УКРАЇНСЬКА ДЕРЖАВНА АКАДЕМІЯ
ЗАЛІЗНИЧНОГО ТРАНСПОРТУ

ФАКУЛЬТЕТ АВТОМАТИКИ, ТЕЛЕМЕХАНІКИ ТА ЗВ'ЯЗКУ

Кафедра «Обчислювальна техніка та системи управління»

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт по Visual Basic for Applications
з дисциплін
«*ОБЧИСЛЮВАЛЬНА ТЕХНІКА ТА ПРОГРАМУВАННЯ*»,
«*ОСНОВИ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ПРОГРАМУВАННЯ*»

Частина 1

Харків 2012

Методичні вказівки розглянуто та рекомендовано до друку на засіданні кафедри „Обчислювальна техніка та системи управління” 29 березня 2010 р., протокол № 8.

Рекомендуються для студентів механічного факультету.

Укладачі:

доц. І.В. Піскачова,
асист. О.В. Казанко

Рецензент

проф. Г.І. Загарій

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт по Visual Basic for Applications
з дисциплін
«*ОБЧИСЛЮВАЛЬНА ТЕХНІКА ТА ПРОГРАМУВАННЯ*»,
«*ОСНОВИ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ПРОГРАМУВАННЯ*»

Частина 1

Відповідальний за випуск Піскачова І.В.

Редактор Еткало О.О.

Підписано до друку 06.09.10 р.

Формат паперу 60х84 1/16 . Папір писальний.

Умовн.-друк.арк. 1,0. Тираж 50. Замовлення №

Видавець та виготовлювач Українська державна академія залізничного транспорту
61050, Харків - 50, майдан Фейєрбаха, 7
Свідоцтво суб’єкта видавничої справи ДК № 2874 від 12.06.2007 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ

УКРАЇНСЬКА ДЕРЖАВНА АКАДЕМІЯ ЗАЛІЗНИЧНОГО ТРАНСПОРТУ

Методичні вказівки до лабораторних робіт по Visual Basic for Applications з дисциплін «Обчислювальна техніка та програмування», «Основи інформаційних технологій та програмування», для студентів механічного факультету частина 1

Харків, 2012

Методичні вказівки розглянуто та рекомендовано до друку на засіданні кафедри "Обчислювальна техніка та системи управління" 29 березня 2010 р., протокол №8.

Укладачі:
доц. І.В. Піскачова,
асист. О.В. Казанко

Рецензент
проф. Г.І. Загарій

ЗМІСТ

ВСТУП

Технологія роботи у середовищі Visual Basic for Applications (VBA) базується на ідеях об’єктно-орієнтованого та візуального програмування. Згідно з [1] об'єктно-орієнтоване програмування (object-oriented programming) - конструювання програм у вигляді ієрархічно впорядкованих класів об'єктів, що описують дані та операції над об'єктами, які можуть взаємодіяти з іншими об'єктами.

Метод об'єктно-орієнтованого програмування випливає з об'єктно-орієнтованого сприйняття світу, що складається з великої кількості об'єктів. Вони є порівняно незалежними, але постійно взаємодіють між собою. Кожний об'єкт має певні властивості та вміє виконувати деякі функції. Об'єктом називається абстракція, що характеризується станом, поведінкою та ідентифікованістю. Ідентифікованість - це така властивість об'єкта, яка відрізняє його від усіх інших об'єктів.

Сукупності схожих об'єктів утворюють клас. Опис окремого об’єкта на основі загального опису класу можна отримати, якщо визначити конкретні значення властивостей. Поведінка визначається тим, як об'єкт функціонує та реагує на зовнішні події.

Ідея об’єктно-орієнтованого програмування полягає в інкапсуляції (об’єднанні) даних і засобів їх опрацювання (методів) у тип, який називається об’єктом. Прикладами об’єктів можуть бути елементи управління: кнопки, списки, текстові поля тощо.

Середовище візуального програмування **Visual Basic (VB)** – це графічна автоматизована оболонка над об’єктно-орієнтованою версією мови Basic. Якщо у мові Basic структурними одиницями є дані та команди, то тут такою структурною одиницею є візуальний об’єкт.

Програмний об'єкт володіє певними властивостями і методами. Властивості - це видимі характеристики об'єкта, а методи - операції перетворення цих даних. Видимими характеристиками називаються дані, які можуть бути доступні поза об'єктом. Властивостями вважаються дані, якими об'єкт маніпулює або які дозволяють контролювати, як об'єкт виглядає або як він себе веде. Наприклад, властивість Value текстового вікна це

текст, який вводиться в це вікно. Властивість шрифту Color управляє тим, як він виглядає, а властивість елемента меню Enabled управляє доступністю вибору цього елемента.

Коли виконується метод, він може змінити значення лише властивості даного об'єкта, але не інших об'єктів.

Програмний об'єкт може бути частиною іншого, більшого програмного об'єкта. Коли один об'єкт є частиною іншого об'єкта, він успадковує всі властивості і методи попереднього, дозволяючи доступ до своїх властивостей і методів.

Мова програмування VBA виникла на основі мови VB. У VB і VBA використовуються ідентичні середовища розробки. Основна відмінність мови програмування Visual Basic і мови Visual Basic for Applications у тому, що на мові VB створюються самостійні додатки, а проекти VBA працюють у середовищі Microsoft Office. Крім того, слід відмітити, що програма VBA не компілюється, а інтерпретується. Це означає, що переведення програми в машинний код здійснюється в процесі її виконання і проекти VBA виконуються тільки з допомогою додатка, який підтримує VBA. При компіляції переведення програми в машинний код відбувається до запуску програми, тому скомпільовані додатки працюють швидше і займають менше пам'яті. Проте, по-перше, вони мають меншу гнучкість і, по-друге, на сьогоднішній день, комп'ютерна техніка настільки прогресивна, що швидкодія і ємність пам'яті не є критичним моментом для офісних додатків.

Visual Basic for Applications - це інструмент розроблення додатків, що дозволяє створити програмні продукти, які можна використовувати, наприклад, для оформлення документів (підготовки текстів) або аналізу даних таблиць (електронних таблиць). Visual Basic for Applications також застосовується для розширення функціональних можливостей додатків, у яких він використовується. Наприклад, можна додати власне меню або функцію до вбудованих засобів Excel. Спільно використовуючи VBA з програмами Microsoft Office, можна вирішувати дуже складні задачі.

Даний курс лабораторних робіт присвячений вивченню основ програмування в середовищі Visual Basic for Applications. Вирішення різних завдань допоможе студентові надалі використовувати отримані навички в професійній діяльності.

Лабораторна робота 1
ВИКОРИСТАННЯ КОМАНДНИХ КНОПОК І ТЕКСТОВИХ ВІКОН У VISUAL BASIC FOR APPLICATIONS

Мета роботи: ознайомитися із інтерфейсом мови Visual Basic for Applications.

Завдання до лабораторної роботи

У даній лабораторній роботі буде складена програма «Hello!». Вона виглядає на екрані так, як показано на рисунку 1.1. Програма працює в такий спосіб: коли користувач ставить покажчик мишки на кнопку «Hello» і натискає ліву кнопку мишки, у прямокутнику текстового вікна з'являється текст «Hello». Якщо користувач клацає мишкою по кнопці «Clear», то текстове вікно очищається від тексту. Клацання по кнопці «Exit» приводить до завершення роботи програми.

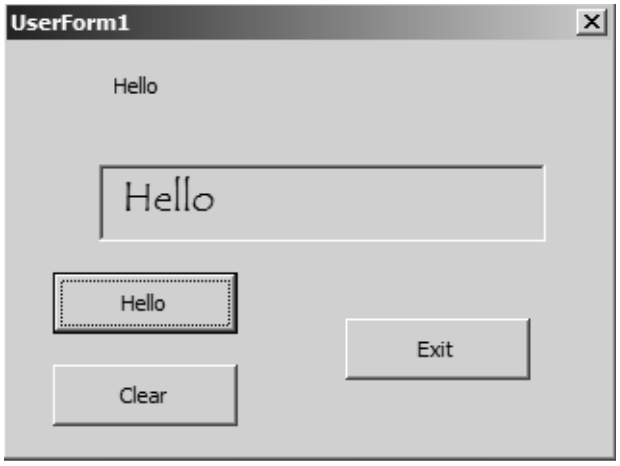


Рисунок 1.1 – Програма «Hello» у роботі

Короткі теоретичні положення

Щоб запустити редактор Visual Basic for Applications з додатка Microsoft Excel потрібно вибрати в меню Сервис команду Макрос→Редактор Visual Basic. З'явиться екран, зображений на рисунках 1.2, 1.3, відкриваються вікно проекту (Project) та вікно властивостей (Properties).

Для того, щоб запустити елемент Форма (UserForm), у меню Вставка (Insert) вибираємо відповідну команду (рисунок 1.2).

Робочий екран VBA складається з таких основних елементів (рисунок 1.3):

1 – елемент управління UserForm (форма). На цьому об'єкті розташовуються елементи управління програмою;

2 – панель інструментів Toolbox. Вона містить елементи управління програмою, які можна розташувати на формі;

3 – таблиця Properties. У ній наведений список усіх властивостей виділеного елемента управління. Властивості програміст може міняти;

4 – ▶ - кнопка для запуску програми.

5 – ■ – кнопка для примусової зупинки програми.

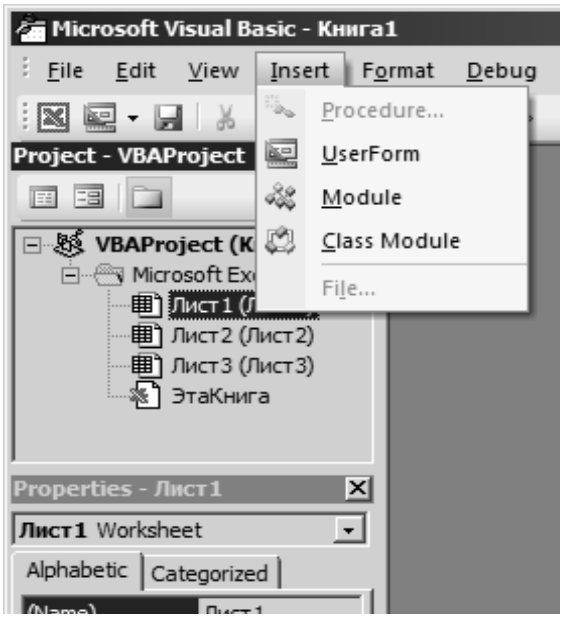


Рисунок 1.2 – Запуск елемента UserForm редактора VBA

Виведення на форму елементів управління

На VBA програми пишуться у два етапи – спочатку програміст складає візуальну частину програми (виводить на форму текстові вікна, кнопки), а потім пише безпосередньо комп'ютерні оператори (команди, які виконує комп'ютер). Перший етап називається візуальним програмуванням, другий – програмуванням у вихідному коді. У даній лабораторній роботі нам знадобляться тільки три управляючих елементи VBA: текстове вікно, командна кнопка й етикетка. Кожний елемент досить складний і за-

стосовується для широкого кола задач.

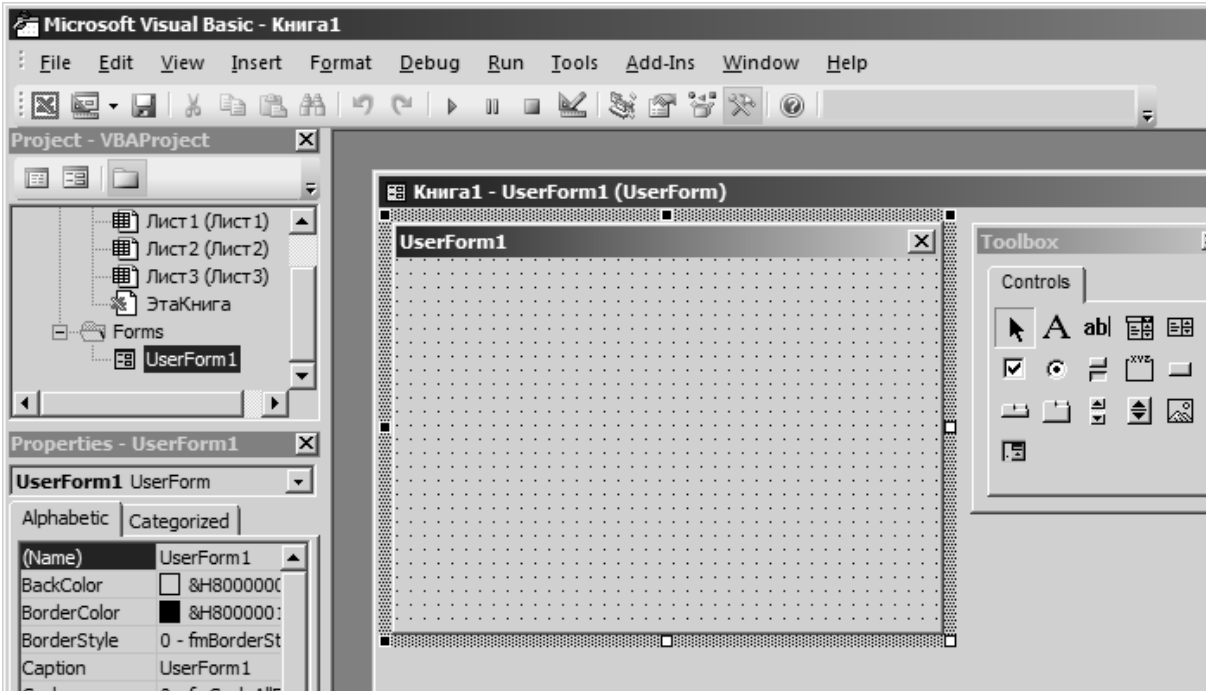


Рисунок 1.3 – Робочий екран редактора VBA

Для того, щоб вивести елемент на форму, потрібно двічі клацнути мишкою по ньому в панелі інструментів Toolbox. Елемент з'явиться в центрі форми. Потім елемент переміщують у ту позицію, у якій за задумом програміста він повинен перебувати. Переміщати елемент можна двома способами.

1-й спосіб переміщення елементів управління. Поставити покажчик мишки на елемент управління, натиснути ліву кнопку мишки і, утримуючи її в натиснутому положенні, перемістити покажчик у потрібне місце, потім відпустити мишку. У процесі переміщення поряд з покажчиком мишки буде рухатися контур переміщуваного елемента.

2-й спосіб переміщення елементів управління. Один раз клацнути мишкою по елементу управління - виділити його (про що будуть свідчити 8 квадратиків, що повинні з'явитися по периметру елемента), після цього натиснути на клавіатурі кнопку Ctrl та, не відпускаючи її, натискати стрілки управління курсором. Кожне натискання на стрілку буде переміщати елемент на один крок точкової сітки, якою розмічена форма.

Елемент управління Label (етикетка) - використовується для

розміщення коментарів поруч із іншими елементами управління, зокрема поруч із текстовими вікнами.

Елемент управління TextBox (текстове вікно) - використовується для введення тексту із клавіатури й виведення тексту на екран.

Елемент управління CommandButton (кнопка) - в основному використовується для ініціювання виконання деяких дій при натисканні на кнопку, наприклад, запуск або зупинка виконання програми, друкування результатів та ін.

Властивості елементів управління

Усі матеріальні й нематеріальні об'єкти мають властивості. Властивості описують ті або інші ознаки об'єкта. Елементи управління – вікно, кнопка, етикетка – теж є об'єктами й мають властивості, які описують їх параметри. У кожного елемента управління VBA багато властивостей. Значення кожної властивості задається автоматично при виведенні елемента на форму. Після цього програміст може самостійно змінити значення будь-якої властивості, якщо в цьому є необхідність. Список відповідних властивостей, їх значення та інструменти їх редагування наведені в таблиці Properties.

Розглянемо тільки ті властивості об’єктів, що необхідні нам при виконанні лабораторної роботи 1.

Значення властивостей командної кнопки

Командна кнопка CommandButton використовується практично у всіх програмах.

Властивість Name – назва елемента. Вона повинна складатися з англійських букв. Першими трьома буквами назви рекомендується ставити cmd.

Властивість Caption - містить текст напису, виведеного на кнопку.

Значення властивостей текстового вікна

Найбільш часто використовуваними властивостями текстового вікна TextBox можна вважати Name і Text.

Властивість Name містить назву вікна. Назва вводиться англійськими буквами, причому першими з них повинні бути txt,

щоб у тексті програми був відразу зрозумілий тип даного елемента управління.

Властивість Text містить текст, який виведений у текстове вікно. Якщо в таблиці Properties у цю властивість увести який-небудь рядок, то вона буде виводитися у вікно при запуску програми. Як правило, на етапі створення інтерфейсу програми дану властивість залишають порожньою.

Значення властивостей етикетки

Елемент Label має 36 властивостей, але для виконання завдань даної лабораторної роботи досить знати й розуміти тільки дві з них – Name і Caption. Перша властивість зберігає назву елемента й повинна починатися з букв lbl, друга зберігає напис на етикетці (Name→lblHello, Caption→Hello).

Програмування у VBA

Комп'ютерна програма – мовна конструкція, що є впорядкованою послідовністю описів і команд, призначена для оброблення даних [1], тобто це список команд, виконуваних комп'ютером. Після того, як програміст створює зовнішній вигляд програми (інтерфейс), він повинен записати команди, які буде виконувати машина. Найпростішою і широко використовуваною при програмуванні у середовищі VBA подією є одинарне клацання мишкою по елементу CommandButton. У створюваній у лабораторній роботі програмі всі події (виведення вітання, очищення текстового вікна, вихід із програми) будуть відбуватися після клацання мишкою по відповідній кнопці.

Для того, щоб запрограмувати ці події, потрібно перейти в редактор коду програми й набрати команди, відповідні до кожної кнопки. Для цього слід двічі клацнути мишкою по кнопці. Після подвійного клацання буде автоматичне завантаження редактора коду й заготовки програми. Наприклад, якщо двічі клацнути по кнопці cmdHello, то форма зникне й на екрані з'явиться текст

```
Private Sub cmdHello_Click()

End Sub
```

Це перший і останній рядки програми, яка повинна виконуватися при клацанні мишкою по даній кнопці. Інші рядки програміст уводить сам (рисунок 1.4).

Для того, щоб повернутися з текстового редактора до форми, слід натиснути кнопки Shift+F7. Якщо перебуваючи на формі натиснути F7, то буде завантажений редактор коду й усі процедури, складені в ньому. Крім цього, переключатися між формою й текстом програми можна послідовно натискаючи комбінацію клавіш Ctrl+Tab.

Порядок виконання лабораторної роботи

Складання програми «Hello» слід виконувати за нижченаведеним алгоритмом.

1 Запустити програму Visual Basic for Applications.

2 Вивести на форму три кнопки, етикетку й текстове вікно, розташувавши їх так, як показано на рисунку 1.4.

3 Присвоїти елементам властивості згідно з таблицею 1.1.

4 Двічі клацнувши по кнопці «Hello», перейти в редактор коду й у виведеній заготовці процедури набрати команду на виведення рядка – «Hello» (рисунок 1.5).

5 Натиснути Shift+F7, повернутися на форму, потім двічі клацнути по кнопці «Clear».

6 У заготовці процедури для кнопки cmdClear набрати команду на очищення текстового вікна (рисунок 1.5).

7 Одним з описаних у пункті 5 способів викликати заготовку процедури для кнопки «Exit» і набрати в ній команду на завершення програми (рисунок 1.5).

8 Клацнувши мишкою по кнопці ▶, запустити програму.

Для того, щоб закрити програму й повернутися до VBA, слід натиснути кнопку «Exit» або клацнути по кнопці зі знаком “×”, яка розташована в правому верхньому куті форми.

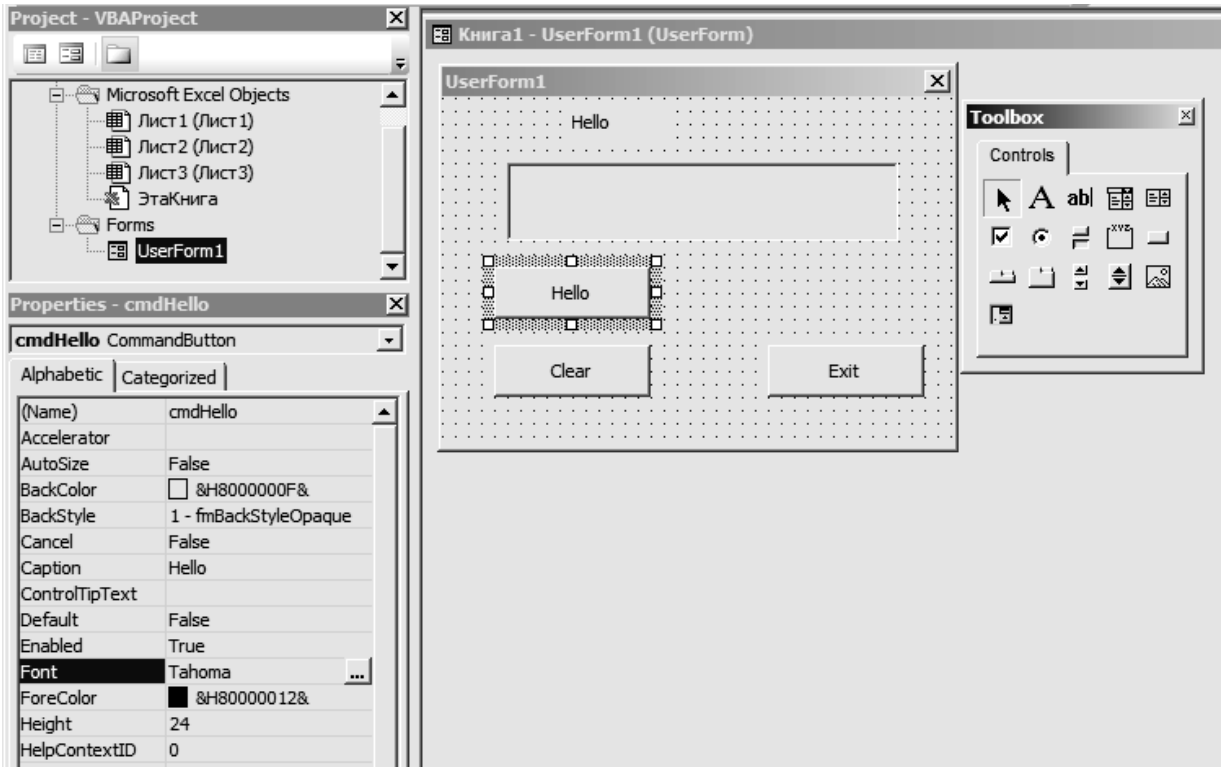


Рисунок 1.4 – Створення інтерфейсу програми

Таблиця 1.1

Елемент	Властивість	Значення
Кнопка «Hello»	Name	cmdHello
	Caption	Hello
Кнопка «Clear»	Name	cmdClear
	Caption	Clear
Кнопка «Exit»	Name	cmdexit
	Caption	Exit
Текстове вікно	Name	txtHello
Етикетка	Name	lblHello
	Caption	Hello

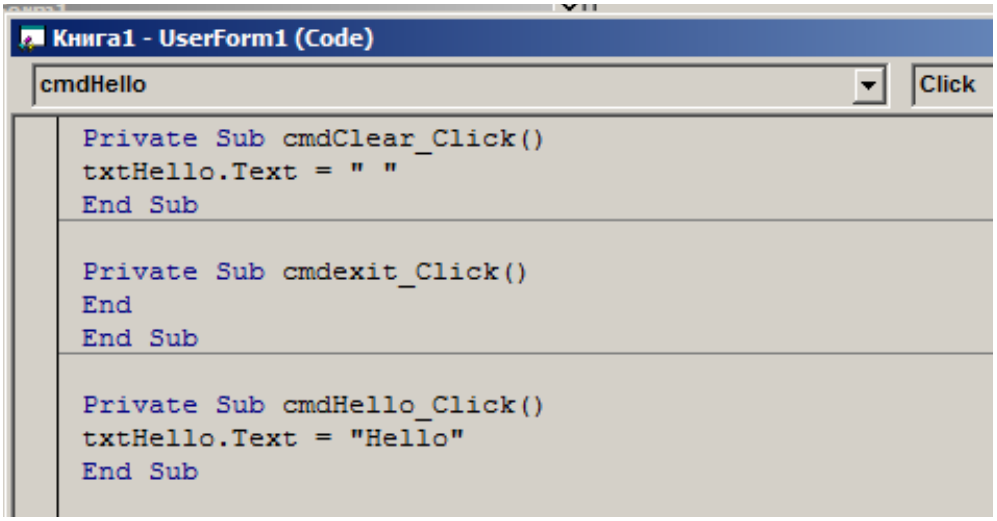


Рисунок 1.5 – Редактор коду із процедурами для кожної кнопки програми

Зміст звіту

У звіті повинно бути:

а) мета завдання;

б) рисунок з елементами управління: форма із кнопками, текстове вікно, етикетки;

в) перелічено основні властивості використаних об'єктів управління;

г) висновки за лабораторною роботою.

Питання до захисту лабораторної роботи 1

1 Що потрібно зробити для того, щоб скласти програму «Hello»?

2 Опишіть призначення основних елементів робочого екрана VBA.

3 Продемонструйте, як можна вивести на форму елемент управління.

4 Продемонструйте, як переміщати елемент управління по формі.

5 Що таке властивість елемента управління?

6 Опишіть основні властивості елемента управління CommandButton.

7 Опишіть основні властивості елемента управління TextBox.

Лабораторна робота 2

ПРАВИЛА ЗАПИСУ МАТЕМАТИЧНИХ ВИРАЗІВ, ВВЕДЕННЯ ВИХІДНИХ ДАНИХ ТА ВИВЕДЕННЯ РЕЗУЛЬТАТІВ У VISUAL BASIC FOR APPLICATIONS

Мета роботи: ознайомитися з правилами запису математичних виразів у VBA.

Завдання до лабораторної роботи

У даній лабораторній роботі необхідно скласти програму, яка розраховує площу рівностороннього трикутника за формулою Герона.

У загальному випадку формула має вигляд

$$S=\sqrt{p\times(p-a)\times(p-b)\times(p-c)},$$

де p=(a+b+c)/2 - півпериметр;

a, b, c – довжини сторін трикутника.

Довжина кожної з них не повинна перевищувати суму довжин двох інших.

У програмі потрібно провести: введення вихідних даних; перевірку, чи існує трикутник з такими сторонами; розрахунок півпериметра трикутника; розрахунок площі трикутника; виведення результату на екран.

Короткі теоретичні положення

Будь-яка комп'ютерна програма розв’язує певну задачу, виконуючи конкретну послідовність дій. Отже, в основі кожної комп'ютерної програми лежить алгоритм. Коли програміст приступає до складання комп'ютерної програми, у першу чергу він будує алгоритм її роботи. Алгоритм - упорядкований скінчений набір чітко визначених правил для розв'язування задач за скінчену кількість кроків [1]. Алгоритми мають такі властивості: дискретність - розчленованість алгоритму на зручні, зрозумілі, доступні частини; масовість (типовість) - можливість використання для інших задач того ж класу; результативність - отримання результату завжди; детермінованість - визначеність, однозначність результату при заданих похідних даних та ін. Для складення

алгоритмів найчастіше використовують блок-схеми. *Блок-схема* – це графічна побудова, на якій за допомогою геометричних фігур показані кроки розв'язання задачі. Для складання блок-схем застосовують фігури, зображені на рисунку 2.1.

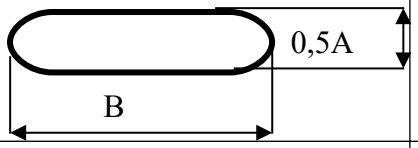
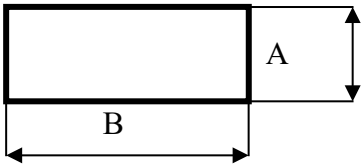
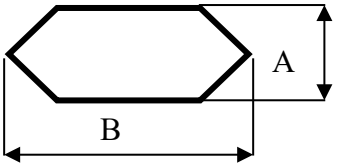
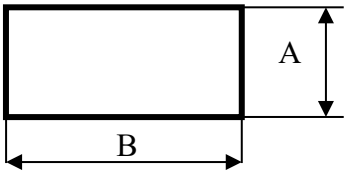
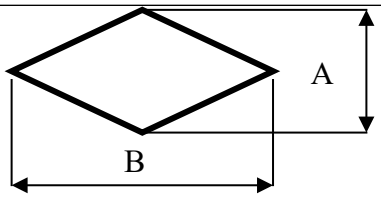
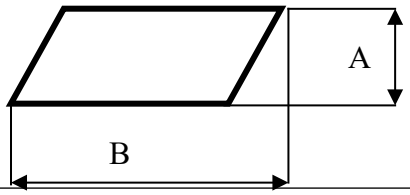
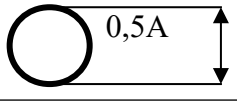
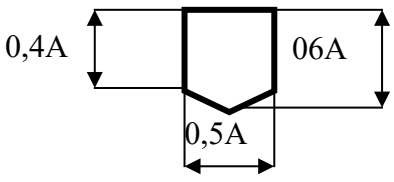
Назва блоку	**Графічне позначення блоку**	**Дії, що виконуються**
Початок-зупинка		Початок, кінець, переривання процесу чи обробки виконання програми
Процес		Виконання операції, у результаті якої змінюється значення, форма чи представлення даних
Модифікація (заголовок циклу)		Виконання операцій, що змінюють команди, групи команд або програму
Визначений процес (підпрограма)		Використання раніше створених або окремо написаних алгоритмів програм
Рішення		Вибір напрямку виконання алгоритму програми в залежності від деяких змінних умов
Введення виведення даних		Перетворення даних у форму, придатну для обробки (уведення) чи відображення результатів обробки (виведення)
З'єднувач сторінковий		Розрив лінії потоку інформації
З'єднувач між-сторінковий		Розрив лінії потоку між сторінками

Рисунок 2.1 – Деякі графічні елементи блок-схеми

У процесі розроблення алгоритму програміст рисує блок-схему, позначаючи кожну дію одним з показаних на рисунку 2.1 знаків. Усередині знака він ставить позначки, що уточнюють дану дію. Усі елементи схеми з'єднуються лініями зі стрілками, які показують напрямок передачі даних і управління.

Існують декілька типів алгоритмів.

1 Лінійний алгоритм.

Алгоритм називається лінійним, якщо він має N кроків і кожен крок виконується послідовно один за одним від початку до кінця.

2 Розгалужений алгоритм.

Алгоритм називається розгалуженим, якщо послідовність виконання кроків алгоритму змінюється в залежності від деяких умов.

Умова - це логічний вираз, який може набувати два значення: "так" - якщо умова правильна або "ні" - якщо умова неправильна. Будь-яка умова складається з трьох частин: ліва частина, знак порівняння, права частина. Наприклад: A<0, X <A, Z = 0.

3 Циклічний алгоритм.

Алгоритм називається циклічним, якщо певна послідовність кроків виконується декілька разів у залежності від заданої величини. Ця величина називається параметром циклу. У будь-якому циклічному алгоритмі повинен бути параметр. Цикл закінчується, коли параметр набуває задане значення.

Правила набору математичних виразів мовою VBA

Математичні вирази в будь-якій мові програмування записуються в один рядок за допомогою спеціальних команд. Спеціальні команди дають вказівки комп'ютеру при розрахунках обчислювати ту або іншу математичну функцію, наприклад синус або квадратний корінь (таблиця 2.1).

Набір математичних виразів слід уводити в нижньому регістрі клавіатури (малими буквами) без уведення пробілів. Це робиться для самоперевірки – після того, як буде набраний рядок із прикладом і натиснута клавіша Enter, у випадку безпомилкового набору, назви функцій автоматично запишуться із великої букви й між арифметичними знаками будуть поставлені пробіли. Якщо ж приклад набраний неправильно, то рядок не зміниться.

Наприклад, вираз

$$Y=\sin(x)+10*x$$

набрано правильно, тому після натискання кнопки Enter і переходу на наступний рядок його буде автоматично перетворено до такого виду:

$$Y = Sin(x) + 10 * x.$$

Функція записана з великої букви, а між знаками “+” і “*” поставлені пробіли. Якщо зміна рядка не відбулася, то це говорить про помилку в його наборі.

Таблиця 2.1 – Правила набору математичних виразів у VBA

Математичний вираз	*Запис у програмі VBA*
$y=x+2$	y=x+2
$y=x^n$	y=x^n
$y=\frac{x}{n}$	y=x/n
$y=\frac{x+2}{x-3}$	y=(x+2)/(x-3)
$y=\sqrt{x}$	y=SQR(x)
$y=\sqrt[n]{x}$	y=x^(1/n)
$y=\sqrt{x+2\times\cos x}$	y=SQR(x+2*COS(x))
$y=\sin x$	y=SIN(x)
$y=\sin^2 x$	y=(SIN(x))^2
$y=\sin x^2$	y=SIN(x^2)

Уведення вихідних даних

Якщо програма призначена для обробки інформації, то цю інформацію необхідно ввести в програму. Як правило, уведення полягає в присвоєнні змінним певних значень.

Найбільш простим способом уведення є присвоєння змінній значення, уведеного користувачем у текстове вікно. Ця дія проводиться за форматом:

Змінна = Val(НазваВікна.Text)

Наприклад, змінній А значення присвоюється в рядку

A = Val(txtA.Text)

Функція Val використовується для переведення числа, уведеного у вікно, з текстового типу в цифровий. Уведення вихідних даних потрібно розміщати на початку програми.

Виведення результату – це остання дія алгоритму.

Результат присвоюється змінній, яка виводиться у відповідне текстове вікно, за форматом:

НазваВікна.Text = змінна

Якщо результат прсвоєний змінній S і повинен бути виведений у вікно з назвою txtS, то ця дія програмується так:

txtS.Text = S

Приклад 2.1

Увести додатні числа А і В. Розрахувати їх середнє арифметичне. Результат вивести в текстове вікно. Скласти блок-схему даного алгоритму.

Блок-схема програми показана на рисунку 2.2.

Опис блок-схеми:

блок 1 – початок розрахунків;

блок 2 – уведення вихідних даних;

блок 3 – розрахунки середнього арифметичного значення;

блок 4 – виведення результату на екран;

блок 5 – завершення розрахунків.

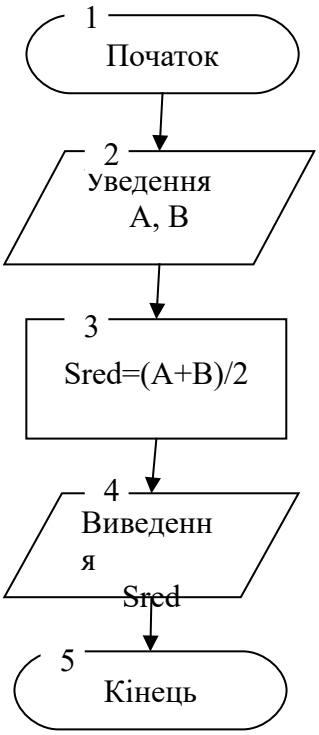


Рисунок 2.2 – Блок-схема для розв’язання прикладу 2.1

Програма. Складемо форму, на якій розташуємо два текстові вікна для введення вихідних даних (txtA і txtB) з етикетками, кнопку для запуску розрахунку (cmdCalculate) і вікно для виведення результату (txtResult) теж з коментарем (рисунки 2.3, 2.4). Присвоюємо елементам властивості згідно з таблицею 2.2.

Таблиця 2.2

Елемент	Властивість	Значення
Кнопка «Розрахувати»	Name	cmdCalculate
	Caption	Розрахувати
Текстове вікно	Name	txtA
Текстове вікно	Name	txtB
Текстове вікно	Name	txtResult
Етикетка	Name	lblA
	Caption	A
Етикетка	Name	lblB
	Caption	B
Етикетка	Name	lblSred
	Caption	Середнє арифметичне

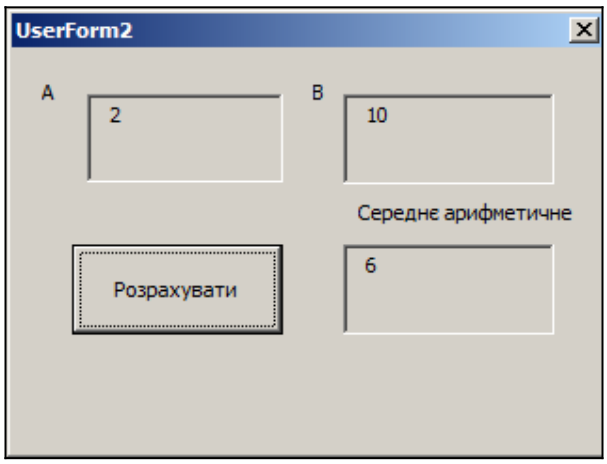


Рисунок 2.3 – Програма розрахунку середнього арифметичного у роботі

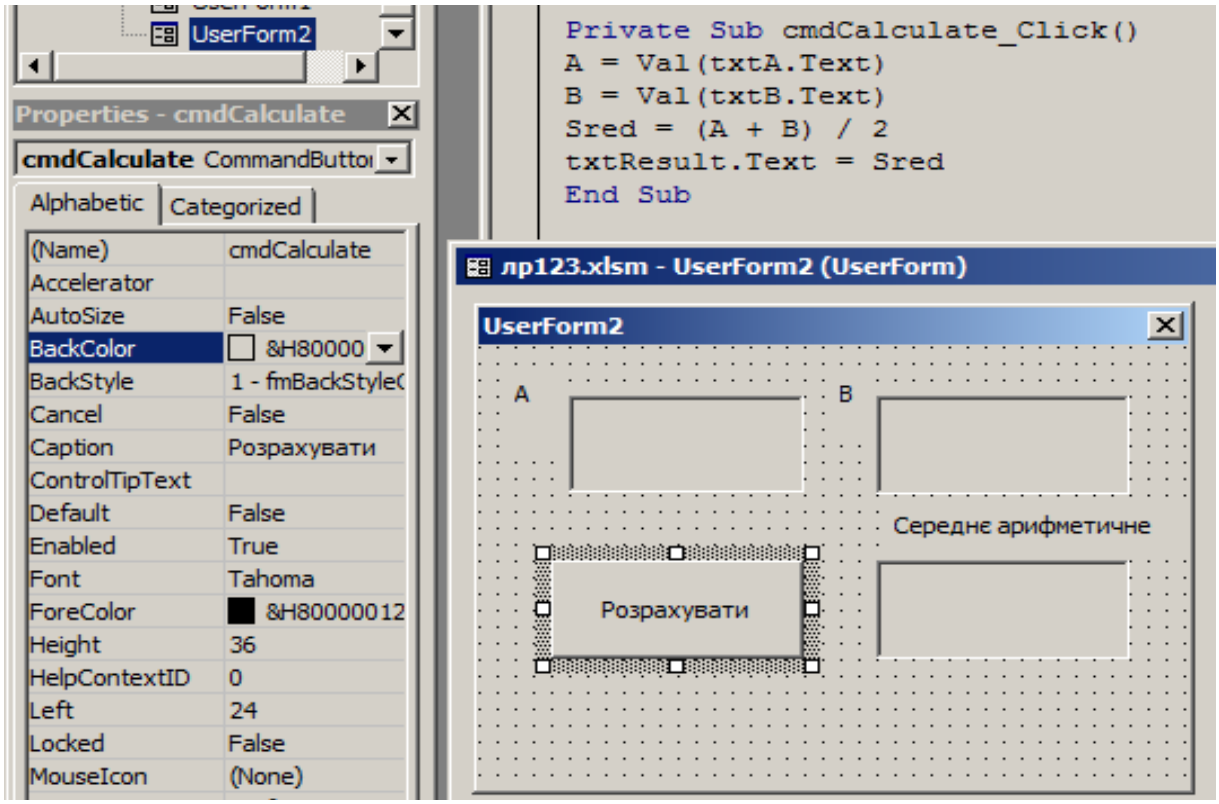


Рисунок 2.4 – створення інтерфейсу програми (приклад 2.1) (форма з елементами управління та програма)

Порядок виконання лабораторної роботи

1 Розрахувати приклад 2.1.

2 Розрахувати площу рівностороннього трикутника за допомогою формули Герона.

Складання програми слід виконувати за нижченаведеним алгоритмом.

1 Запустити програму Visual Basic for Applications.

2 Вивести на форму дві кнопки, чотири етикетки й чотири текстових вікна, розташувавши їх так, як показано на рисунку 2.5.

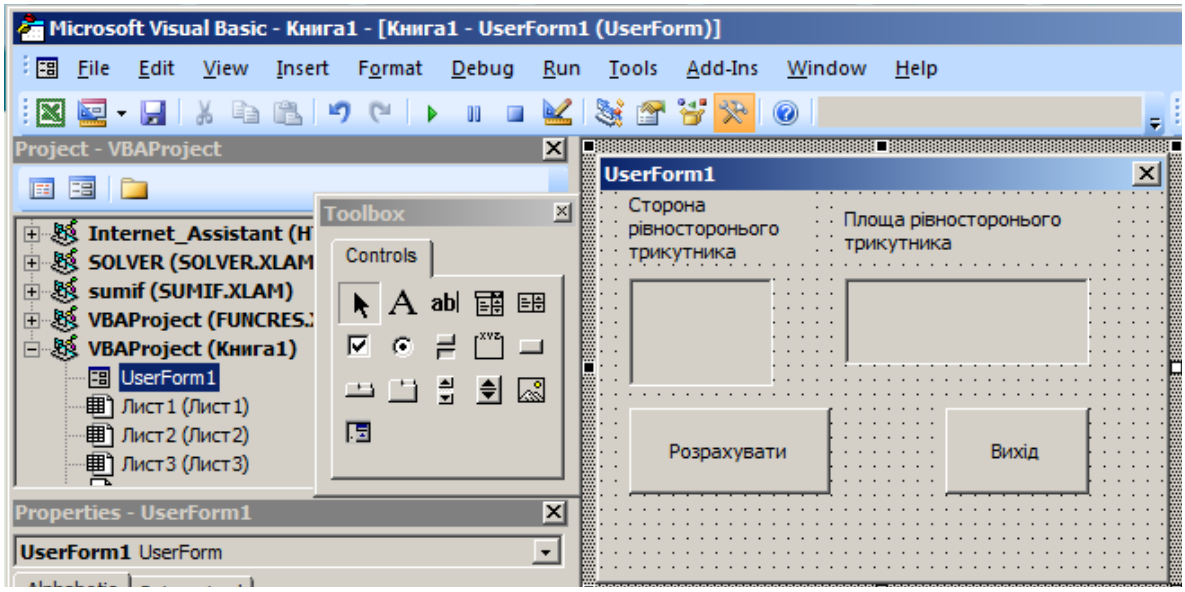


Рисунок 2.5

3 Присвоїти елементам властивості згідно з таблицею 2.3.

Таблиця 2.3

Елемент	Властивість	Значення
Кнопка «Розрахувати»	Name	cmdCalculate
	Caption	Розрахувати
Кнопка «Вихід»	Name	cmdExit
	Caption	Вихід
Текстове вікно	Name	txta
Текстове вікно	Name	txtResult
Етикетка	Name	lbla
	Caption	Сторона рівностороннього трикутника
Етикетка	Name	lblS
	Caption	Площа рівностороннього трикутника

4 Двічі клацнувши по кнопці «Розрахувати», перейти в редактор коду й у виведеній заготовці процедури скласти програму, яка розраховує площу трикутника за формулою Герона.

5 Викликати заготовку процедури для кнопки «Вихід» і набрати в ній команду на завершення програми.

6 Клацнувши мишкою по кнопці ▶, запустити програму.

Зміст звіту

У звіті повинно бути:

а) мета завдання;

б) розв'язання прикладу 2.1. Наведено рисунок з елементами управління: форма із кнопками, текстове вікно, етикетки;

в) розрахунок площі рівностороннього трикутника за допомогою формули Герона при деяких значеннях його сторін. Наведено рисунок з елементами управління: форма із кнопками, текстове вікно, етикетки). У звіті необхідно навести алгоритм розв'язання, а також програму обчислень;

г) обчислення пункту 3 з питань до захисту лабораторної роботи 2

д) висновки за лабораторною роботою.

Питання до захисту лабораторної роботи 2

1 Опишіть порядок уведення даних у програму.

2 Опишіть порядок виведення результату в текстове вікно.

3 Складіть алгоритм та комп'ютерну програму розв'язання таких математичних виразів на VBA:

$$y = (a + 3) \times (a + 5),$$

$$y = \frac{a + 4}{a^2 + 5} \times (a + 5),$$

$$y = \frac{\sqrt{a^6 + b^4 + c^2}}{a^2 + b^2 + c^2}.$$

Лабораторна робота 3
ВИКОРИСТАННЯ ВБУДОВАНИХ МАТЕМАТИЧНИХ ФУНКЦІЙ. ОСОБЛИВОСТІ РОБОТИ З ТРИГОНОМЕТРИЧНИМИ ТА ЛОГАРИФМІЧНИМИ ФУНКЦІЯМИ У VISUAL BASIC FOR APPLICATIONS

Мета роботи: ознайомитися із записом математичних функцій у Visual Basic for Applications.

Завдання на лабораторну роботу

У даній лабораторній роботі необхідно скласти програму, яка обчислює два математичних приклади, наведених у варіантах самостійних завдань до лабораторної роботи 3 при довільних значеннях змінних.

Короткі теоретичні положення

У Visual Basic for Applications вбудовано набір спеціальних функцій, які призначені для розрахунку відповідних математичних виразів. Форми запису цих функцій відрізняються одна від одної лише назвою самого оператора та вимогами до вихідних аргументів. Тому всі функції можна подати у вигляді таблиці. При роботі з нею слід ураховувати , що обов'язковий аргумент «м.в.» – це математичний вираз, яким може бути число, змінна або математична формула, що містить числа, змінні та інші математичні функції (таблиця 3.1).

Якщо аргументом функції є вираз, значення якого не задовольняє вимоги, що висуваються до аргументів даної функції, то програма закінчує свою роботу та видає повідомлення про помилку – "Illegal procedure call or argument" (неправильний аргумент або виклик функції). Наприклад, якщо аргументом квадратного кореня є від'ємне число, відбувається аварійна зупинка. Результат обчислення, проведеного функцією, можна присвоїти змінній, використовувати як частину загальної формули або як аргумент оператора.

Таблиця 3.1 – Деякі математичні функції мови VBA

Функція	Опис функції
ABS(м.в)	Модуль математичного виразу
EXP(м.в)	Основа натурального логарифма в степені заданого натурального логарифма в степені заданого математичного виразу або аргументу ($e^{м.в.}$)
LOG(м.в)	Натуральний логарифм (ln(м.в)). Аргумент повинен бути більше нуля
SQR(м.в)	Квадратний корінь математичного виразу. Значення виразу повинно бути більше або дорівнювати нулю
TAN(м.в)	Тангенс математичного виразу. Значення виразу розглядається як кут у радіанах
ATN(м.в)	Арктангенс математичного виразу. Значення виразу розглядається як кут у радіанах
COS(м.в)	Косинус математичного виразу. Значення виразу розглядається як кут у радіанах
SIN(м.в)	Синус математичного виразу. Значення виразу розглядається як кут у радіанах
SGN(м.в)	Повертає знак математичного виразу: м.в.<0→ SGN(м.в)=-1 м.в.=0→ SGN(м.в)=0 м.в.>0→ SGN(м.в)=1

Приклад 3.1
Фрагмент програми з використанням математичних функцій:

```
Pi = 3.1415926
 y = Cos(2 * Pi)
txt1.Text = y                  ‘Відповідь: 1
z = Tan(3 * PI / 4) - Sin(Pi / 2)
txt2.Text = z                  ‘Відповідь: -2
txt3.Text = Sqr(9)             ‘Відповідь: 3
txt4.Text =Sqr(ABS(-16))       ‘Відповідь: 4
```

За допомогою наведених базових команд можна вводити в

програму доволі складні вирази та функції. Виведення результатів обчислень відбувається у тестові вікна, що мають імена txt1, txt2, txt3, txt4 відповідно.

Особливості роботи з тригонометричними функціями

Аргументи тригонометричних функцій задають кут у радіанах, тому для отримання коректного результату звичні градуси необхідно переводити в радіани. Якщо цього не зробити, то машина не видасть повідомлення про помилку, просто буде обчислювати неправильно. Зокрема, якщо записати y=SIN(90º), то комп'ютер буде вважати , що необхідно розрахувати синус 90 радіан та замість 1 отримаємо 0,8939. Це логічна помилка, а помилки такого роду сам комп'ютер не знаходить.

Співвідношення між градусами та радіанами виражається формулою рад=(π/180)·град, з якої випливає, що для переведення градусів у радіани їх слід помножити на π/180. Для того, щоб здійснити зворотне переведення – із радіанів у градуси – необхідно градуси помножити на число 57,2958.

Наприклад, переведемо 90о в радіани, а потім радіани в градуси.

У радіани: 90×π/180=1,570796.

У градуси: 1,570796×57,2958=90,00001.

Невелика неточність при переведенні в градуси обумовлена системою округлення результатів, що виводяться.

У VBA взагалі чотири тригонометричних функції: TAN(м.в), ATN(м.в), COS(м.в), SIN(м.в).

Інші стандартні тригонометричні функції обчислюються на їх основі за формулами:

Sec(м.в.)= 1 / Cos(м.в.)
Sec(м.в.)= 1 / Cos(м.в.)
Cosec(м.в.)= 1 / Sin(м.в.)
Cotan(м.в.) = 1 / Tan(м.в.)
Arcsin(м.в.) = Atn(м.в. / Sqr(-м.в. * м.в. + 1))
Arccotan(м.в.) = 1.570796 – Atn(м.в.)
Arccos(м.в.) = Atn(-м.в. / Sqr(-м.в. * м.в. + 1)) + 2 * Atn(1)

Приклад 3.2

Скласти програму розрахунку арксинуса кута X.

Наведемо фрагмент програми:

```
X = Val(txtArgument.Text) ‘Вводимо градуси
k = 3.1415926 / 180
X = k * X                          ‘Переводимо градуси в радіани
ArcsinX=Atn(X/Sqr(-X/X+1)
txtRes.Text = "ArcsinX = " + Str(ArcsinX)
```

Функція Str переводить значення із числової в символьну форму – текст. Це дозволяє командою

```
txtRes.Text = "ArcsinX = " + Str(ArcsinX)
```

виводити в текстове вікно txtRes текст та числа, як єдиний текстовий рядок.

Особливості робіт із логарифмічними функціями

У практиці поряд з натуральними (ln(м.в.)) доводиться мати справу з десятковими (lg(м.в.)) та логарифмами з довільною основою ($\log_n$(м.в.)). Їх можна виразити через функцію Log(м.в.) (таблиця 3.2).

Таблиця 3.2

Вираз	У програмі VBA
y=ln(x)	y=Log(x)
y=lg(x)	y=Log(x)/Log(10)
$y=\log_n(x)$	y=Log(x)/Log(n)

Приклад 3.3

Скласти програму розрахунку Ln(x), Lg(x) и $\text{Log}_n(x)$.

Наведемо фрагмент програми:

```
  'Введення вихідних даних
x = Val(txtArgument.Text)
n = Val(txtOsnovanie.Text)
```

```
‘Обчислення
        Y1 = Log(x)
        Y2 = Log(x) / Log(10)
        Y3 = Log(x) / Log(n)
 ‘Виведення результатів на екран
txtRes1.Text = "Ln = " + Str( Y1)
txtRes2.Text = "Lg = " + Str( Y2)
 txtRes3.Text = "Log" + Str(n) + " = " + Str( Y3)
```

Порядок виконання лабораторної роботи

1 Розрахувати вирази згідно зі своїм варіантом, визначеним викладачем. Наприклад,

$$A=\frac{3\cos(x-\pi/3)}{2+\sin^4(y)},$$

$$b = 1 + \frac{z^2}{1 + z^5}.$$

2 Запустити програму Visual Basic for Applications.

3 Вивести на форму дві кнопки, п’ять етикеток і п’ять текстових вікон. Зовнішній вигляд форми наведено на рисунку 3.1.

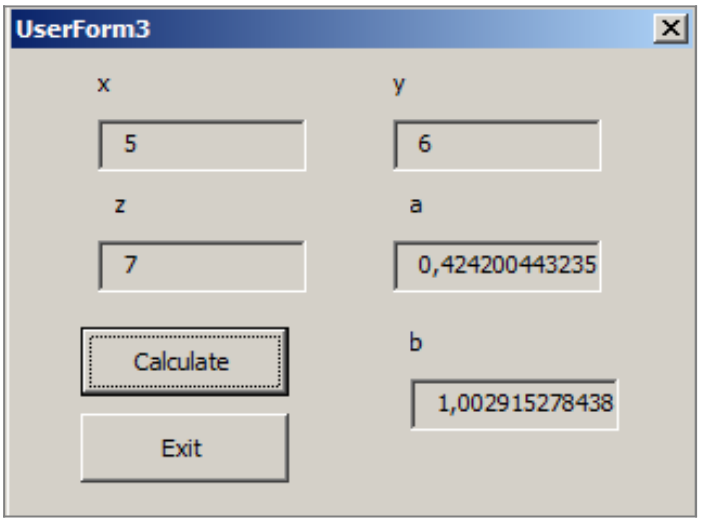


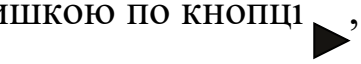
Рисунок 3.1

3 Програма повинна містити введення вхідних даних, розрахунок формул та виведення результатів на екран.

4 Властивості елементів управління наведено в таблиці 3.3.

5 Двічі клацнувши по кнопці «Calculate», перейти в редактор коду й у виведеній заготовці процедури скласти програму, яка розраховує математичні вирази.

6 Викликати заготовку процедури для кнопки «Exit» і набрати в ній команду на завершення програми.

7 Клацнувши мишкою по кнопці  , запустити програму.

Таблиця 3.3 – Властивості елементів

Елемент	Властивість	Значення
Кнопка «Calculate»	Name	cmdCalculate
	Caption	Calculate
Кнопка «Exit»	Name	cmdexit
	Caption	Exit
Текстове вікно	Name	txta
Текстове вікно	Name	txtb
Текстове вікно	Name	txtx
Текстове вікно	Name	txty
Текстове вікно	Name	txtz
Етикетка	Name	lbla
	Caption	a
Етикетка	Name	lblb
	Caption	b
Етикетка	Name	lblx
	Caption	x
Етикетка	Name	lbly
	Caption	y
Етикетка	Name	lblz
	Caption	z

Зміст звіту

У звіті повинно бути:

а) мета завдання;

б) розв’язання прикладів 3.1 - 3.3;

в) обчислення виразів згідно з обраним варіантом при деяких значеннях змінних;

б) рисунки з елементами управління: форма із кнопками, текстові вікна, етикетки;

в) результати обчислень;

г) висновки за лабораторною роботою.

Варіанти самостійних завдань до лабораторної роботи 3

Обчислити вирази при довільних значеннях змінних

Варіант	Функція
1	$y = \frac{\left(1-\sin^2 x\right)^2}{\sin^2 x^3 + 5} + 5\sin^2 x^2$
2	$p = \frac{\sin^6 z + \cos^6 z}{\sin^4 z - \cos^2 z \sin^2 z + \cos^4 z}$
3	$y = \frac{\left(1-\cos 2x\right)^2}{2\sin^2 x + 1} + 2\cos^2 x$
4	$y = \frac{s^2 + \sin 2s}{\sin^2 s + 10}$
5	$y = \cos b - \frac{\left(1-\cos^2 b\right)^2}{5} + 2\sin b^2$
6	$z = \frac{\left(1-\cos 2a\right)^2}{2\sin^2 a + 1} + tga$
7	$c = \frac{\left(3+\sin^2 a\right)^2}{\sin^2 b^3 + 5\sin b} + 25\sin^2 a^2$
8	$y = \frac{\left(\sin^2 x + \cos^2 x\right)^2}{\sin^2 x^2 + \sin x} + \sin^2 x$
9	$y = \frac{\left(10-\sin^2 x\right)^2}{3+\sin^2 x} + x^2$
10	$y = \frac{x^2}{\sqrt{\sin^2 x^3 + 5}}$
11	$y = \frac{\left(1+tg^2 x\right)^2}{tg^2 x^3 + 2} + 5tg^2 x^2$
12	$p = \sin\left(\sqrt[3]{\frac{x+y}{\frac{z^3 + \cos x + tgy}{100}}}\right)$
13	$y = \frac{\left(1-\sin^2 x\right)^2}{\sin^2 x^3 + 5} + 5\sin^2 x^2$
14	$p = \cos\left(\sqrt[3]{\frac{x^2 + y^2}{\cos x + tgy + 6}}\right)$
15	$q = \frac{tgx^2 + tg^2 x}{arctg^y 2 + arctg 2^y}$

Варіант	Функція
16	$y = \frac{\left(1 - \sin^2 x\right)^2}{\sin^2 x^3 + 5} + 5\sin^2 x^2$
17	$z = \frac{\left(10 + \sqrt{\sin^2 x}\right)^2}{\sin^2 x^3 + 7\cos y^2}$
18	$d = \frac{tga^2 + tg^2 a^2}{arctg^2 b + arctg^2 b}$
19	$s = tg\left(\sqrt[5]{\frac{z + x + y}{\sin z^5 + \cos x + tgy}}\right)$
20	$q = \frac{\sin x^2 + \cos^2 x}{arctg^x 2 + arctg 2^x},$

Питання до захисту лабораторної роботи 3

1 Для чого і як використовуються функції Abs(x), Exp(x), Sqr(x), Log(x)?

2 Для чого і як використовуються функції Tan(x), Atn(x), Cos(x), Sin(x)?

3 Наведіть формулу переведення градусів у радіани.

4 Наведіть формулу переведення радіанів у градуси.

5 Як у програмі записати обчислення lg(x) і $\log_n(x)$?

Лабораторна робота 4
ПРОГРАМУВАННЯ АЛГОРИТМІВ РОЗГАЛУЖЕНОЇ СТРУКТУРИ У VISUAL BASIC FOR APPLICATIONS

Мета роботи: ознайомитися з перевіркою умов у VBA, оператором If...Then... Else.

Завдання на лабораторну роботу

У даній лабораторній роботі необхідно скласти програму, яка обчислює завдання наведених у варіантах самостійних завдань до лабораторної роботи 4 при довільних значеннях змінних.

Програма повинна передбачати введення значень, аналіз умов, розрахунок відповідного прикладу та виведення результату на екран.

Короткі теоретичні положення

Оператори умовного переходу - одні з найважливіших і часто використовуваних елементів в мовах програмування. Загальний принцип їх роботи простий: перевіряється відповідність якимсь умовам (істинність або хибність яких-небудь виразів) і, залежно від цього, виконання програми направляється по одній або іншій гілках. У VBA передбачено два оператори умовного переходу: If... Then... Else і Select Case.

Оператор If... Then... Else – найбільш популярний у програмістів. Повний його синтаксис виглядає так:

```
If <Умова> Then
<Дія_1>
[Else
<Дія_2>]
End If
```

Умова - вираз, що перевіряється на істинність. Якщо він істинний, то виконується Дія_1, якщо хибний – Дія_2.

Умовою у всіх форматах If Then є вираз зі знаком порівняння. Він може мати тільки два значення: True (істина) і False (хибність). Якщо умова виконується, то його значенням є True , якщо не виконується – False (хибність).

В умовах можна використовувати такі знаки порівняння:

> - більше;

< - менше;

>= - більше або дорівнює;

<= - менше або дорівнює;

< > - не дорівнює.

Наприклад, A > B або D <= 3. Справа і зліва від знака порівняння можуть знаходитися не тільки цифри і змінні, але й цілі математичні вирази.

Зокрема (Sqr(A + 5) + A ^ 2) >= Abs(SIN(2 * Pi))

Умова може задаватися досить складним чином і складатися з декількох блоків. Для уведення таких виразів використовуються спеціальні оператори - And, Or, Not.

Формат 1 - якщо умова виконується, то машина спочатку виконує Дію_1, потім Дію_2. Якщо умова не виконується, то машина виконує тільки Дію_2. Блок-схема зображена на рисунку 4.1.

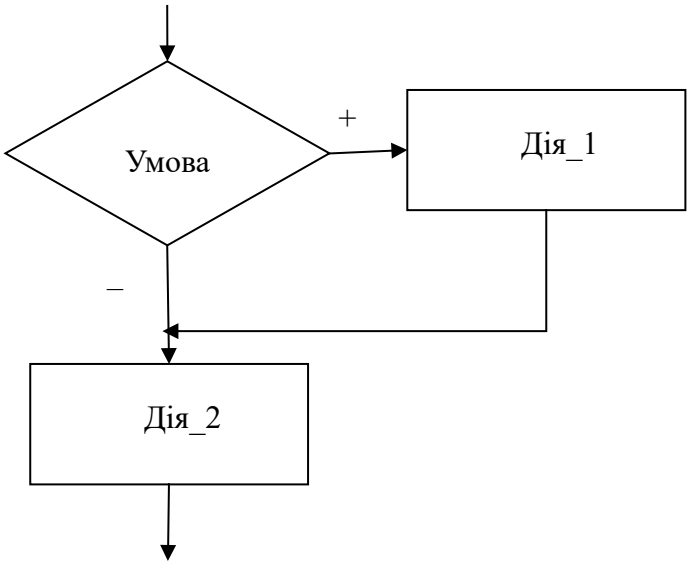


Рисунок 4.1 – Блок-схема формату 1 оператора If Then

Розглянемо коди найпростіших програм.

Приклад 4.1

Ввести число X, якщо воно менше нуля, то помножити на (-1) та вивести на екран. Алгоритм розв'язання у вигляді блок-схеми має вигляд, наведений на рисунку 4.2.

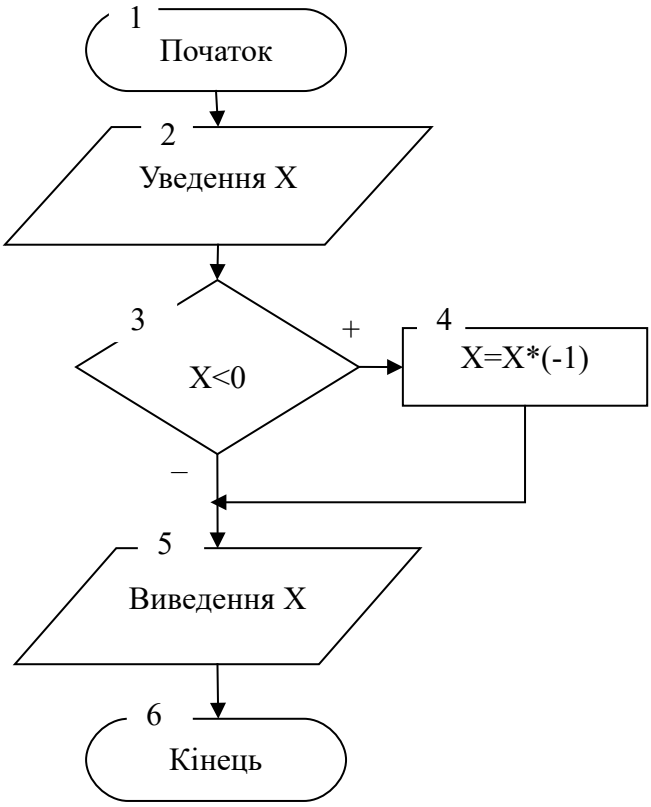


Рисунок 4.2 – Блок-схема до прикладу 4.1

Опис блок-схеми:

блок 1 - початок розрахунку;

блок 2 - уведення X;

блок 3 - перевірка умови;

блок 4 - множення X на (-1) у випадку, якщо умова виконується;

Блок 5 - виведення X на екран;

Блок 6 - завершення розрахунку.

Запишемо алгоритм, зображений на блок-схемі, у вигляді комп'ютерної програми.

```
X = Val(txtX.Text)              'Уводимо X
If X < 0 Then X = X*(-1)        'Якщо X<0, то множимо його на (-1)
txtResult.Text = X              'Виводимо X на екран
```

Робота програми

Після уведення X оператор If … Then аналізує його значення. Якщо X виявиться від’ємним, то програма виконує дію

X=X*(-1), потім виводимо txtResult.Text = X. Якщо X=0 або X>0, то буде виконана тільки Дія_2 - txtResult.Text = X.

Особливістю формату оператора If ... Then є те, що Дія_2 виконується завжди, незалежно від результату аналізу умови. На цьому заснована дія алгоритмів, що включають послідовне використання оператора If…Then.

Приклад 4.2

Скласти блок-схему й програму розв'язання завдання:

$$y=\begin{cases} x^2+|x+1|, & \text{при } x<0, \\ 10a+\cos x, & \text{при } x=0, \\ (x+2)\cdot(b+x^2), & \text{при } x>0. \end{cases}$$

Зовнішній вигляд форми наведено на рисунку 4.3.

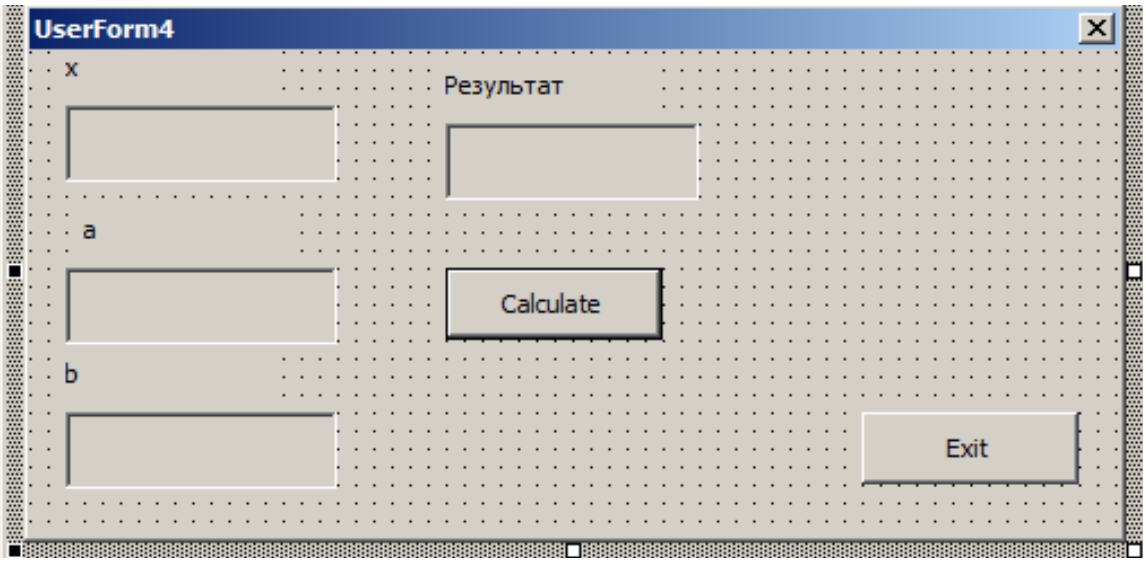


Рисунок 4.3

Програма повинна передбачати введення трьох значень, аналіз умов, розрахунок відповідного прикладу та виведення результату на екран.

Розв'язання

Алгоритм програми записано у вигляді блок-схеми, що наведена на рисунку 4.4.

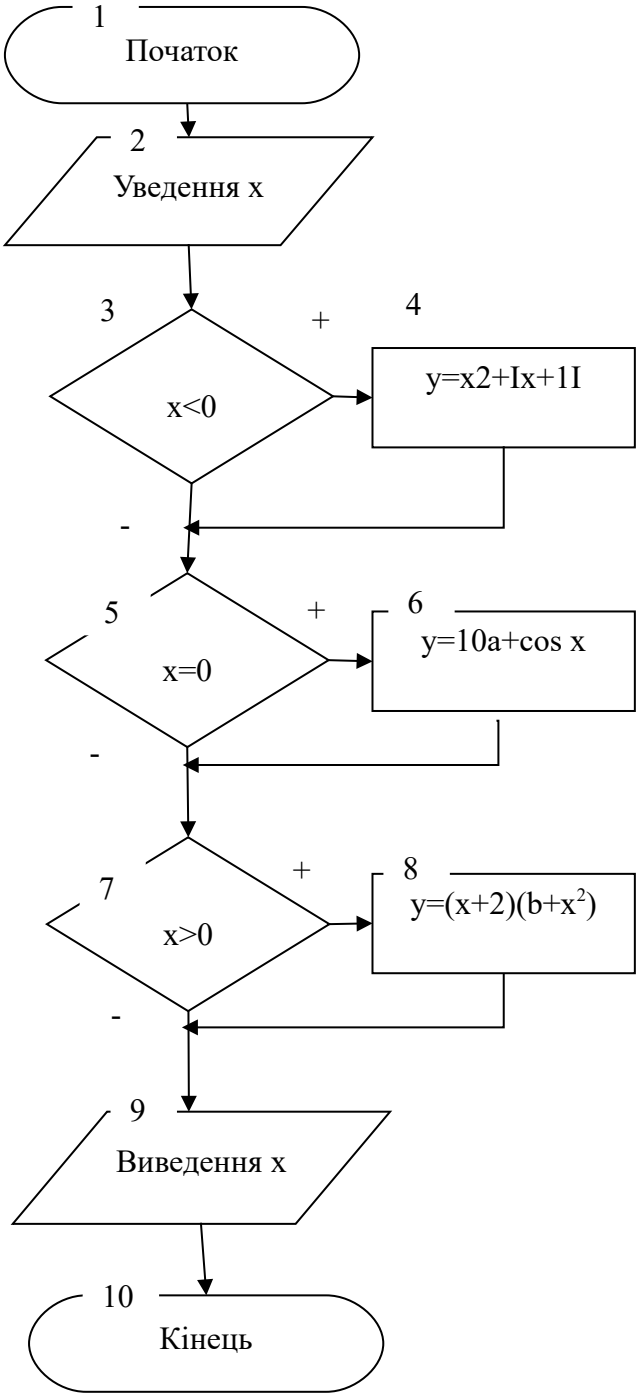


Рисунок 4.4

Опис блок – схеми:

блок 1 – початок роботи програми;

блок 2 – введення x;

блок 3 – перевірка умови x<0.

Якщо умова виконується, то відбувається розрахунок виразу в блоці 4 з подальшим переходом у блок 5.

Якщо умова не виконується, то управління переходить до блоку 5.

У блоці 5 здійснюється перевірка умови x = 0. У випадку виконання умови відбувається розрахунок формули в блоці 6 та перехід до блоку 7.

Якщо умова не виконується, то управління переходить до блоку 7.

У блоці 7 розглядається умова x > 0.

У випадку його виконання розраховуємо вираз у блоці 8 та виводимо результат на екран (блок 9).

Якщо умова у блоці 7 не виконується, то виводиться на екран значення y.

Блок 10 – завершення роботи програми.

Запишемо алгоритм , зображений на блок-схемі, у вигляді комп'ютерної програми:

```
a = Val(txta.Text)
b = Val(txtb.Text)
x = Val(txtx.Text)
    If x < 0 Then y = x ^ 2 + Abs(x + 1)          'Умова №1
    If x = 0 Then y = 10*a + Cos(x)               'Умова №2
    If x > 0 Then y = (x + 2)*(b + x ^ 2)         'Умова №3
txtResult.Text = "y = " + Str(y)
```

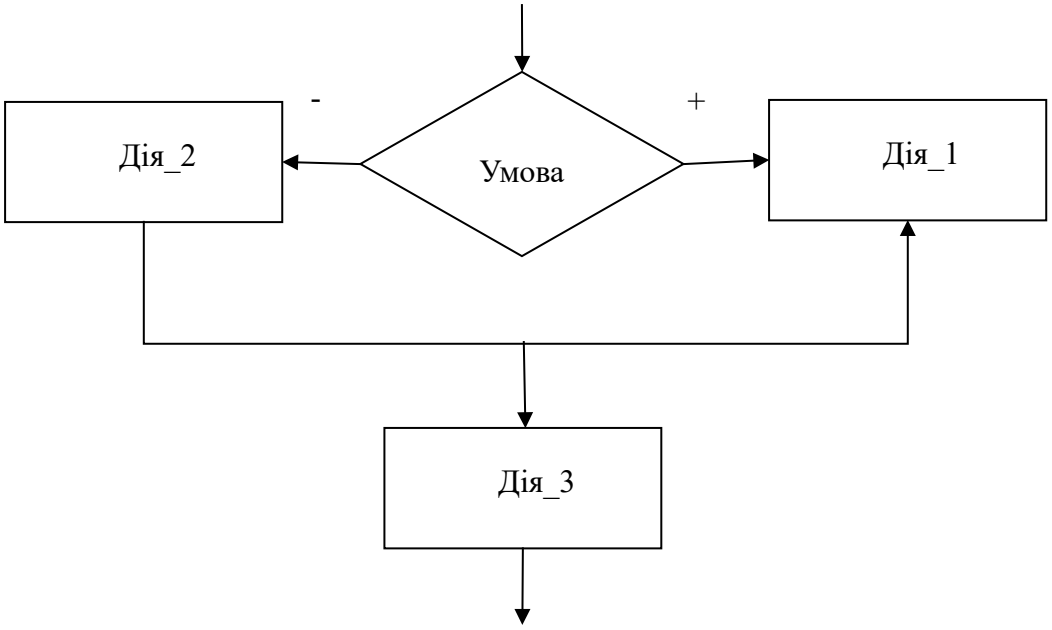
Формат 2 If ... Then ... Else дозволяє виконати або першу або другу дії залежно від умови. Його формат має вигляд:

```
If умова Then Дія_1 Else Дія_2
Дія_3
```

Якщо умова виконується, то машина робить Дію_1, потім Дію_3. Якщо умова не виконується, то машина робить Дію_2, а потім Дію_3.

Блок-схема формату If ... Then ... Else … наведена на рисунку 4.5.

Ри



сунок 4.5 – Блок-схема конструкції If … Then … Else…

Приклад 4.3

Увести X. Якщо його значення парне або дорівнює 0, то додати 2, якщо непарне, то додати 1.

Фрагмент програми:

```
X = Val(txtX.Text)
If X / 2 = X \ 2 Then X = X + 2 Else X = X + 1
txtResult.Text = “X = “ + Str(X)
```

Робота програми

У прикладі 4.3 Дія_1 - це вираз X=X+2, а Дія_2 - вираз X=X+1. Вони не можуть виконуватися послідовно, тому що формат 2 оператора If Then не дозволяє цьому відбутися. Може бути виконана або Дія_1, при виконанні умови, або Дія_2, якщо умова не виконується.

Порядок виконання лабораторної роботи

1 Запустити програму Visual Basic for Applications.

2 Вивести на форму кнопки, етикетки й текстові вікна.

3 Програма повинна містити введення вхідних даних, розрахунок формул та виведення результатів на екран.

4 Властивості елементів управління для прикладу 4.2

наведено в таблиці 4.1.

5 Двічі клацнувши по кнопці «Calculate», перейти в редактор коду й у виведеній заготовці процедури скласти програму, яка розраховує математичні вирази.

Таблиця 4.1 - Властивості елементів для прикладу 4.2

Елемент	Властивість	Значення
Кнопка «Calculate»	Name	cmdCalculate
	Caption	Calculate
Кнопка «Exit»	Name	cmdexit
	Caption	Exit
Текстове вікно	Name	txta
Текстове вікно	Name	txtb
Текстове вікно	Name	txtx
Текстове вікно	Name	txtResult
Етикетка	Name	lbla
	Caption	a
Етикетка	Name	lblb
	Caption	b
Етикетка	Name	lblx
	Caption	x
Етикетка	Name	lblResult
	Caption	Результат

6 Викликати заготовку процедури для кнопки «Exit» і набрати в ній команду на завершення програми.

7 Клацнувши мишкою по кнопці ▶, запустити програму.

Зміст звіту

У звіті повинно бути:

а) мета завдання;

б) розв’язання прикладів 4.1 - 4.3;

в) обчислення завдання, згідно з варіантом;

г) рисунки з елементами управління: форма із кнопками, текстові вікна, етикетки;

д) результати обчислень;

г) висновки за лабораторною роботою.

Варіанти самостійних завдань до лабораторної роботи 4

Варіант	Завдання
1	$y=\begin{cases}10x^2-7/x^2, & \text{якщо } x<0,\\ 3, & \text{якщо } x=0,\\ x^5+7\sqrt{x}, & \text{якщо } x>0.\end{cases}$
2	$y=\begin{cases}5x^2+2x+1, & \text{якщо } x<1,\\ 3/x+\sqrt{x^2+1}, & \text{якщо } x=1,\\ (4+2x)/\sqrt{x^2+1}, & \text{якщо } x>1.\end{cases}$
3	$y=\begin{cases}1.5x-x^3, & \text{якщо } x<5,\\ (x-2)^2+6x, & \text{якщо } x=5,\\ 3x+\sqrt{3+x}, & \text{якщо } x>5.\end{cases}$
4	Дано три відрізки a, b, c. Перевірити, чи існує трикутник з такими сторонами
5	Дано дійсні числа x, y (x ≠y). Менше із цих двох чисел замінити половиною їх суми, а більше - їх подвоєним добутком
6	$z=\begin{cases}x^2+3x, & \text{якщо } x\le 0,\\ \sqrt{x^2+4x}, & \text{якщо } x>0.\end{cases}$
7	$y=\begin{cases}x^3+x^2-5, & \text{якщо } x<1,\\ 3+\sqrt{x^3+1}, & \text{якщо } x=1,\\ 2x/\sqrt{x^3+1}, & \text{якщо } x>1.\end{cases}$
8	З клавіатури вводяться довільні дійсні числа a, b, c. Проаналізувати, чи можна утворити трикутник з такими сторонами і вивести на екран одне із повідомлень: „Трикутник з такими сторонами не можна утворити”, „Це рівнобічний трикутник”, „Це не рівнобічний трикутник”
9	З клавіатури вводяться довільні дійсні числа a, b, c. Проаналізувати, чи можна утворити трикутник з такими сторонами і вивести на екран одне із повідомлень: „Трикутник з такими сторонами не можна утворити”, „Це прямокутний трикутник”, „Це не прямокутний трикутник”
10	З клавіатури вводяться три числа. Якщо вони утворюють послідовність, що збільшується, – вивести на екран повідомлення „Числова послідовність збільшується”. Якщо вони утворюють послідовність, що зменшується, – вивести на екран повідомлення „Числова послідовність зменшується”. У протилежному випадку вивести на екран повідомлення

Варіант	Завдання
	„Невизначена числова послідовність”
11	$y=\begin{cases}15x^4-8x^3, & \text{якщо } x<2,\\ 8+x, & \text{якщо } x=2,\\ x^2+8\sqrt{x}, & \text{якщо } x>2.\end{cases}$
12	Змінній D присвоїти максимальне значення дійсних змінних A,B,C. Вивести на екран усі значення
13	З клавіатури вводяться довільні дійсні числа a, b, c. Проаналізувати, чи можна утворити трикутник з такими сторонами і вивести на екран одне із повідомлень: „Трикутник з такими сторонами не можна утворити”, „Це рівносторонній трикутник”, „Це не рівносторонній трикутник”
14	Написати програму перевірки таблиці множення від 1 до 3. Пропонувати користувачеві вводити результат множення і виводити на екран повідомлення „Правильно” або „Помилка”
15	Змінній D присвоїти мінімальне значення змінних цілого типу A, B, C. Вивести на екран усі значення
16	$z=\begin{cases}y^2+3, & \text{якщо } x\le 0,\\ \sqrt{y^2+5}, & \text{якщо } x>0.\end{cases}$
17	$y=\begin{cases}18\lvert x^3\rvert, & \text{якщо } x<2,\\ 7, & \text{якщо } x=2,\\ 5\sqrt{x}, & \text{якщо } x>2.\end{cases}$
18	$y=\begin{cases}3\lvert x\rvert^3+5x^2, & \text{якщо } x<1,\\ 5, & \text{якщо } x=1,\\ 2/\sqrt{x^3+1}, & \text{якщо } x>1.\end{cases}$
19	$y=4\begin{cases}4x^3+4x^2, & \text{якщо } x<0,\\ 3, & \text{якщо } x=0,\\ \sqrt{x^2+1}, & \text{якщо } x>0.\end{cases}$
20	$y=\begin{cases}\lvert x+x^2\rvert+2, & \text{якщо } x<10,\\ 3+\sqrt{x^3+1}, & \text{якщо } x=10,\\ 2x/\sqrt{x^3+4}, & \text{якщо } x>10.\end{cases}$

Питання до захисту лабораторної роботи 4

1 Наведіть формат оператора If … Then і опишіть його роботу.

2 Наведіть блок-схему роботи оператора If … Then.

3 Які знаки порівняння можна використовувати при записі умов в операторі If … Then.

4 Наведіть приклад використання оператора If, якщо треба, щоб після then виконувались декілька операторів.

Лабораторна робота 5

ВИКОРИСТАННЯ ОПЕРАТОРА SELECT CASE В АЛГОРИТМАХ РОЗГАЛУЖЕНОЇ СТРУКТУРИ

Мета роботи*:* ознайомитися з оператором Select Case.

Завдання на лабораторну роботу

У даній лабораторній роботі необхідно скласти програму (варіант встановлює викладач), яка розраховує значення Y, відповідно до введеного X за допомогою конструкції Select Case . Крім цього, програма повинна виводити текст умови, що спрацювала, містити кнопку очищення всіх вікон і кнопку виходу з програми.

Короткі теоретичні положення

Крім If Then, широко розповсюджений оператор умовної передачі управління Select Case. Вони близькі за логікою роботи, взаємозамінні при розв'язанні більшості завдань, але бувають ситуації, у яких одна все-таки переважніша за іншу. Відмінність цих конструкцій обумовлена їх форматами.

Розглянемо формат конструкції Select Case.

Формат Select Case не відрізняється таким різноманіттям можливих конструкцій, як формат If Then. Він відразу представляється багатоповерховою формою, яка може містити вкладені блоки. Формат оператора має такий вигляд:

```
Select Case <вираз>
  Case <умовний_вираз_1>
  <Блок_операторів_1>
  Case <умовний_вираз_2>
  <Блок_операторів_2>
  [Case Else
  Блок_операторів_ELSE ]
End Select
```

Робота оператора. Конструкція спочатку розраховує вираз (як вираз може бути число, змінна, математичний вираз або функція) і, залежно від його значення, виконує той або інший блок

операторів. Значення виразів відслідковуються за допомогою умов, записаних за службовими словами Case. Якщо значення відповідає умові, то програма виконує блок операторів даного Case і передає управління за межі конструкції – першому операторові за End Select. У тому випадку, якщо значення виразу не задовольняє жодну з умов у всіх Case, програма виконує блок операторів, розташованих за Case Else. Службові слова Case Else і оператори за ними використовувати не обов'язково, тому вони записані у квадратних дужках. Блок-схема конструкції Select Case зображена на рисунку 5.1.

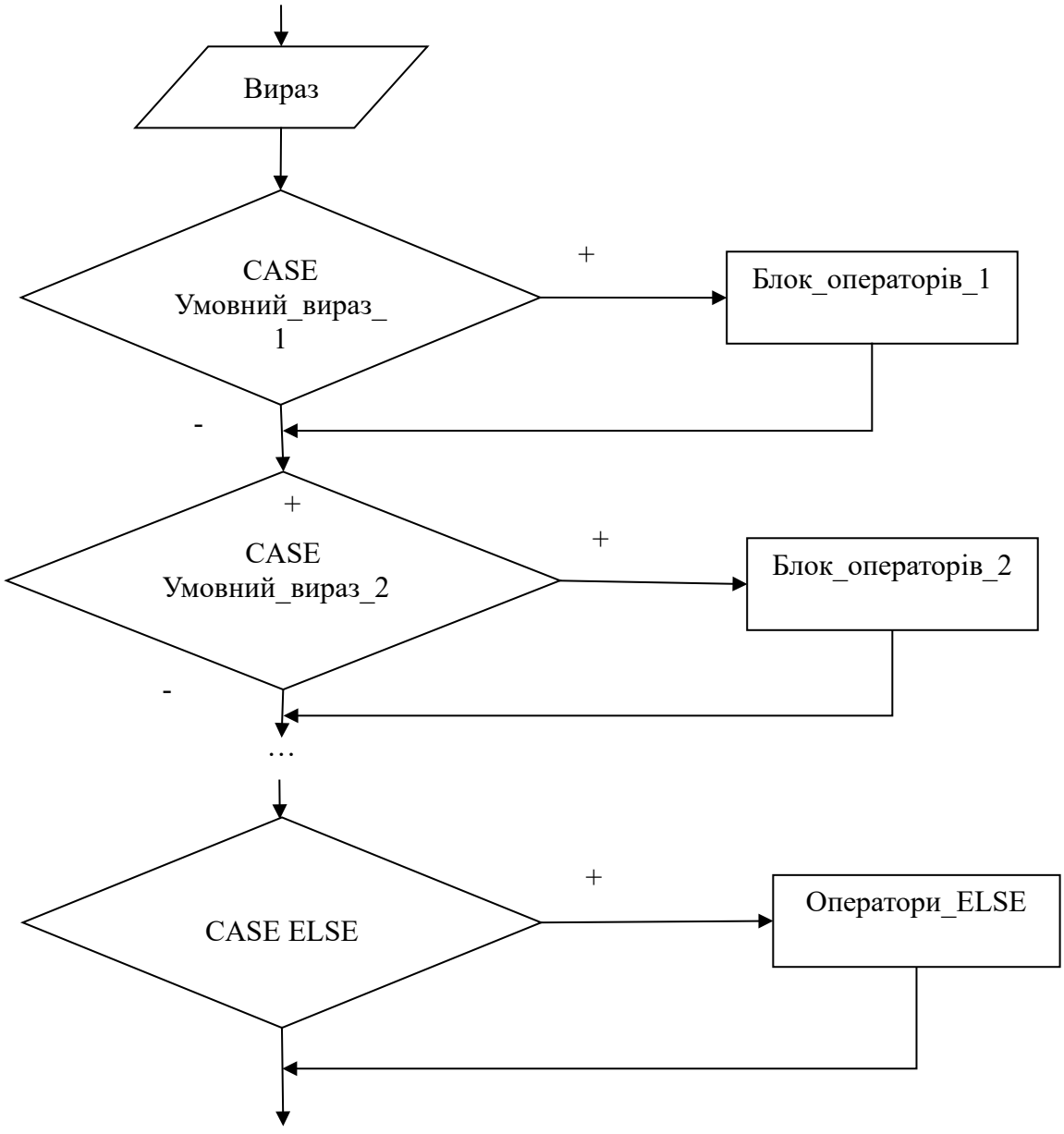


Рисунок 5.1 – Блок-схема конструкції Select Case

Розглянута конструкція має ряд тонкощів.

1 Якщо в декількох Case повторюються ті самі умови, то машина виконає блок операторів тільки за першим з таких Case.

2 Case умова – це дія, тому оператори даного Case повинні записуватися з наступного рядка або в цей же рядок, але через знак «:». Машина виконує всі оператори від Case, з яким спрацювала умова, до наступного Case або до End Select, якщо це був останній Case.

3 Між Select Case і першим Case не можна розміщати нічого, крім текстових коментарів, інакше машина виведе повідомлення про помилку “Statements and labels invalid between Select Case and first Case” (“Оператори й мітки неприпустимі між Select Case і першим Case ”).

Умовні вирази за Case можуть бути записані в одному із чотирьох форматів:

1 Число. Наприклад, Case 5.

2 Діапазон від меншого значення до більшого. Наприклад, Case 10 To 20. Якщо діапазон буде записаний від більшого значення до меншого, то машина не видасть повідомлення про неправильний запис, але буде перевіряти тільки перше значення. Помилки такого роду можуть виникнути при записі діапазонів у від’ємній області.

3 Умова. Підтримуються 6 знаків порівняння. Логічні оператори прямо стосовно параметра, що відслідковується, використовувати не можна. У записі умови назву параметра обов'язково потрібно заміняти словом “Is”. Наприклад, умова X>10 повинна бути записана як Case IS > 10. Причому Is ставиться завжди ліворуч від знака порівняння.

4 Комбінація форматів 1 – 3, розділених комами. Наприклад, Case 5, 15, 20 To 30, IS > 100. Програма відпрацює оператори, що відповідають Case, якщо виконається як мінімум одна умова з комбінації. Розглянемо приклад використання оператора.

Приклад 5.1

Розрахувати значення Y залежно від значення X.

$$Y = \begin{cases} 10, & \text{при } X=2; \\ 20, & \text{при } X=5, 7; \\ 30, & \text{при } X=10\text{-}20; \\ 40, & \text{при } X>100; \\ 50, & \text{при інших значеннях X.} \end{cases}$$

```
X = Val(txtArgument.Text)
Select Case X
Case 2
  Y = 10
Case 5, 7
  Y = 20
Case 10 To 20
  Y = 30
Case Is > 100
  Y = 40
Case Else
  Y = 50
End Select
  txtResult.Text = Y
```

Робота програми

Конструкція Select Case відслідковує значення змінної X.

Якщо її значення задовольняє яку-небудь умову за одним з Case, то виконується дія за цим оператором. Зокрема при X=5, спрацює другий Case і змінній у буде присвоєне число 20.

Якщо значення x буде більше 100, то спрацює четвертий Case і змінна у одержить значення 40.

У тому випадку, коли значення x не буде задовольняти жодному з умов в Case (наприклад X=80), будуть виконані дії, розташовані за Case Else.

Порядок виконання лабораторної роботи

1 Запустити програму Visual Basic for Applications.

2 Вивести на форму дві кнопки, чотири етикетки й чотири текстових вікна. Зовнішній вигляд форми наведено на рисунку 5.2.

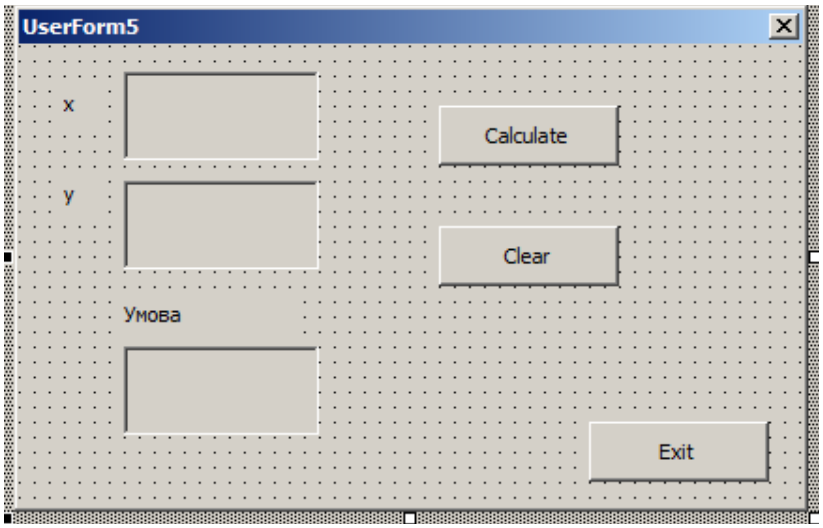


Рисунок 5.2

3 Програма повинна містити введення вхідних даних, обчислення системи рівнянь та виведення результатів на екран.

4 Властивості елементів управління наведено в таблиці 5.1.

Таблиця 5.1 - Властивості елементів управління

Елемент	Властивість	Значення
Кнопка «Calculate»	Name	cmdCalculate
	Caption	Calculate
Кнопка «Exit»	Name	cmdExit
	Caption	Exit
Кнопка «Clear»	Name	cmdClear
	Caption	Clear
Текстове вікно	Name	txtx
Текстове вікно	Name	txty
Текстове вікно	Name	txtComment
Етикетка	Name	lblx
	Caption	x
Етикетка	Name	lbly
	Caption	y
Етикетка	Name	lblComment
	Caption	Умова

5 Двічі клацнувши по кнопці «Calculate», перейти в редактор коду й у виведеній заготовці процедури скласти програму, яка розраховує математичні вирази.

6 Викликати заготовку процедури для кнопки «Exit» і набрати в ній команду на завершення програми.

7 Клацнувши мишкою по кнопці ▶, запустити програму.

Розглянемо розв'язання завдання на прикладі.

Приклад 5.2

Вивести на екран результат та спрацьовану умову. Запрограмувати кнопку очистки всіх текстових вікон та кнопку виходу з програми (рисунок 5.2).

$$y = \begin{cases} x+1, & \text{при } x=2; \\ x+2, & \text{при } x=1, 5, 8; \\ x+3, & \text{при } x>0; \\ x+4, & \text{при } x \text{ від } 15 \text{ до } 35; \\ x+5, & \text{при інших значеннях } x. \end{cases}$$

Код програми:

```
Private Sub cmdCalculate_Click()
    ‘ Обчислення по умові
    x = Val(txtx.Text)
    Select Case x
    Case 2
      txty.Text = x + 1
      txtComment.Text = "x = 2"
    Case 1, 5, 8
      txty.Text = x + 2
      txtComment.Text = "1, 5, 8"
    Case Is < 0
      txty.Text = x + 3
      txtComment.Text = "x < 0"
    Case 15 To 35
      txty.Text = x + 4
      txtComment.Text = "від 15 до 35"
```

```
    Case Else
      txty.Text = x + 5
      txtComment.Text = "Друге значення"
    End Select
End Sub
```

```
    Private Sub cmdClear_Click()
    ‘ Очистка текстових вікон
      txtx.Text = " "
      txty.Text = " "
      txtComment.Text = " "
    End Sub
```

```
    Private Sub cmdExit_Click()
    ‘ Завершення програми
      End
    End Sub
```

Зміст звіту

У звіті повинно бути:

а) мета завдання;

б) розв’язання завдання, згідно з варіантом;

в) рисунки з елементами управління: форма із кнопками, текстові вікна, етикетки;

д) результати обчислень;

г) висновки за лабораторною роботою.

Варіанти самостійних завдань до лабораторної роботи 5

Варіант	Функція	Умова
1	$y = \begin{cases} x+1000, \\ x+2000, \\ x+3000, \\ x+4000, \\ x+5000, \end{cases}$	при x=50; при x=40, -50, 10; при x<0; при x від 3 до 10; при інших значеннях

Варіант	Функція	Умова
2	$y = \begin{cases} x+10, \\ x+20, \\ x+30, \\ x+40, \\ x+50, \end{cases}$	при x=5; при x=4, -5, 0; при x>80; при x від 30 до 50; при інших значеннях
3	$y = \begin{cases} x+100, \\ x+200, \\ x+300, \\ x+400, \\ x+500, \end{cases}$	при x>80; при x=4, 5, 9; при x від 10 до 20; при x=2; при інших значеннях
4	$y = \begin{cases} x+10, \\ x+20, \\ x+30, \\ x+40, \\ x+50, \end{cases}$	при x=1, 5; при x=2, -2, 20; при x від -50 до -10; при x>100; при інших значеннях
5	$y = \begin{cases} x-1, \\ x-2, \\ x-3, \\ x-4, \\ x-5, \end{cases}$	при x=-2; при x=8, -5, 10; при x<=0; при x від 30 до 40; при інших значеннях
6	$y = \begin{cases} x+5, \\ x+6, \\ x+7, \\ x+8, \\ x+9, \end{cases}$	при x=0.4, -2.5, 10; при x від 3 до 5; при x>90; при x=-2.5; при інших значеннях
7	$y = \begin{cases} x-25, \\ x-35, \\ x-45, \\ x-55, \\ x-65, \end{cases}$	при x=-2; при x=4, -5, 0.11; при x<-5; при x від 35 до 55; при інших значеннях

Варіант	Функція	Умова
8	$y = \begin{cases} x + 0.1 \\ x + 0.2, \\ x + 0.3, \\ x + 0.4, \\ x + 0.5, \end{cases}$	при x=400, -500, 10000; при x=55; при x від -300 до 50; при $x>10^6$; при інших значеннях
9	$y = \begin{cases} x + 15, \\ x + 25, \\ x + 35, \\ x + 45, \\ x + 55, \end{cases}$	при x від 3 до 15; при x=-4.5; при x>=20; при x=40, 0.5, 0; при інших значеннях
10	$y = \begin{cases} x - 1.5, \\ x - 2.5, \\ x - 3.5, \\ x - 4.5, \\ x - 5.5, \end{cases}$	при x=2, -2, 10; при x=1; при x від 3 до 50; при x<=-18; при інших значеннях
11	$y = \begin{cases} x + 11, \\ x - 12, \\ x + 21, \\ x - 22, \\ x + 50, \end{cases}$	при x=1; при x=5, 18; при x>40; при x від 19 до 35; при інших значеннях
12	$y = \begin{cases} x + 33, \\ x + 34, \\ x + 35, \\ x + 36, \\ x + 37, \end{cases}$	при x=-1; при x=-15, 108; при x<-40; при x від 2 до 5; при інших значеннях
13	$y = \begin{cases} x + 10.2, \\ x + 20.2, \\ x + 30.2, \\ x + 40.2, \\ x + 50.2, \end{cases}$	при x=5; при x=4, 0; при x>6; при x від 1 до 3; при інших значеннях

Варіант	Функція	Умова
14	$y = \begin{cases} x + 150, \\ x + 160, \\ x + 170, \\ x + 180, \\ x + 190, \end{cases}$	при x=5, 8; при x=6, 2, 0; при x<0; при x від 100 до 500; при інших значеннях
15	$y = \begin{cases} x - 10, \\ x - 20, \\ x - 30, \\ x - 40, \\ x - 50, \end{cases}$	при x=45, 55, 899; при x=555; при x<=0; при x від 3 до 50; при інших значеннях
16	$y = \begin{cases} x + 33, \\ x + 44, \\ x + 55, \\ x + 66, \\ x + 77, \end{cases}$	при x=3; при x=1, -2, -7; при x<=-10; при x від 10 до 1000; при інших значеннях
17	$y = \begin{cases} x * 10, \\ x * 20, \\ x * 30, \\ x * 40, \\ x * 50, \end{cases}$	при x=25; при x=4, 5; при x>99; при x від 40 до 50; при інших значеннях
18	$y = \begin{cases} x + 11, \\ x + 12, \\ x + 13, \\ x + 14, \\ x + 15, \end{cases}$	при x=7; при x=8, 9; при x>10; при x від -30 до -5; при інших значеннях
19	$y = \begin{cases} x * 15, \\ x * 20, \\ x * 25, \\ x * 30, \\ x * 35, \end{cases}$	при x=1; при x=6, 8; при x>15; при x від 20 до 45; при інших значеннях

Варіант	Функція	Умова
20	$y = \begin{cases} x/10, \\ x/20, \\ x/30, \\ x/40, \\ x/50, \end{cases}$	при x=50; при x=40, 60; при x>90; при x від 20 до 35; при інших значеннях

Питання до захисту лабораторної роботи 5

1 Наведіть формат оператора Select Case і опишіть його роботу.

2 Наведіть блок-схему оператора Select Case.

3 Опишіть особливості оператора Select Case.

4 Наведіть формати запису умов за командами Case.

Список літератури

1 ДСТУ 2873-94. Системи оброблення інформації. Програмування. Терміни та визначення. – К.: Держстандарт України, 1994.

2 Гарнаев А.Ю. Microsoft Excel 2002: Разработка приложений. – СПб.: BHV-Санкт-Петербург, 2002. – 763 с.

3 Гарнаев А.Ю. Самоучитель VBA. – 2-е изд. – СПб.: BHV-Петербург, 2004. – 540 с.

4 Гарнаев А.Ю. VBA: в подлиннике. – СПб.: BHV-Санкт-Петербург, 2005. – 848 с.

5 Титаренко Г. Visual Basic 6.0 / Г. Титаренко. − К.: BHV, 2002. − 447 с.

6 Створення програмних додатків в середовищі MS Office Visual Basic for Application (для самостійного вивчення): Навч. посібник / Укладачі Ш.Ф. Григоришин, С.В. Косяченко, Л.В. Кульбаба. – К.: Дакор: КНТ, 2007. – 208 с.

7 Карпов Б.И. Специальный справочник. VBA (Visual Basic for Applications). Скриптовый язык приложений Microsoft Office. – СПб.: Питер, 2002. – 416 с.

8 Слепцова Л. Д. Программирование на VBA в Microsoft Office 2007. Самоучитель. — М.: Диалектика, 2007. – 432 с.

9 Кузьменко В.Г. Программирование на VBA 2002. – М.: Бином-Пресс, 2003. – 880 с.

10 Уокенбах Д. Профессиональное программирование на VBA в Excel 2002. – М.: Вильямс, 2003. – 784 с.