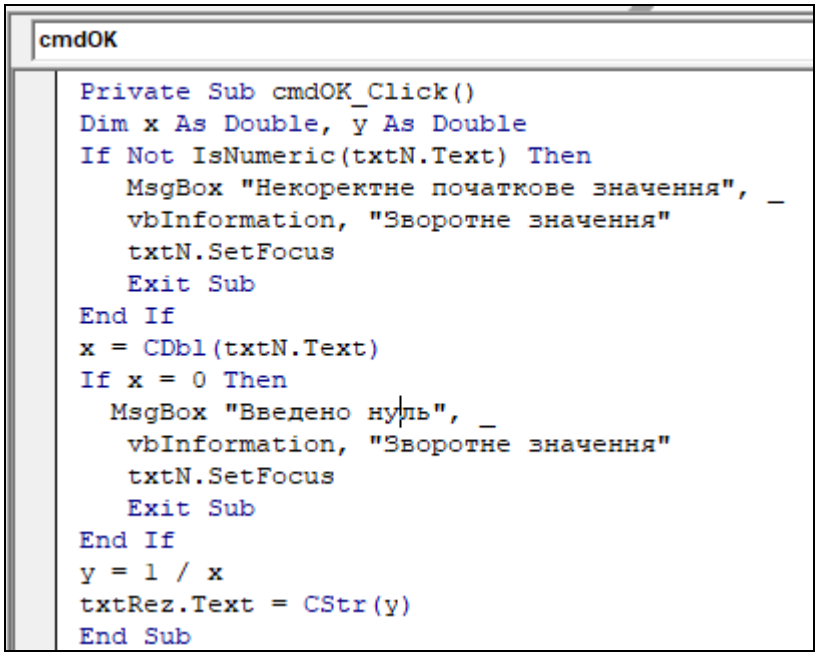
Додаємо на форму 2 елементи **Label**, 2 елементи TextBox, та елемент **CommandButton**.

У вікні Властивості (**Properties**) форми встановлюємо наступну властивість: Caption (заголовок) – **Зворотне значення** (для UserForm1); у вікні Властивості Кнопки (CommandButton1) – Caption (заголовок) – **Розрахунок**; у вікні Властивості Мітки (Label1) – Caption (заголовок) – **Введіть число**; у вікні Властивості Мітки (Label2) – Caption (заголовок) – **Зворотне значення**.

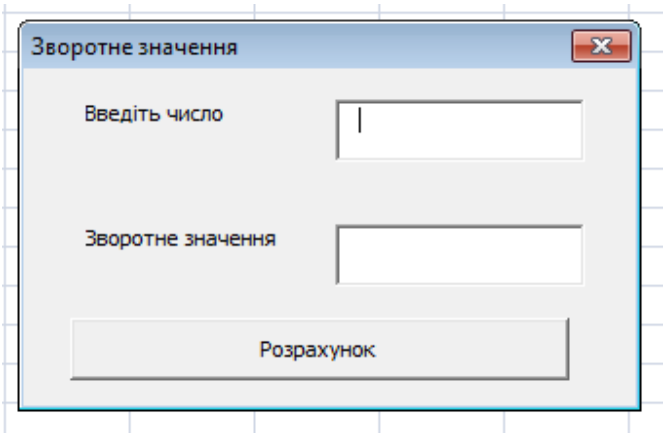
Отримаємо:



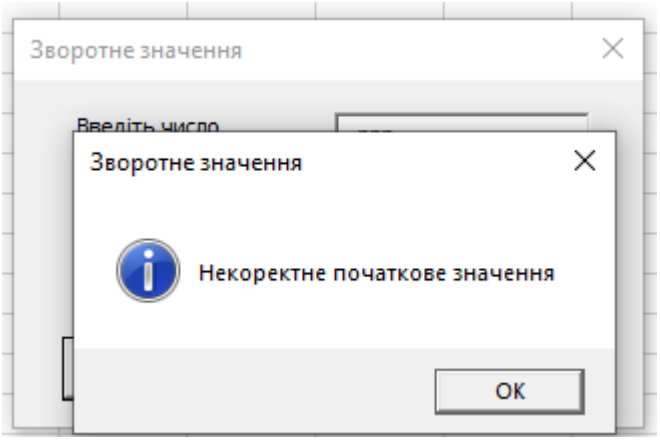
У вікні Коду (**Code**) для подій Click елементу CommandButton1 записуємо код:



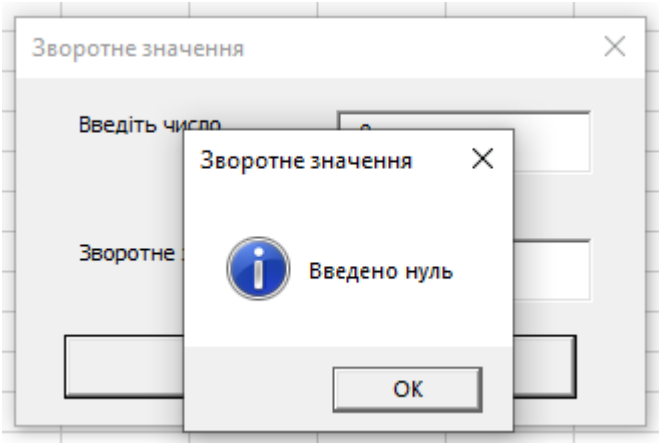
Після запуску програми на екран буде виведена форма:



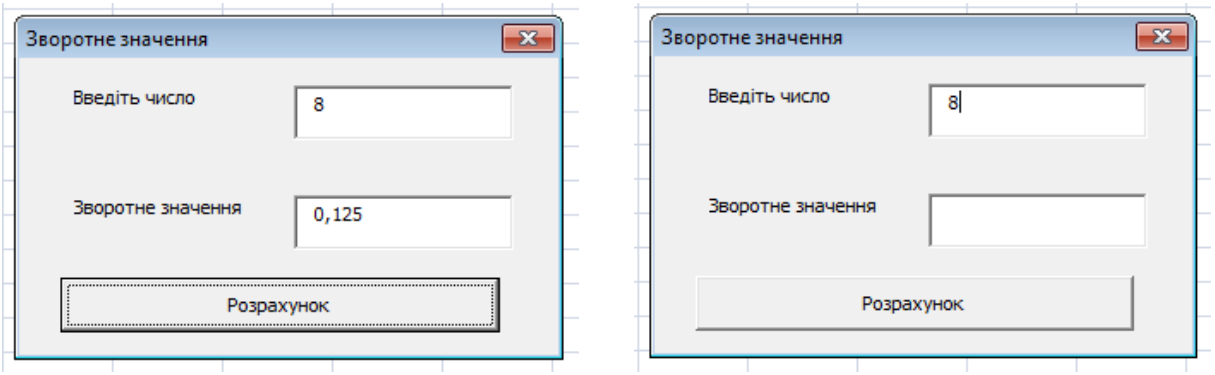
Якщо в поле **Введіть число** буде введений текст або воно залишиться пустим, то при натисканні кнопки **Розрахунок**, то з’явиться вікно:



Якщо в поле **Введіть число** буде введений 0, то при натисканні кнопки Розрахунок, то з’явиться вікно:



У всіх інших випадках при введенні числа в поле **Введіть число** в поле **Зворотне значення** буде виведений результат.

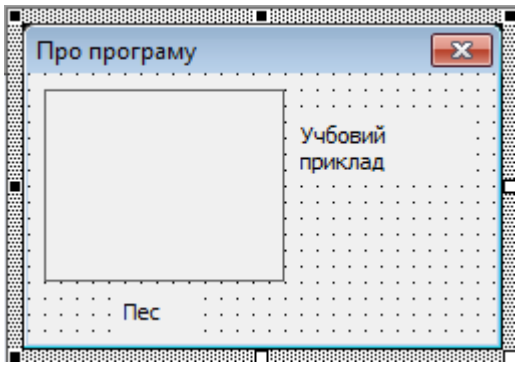


Приклад 14 (робота з екранною формою, де завантажується картинка, яка має підпис)

Створити екранну форму, де завантажується картинка, яка має підпис.

Додаємо на форму 2 елементи **Label** та елемент **Image**.

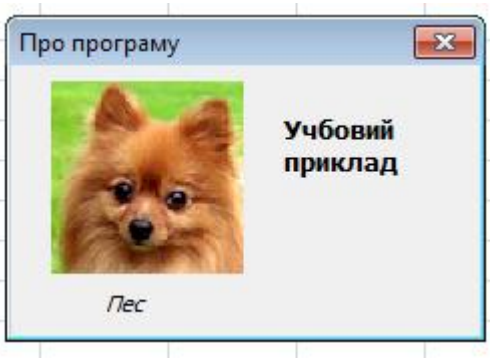
У вікні Властивості (**Properties**) встановлюємо значення наступних властивостей: Caption (заголовок) – **Про програму** (для UserForm1); Caption (заголовок) – **Учбовий приклад** (для Label1); Caption (заголовок) – **Пес** (для Label2). Отримаємо:



У вікні Коду (**Code**) для події Initialize елементу UserForm записуємо код:

```
UserForm
Private Sub UserForm_Initialize()
Image1.BorderStyle = fmBorderStyleNone
Image1.Picture = LoadPicture("d:\пес.bmp")
Image1.PictureSizeMode = fmPictureSizeModeZoom
Label1.Font.Size = 10
Label1.Font.Bold = True
Label2.Font.Size = 8
Label2.Font.Italic = True
End Sub
```

Після запуску програми на екран буде виведена форма:



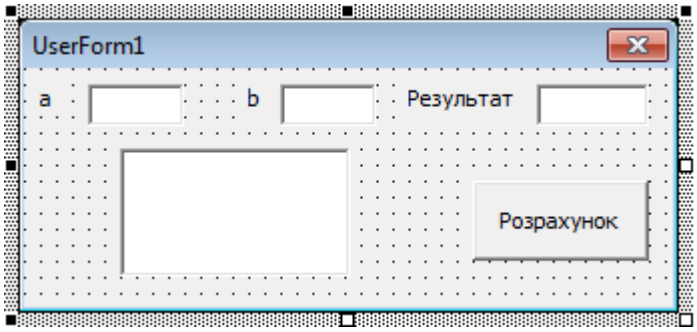
Примітка. В програмному коді вказаний шлях до файлу малюнку Пес, де зберігається рисунок: d:\nec.bmp. Для коректної роботи програми в кореневому каталозі диску d: повинен знаходитись цей файл.

Приклад 15 (простий калькулятор)

Створити екранну форму, де за допомогою значення, що введено в Створити екранну форму, де виконуються дії додавання, віднімання, множення та ділення з вибором цих дій.

Додаємо на форму 3 елементи **Label**, 2 елементи **TextBox**, та елемент **CommandButton**.

У вікні Властивості (**Properties**) встановлюємо значення наступних властивостей: Caption (заголовок) – **a** (для Label1); Caption (заголовок) – **b** (для Label2); Caption (заголовок) – **Результат** (для Label3); Caption (заголовок) – **Розрахунок** (для CommandButton1); Name(ім'я) – **txtA** (дляTextBox1); Name(ім'я) – **txtB** (для TextBox2); Name(ім'я) – **txtRez** (для TextBox3); Name(ім'я) – **lstV** (для ListBox); Name(ім'я) – **cmdOk** (для CommandButton1). Отримаємо:



У вікні Коду (**Code**) для події Click елементу cmdOk записуємо код:

cmdOk

```
Private Sub cmdOk_Click()
Dim a As Double, b As Double, rez As Double
a = txtA.Text
b = txtB.Text
Select Case lstV.ListIndex
Case 0
   rez = a + b
Case 1
   rez = a - b
Case 2
   rez = a * b
Case 3
   rez = a / b
End Select
txtRez.Text = rez
End Sub
```

Для події Click елементу lstV записуємо код:

```
Private Sub lstV_Click()
UserForm1.Caption = lstV.Text
End Sub
```

Для події Initialize елементу UserForm1 записуємо код:

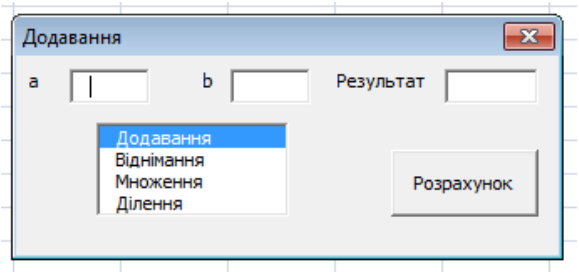
```
Private Sub UserForm_Initialize()
Dim r As Variant
r = Array("Додавання", "Віднімання", "Множення", "Ділення")
lstV.List = r
lstV.ListIndex = 0
txtRez.Locked = True
End Sub
```

Лістинг програми:

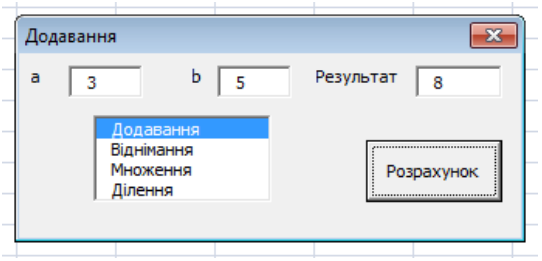
cmdOk

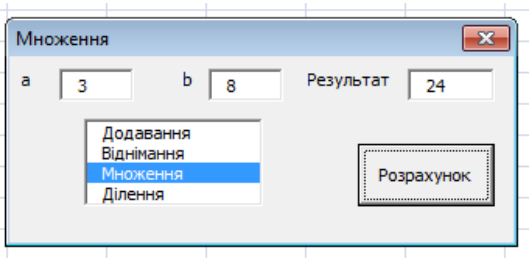
```
Private Sub cmdOk_Click()
Dim a As Double, b As Double, rez As Double
a = txtA.Text
b = txtB.Text
Select Case lstV.ListIndex
Case 0
   rez = a + b
Case 1
   rez = a - b
Case 2
   rez = a * b
Case 3
   rez = a / b
End Select
txtRez.Text = rez
End Sub
Private Sub lstV_Click()
UserForm1.Caption = lstV.Text
End Sub
Private Sub UserForm_Initialize()
Dim r As Variant
r = Array("Додавання", "Віднімання", "Множення", "Ділення")
lstV.List = r
lstV.ListIndex = 0
txtRez.Locked = True
End Sub
```

Після запуску програми на екран буде виведена форма:



Вводимо два числа **a** та **b** і вибираємо потрібну дію. Після натискання кнопки **Розрахунок**, у вікно Результат буде виведений результат розрахунку.

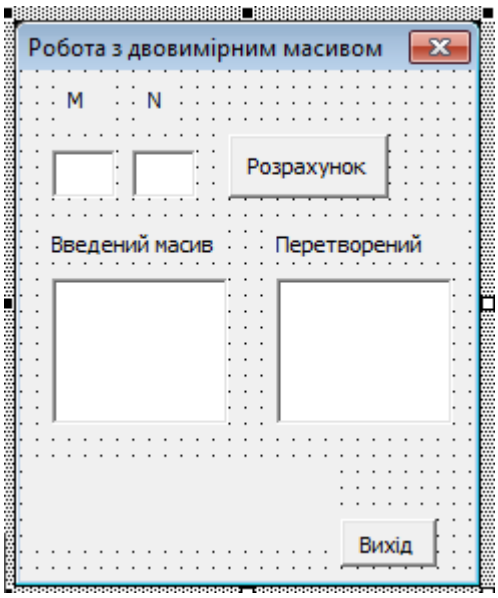




Приклад 16

Створити екранну форму, де в двовимірному масиві розміром m×n (не більше 6×6) міняються місцями перший та останній рядки масиву. Масив заповнюється випадковими числами.

Додаємо на форму чотири елементи **TextBox**, два елементи **CommandButton** та п‘ять елементів **Label**.



У вікні Властивості (**Properties**) різних елементів управління встановлюємо значення наступних властивостей:

– MultiLine (багато рядків) – True (для TextBox3);
– MultiLine (багато рядків) – True (для TextBox4);
– Caption (заголовок) – Робота з масивом (для UserForm1);
– Caption (заголовок) – M (для Label1);
– Caption (заголовок) – N (для Label2);
– Caption (заголовок) – Введений масив (для Label3);
– Caption (заголовок) – Перетворений (для Label4);
– Caption (заголовок) – (для Label5);
– Caption (заголовок) – Розрахунок (для CommandButton1);
– Caption (заголовок) – Вихід (для CommandButton2);
– Name (ім’я) – txtM (для Label1);
– Name (ім’я) – txtN (для Label2);
– Name (ім’я) – txtA (для Label3);
– Name (ім’я) – lblText (для Label4);
– Name (ім’я) – CmdRun (для CommandButton1);
– Name (ім’я) – CmdExit (для CommandButton2).

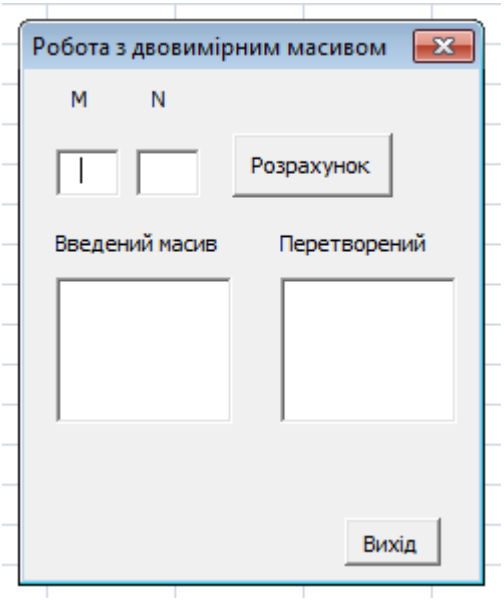
Примітка. Коли ми вводимо ім’я елемента, то необхідно слідкувати за іншими властивостями враховуючи це ім’я.

У вікні Коду (**Code**) для події Click елементу **cmdRun** записуємо код:

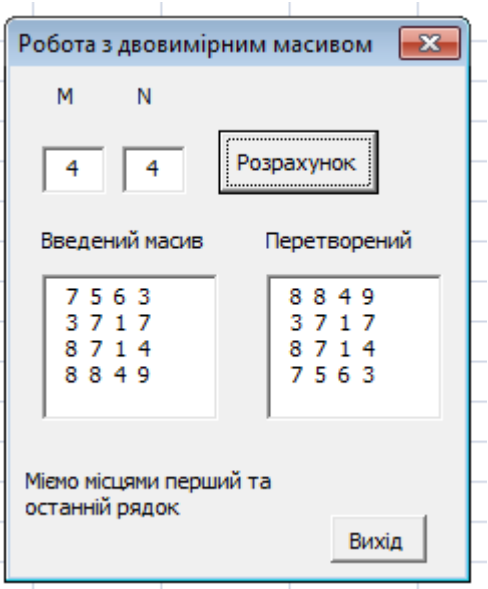
UserForm

```
Private Sub CmdExit_Click()
 UserForm1.Hide
End Sub
Private Sub cmdRun_Click()
' Кнопка Розрахунок
Dim m As Integer, n As Integer, r As Integer
Dim i As Integer, j As Integer
Dim a() As Integer
txtA.Value = ""
txtRez.Value = ""
m = CInt(txtM.Value)
n = CInt(txtN.Value)
ReDim a(m, n) As Integer
For i = 1 To m
 For j = 1 To n
' Введення елементів масиву
  a(i, j) = Int((9 * Rnd) + 1)
'Виведення в елемент TextBox
  txtA.Value = txtA.Value & CStr(a(i, j)) & "  "
 Next j
txtA.Value = txtA.Value & vbCrLf
Next i
' Міємо місцями перший та останній рядок
For j = 1 To n
 r = a(1, j)
 a(1, j) = a(m, j)
 a(m, j) = r
Next
' Виведення елементів перетвореного масиву
For i = 1 To m
 For j = 1 To n
  txtRez.Value = txtRez.Value & CStr(a(i, j)) & "  "
 Next j
 txtRez.Value = txtRez.Value & vbCrLf
 lblText.Caption = "Міємо місцями перший та останній рядок"
Next i
End Sub

Private Sub UserForm_Click()

End Sub
```

Після запуску програми на екран буде виведена форма:



Вводимо кількість рядків та кількість стовпчиків. Натискаємо кнопку Розрахунок.



Приклад 17. Розробити форму «Анкета клієнта», де клієнт туристичної фірми вносить необхідні дані, а потім може отримати розрахунок вартості поїздки, можливу знижку і отримати сформовану анкету.

Форма має вигляд (рис. Ж1):

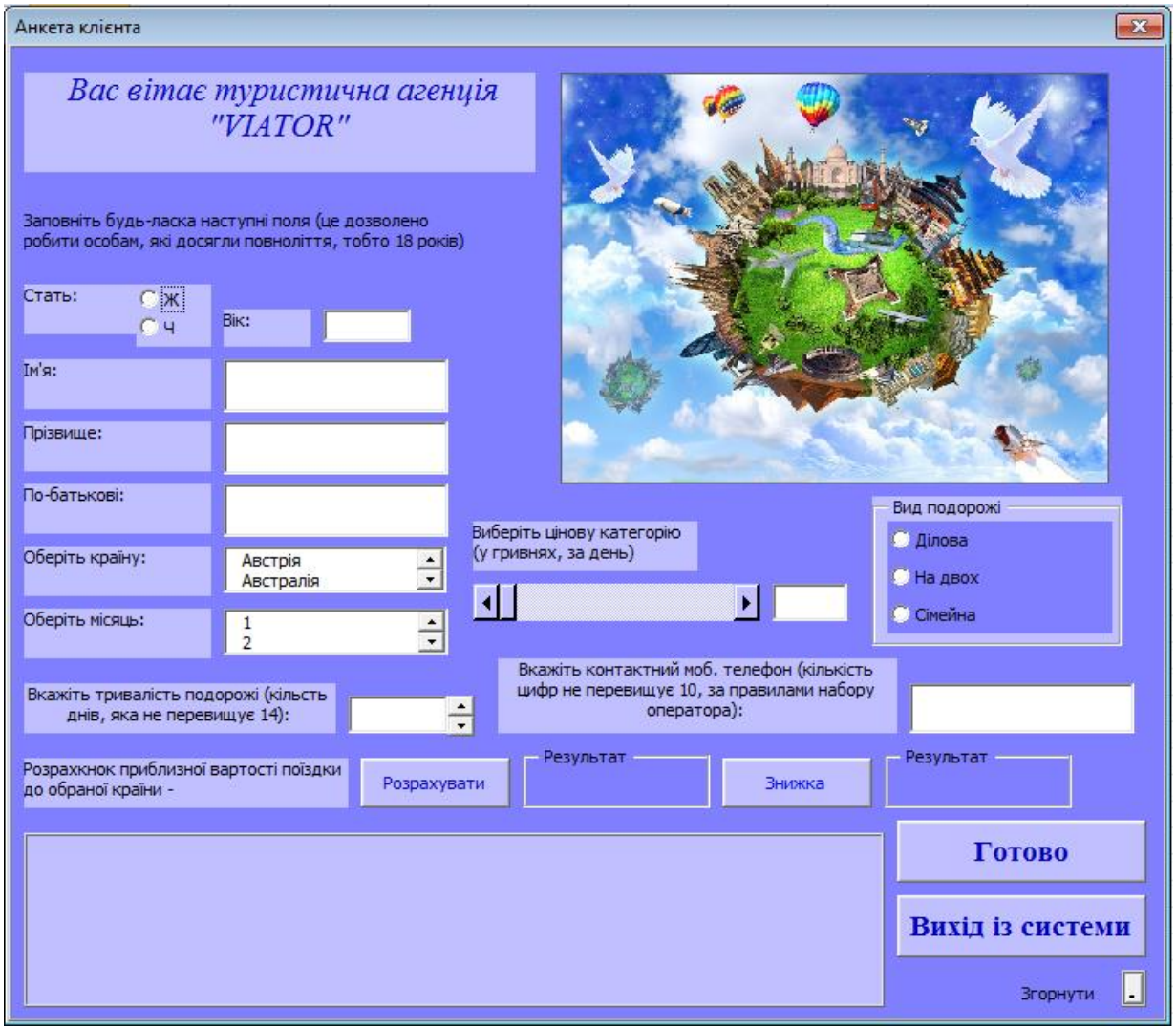


Рис. Ж1. Загальний вигляд форми

Вікно властивостей форми наведено на рис. Ж2.

Користувач (клієнт фірми) заповнює наступні дані:

1. За допомогою перемикача встановлює стать (Ж/Ч).

2. В перше текстове поле вносить вік.

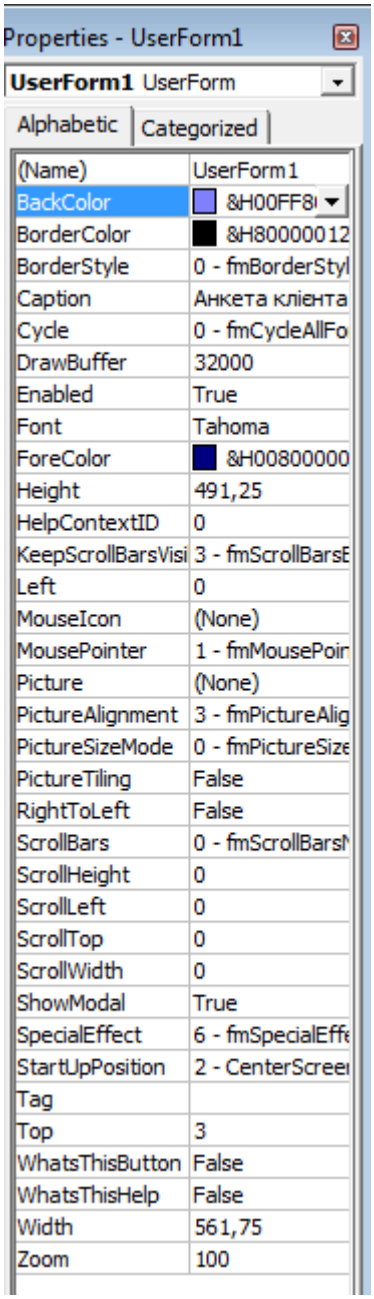
3. Заповнює текстові поля: Прізвище, Ім'я , По батькові.

4. За допомогою смуги прокрутки обирає країну із запропонованих.

5. За допомогою смуги прокрутки обирає місяць поїздки.

6. За допомогою смуги прокрутки обирає тривалість поїздки (не перевищує 14 днів).

7. За допомогою смуги прокрутки обирає цінову категорію, що відображається у текстовому полі.



8. За допомогою перемикача встановлює вид подорожі.

9. Вказує в текстовому полі контактний телефон.

10. Натискання на кнопку **Розрахувати** дозволяє виконати розрахунок вартості подорожі (результат розрахунку буде виведено в поле **Результат**).

11. Натискання на кнопку **Знижка** дозволяє отримати знижку до розрахункової вартості подорожі(результат розрахунку буде виведено в поле **Результат**).

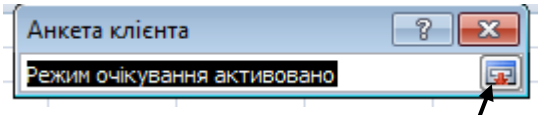
12. Натискання на кнопку **Готово** дозволяє отримати у відповідному полі сформовану Анкету клієнта.

13. Натискання на кнопку **Вихід із системи** прибирає форму з екрана.

14. Натискання на кнопку **Згорнути** дозволяє згорнути вікно

Рис. Ж2. Вікно властивостей форми

Згорнуте вікно виглядає наступним чином:



Клацання мишею по кнопці призводить до розгортання вікна.

Також на формі передбачено поле з рисунком (логотип туристичної фірми).

Після внесення даних сформовано форма має вигляд (рис. Ж3)

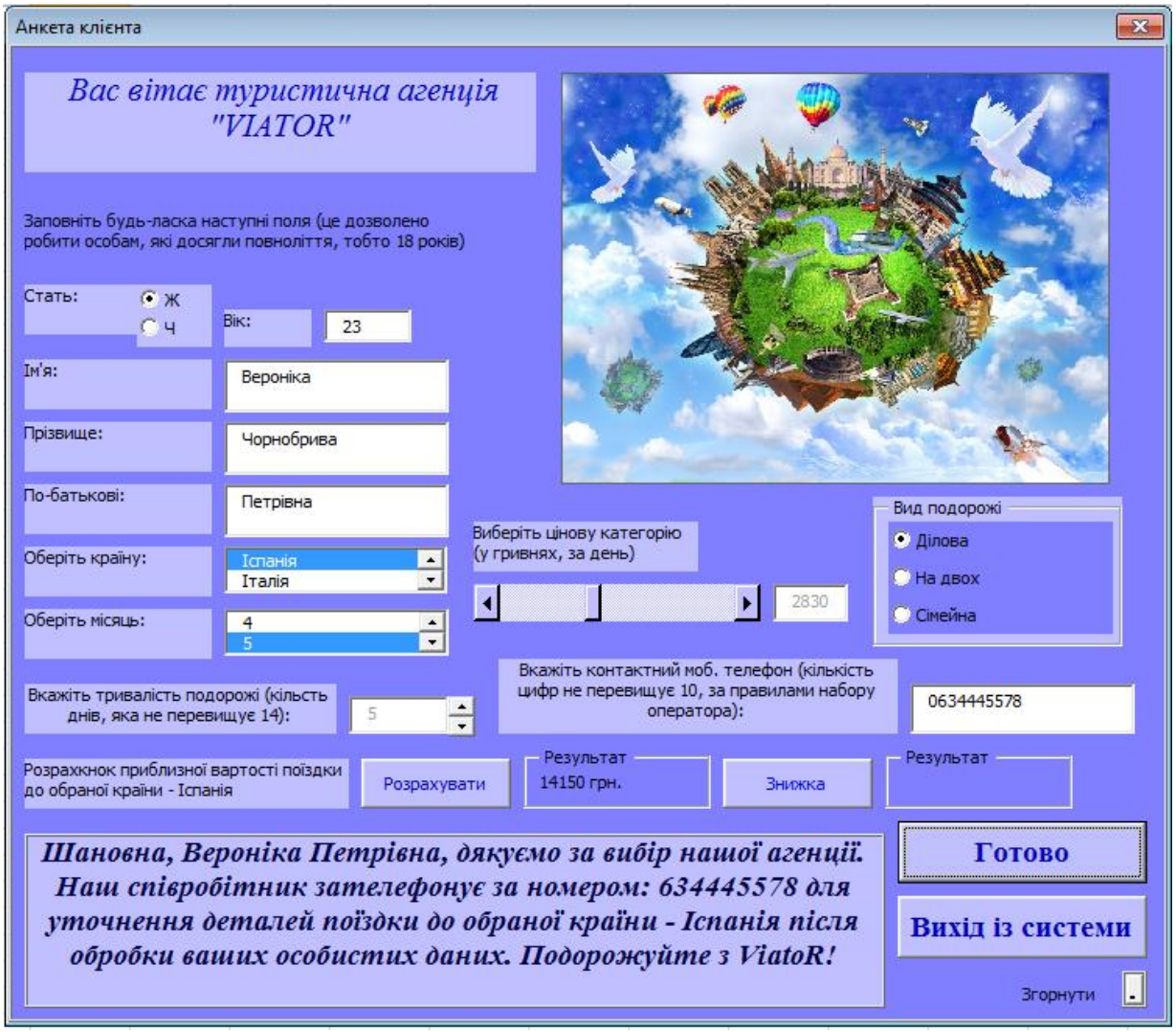


Рис. Ж3. Вигляд форми з внесеними даними і сформованим повідомленням.

Програмний код наведено нижче:

CommandButton1 | Click

```
Private Sub CommandButton1_Click()
'Плануємо подорож
Dim im As Variant, fm As Variant, pb As Variant
Dim nom As Variant, kra As Variant
im = TextBox2.Value
pb = TextBox4.Value
nom = TextBox6.Value
kra = ListBox2.Value
If OptionButton1.Value = True Then
Label18.Caption = "Шановний, " + Format(im) + " " + Format(pb) + ", дяк
End If
If OptionButton2.Value = True Then
Label18.Caption = "Шановна, " + Format(im) + " " + Format(pb) + ", дяку
End If
End Sub
Private Sub CommandButton2_Click()
UserForm1.Hide
End Sub
Private Sub CommandButton3_Click()
Dim dni As Variant, vart As Variant
Dim rez As Variant
'Визначаємо дані у  обраних користувачем рядках
dni = TextBox5.Value
vart = TextBox7.Value
'Підраховуємо вартість поїздки
rez = dni * vart
Label16.Caption = Format(rez) + " грн."
End Sub
Private Sub CommandButton4_Click()
Dim dniz As Variant, vartz As Variant
Dim rezz As Variant, zn As Variant
'Визначаємо дані у  обраних користувачем рядках
dniz = TextBox5.Value
vartz = TextBox7.Value
'Підраховуємо загальну вартість поїздки
rezz = dniz * vartz
'Підраховуємо вартість поїздки з урахуванням знижки
'Знижка на ділову подорож складає 3%
'Розраховуємо суму знижки
If OptionButton3.Value = True Then zn = dniz * vartz * 0.03
'Знижка на ділову подорож складає 5%
'Розраховуємо суму знижки
```

```
If OptionButton5.Value = True Then zn = dniz * vartz * 0.07
rezz = rezz - zn
Label17.Caption = Format(rezz) + " грн."
End Sub
Private Sub ListBox2_Click()
Dim kr As Variant
kr = Label15.Caption
With Label15
.Caption = Format(kr) + ListBox2.Value
End With
End Sub
Private Sub ScrollBar1_Change()
TextBox7.Value = ScrollBar1.Value
End Sub
```

```
Private Sub SpinButton1_Change()
 TextBox5.Value = SpinButton1.Value
End Sub
Private Sub SpinButton2_Change()
 TextBox8.Value = SpinButton2.Value
End Sub
Private Sub TextBox1_AfterUpdate()
If TextBox1.Value < 18 Then MsgBox "Вибачте, але ви не можете заповнювати дану анкету!!!"
If TextBox1.Value < 18 Then UserForm.Hide
End Sub
Private Sub UserForm_Activate()
With ListBox3
.List = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)
End With
With ListBox2
.RowSource = "Лист1!B50:B83"
End With
End Sub
```

Автори дякують студентці ХТФ гр. ХЕ-01 Чалій Інні за якісне програмування цієї задачі.

Додаток К. Індивідуальні завдання лабораторної роботи №3

Тема: «Створення користувацького інтерфейсу. Форми користувача».

Набір індивідуальних завдань 1

Використовуючи MS Excel створити на мові VBA форму користувача. На формі передбачити поля введення початкових даних (елемент **TextBox**), їх текстове позначення (елемент **Label**), а також дві командні кнопки (елемент **CommandButton**): для запуску розробленої програми обробки даних згідно індивідуального завдання і приховування форми з екрана. Результати обчислень виводити в заздалегідь підготовлені на формі поля з їх текстовим позначенням. Назва форми повинна відповідати умові завдання.

1. Створити форму, яка обчислює тригонометричні функції sin, cos і tan введеного кута α (рад).
2. Створити форму, яка обчислює суму і різницю двох введених чисел a і b.
3. Створити форму, яка обчислює добуток(a*b) і результат ділення (a/b) двох введених чисел a і b.
4. Створити форму, яка обчислює середнє арифметичне ($\frac{a+b}{2}$) і середнє геометричне ($\sqrt{a \cdot b}$) двох введених чисел a і b.
5. Створити форму, яка обчислює середнє арифметичне ($\frac{a+b}{2}$) і середнє гармонічне ($\frac{a \cdot b}{a+b}$) двох введених чисел a і b.
6. Створити форму, яка обчислює площу трикутника за формулою Герона ($S = \sqrt{p \cdot (p-a) \cdot (p-b) \cdot (p-c)}$, де $p = \frac{a+b+c}{2}$) за значеннями введених його сторін a, b і c.

7. Створити форму, яка обчислює площу трикутника за значеннями введених його сторін a, b і кута між ними α (рад) за формулою $S = \frac{1}{2} \cdot a \cdot b \cdot \sin \alpha$.

8. Створити форму, яка обчислює площу квадрата зі стороною a і прямокутника зі сторонами a і b.

9. Створити форму, яка обчислює середнє арифметичне трьох введених чисел

a, b і c.

10. Створити форму, яка за координатами двох точок $A(x_1,y_1)$ і $B(x_2,y_2)$ обчислює відстань між ними ($l = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$).

11. Створити форму, яка обчислює площу квадрата зі стороною a і площу круга діаметра d.

12. Створити форму, яка для введеного числа n обчислює 2^n, 1/n і n^2.

13. Створити форму, яка обчислює відстані від початку координат до точок $A(x_1,y_1)$ і $B(x_2,y_2)$. ($l = \sqrt{x^2 + y^2}$ - відстань від початку координат до точки).

14. . Створити форму, яка обчислює площу трапеції зі сторонами a, b і висотою h ($S = \frac{a+b}{2} \cdot h$).

15. Створити форму, яка обчислює об'єм кулі радіуса r ($V = \frac{4}{3} \cdot \pi \cdot r^3$) і об'єм конуса радіуса r і висотою h ($V = \frac{1}{3} \cdot \pi \cdot r^2 \cdot h$).

16. Створити форму, яка обчислює об'єм циліндра ($V = \pi \cdot r^2 \cdot h$) і об'єм конуса ($V = \frac{1}{3} \cdot \pi \cdot r^2 \cdot h$), які мають в основі круг радіуса r і однакову висоту h.

17. Створити форму, яка обчислює натуральний і десятковій логарифми введеного числа a.

18. Створити форму, що визначає, який відсоток становить менше з двох введених чисел a і b від більшого.

19. Створити форму, що визначає у скільки разів більше одне (більше число) з двох введених чисел a і b від іншого (менше число).

20. Створити форму, яка обчислює корінь рівняння ($a \cdot x + b = 0$) при введених коефіцієнтах a і b.

21. Створити форму, яка обчислює значення критерію Рейнольдса ($\mathrm{Re} = \frac{W \cdot d}{\upsilon}$) для потоку рідини за введеними значеннями швидкості потоку W, діаметру трубопроводу d, і в'язкості рідини υ (наприклад, W= 0.754; d= 0.032;

υ= $0.402 \cdot 10^{-6}$).

22. Створити форму, яка обчислює коефіцієнт теплопередачі ($\mathrm{K} = \frac{1}{\frac{1}{\alpha_1} \cdot \frac{\delta}{\lambda} \cdot \frac{1}{\alpha_2}}$) за введеними значеннями α_1 , α_2, λ і δ (наприклад, $\alpha_1 = 3817$; $\alpha_2 = 4376$; $\lambda = 45$; $\delta = 1.5 \cdot 10^{-3}$).

23. Створити форму, яка обчислює коефіцієнт тепловіддачі α_2 ($\alpha_2 = \frac{Nu \cdot \lambda}{d}$) за введеними значеннями Nu, λ і d (наприклад, $Nu = 81{,}3$; λ=0,6; d= 0.032).

24. Створити форму, яка визначає, яка з двох точок A(x_1,y_1) чи B(x_2,y_2) знаходиться ближче до початку координат і виводить повідомлення про це в поле виводу.($l = \sqrt{x^2 + y^2}$ - відстань від початку координат до точки).

25. Створити форму, яка визначає, яка з двох точок A(x_1,y_1) чи B(x_2,y_2) розташована на більшій відстані від початку координат і виводить повідомлення про це в поле виводу.($l = \sqrt{x^2 + y^2}$ - відстань від початку координат до точки).

Набір індивідуальних завдань 2

1. Виконати перерахунок швидкості, що задана в км/год в м/с. Вказати біля значень і розмірність величини.
2. Виконати перерахунок часу, що заданий в годинах в секунди. Вказати біля значень і розмірність величини.
3. Виконати перерахунок часу, що заданий в секундах в години. Вказати біля значень і розмірність величини.
4. Виконати перерахунок швидкості, що задана в м/с в км/год. Вказати біля значень і розмірність величини.
5. Виконати перерахунок відстані, що задана в м в км. Вказати біля значень і розмірність величини.
6. Виконати перерахунок відстані, що задана в км в м. Вказати біля значень і розмірність величини.
7. Виконати перерахунок відстані, що задана в м в см. Вказати біля значень і розмірність величини.
8. Виконати перерахунок відстані, що задана в міжнародних морських милях в метри(міжнародна морська миля = 1852 м). Вказати біля значень і розмірність величини.
9. Виконати перерахунок відстані, що задана в статутних милях в метри(статутна миля =1609,344 м). Вказати біля значень і розмірність величини.
10. Виконати перерахунок ваги, що задана в тонах в кілограми. Вказати біля значень і розмірність величини.
11. Виконати перерахунок ваги, що задана в центнерах в кілограми. Вказати біля значень і розмірність величини.
12. Виконати перерахунок ваги, що задана в тонах в центнери. Вказати біля значень і розмірність величини.
13. Виконати перерахунок діаметру що заданий в дюймах в см(1дюйм =25,4 мм). Вказати біля значень і розмірність величини.

14. Виконати перерахунок діаметру що заданий в дюймах в м(1дюйм =25,4 мм). Вказати біля значень і розмірність величини.
15. Виконати перерахунок діаметру що заданий в ярдах в м(1 ярд=0,9144 м). Вказати біля значень і розмірність величини.
16. Виконати перерахунок діаметру що заданий в ярдах в см(1 ярд=0,9144 м). Вказати біля значень і розмірність величини.
17. Виконати перерахунок об'єму що заданий в $м^3$в літри. Вказати біля значень і розмірність величини.
18. Виконати перерахунок об'єму що заданий в $м^3$в мілілітри. Вказати біля значень і розмірність величини.
19. Виконати перерахунок об'єму що заданий в галонах в літри. Вказати біля значень і розмірність величини.
20. Виконати перерахунок об'єму що заданий в галонах в $м^3$(1 галон = 4,54609 л). Вказати біля значень і розмірність величини.
21. Виконати перерахунок об'єму що заданий в барелях в літри (1барель нафтовий = 158,988 л). Вказати біля значень і розмірність величини.
22. Виконати перерахунок об'єму що заданий в барелях в $м^3$(1барель нафтовий = 158,988 л). Вказати біля значень і розмірність величини.
23. Виконати перерахунок об'єму що заданий в пінтах в літри. Вказати біля значень і розмірність величини.
24. Виконати перерахунок об'єму що заданий в пінтах в $м^3$(1 пінта рідинна = 0,473179 л). Вказати біля значень і розмірність величини.
25. Виконати перерахунок української гривні в долари США. Вказати біля значень і розмірність величини.
26. Виконати перерахунок української гривні в Євро. Вказати біля значень і розмірність величини.

Набір індивідуальних завдань 3

1. За заданими значеннями відстані та часу визначити швидкість руху.
2. За введеною кількістю балів, що набрані студентом в семестрі, визначити його оцінку за наступною шкалою (від 95 до 100 балів – оцінка «Відмінно»; від 85 до 94 балів – оцінка «Дуже добре»; від 75 до 84 балів – оцінка «Добре»; від 74 до 65 балів – оцінка «Задовільно»; від 64 до 60 балів – оцінка «Достатньо»; менше 60 балів оцінка «Незадовільно»).
3. За значенням числа (ціле число) знайти його факторіал.
4. За значенням температури, що задана у градусах Цельсія виконати її перерахунок у градуси Кельвіна (n град. Цельсія =n град. Кельвіна – 273) Вказати біля значень і розмірність величини.
5. За значенням температури, що задана у градусах Цельсія виконати її перерахунок у градуси Фаренгейта (n град. Цельсія =5/9(n град. Фаренгейта – 32)). Вказати біля значень і розмірність величини.
6. За значенням температури, що задана у градусах Цельсія виконати її перерахунок у градуси Реомюра (n град. Цельсія =5/4n град. Реомюра). Вказати біля значень і розмірність величини.
7. За значенням температури, що задана у градусах Фаренгейта виконати її перерахунок у градуси Цельсія (n град. Цельсія =5/9(n град. Фаренгейта – 32)). Вказати біля значень і розмірність величини.
8. Виконати перерахунок кута, що заданий у градусах в радіани (αград=αрад180/π). Вказати біля значень і розмірність величин.
9. Виконати перерахунок кута, що заданий в радіанах у градуси (αград=αрад180/π). Вказати біля значень і розмірність величин.
10. Створити форму, яка обчислює довжину кола радіуса r ($L=2\pi r$) та його площу ($S=\pi r^2$).
11. Створити форму, яка введене число підносить до квадрату та кубу.
12. Створити форму, яка введене число підносить до квадрату та розраховує зворотне значення.

13. Створити форму, яка для позитивного введеного числа шукає натуральний та десятковий логарифми.

14. Створити форму, яка для позитивного введеного числа шукає квадратний та кубічний корені.

15. Створити форму, яка для двох введених чисел **a** та **b** виконує їх порівняння за результатами якого виводить одне з повідомлень: «**a** менше **b**»; «**b** менше **a**»; «**a** дорівнює **b**» в мітку.

16. Створити форму, яка для двох позитивних введених чисел **a** та **b** шукає $\mathbf{a^b}$ та $\mathbf{b^a}$.

17. Створити форму, яка для двох позитивних введених чисел **a** та **b** шукає $\mathbf{e^b}$ (EXP(**b**) та $\mathbf{e^a}$(EXP(**a**)).

18. Створити форму на якій вводиться вік людини, а потім виводиться наступний текст (наприклад, «Вам вже 18 років»).

19. Створити форму, яка для позитивного введеного числа шукає вказану кількість процентів.

20. Створити форму в текстове поле якої вводиться числовий пароль (6 символів)і виконується його перевірка. При правильно введеному паролі виводиться повідомлення «Пароль вірний», а в іншому випадку – « Пароль не вірний. Спробуйте ще».

21. Створити форму, яка надає можливість вибору декількох елементів із списку та відображення вибраних елементів на формі. В список занести всі пори року. Використовуємо елемент управління Список (ListBox).

22. Створити форму, яка надає можливість вибору декількох елементів із списку та відображення вибраних елементів на формі. В список занести п’ять студентів групи. Використовуємо елемент управління Список (ListBox).

23. Створити форму, яка надає можливість вибору декількох елементів із списку та відображення вибраних елементів на формі. В список занести п’ять країн. Використовуємо елемент управління Список (ListBox).

24. За заданими значеннями опорів R_1 та R_2 та видом з’єднання знайти опір ланцюга R ($R = R_1 + R_2$– при послідовному з'єднанні; $R = R_1R_2 / (R_1 + R_2)$ – при паралельному з'єднанні). Також вказати при якому виді з’єднання виконаний розрахунок.