Глинський Я. М., Анохін В. Є.,
Ряжська В. А.

БЕЙСИК

Від QBASIC до VISUAL BASIC .NET

Навчальний посібник

5-те вид., доп.

Львів – 2006

ББК 32.973я7
Г 54
УДК 681.3

Рекомендовано Міністерством освіти і науки України як навчальний посібник для студентів нетехнічних спеціальностей вищих навчальних закладів
(Лист МОНУ від 09.07.2003 р. № 14/18.2-1214)

Рецензенти:
д-р фіз.-мат. наук, проф. *Каленюк П. І.*,
д-р фіз.-мат. наук, проф. *Савула Я. Г.*,
канд. фіз.-мат. наук, доц. *Монцібович Б. Р.*

Глинський Я. М., Анохін В. Є., Ряжська В. А.
Г54 Бейсик. Від Qbasic до Visual Basic .NET: Навч. посіб. 5-те вид., доп. — Львів: СПД Глинський, 2006. — 192 с.
ISBN 966-2934-03-0

Описано можливості сучасних версій мови Бейсик: Qbasic, Visual Basic 6, VBA, Visual Basic .NET. Виклад теоретичних відомостей супроводжується значною кількістю прикладів і зразками розв'язування типових навчальних задач. Для організації ефективної індивідуальної роботи учнів і студентів, підготовки і проведення практичних занять наведено багатоваріантний збірник задач.

Завдяки лаконічному стилю, оригінальній апробованій методиці викладу навчання буде доступним, зручним та приємним.

Для учнів шкіл, студентів та широкого кола читачів.

Г $\frac{1404000000-003}{2006}$ Без оголошення **ББК 32.973я7**

ISBN 966-2934-03-0

Розділ 1. QBASIC

§ 1. ОСНОВНІ ПОНЯТТЯ

1. Призначення та версії мови. Мова Бейсик є мовою програмування високого рівня. Вона дає змогу описувати алгоритми розв'язування задач у стислій формі, використовуючи слова розмовної англійської мови.

Мова Бейсик є зручною для вивчення основ програмування та початкового освоєння сучасної обчислювальної техніки. Її можна застосувати практично на всіх моделях комп'ютерів.

Мовою Бейсик записують програми для розв'язування як різноманітних задач обчислювального характеру, так і інформаційно-пошукових задач, тобто задач опрацювання текстової інформації.

Мова є діалоговою. Бейсик-програми зрозумілі та наочні. Ось як просто виглядає програма, в результаті виконання якої на екрані монітора буде виведено число 5:

```
PRINT 5
```

Вважають, що назва мови утворена з перших букв англійських слів *Beginner's All-purpose Sumbolic Instruction Code*. У вільному перекладі це означає "мова програмування для початківців". Англійське слово *Basic* перекладають як елементарний, базовий (наприклад, набір знань). Перша версія мови налічувала 14 службових слів.

Мову Бейсик розробили співробітники Дартмудського коледжу (США) у 1964 р. з метою навчання студентів, і вона відразу здобула широке визнання. У 80-х роках інтерес до мови зменшився, оскільки її різноманітні версії не задовольняли вимог структурного програмування. Цей недолік було подолано у версіях мови Turbo Basic, Quick Basic (Швидкий Бейсик) та Power Basic. Спрощена версія Швидкого Бейсика – Qbasic – увійшла до комплекту програм операційної системи MS-DOS корпорації Microsoft.

Усі зазначені вище версії мови мають багато спільних елементів, які ми розглянемо в цьому розділі під загальною назвою Бейсик. Ознайомившись з ним, можна працювати практично з будь-якою версією. Відмінності та особливості різних версій мови описані в довідниках. Можливості найновіших версій (компіляторів, що дають змогу створювати exe-файли) настільки великі, що в багатьох випадках вони успішно конкурують з мовами Паскаль та Сі.

Мова Бейсик постійно розвивається. Вона суттєво оновилася після створення корпорацією Microsoft програмного середовища Visual Basic (VB), яке дає змогу користувачам засобами візуального програмування складати власні програми для операційної системи

Windows. Більшість програмних засобів у Microsoft створені засобами цієї мови. Для розширення можливостей таких програм, як MS Word, MS Excel, MS Access та інших корпорація Microsoft розробила версію VB під назвою Visual Basic for Applications (VBA).

Наступне удосконалення мови – Visual Basic .NET, що є складовою інтегрованого середовища створення програм Visual Studio .NET. *Середовище* – це програмний комплекс, що автоматизує процеси створення, налагодження та виконання програм користувача.

Visual Basic .NET дає змогу не лише створювати типові навчальні програми, що можна робити і в середовищі Qbasic, чи типові Windows-проекти, для чого достатньо середовища Visual Basic, але й високопрофесійні програми для роботи в локальній мережі, а також в інтернеті з характерним web-інтерфейсом, що запускаються з сервера, доступ до яких здійснюється за допомогою броузера.

Мову Бейсик у цьому розділі викладатимемо з позицій структурного підходу з метою вироблення правильних навичок складання алгоритмів і написання програм. Усі наведені нижче програми були протестовані в середовищі Qbasic. Після деяких змін більшість з них можна реалізувати в середовищі Visual Basic 6, а після незначного доопрацювання – у Visual Basic .NET.

2. Алфавіт мови. Алфавіт мови складається з таких символів:

- латинських літер від A до Z та літер кирилиці;
- цифр від 0 до 9;
- символів математичних операцій (+, –, *, /, ^);
- символів відношення (=, <, > та інших);
- розділових символів (крапка, кома, двокрапка, крапка з комою, лапки, круглі дужки, пропуск);
- спеціальних символів (!, #, $, %, ^, & та інших).

Алфавіти можуть несуттєво відрізнятися в різних версіях мови. За допомогою символів алфавіту будують елементи мови: службові слова, імена змінних, числа, вирази, імена функцій та ін.

3. Службові слова та команди мови. Програми складаються з команд. Іноді команди називають операторами, вказівками, реченнями. За призначенням команди поділяють на такі групи:

- описові команди, які використовують для опису даних, типів змінних, розмірів масивів, нестандартних функцій тощо;
- команди введення-виведення даних, які використовують для введення-виведення інформації, зокрема, для організації діалогу користувача з комп'ютером;
- команди для обчислень і керування процесами опрацювання інформації: команди присвоєння, переходу, розгалуження, циклів тощо;
- інші команди, які забезпечують додаткові можливості: команди для роботи з файлами даних, команди для графічних побудов, отримання звукових ефектів тощо.

Команди складаються з неподільних елементів мови (літералів): службових слів, чисел, символів операцій тощо. Службові слова прийнято записувати прописними літерами. Якщо службове слово набрати рядковими літерами, то програма Qbasic автоматично переписує його прописними. Основні *службові* слова мови QBasic:

DATA – дані	NEXT – наступний
DIM – оголошення змінних	ON – при
END – кінець	OPTION BASE – задати базу
FOR-TO-STEP – для-до-крок	PLAY – грати
FUNCTION – опис функції	PRINT – друкувати
GOTO – перейти до	READ – читати
IF-THEN – якщо-то	REM – пояснення
INPUT – ввести	RESTORE – відновити
LET – нехай	RETURN – повернутися
DO-LOOP – цикл	WHILE-WEND – цикл «доки»

4. Структура Qbasic-програми. Бейсик-програма складається з програмних рядків. Один програмний рядок не повинен бути довшим, ніж 255 символів. У середовищі Qbasic під час створення довшого, ніж 80 символів програмного рядка відбувається скролінг – зміщення тексту ліворуч, тому використовувати довгі рядки не рекомендують.

У середовищі VB частину команди можна переносити. У цьому випадку використовують символ переносу «_», який розташовують у кінці рядка після лексеми (неподільного елемента мови) і пропуска. Для переходу на наступний рядок натискають на клавішу вводу.

В одному рядку можна розмістити одну або декілька команд, відокремлених одна від одної двокрапкою (:).

Довідка. У ранніх версіях мови програмний рядок починався з номера. Номер програмного рядка – це ціле число з деякого діапазону, наприклад, від 1 до 65535. Команди виконувались за зростанням номерів відповідних рядків. Для зручності програмні рядки нумерували числами з певним інтервалом, наприклад, з інтервалом 10: 10, 20, 30... Це давало змогу системі автоматично вставляти в середину програми пропущені й згодом додатково введені програмні рядки з такими номерами: 15, 17, 25 тощо. У сучасних версіях номери рядків можна не писати. Тут вони не впливають на черговість виконання команд програми. Якщо ж номери є, то система трактує їх як мітки (позначки). Після номера рядка можна ставити двокрапку.

Задача 1. Скласти програму для визначення сторони (C), периметра (P) та площі (S) прямокутного трикутника за відомими катетами: A=5; B=3,6.

```
REM       Моя перша Бейсик-програма
A=5 : B=3.6            'У рядку є дві команди і коментар
C = SQR(A^2 + B^2) 'Застосовуємо теорему Піфагора
P = A + B + C
PRINT "P="; P
S = 1 / 2 * A * B
PRINT "S="; S
END
```

Пояснення. Текст після символу ' чи службового слова REM є коментарем (поясненням). Символ * позначає операцію множення, / – ділення, ^ – піднесення до степеня, + – додавання, SQR( ) – це функція обчислення квадратного кореня з A^2+B^2, тобто гіпотенузи C. Аргумент функції записують у круглих дужках. Команда PRINT призначена для виведення результатів на екран (тут з поясненням змісту результатів: P= і S=). У дійсних числах для відокремлення дробової частини числа від цілої використовують крапку, а не кому. Команда END є необов'язковою.

Виконання. Після виконання програми у середовищі Qbasic (за допомогою команди Run) на екрані отримаємо

```
P= 14.76117
S= 9
```

5. Структура Бейсик-коду у VB-програмі. Для створення Бейсик-коду в середовищі VB потрібно після відкриття нового проекту двічі клацнути на формі. Отримаємо вікно, що вже міститиме заготовку процедури Form_Load():

```
Private Sub Form_Load()
    <сюди вводять текст Бейсик-коду>
End Sub
```

Бейсик-код для задачі про трикутник має такий вигляд:

```
Show
Rem Моя перша програма
A = 5: B = 3.6 'У рядку є дві команди і коментар
C = Sqr(A ^ 2 + B ^ 2)
P = A + B + C
Print "P="; P
S = 1 / 2 * A * B
Print "S="; S
```

Порівняйте програми для двох різних середовищ: Qbasic та VB. У VB-програмі дописано команду Show і забрано команду END. Виконаємо VB-програму. На екрані отримаємо

```
P= 14.7611687202997
S= 9
```

Наведена вище програма не працюватиме в середовищі VB .NET. Інший підхід до складання програм, сумісний з VB .NET, розглянемо у розділі 4.

Завдання. Виконайте програму про трикутник, якщо A=8,2; B=6,3. Який результат отримали? Модифікуйте програму, щоб: 1) вивести на екран значення сторони C; 2) обчислити і вивести на екран значення тангенса гострого кута A (BAC); 3) обчислити висоту, опущену з вершини прямого кута на гіпотенузу; 4) обчислити радіус кола, описаного навколо трикутника.

Запитання

1. Хто і коли створив мову Бейсик?
2. Яке призначення мови Бейсик?
3. Назвіть декілька версій мови.
4. Опишіть алфавіт мови.
5. На які групи поділяють команди мови?
6. З чого складається програмний рядок?
7. Який символ слугує для відокремлення команд в одному рядку?
8. Чому в старих версіях програмні рядки рекомендують нумерувати числами з деяким інтервалом?

§ 2. ТИПИ ДАНИХ

1. Текстові дані. *Дане* – це деяка інформація, наприклад, числова чи текстова, яку опрацьовують за допомогою комп'ютера.

Залежно від характеру інформації дані поділяють на типи. ***Тип*** визначає допустимі значення даного, а також операції, які можна над ним виконувати.

Розглянемо два основні типи даних: ***текстові*** (інші назви – символьні, літерні, рядки) та ***числові***.

Текстові дані – це тексти, записані у подвійних лапках, наприклад, "Інформатика", "Маса", "2006".

Текстові дані застосовують, зокрема, в команді виведення PRINT для створення заголовків та пояснення результатів, наприклад, PRINT "Це розв'язок задачі" або PRINT "P=", P. Лапки на екран не виводяться.

2. Числові дані. *Числові дані* призначені для роботи з числовою інформацією. В інформатиці розрізняють числа ***цілі*** (1, –25, 100) та ***дійсні*** (–1.0, –15.2, 12.25). Дійсні дані зазвичай містять у своєму зображенні десяткову крапку. Цілі числа зображають цифрами без крапки.

Дійсні числа. Є дві форми зображення дійсних чисел: 1) з фіксованою крапкою, 2) з плаваючою (рухомою) крапкою.

У зображеннях чисел з фіксованою крапкою розміщення десяткової крапки безпосередньо вказує на величину числа. Наприклад: 2., –10., 0.00006, 9000000.

У математиці для зручного написання дуже великих або дуже малих чисел є показникова (наукова) форма їх зображення:

$$0{,}000006=0{,}6\cdot10^{-5}\ ,\quad 900000000=9\cdot10^{8}\ .$$

Число у показниковій формі має загальний вигляд $\pm m \cdot 10^{\pm p}$, де $\pm m$ – мантиса, а $\pm p$ – порядок числа. Знак "+" можна опускати. Знак "–" опускати не можна.

У мовах програмування відповідне число зображають так:

±mE±p

Наприклад, одне й те ж число 0,000006 можна записати різними способами: 0.6E–5, 6.0E–6, .06E–4.

Оскільки розміщення десяткової крапки в мантисі явно не вказує на величину числа, то кажуть, що "крапка плаває". Тому таке зображення числа називають зображенням з плаваючою крапкою. Наприклад, число 12,5 можна записати так:

12.5, 12.5E0, 1.25E+1, 125.E–1, 0.125E+2.

3. Типи даних. *Цілі дані* поділяють на ***короткі*** (назва типу INTEGER): –45, 135, 32765 та ***довгі*** (LONG): 32769, –23465500.

Оперативну пам'ять комп'ютера вимірюють байтами. В одному байті може зберігатися один символ алфавіту або число зі значенням від 0 до 255, або від –127 до 128.

У Qbasic значення коротких цілих даних (INTEGER) займають по 2 байти в пам'яті комп'ютера, а довгих цілих (LONG) – по 4. Тому цілі короткі дані мають значення в діапазоні від –32768 до 32767, а цілі довгі – орієнтовно до плюс-мінус двох мільярдів. У VB і VB .NET дещо інакше: цілі типу SHORT займають 2 байти, типу INTEGER – 4, а типу LONG – 8 байтів.

Дійсні дані. Розрізняють такі типи числових дійсних даних: 1) *короткі* (одинарні, чотирибайтні, назва типу SINGLE) – до 7 цифр у мантисі; 2) *довгі* (подвійні, з подвійною точністю, назва типу DOUBLE) – до 15 цифр. Для зберігання довгих даних система відводить удвічі більше пам'яті (8 байтів), ніж для коротких.

4. Константи (сталі). Числам чи текстам можна надавати імена за допомогою команди оголошення сталих CONST

CONST <ім'я 1> = <вираз 1>, <ім'я 2> = <вираз 2>, ...

Команду записують в одному програмному рядку або, якщо сталих багато, кожний наступний рядок має починатися зі службового слова CONST. Кутові дужки < > тут призначені для формального запису команд, у фактичних командах їх не пишуть. Наприклад,

CONST Pi = 3.1415926, A = "Програмування".

Дія команди. Обчислюються значення виразів і результати присвоюються відповідним сталим. Дані Pi та A називають сталими.

Сталі дані не можуть змінювати свого значення під час виконання програми. За цією ознакою дані в програмуванні поділяються на ***сталі*** та ***змінні***.

5. Змінні. У математиці та фізиці величинам дають імена: сторони – a, висоті – h, часу – t, площі чи шляху – s тощо. Аналогічно у програмуванні. Фізичні величини зазвичай можуть міняти свої значення, тобто є *змінними*. Саме цим терміном у програмуванні прийнято називати дані, які можуть міняти свої значення під час виконання програми.

Змінна – це поіменована ділянка оперативної пам'яті комп'ютера, де зберігається значення деякої величини. Змінна має такі властивості: назву, значення, тип.

Назву змінній надає користувач. У ранніх версіях мови використовували короткі імена, які складалися з одної літери або з літери і цифри. Приклади коротких назв змінних: A, A1, A6, C6.

У нових системах імена можуть бути довшими і складатися як з великих, так і з малих латинських літер (і цифр), наприклад, Myname1, Myname2. Цифра не може бути першим символом в імені. Регістр значення не має. Імена d і D позначають одну і ту ж змінну.

Під час складання програми імена змінним варто надавати, керуючись змістом задачі, наприклад, суму можна позначати буквою S, добуток – D, аргументи x_1, x_2 деякої функції – X1, X2.

6. Типи змінних. Змінні бувають числові та текстові. Поділ змінних на типи користувач може зафіксувати у програмі різними способами. Їх є чотири:

1) принцип замовчування;
2) спеціальні символи: $, % чи &, ! чи #;
3) команди описування типів DEF;
4) команди оголошення змінних DIM.

Використання принципу замовчування. Якщо в програмі немає описів типів, то Qbasic розглядає змінні як короткі дійсні числові, а VB – як довгі дійсні числові. Однак текстові змінні потрібно описувати *завжди*. Для цього до імені текстової змінної дописують спеціальний символ $, наприклад, A$, A1$.

Використання спеціальних символів. Щоб зазначити явно типи змінних, до їхніх імен дописують спеціальні символи (суфікси).

До імені змінної цілого типу можна дописати символ % (коротке дане) або символ & (довге дане), наприклад, A1%, C&.

Оголосити короткі чи довгі дійсні змінні можна за допомогою символів ! чи # відповідно. Наприклад, D!, X1#.

Імена текстових змінних закінчуються суфіксом $, наприклад, B5$, C$, M$(8).

Спеціальні символи можна дописувати до чисел, наприклад: 5%, 5234455&, 5.2!, 523234.56455#, але це робити необов'язково.

Використання команд описування типів. У більшості сучасних мов програмування спеціальні символи для описування типів змінних не використовують, тому в мові Бейсик варто надавати перевагу командам опису типів або принципу замовчування.

Команди описування типів записують на початку програми. У Qbasic використовують такі команди:

Опис змінних	Призначення
DEFINT список літер	Для змінних цілого короткого типу
DEFLNG список літер	Для змінних цілого довгого типу
DEFSNG список літер	Для дійсних коротких змінних
DEFDBL список літер	Для дійсних довгих змінних
DEFSTR список літер	Для змінних символьного типу

Приклад 1. Цілі змінні A, B, C, D можна описати так:

```
DEFINT A, B, C, D
```

або так:

```
DEFINT A, B : DEFLNG C, D.
```

Приклад 2. Щоб проінформувати систему, що всі змінні, які починаються із символів A, B, C, D, K, M, є змінними цілого типу, із символів E, F дійсного короткого типу, із P, Q текстового типу, на початку програми записують такі команди:

```
DEFINT A–D, K, M
DEFSNG E, F
DEFSTR P, Q.
```

Зверніть увагу на дефіс у першій команді. Тут він замінює перелік елементів списку через кому (A, B, C, D).

Використання команди оголошення змінних DIM. Для оголошення змінних у сучасних версіях мови застосовують команду DIM

```
DIM <змінна 1> AS <тип 1>, <змінна 2> AS <тип 2>
DIM <змінна 3> AS <тип 3>, ...
```

Приклад 3. Цілі змінні A, B, дійсні C, D та текстову H можна оголосити так:

```
DIM A AS INTEGER, B AS INTEGER, C AS DOUBLE
DIM D AS DOUBLE, H AS STRING*10.
```

Текстові дані (назва типу STRING) можуть містити у Qbasic до 255 літер і займати в пам'яті до 256 байтів (у VB ці дані значно довші). Якщо заздалегідь відомо, що кількість літер у тексті не перевищує числа *n*, то тип цього даного описують як STRING*n.

Задача 1. Розглянемо програму для обчислення площі поверхні (S) та об'єму (V) кулі, радіус (R) якої дорівнює 8,95.

```
' Програма Куля
CONST PI = 3.1415926
DIM R AS SINGLE, S AS SINGLE, V AS SINGLE
CLS   'Очищує екран
R = 8.95
S = 4 * PI *R^2
V = 4* PI *R ^ 3 /3
PRINT "S ="; S
PRINT "V ="; V
END
```

Команда CLS призначена для очищення екрану під час виведення результатів. Після виконання команди на екрані отримаємо

```
S = 1006.598
V = 3003.016
```

Завдання. Виконайте програму Куля, якщо R=8,93. Який результат отримали? Обчисліть діаметр кулі.

Запитання та вправи

1. Які значення чисел:
   а) 1.E0; 0.005E+2; 25E–1; –1.7E+1;
   б) –0.052E–2; 2.53E+1; –0.172E4; 1E–5;
   в) 0.152E–1; 0.152E+01; 0.152E+02; 0.15E+3;
   г) 2.25E+01; –2.25E–01; 2.25E–02; 2.25E+02
   д) 3.45E+03; 3.45E+4; 3.45E5; –3.45E–1?
2. Записати числа мовою Бейсик:
   а) 1,5; 0,005; $-1{,}5 \cdot 10^{-2}$; 16000; 35,25;
   б) 2,64; $1{,}2 \cdot 10^{-3}$; 0,0002; 54300000; $0{,}823 \cdot 10^{-1}$.
3. Назвати типи таких даних:
   а) 3, 3., 3&, .3E+1, "3";
   б) 5., 5%, 5!, .5E+1, "5";
   в) 8%, "8", 8!, 8&, 8# .
4. Виписати правильно утворені імена змінних з такого списку:
   а) A, A1%, 5A, A%$, %A, A$;
   б) A1, 2A, B–2, AA2, A1$, B%, A%2;
   в) C, C%, C2!, 2C!, Cina, Ціна.
5. Записати команди опису типів для таких змінних:
   A, A1, A2, B, C – цілі;
   F1, F2, F5, F9, G, H – короткі дійсні;
   H1, H2, H3, H4, J1, X, V5 – текстові.
6. Описати типи змінних:
   K, M, M1, M2, N, L – цілі;
   F1, F2, F5, F9, G, H – довгі дійсні;
   H1, H2, H3, H4, J1, X, V5 – короткі дійсні.
7. Описати типи змінних з вправи 5, використовуючи спеціальні символи.
8. Описати типи змінних з вправи 6, використовуючи спеціальні символи.
9. Розв'язати задачу №1 свого варіанта з розділу "Задачі".

§ 3. КОМАНДА ПРИСВОЄННЯ. ФУНКЦІЇ. ВИРАЗИ

Прості (лінійні) програми складаються з команд присвоєння, введення-виведення даних та викликів процедур.

1. Команда присвоєння. Команду присвоєння використовують для надання початкового значення змінній або для зміни її поточного значення. Команда присвоєння має вигляд

LET <A> = <B>

Тут A – назва змінної, B – вираз, символ = тут є символом присвоєння. Службове слово LET є необов'язковим.

Вираз призначений для описування формул, за якими виконуються обчислення. Вираз може складатись з чисел, змінних, назв функцій, які з'єднані символами операцій.

Дія команди. Обчислюється значення виразу B і результат присвоюється змінній A. Попереднє значення змінної A втрачається.

Зауваження. Використання службового слова LET у всіх версіях мови є необов'язковим і його надалі писати не будемо.

Приклад 1. Розглянемо фрагмент програми:

```
A = 5 : B = 10 : C = -2.6
D = (A + B) / (B - A) + C.
```

Яке значення буде присвоєне змінній D? Відповідь: 0,4.

Приклад 2. У багатьох задачах потрібно лічити кількість деяких об'єктів (позначимо їх іменем K) або суму (S) величин Y. Відповідні команди присвоєння матимуть такий вигляд:

```
K = K + 1 'так обчислюють кількість
S = S + Y 'так обчислюють суму S значень Y.
```

2. Перетворення типів. Якщо числовий тип змінної A в команді присвоєння не відповідає числовому типу виразу B, то після обчислення виразу результат автоматично перетворюється до типу змінної A. Наприклад, якщо в програмі про кулю написати команду

```
S% = 4 * Pi * R^2,
```

то результат заокруглиться до цілого значення – 1006.

Не можна числовій змінній присвоювати текстове значення, а текстовій – числове.

3. Арифметичні вирази. Арифметичні вирази призначені для описування послідовності дій (операцій) з числовими даними. Вони дають змогу за допомогою позначень, близьких до математичних, описувати у програмі складні формули. Результатом обчислення арифметичного виразу є число.

Арифметичні вирази будує користувач відповідно до умови задачі, комбінуючи сталі, змінні, функції за певними правилами і поєднуючи їх за допомогою символів операцій.

Над числовими даними виконують такі операції:

- піднесення до степеня (^);
- множення (*) та ділення (/);
- додавання (+) та віднімання (–).

4. Операції \ та MOD. Над цілими даними виконують ще дві операції:
- цілочислового ділення (символ \);
- обчислення остачі від ділення двох чисел (слово MOD).

Перша операція відкидає дробову частину результату, наприклад, 5 \ 2 = 2, 359 \ 100 = 3, а 5 MOD 2 = 1, 17 MOD 3 = 2.

Задача 1. Дано тризначне ціле число А. Обчислити суму його цифр.

Позначимо шукані цифри С1, С2, С3, а суму – S. Розглянемо будь-яке тризначне число, наприклад 357, і переконаємося, що такі команди присвоєння забезпечують відшукання суми (15) його цифр:

```
' Програма Сума чисел
A = 357
C1 = A \ 100
C2 = A \ 10 MOD 10
C3 = A MOD 10
S = C1 + C2 + C3
PRINT "S ="; S
```

5. Послідовність виконання операцій у виразах така ж, як у математиці. Її описують правилом пріоритетів (порядком виконання операцій).

Є чотири рівні пріоритетів:

1) обчислюються значення функцій, якщо вони є у виразі;

2) виконуються всі наявні операції піднесення до степеня;

3) виконуються операції множення та ділення;

4) виконуються операції додавання та віднімання.

Операції одного рівня виконуються послідовно зліва направо. Для зміни природної послідовності виконання операцій використовують круглі дужки. *Кількість відкритих і закритих дужок у виразі повинна бути однаковою.* Спочатку обчислюються вирази у дужках. Якщо є вкладені дужки, то спочатку обчислюються вирази у внутрішніх дужках, а потім у зовнішніх та ін.

Оформлення виразів. Вирази записують у Qbasic в одному або (якщо допускає версія мови, наприклад VB) у кількох рядках без дублювання символів арифметичних операцій у випадку переносу. Є таке обмеження на оформлення виразів під час написання програми: показник степеня, індекси, чисельники та знаменники розміщують у горизонтальному рядку. Потрібно уважно записувати знаменники. У багатьох випадках їх беруть у дужки. Зайва пара дужок у складному виразі не призводить до помилки.

Тип результату. В арифметичному виразі використовують числові сталі та числові змінні різних типів – цілі, дійсні. Тип результату буде цілим тільки тоді, коли всі операнди будуть цілого типу. Наприклад, результат А% + В% буде цілого типу, а результати дій

A + B% чи A + B – дійсного, якщо заздалегідь не оголошено, що змінні A та B є цілого типу. Результат ділення двох чисел A / B завжди є даним дійсного типу.

6. Стандартні математичні функції. Функції поділяють на стандартні та нестандартні. Аргумент функції завжди беруть у круглі дужки. Аргументом може бути стала, змінна, арифметичний вираз, інша стандартна функція. Основні стандартні функції наведені в табл. 1.

Табл. 1. Стандартні функції

Функція	Математичний запис	Коментар
SIN(X)	$\sin x$	x задають у радіанах
COS(X)	$\cos x$	x задають у радіанах
TAN(X)	$\operatorname{tg} x$	x задають у радіанах
ATN(X)	$\operatorname{arctg} x$	Арктангенс x
ABS(X)	$\lvert x \rvert$	Модуль (абсолютна величина) x
SQR(X)	$\sqrt{x}$	Квадратний корінь x
LOG(X)	$\ln x$	Логарифм натуральний x
EXP(X)	e^x	Експонента
INT(X)	$[x]$	Найбільше ціле число, яке не перевищує x
FIX(X)		Ціле число, що утворюється відкиданням дробової частини числа x
RND(X)		Випадкове число з проміжку [0;1)
SGN(X)	$\operatorname{sign} x$	Знак числа $\operatorname{sign} x = \begin{cases} 1, & \text{якщо } x > 0, \\ 0, & \text{якщо } x = 0, \\ -1, & \text{якщо } x < 0, \end{cases}$

Приклад 3. Значеннями функцій TAN(0), ABS(–2), SQR(3*3–5), INT(2.5), INT(–2.5) є відповідно 0, 2, 2, 2, –3.

Задача 2. Розглянемо програму для обчислення периметра (P) та площі (S) трикутника за формулою Герона. Сторони трикутника такі: A = 5; B = 3,6; C = 4,2.

```
' Програма Трикутник-2
DIM A AS SINGLE, B AS SINGLE, C AS SINGLE
DIM P AS SINGLE, S AS SINGLE
A = 5 : B = 3.6 : C = 4.2
P = A + B + C
PRINT "P="; P
P = P / 2
```

```
S = SQR(P * (P - A) * (P - B) * (P - C))
PRINT "S="; S
END
```

Після виконання команди на екрані отримаємо

```
P= 12.8
S= 7.429237
```

Нагадаємо, що в QBasic-програмі можна не оголошувати змінні, тобто не писати команд DIM. У VB їх також можна опустити, але за замовчуванням усі змінні трактуватимуться системою як дійсні довгі. VB-код матиме такий вигляд:

```
' Програма Трикутник-2
Show
A = 5 : B = 3.6 : C = 4.2
P = A + B + C
Print "P="; P
P = P / 2
S = Sqr(P * (P - A) * (P - B) * (P - C))
Print "S="; S
```

Завдання. Виконайте програму, якщо A = 8,2; B = 6,3; C = 9,1. Обчисліть висоту трикутника, опущену на сторону A.

Запитання та вправи

1. Які є стандартні функції?
2. Сформулювати правило пріоритетів.
3. Сформулювати правило дужок.
4. Як оформляють арифметичні вирази?
5. Від чого залежить тип результату під час обчислення арифметичного виразу?
6. Від чого залежить тип результату після виконання команди присвоєння?
7. Яких значень набудуть функції:
   а) SIN(0); б) ABS(−1); в) SQR(25); г) INT(4.8); д) SGN(−5)?
8. Яких значень набудуть функції:
   а) INT(5/2); б) ABS(4−SQR(36)); в) COS(INT(1/2));
   г) EXP(2−2); д) SQR(ABS(−4))?
9. Записати мовою Бейсик такі вирази:
   а) $5x^2 + 2x + 3{,}5$; б) $3x^2 - 6x + \cos 2x$; в) $2{,}5x^2 + 7{,}2x + 3\sin 1{,}5x$;

   г) $\dfrac{x^2 + 7{,}4x}{x - 3{,}91}$; д) $\dfrac{1{,}4x^2 + 4{,}2x - 5{,}3}{\left(x^2 - 3\right)\left(x - 2{,}8\right)}$.

10. Записати мовою Бейсик такі вирази:

    а) $a\,x^2 + b\,x + c$; б) $\sqrt{a+b}\sin x + \cos 3{,}3x + \mathrm{tg}^2\,x - 2$;

    в) $2\,x^y + (x^y)^x$; г) $\sin^5 x^2 + \sqrt[4]{3y}$; д) $3\sin\sqrt{3{,}1x} + \sqrt{4{,}2y}\cos x^3$.

11. Які помилки допущені в записах арифметичних виразів:
    а) SIN(5X) + COSX; б) 3*X + 2Y/(5*X − 2*Y;

в) – A*–5/–2; г) SIN(ABS(2*X); д) 2,51X + 7A + A8 + .5E1?

12. Яких значень набудуть змінні в результаті виконання команди присвоєння, якщо раніше виконувались команди A=4, B=2:
а) A1=A + 2*B ; б) A2=A*A + 2/B ; в) A3=SQR(A) – B*A;
г) A4=A^2 + B^4 ; д) A5=1.4E1*A + A*B/0.2E1?

13. Яких значень набудуть змінні в результаті виконання команди присвоєння, якщо A=2, B=5, C=0:
а) A1=(2*A – 3*B)/(.6E1–B);
б) A2=SIN(2*C)/COS(A+B);
в) A3=A/.2E1+B +5*A/(B+C);
г) A4=INT(B/A)+B/A+SGN(B/A);
д) A5=A+B*A/(C+A*B)?

§ 4. ВВЕДЕННЯ ДАНИХ

У Qbasic є такі способи надання значень змінним:
1) за допомогою команди присвоєння;
2) за допомогою команди INPUT – уведення даних з клавіатури в режимі діалогу з комп'ютером;
3) за допомогою команди READ – зчитування та надання значень з блоку даних.

1. Використання команди присвоєння. Команду присвоєння використовують також для надання значень змінним.

Приклад 1. Нехай змінній A потрібно надати значення 5, змінній B – 10, змінній C – -2.5, змінній D$ – значення "Україна". Відповідні команди матимуть такий вигляд:

A = 5 : B = 10 : C = –2.5 : D$ = "Україна"

Надання значень змінним за допомогою команди присвоєння є простим способом, але має два суттєві недоліки. По-перше, якщо є велика кількість вхідних даних, то потрібно записувати відповідну кількість команд присвоєння. По-друге, для повторного розв'язування задачі з іншими даними потрібно вносити зміни в усі команди присвоєння. Цих недоліків не мають інші способи.

2. Команда INPUT. Команда уведення даних з клавіатури INPUT дає змогу не вносити змін у програму під час повторного її виконання. Вона має вигляд:

INPUT <список змінних>

У списку через кому чи крапку з комою пишемо імена змінних, значення яких задаватимемо з клавіатури, наприклад,

INPUT A, B, C, D$

Дія команди. Виконання програми тимчасово припиняється. На екрані монітора з'являється запит комп'ютера у вигляді знака запитання (?) і система чекає введення даних. Користуючись клаві-

атурою, набираємо список значень, розмежовуючи їх комами, і натискаємо на клавішу вводу.

Приклад 2. Розглянемо дію попередньої команди INPUT. Відповімо на запит комп'ютера таким чином:

? 5, 10, –2.5, "Україна" (Натискаємо на клавішу вводу).

Змінна A отримає значення 5, B – значення 10, C – значення –2.5, а D$ – значення "Україна". Лапки в текстових сталих можна не писати.

Для уникнення помилок під час уведення даних з клавіатури в команді INPUT можна використовувати текстову підказку:

INPUT "Введіть A, B, C, ТЕКСТ"; A; B; C; D$.

Тепер комп'ютер підкаже, що потрібно ввести. Запит матиме такий вигляд: Введіть A, B, C, ТЕКСТ?

Знак запитання тут зайвий. Щоб він не висвітлювався, потрібно використати кому замість крапки з комою після текстової підказки в команді INPUT, а саме:

INPUT "Введіть A, B, C, ТЕКСТ", A; B; C; D$.

Отже, за допомогою команди INPUT значення змінних вводять у відповідь на запит комп'ютера, тобто *у режимі діалогу користувача із системою.*

3. Команда READ (Qbasic). Команда READ надає значення змінним з блоку даних (блок розглянемо нижче). Вона має вигляд

READ <список змінних>

Список складається з імен змінних, записаних через кому, наприклад:

READ A,B,C,D$.

Значення змінним надають з блоку даних.

Блок даних – це список даних (наприклад: 2, 3, 4), значення яких присвоюють відповідним змінним зі списку команди READ. Блок даних потрібно задати. Для цього використовують команду оголошення блоку даних, яка має вигляд

DATA <список даних>

Команда DATA не виконує активної дії. Вона є описовою. У список заносимо числові та текстові дані, наприклад:

DATA 5, 10, –2.5, "Україна".

Взаємодія команд READ-DATA. Змінним зі списку команди READ надають відповідні значення зі списку DATA.

Наприклад, взаємодія команд READ–DATA з наведених прикладів дає такий же результат, як у випадку застосування чотирьох команд присвоєння чи команди INPUT, описаних раніше, а саме: змінна A отримає значення 5, B – значення 10, C – значення –2.5, D$ – значення "Україна".

Потрібно дотримуватися таких правил.

1. Типи відповідних елементів у обох списках повинні збігатися.

2. Команда READ може читати дані з декількох списків DATA і навпаки: декілька команд READ можуть читати дані з одного списку, починаючи з того місця, де закінчила читати попередня команда READ. Важливо, щоб кількість змінних у команді READ не була більшою від кількості даних в усіх командах DATA.

Блок даних, якщо потрібно, можна використати повторно. Для цього його треба відновити, тобто підготувати до повторного використання ***командою відновлення блоку даних***, яка має вигляд

```
RESTORE
```

Дія команди. Дані з уже використаного блоку даних стають знову доступними для надання значень іншим змінним.

Приклад 3. Розглянемо такі команди:

```
READ A, B, C, D$
DATA 5, 10, –2.5, "Україна"
RESTORE
READ A1, B1
READ C1, F$
```

Яких значень набудуть нові змінні: A1, B1, C1, F$? Переконайтеся, що A1 = 5, B1 = 10, C1 = –2.5, F$ = "Україна".

Використання команд READ – DATA забезпечує універсальність програми. У випадку повторного виконання програми з іншими значеннями вхідних даних зміни вносяться тільки в списки даних, які можна згрупувати для зручності на початку програми.

4. Діалогове введення даних у VB і VB .NET. У VB команда введення даних, що відповідає команді INPUT A, має вигляд

```
A = InputBox(<підказка>,<заголовок вікна>,<значення за замовчуванням>,<X-коорд. вікна>,<Y-коорд. вікна>)
```

У функції InputBox() усі параметри, крім першого, необов'язкові. Коротка форма команди така:

```
A = InputBox(<підказка>).
```

Дія команди. З'явиться діалогове вікно, в якому треба ввести потрібне значення або залишити без змін значення за замовчуванням і натиснути кнопку ОК. Якщо координати вікна не задані, то вікно з'являється посередині екрана. Вікно може мати заданий заголовок або заголовок поточного проекту. Наприклад,

```
A = InputBox("Введіть значення A")
B = InputBox("Введіть значення B", "Трикутник", 10)
D$ = InputBox("Введіть назву країни",, "Україна").
```

Якщо в середині списку пропускаємо необов'язковий параметр, то відповідна кома залишається у списку.

Запитання та вправи

1. Назвати три способи надання значень змінним. Які їхні переваги та недоліки?
2. Яким чином після знака запитання треба записати числа під час виконання команди INPUT A, B, C, якщо змінним потрібно надати такі значення:
   а) A=2, B=3, C=−4; б) A=0, B=253, C=−241; в) A=0,0027, B=−125, C=8;
   г) A=3 , B=$-2\cdot10^{7}$, C=−30,2; д) A=$-1{,}2\cdot10^{-6}$, B=8 , C=125,62?
3. Яких значень набудуть змінні в результаті виконання команд:

   а)
   ```
   READ A1,A2
   DATA 3,15
   ```

   б)
   ```
   READ A,B
   READ P,C
   DATA 21,7,5,1
   ```

   в)
   ```
   READ A,B
   RESTORE
   READ C,P
   DATA 33, 35
   DATA −17, −8
   ```

   г)
   ```
   DATA 1,2,3,4,5
   READ A,B,C
   RESTORE
   READ A1,A2,A3
   ```

   д)
   ```
   READ A
   DATA 5,2
   RESTORE
   READ B,C,D
   DATA 3,5 ?
   ```

4. Яких значень набудуть змінні в результаті виконання команд:

   а)
   ```
   READ A,B
   READ P,C
   DATA .3E1, .7E2, 5., 1.E−1
   ```

   б)
   ```
   READ A1,A2,A3
   DATA .1E−1, .1E1, .25E3
   ```

   в)
   ```
   READ A
   DATA 5.2, 2.5
   RESTORE
   READ B, C, D
   DATA 3,5
   ```

   г)
   ```
   DATA 1,"СИР",3,4,"МАСЛО",5
   READ A,B$,C
   RESTORE
   READ A1,A2$,A3
   READ A1,A2$,A3 ?
   ```

§ 5. ВИВЕДЕННЯ РЕЗУЛЬТАТІВ

Концепцію виведення результатів для середовищ VB та VB .NET розглядатимемо в розділах 3 і 4.

1. Команда PRINT. Для виведення результатів виконання програми на екран у Qbasic або на форму у VB використовують команду

PRINT <список виведення>

Елементами списку виведення можуть бути числа, сталі, змінні або вирази. Їх відокремлюють комами або крапками з комами.

Дія команди.

1. На екран виводяться відповідні значення сталих, змінних та арифметичних виразів.

2. Якщо елементи списку відокремлені крапкою з комою, то числові значення виводяться через один пропуск, а текстові – без пропуску. Додаткові пропуски забезпечує функція SPC (<кількість пропусків>), якщо її зазначити у списку.

3. Якщо елементи списку відокремлені комами, то для кожного

значення надається 16 позицій, які називають зоною. В одному рядку виведення може бути до п'яти зон. Дані вирівнюються до лівого краю зони.

4. Під час виведення числових сталих одна позиція надається знаку числа. Знак плюс не виводиться. Його замінює пропуск.

5. Якщо списку нема, то виконується пропуск одного рядка.

6. Наступна команда PRINT виводитиме інформацію в наступному рядку. Але якщо список попередньої команди закінчується комою чи крапкою з комою, то наступна команда PRINT буде виводити дані в тому ж рядку.

7. Текстові сталі у списку використовують для виведення текстів, оформлення результатів, побудови таблиць. Крапку з комою після текстової сталої можна не писати.

8. Якщо у списку стоять дві коми підряд, то пропускається одна зона, три коми – дві зони і т. д.

Довідка 1. Команда виведення даних на пристрій друку має вигляд

LPRINT <список виведення>

Замість службового слова PRINT використовують також символ запитання (?).

Розглянемо приклади застосування команди виведення та їх відповідні зображення на екрані монітора (зона має 16 позицій). Символів, які позначають пропуски, на екрані не видно.

PRINT A, B	5	10
? A; B, –2	5 10	–2
PRINT "A="; A,"B="; B–20	A= 5	B=–10
PRINT "Таблиця 1"	Таблиця 1	
PRINT "*** ***"	*** ***	

2. Діалогові програми. Команди PRINT та INPUT дають змогу складати діалогові програми. Розглянемо програму.

```
' Програма Діалог з Електронікою
CLS
INPUT "ЯК ТЕБЕ ЗВАТИ"; A$
PRINT "ПРИВІТ,"; A$
PRINT "МЕНЕ ЗВАТИ ЕЛЕКТРОНІКА"
INPUT "СКІЛЬКИ ТОБІ РОКІВ"; N
PRINT "Я МОЛОДША ВІД ТЕБЕ"
PRINT " ДАВАЙ ДРУЖИТИ!"
END
```

Під час виконання програми на екрані з'явиться запитання:

ЯК ТЕБЕ ЗВАТИ?

Слід набрати своє ім'я й натиснути на клавішу вводу. Комп'ютер привітається, відрекомендується і задасть запитання

СКІЛЬКИ ТОБІ РОКІВ?

Уведіть відповідне число. Після цього виведеться наступний текст, що імітуватиме діалог.

Завдання. Додайте до програми кілька запитань на свій розсуд.

Команда CLS. Команда CLS очищує з екрана монітора отримані раніше результати. Вона має такий вигляд:

```
CLS
```

Команда LOCATE. Якщо якесь повідомлення потрібно вивести у зазначеному місці екрана, то перед командою PRINT використовують команду

```
LOCATE A, B
```

Ця команда розміщує курсор у рядку екрана з номером A та стовпці з номером B. Усіх рядків є 25, стовпців – 80. Відлік ведуть від верхнього лівого кута екрана.

Пауза. Для організації паузи під час роботи програми використовують команду

```
SLEEP(n)
```

Вона затримує виконання програми на *n* секунду. Крім того, можна використати "холосту" команду введення, наприклад: INPUT P$, де P$ – деяка змінна. Якщо паузу потрібно закінчити, то слід натиснути алфавітно-цифрову клавішу та клавішу вводу.

Функція INKEY$. Ще один спосіб організації паузи – це використання конструкції

```
<номер> P$=INKEY$: IF P$="" THEN <номер>
```

Настане пауза. Вона триватиме доти, доки не натиснути будь-яку клавішу. Текстова функція INKEY$ набуває значення натиснутої клавіші. Поки клавіша не натиснута, значенням функції є "" і команда конструкції виконуються повторно.

3. Форматне виведення числових даннх. Результати виконання програми рекомендують виводити на екрані у зручному для читання та подальшого опрацювання вигляді, тобто у потрібному форматі. З цією метою використовують *команду виведення з форматом*, яка має вигляд

```
PRINT USING <формат>; <список виведення>
```

Формат – це текстове дане, що містить *символи керування форматом виведення.* Символи бувають звичайні або спеціальні.

Для виведення числових даних використовують *спеціальні символи:* #, 0, ^, $, кома, крапка.

Символ # резервує одну позицію для цифри. Якщо число велике, то використовують групу символів # із запасом (наприклад,

###), що задає поле виведення. Поле виведення складається з поля виведення цілої частини та поля виведення дробової частини числа. На позиції зайвого # (для цілої частини числа) буде пропуск. Символ 0 позначає позицію, де обов'язково має стояти значуща цифра або число 0. Група може містити крапку, яка позначатиме місце розташування десяткової крапки.

Дія команди.

1. Ціле число або ціла частина числа вирівнюється до правого краю відведеного поля, а зайві позиції ліворуч заповнюються пропусками.

2. Розміщення дійсного числа визначається крапкою.

3. Зайві позиції після крапки заповнюються нулями.

4. Якщо дробова частина більша, ніж відповідне поле, то число округлюється.

5. Якщо ціла частина числа або знак (–) не вміщуються у відведеному полі, то формат ігнорується, виводиться символ % і все число.

6. В одному форматі можна описати одне або декілька форматних полів. Кожному форматному полю відповідає наступний елемент списку виведення.

7. Звичайні символи використовують для наочного оформлення результатів. Вони виводяться на екран у тих позиціях, які зазначені у форматі. У програмі, наведеній нижче, це такі символи:

A1 =, B1 =, A2 =, B2 =.

Приклад 1. Розглянемо виконання програми

```
' Програма  Виведення даних
A1 = 5.6426
B1 = 2064.2
A2 = 50.2
B2 = -20.4131
PRINT USING "A1=##.##B1=####.##"; A1; B1
PRINT USING "A2=##.##B2=####.##"; A2, B2
```

На екрані отримаємо (не посередині екрана, а ліворуч):

```
A1=5.64B1=2064.20
A2=50.20B2 =  -20.41
```

Завдання. Модифікуйте програму так, щоб: а) всі змінні виводились в різних рядках; 2) між значеннями A1 та B1 і між A2 та B2 були пропуски; 3) щоб змінним A1, A2, B1, B2 можна було надавати значень з клавіатури.

8. Зайві форматні поля система ігнорує. Якщо ж у списку виведення елементів більше, ніж форматних полів, то система використовує форматні поля повторно.

Приклад 2. Команда

```
PRINT USING "####"; 1350; 350, 50
```

дає на екрані такий результат: 1350 350 50

9. Кому у форматному полі використовують для відокремлення у великому числі тисяч та мільйонів.

10. Якщо в кінці форматного поля стоять чотири символи ^^^^, то на екрані до числа дописується символ E, знак порядку і порядок числа, тобто число виводиться з плаваючою крапкою.

11. Якщо на початку форматного поля стоїть один символ $, то перед числом виводиться символ грошової одиниці. Наприклад,

Команда	Результат
PRINT USING "$###,###.##"; 2115.5	$ 2,115.50

12. Щоб уникнути "небезпечних" для банківської справи пропусків між символом $ та числом, у форматі зазначають не один, а два символи $$:

Команда	Результат
PRINT USING "$$###,###.##"; 2115.5	$2,115.50.

Запитання та вправи

1. Записати результати виконання таких команд:

а)
```
T = 5
PRINT T
```

б)
```
F = 5
PRINT "F="F
```

в)
```
A = 9
B = 420 : B = 4
PRINT A * B, 5
```

г)
```
L = -2
M = 4
PRINT L, M
Y = Y + X
PRINT X; Y
```

д)
```
X = 5
Y = 12
PRINT X, Y
```

2. Що буде виведене на екран у результаті виконання таких команд:

а)
```
T = 0
L = 3*T+T+3
PRINT T; L
```

б)
```
A = 9
B = 14-A-SQR(A)
PRINT A * B
```

в)
```
L = 9
M = 4 + L * L
PRINT L, M
```

г)
```
F = 5
F = F + 7
PRINT F, 2 * F
```

д)
```
X = 5
Y = X * X + .1E2
PRINT X, Y
X = X + 1 : Y = Y + X * X
PRINT X, Y ?
```

3. Що буде виведене на екран дисплея в результаті виконання команд:

а)
```
X = 2 : Y = 3
PRINT X; -Y, X
```

б)
```
A = 5.2 : B = 8.2
PRINT "A = "; A; "B = ", B
```

в)
```
A = 5 : B = -6
PRINT "A = ", A;
PRINT "B = ", B
```

г)
```
X = 5 : Y = 7
PRINT "РЕЗУЛЬТАТИ",
PRINT "X=" X, "Y="Y; "X-Y="; X-Y
```

д)
```
A = 2 : B = 5
PRINT A, A * B, "B = " B ?
```

4. Що буде виведене на екран у результаті виконання команд:

а) PRINT USING "###.##"; 25.31; 16.5 ;

б) PRINT USING "######"; 1995; 1996; 20.4 ;

в) PRINT USING "##.## #.#"; 3.4; 2.3 ;

г) PRINT USING "$###.##"; 1.25; 5.5 ;
д) PRINT USING "$$###.##"; 1.25; 5.5 ?

5. Що буде виведене на екран, якщо A = 148.8 : B = .08:
а) PRINT USING "###.##"; 12.341; B; A ;
б) PRINT USING "###"; 1995
в) PRINT USING "A=##.##"; A; B ;
г) PRINT USING "A=#.#"; A; B ;
д) PRINT USING "A=##.## B=##.#"; A; –B ?

6. Для чого використовують кому та крапку з комою у списках виведення?

7. Як роблять і для чого потрібні паузи під час виконання програм?

§ 6. ПРОСТІ ПРОГРАМИ

1. Структура простої програми. Проста (лінійна) програма складається переважно з команд уведення–виведення даних (READ, INPUT, PRINT), команди присвоєння (LET), зупинки (STOP) та виходу (END) з програми, з команд-коментарів та ін. Ці команди називають простими. Розглянемо деякі з них.

Команду-коментар REM використовують для внесення пояснень у текст програми:

```
REM <текст пояснення>
```

або

```
' <текст пояснення>
```

Ця команда може розміщуватися у будь-якому місці програми, але в рядку має бути єдиною або останньою. Вона не впливає на виконання програми.

Команда зупинки STOP. Команда зупинки виконання програми може бути у будь-якому місці програми:

```
STOP
```

Ця команда зупиняє виконання програми і на екран виводиться повідомлення про це. Часто цю команду використовують, щоб переписати проміжні результати з екрана монітора. Виконання програми можна продовжити за допомогою команди Continue з меню середовища QBasic (або натискають на клавішу **F5**).

Команда END. Команда закінчення роботи програми має вигляд

```
END
```

Ця команда є необов'язковою у програмі. Її можна не писати.

Задача 1. Задано координати $(x_1; y_1)$, $(x_2; y_2)$, $(x_3; y_3)$ трьох вершин трикутника у площині. Обчислити медіану m_b та радіус описаного кола r.

Розв'яжемо задачу для трикутника з координатами вершин (1; 1), (2; 2), (−1; 2), які введемо за допомогою команди READ і блоку даних. Розглянемо таку програму:

```
' Програма  Ще один приклад простої програми
DATA 1, 1, 2, 2, −1,2
READ X1, Y1, X2, Y2, X3, Y3
A = SQR((X2−X1)^2 + (Y2−Y1)^2)      ' Обчислимо довжини
B = SQR((X3−X1)^2 + (Y3−Y1)^2)      ' сторін трикутника
C = SQR((X3−X2)^2 + (Y3−Y2)^2)
X = (X1 + X3) / 2                   ' Обчислимо координати
Y = (Y1 + Y3) / 2                   ' середини сторони b
Mb = SQR((X−X2)^2 + (Y−Y2)^2)       ' Обчислимо медіану Mb
P = (A + B + C) / 2                 ' Обчислимо півпериметр
S = SQR(P * (P−A) * (P−B) * (P−C))  ' Обчислимо площу
R = A * B * C / (4 * S)             ' Обчислимо радіус
PRINT " Mb ="; Mb                   ' Виведемо медіану
PRINT " R = "; R                    ' і радіус
END
```

Завдання. Модифікуйте програму так, щоб обчислити довжини медіан, проведених до сторін A та C. Виконайте програму. Які результати ви отримали? Обчисліть одну з висот трикутника. Виконайте програму для інших вхідних даних. Виконайте програму, використавши команду INPUT.

2. Команда SWAP. Обмін значеннями між двома змінними можна виконати за допомогою спеціальної команди, яка має вигляд

```
SWAP  A, B
```

Тут A та B – імена змінних, які обмінюються значеннями.

Приклад 1. У результаті виконання програми

```
A = 7 : B = 5
SWAP A, B
PRINT A, B
```

на екрані монітора будуть виведені значення 5 і 7.

Без цієї команди для обміну даними необхідно вводити проміжну змінну C. Програма виглядатиме так:

```
A = 7 : B = 5
C = A : A = B : B = C
PRINT A, B
```

Вправи

1. Що отримаємо на екрані монітора в результаті виконання програми:

а)
```
REM Задача А
LET B = 4
```

б)
```
' Задача Б
  A = 8
```

```
LET A = 5 + B ^ 2
PRINT A
END
```

```
B = A - 4
PRINT B * 2
END
```

в)
```
REM Задача В
LET B = 3
LET B = 10 + B * 2
PRINT B
END
```

г)
```
' Задача Г
B = 4
B = B * 2 + B / 4
PRINT B
END ?
```

2. Розв'язати задачу № 2 з розділу "Задачі" свого варіанта, використовуючи: а) блок даних, б) команду INPUT.

Довідка. Для розв'язування типових задач про трикутник наведемо формули обчислення деяких величин:

відстань між точками (x_1, y_1), (x_2, y_2): $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$;

координати середини відрізка: $x = (x_1 + x_2) / 2$, $y = (y_1 + y_2) / 2$,

півпериметр трикутника: $p = (a + b + c) / 2$;

площа трикутника: $s = \sqrt{p(p-a)(p-b)(p-c)}$;

висоти трикутника: $h_a = 2s / a$, $h_b = 2s / b$, $h_c = 2s / c$;

бісектриси трикутника: $W_\alpha = \frac{2}{(b+c)}\sqrt{bc(p-a)}$,

$W_\beta = \frac{2}{(a+c)}\sqrt{ac(p-b)}$, $W_\gamma = \frac{2}{(a+b)}\sqrt{ab(p-c)}$;

радіус описаного кола: $R = abc / (4s)$;

радіус вписаного кола: $r = s / p$, де a, b, c – сторони трикутника.

§ 7. РОЗГАЛУЖЕННЯ

Розгалуження – це алгоритмічна конструкція, де перевіряється умова (значення логічного виразу), і залежно від її істинності чи хибності виконується та чи інша серія команд. Вивчатимемо такі види розгалужень: 1) повне; 2) неповне; 3) вибір.

1. Логічний вираз. ***Логічний вираз*** – це засіб записування умов. Логічний вираз може набувати значення істинність або хибність. Хибному логічному виразу відповідає числове значення 0, а істинному – будь-яке інше число. Qbasic істинний логічний вираз позначає числом -1.

Логічні вирази бувають прості та складені. Простий логічний вираз – це два арифметичні вирази, з'єднані символом відношення (=, >, <, >=, <=, <>), а складений – це прості логічні вирази, з'єднані назвами логічних операцій: NOT (не), AND (і) та OR (або).

Пріоритет (порядок) виконання логічних операцій такий:

1) NOT, 2) AND, 3) OR.

Розглянемо означення логічних операцій.

Означення 1. Операція NOT A дає істинний результат, якщо логічний вираз A є хибний і навпаки.

Означення 2. Операція A AND B дає істинний результат лише тоді, коли обидва логічні вирази є істинні.

Означення 3. Операція A OR B дає істинний результат, якщо хоча би один з логічних виразів є істинний.

Приклад 1. Нехай X = 3, Y = –9. Розглянемо деякі логічні вирази та їхні значення.

Прості вирази	Значення	Складені вирази	Значення
X = 3	-1	NOT Y <= -50	-1
X < Y	0	1 < X AND X < 5	-1
7 MOD 3 = 1	-1	X > 4 OR Y < -15	0
Y \ 2 = 4	0	X > 4 OR Y > -15	-1

Подвійну нерівність 1 < X < 5 як складений логічний вираз записують так: 1 <X AND X < 5. Сукупність нерівностей вигляду x < 1; x > 5 подають так: X < 1 OR X > 5.

Довідка 1. Можна порівнювати текстові дані. Наприклад,

T$ = "ТАК", C$ <> "С" .

Перша умова справджується, якщо значення змінної T$ збігається зі словом "ТАК", а друга, якщо значення змінної C$ не збігається із символом С.

Довідка 2. Умовою може бути навіть арифметичний вираз. Умова набуває значення хибність, якщо арифметичний вираз має значення 0, і значення істинність в усіх інших випадках.

2. Команда розгалуження IF. Повна форма умовної команди. Команду розгалуження іноді ще називають умовною командою. Вона має дві форми: повну та коротку. Загальний вигляд повної команди розгалуження такий:

IF <логічний вираз> THEN <серія 1> ELSE <серія 2>

Дія команди. Якщо значення логічного виразу істинне, то виконується серія 1, якщо воно хибне, то виконується серія 2. Повну умовну команду записують в одному рядку.

Приклад 2. У результаті виконання команди

IF X > 0 THEN PRINT X ELSE PRINT –X

на екрані отримаємо значення X, якщо X додатне, або –X в протилежному випадку.

Приклад 3. Обчислення складеної функції

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

і виведення результатів програмуємо так:

```
INPUT X
IF X >= 0 THEN Y = SIN(X) ELSE Y = COS(X)
PRINT X, Y
```

3. Блок IF – END IF. Повне розгалуження можна реалізувати за допомогою блокової конструкції IF-END IF так (потрібно дотримуватися такого шаблону багаторядкового написання):

```
IF <умова> THEN
        <серія 1>
ELSE
        <серія 2>
END IF
```

Прикладу 3 відповідає такий запис команд:

```
INPUT X
IF X >= 0 THEN
  Y = SIN(X)
ELSE
  Y = COS(X)
END IF
PRINT X, Y
```

Задача 1. Обчислити і вивести значення складеної функції y у деякій заданій користувачем точці x, якщо

$$y = \begin{cases} \ln|x|, & x < -1, \\ \sin(x), & -1 \le x < 1, \\ \cos(x), & x \ge 1. \end{cases}$$

```
REM Програма  Обчислення складеної функції
INPUT "X="; X
IF X < -1 THEN
    Y = LOG(ABS(X))
ELSE
    IF X >= -1 AND  X < 1 THEN
        Y = SIN(X)
    ELSE
        Y = COS(X)
    END  IF
END IF
PRINT X, Y
```

Завдання. Модифікуйте програму на випадок, коли функція залежить від чотирьох умов (придумайте четверту умову).

4. Коротка форма команди розгалуження. Коротка форма команди розгалуження має вигляд

```
IF <логічний вираз> THEN <серія команд>
```

Тут серія команд – це одна або декілька команд, які є в *одному* зі словом IF рядку програми. Команди в серії відокремлюють одну від одної двокрапкою.

Дія команди. Якщо значення логічного виразу істинне, то виконується зазначена серія команд. Якщо логічний вираз хибний, то серія команд ігнорується, і виконується наступний після IF рядок програми.

Приклад 4. Розглянемо команди, за допомогою яких можна: 1) вивести на екран додатне число; 2) перевірити, чи задане число N є цілим; 3) перевірити, чи число ділиться без остачі на 3:

```
IF N > 0 THEN PRINT N
IF N = INT(N) THEN PRINT "N – ціле"
IF N MOD 3 = 0 THEN PRINT "N ділиться на 3"
```

Задача 2. Скласти програму, яка для туриста дає довідку про назву столиці та кількість населення конкретної держави з такого переліку: Угорщина, Італія, Греція, Туреччина, Єгипет, Непал, Бельгія.

```
REM Програма  Столиці держав
REM Використано такі імена змінних: Kr$, St$, Nas
INPUT "Уведіть назву держави"; Kr$
IF Kr$ = "Угорщина" THEN St$ = "Будапешт": Nas = 11
IF Kr$ = "Італія" THEN St$ = "Рим": Nas = 60
IF Kr$ = "Греція" THEN St$ = "Афіни": Nas = 10
IF Kr$ = "Туреччина" THEN St$ = "Анкара": Nas = 55
IF Kr$ = "Єгипет" THEN St$ = "Каїр": Nas = 53
IF Kr$ = "Непал" THEN St$ = "Катманду": Nas = 18
IF Kr$ = "Бельгія" THEN St$ = "Брюссель": Nas = 10
PRINT "Столиця - "; St$; ", населення - "; Nas; "млн осіб"
END
```

Якщо турист зібрався штурмувати Еверест, то його діалог з комп'ютером виглядатиме так:

```
Введіть назву держави? Непал
        Столиця - Катманду, населення - 18 млн осіб.
```

Завдання. Додайте до програми дані ще про три країни.

5. Команда переходу GOTO. Для передачі керування у програмі використовують команду переходу

GOTO <номер рядка або позначка>

Дія команди. Відбувається перехід до рядка із зазначеним номером або позначеного алфавітно-цифровою позначкою.

Описану команду ще називають командою безумовного переходу. Після номера рядка може стояти двокрапка. Після алфавітно-цифрової позначки двокрапка обов'язкова.

Приклад 5. Для переходу до рядка з номером 50 чи позначкою Perehid1 використовують команду GOTO 50 або GOTO Perehid1.

Зауваження 1. Не рекомендують часто і безсистемно використовувати команду переходу, особливо до попередньої частини програми.

6. Команда умовного переходу. Команда умовного переходу є частинним випадком короткої форми команди розгалуження. Вона має вигляд

IF <логічний вираз> THEN GOTO <номер або позначка>

Дія команди. Якщо значення логічного виразу істинне, то відбувається перехід до зазначеного рядка. Якщо воно хибне, то виконується наступний рядок програми.

Службове слово THEN або GOTO можна не писати. Є дві короткі форми команди:

1) IF <умова> THEN <номер> 2) IF <умова> GOTO <номер>

Приклад 6. Наступні дві команди рівнозначні:

1) IF X > 0 THEN 50 ; 2) IF X > 0 GOTO 50.

Приклад 7. Увести з клавіатури деяке ціле число. Доки не введено 0, роботу продовжувати. Додатне число вивести на екран, в протилежному випадку вивести повідомлення "ВІД'ЄМНЕ ЧИСЛО". Програма має надавати можливість вводити та аналізувати декілька чисел, доки не буде введено число 0 – ознака закінчення роботи програми. Відповідна програма матиме вигляд:

```
' Програма  Це вже приклад циклічної програми
DEFINT A
30: INPUT "A"; A                    'або 30 INPUT "A"; A
IF A = 0 THEN finish
IF A > 0 THEN PRINT A
IF A < 0 THEN PRINT "ВІД'ЄМНЕ ЧИСЛО"
GOTO 30
finish: END
```

Висновок. За допомогою короткої умовної команди можна змінювати порядок виконання команд у програмі, організовувати розгалуження та повторення (цикли).

Запитання та вправи

1. Для чого використовують логічні вирази?
2. Що таке простий логічний вираз?
3. Які є символи відношень між величинами?
4. Що таке складений логічний вираз?
5. Які є логічні операції?
6. Дайте означення логічної операції **not**.
7. Дайте означення логічної операції **and**.
8. Дайте означення логічної операції **or**.

9. Який пріоритет логічних операцій?
10. Чи істинний простий логічний вираз X > 10, якщо:
    а) x=0 (відповідь: ні); б) x=2; в) x=10; г) x=5; д) x=15?
11. Чи буде хибним вираз X >= 10, якщо:
    а) x=1 (відповідь: так); б) x=3; в) x=10; г) x=12; д) x=25?
12. Який результат виконання програми, якщо ввести такі значення K:
    а) 3; б) 4; в) 0; г) 2; д) −2?

```
INPUT K
IF K > 2 THEN Y = K * K : GOTO 50
IF K < 2 THEN Y = K + 5 : GOTO 50
Y = K + 7
50: PRINT K, Y
END
```

13. Записати результат виконання програми, увівши такі значення Z:
    а) 0; б) 10; в) −2; г) −12; д) 12.

```
INPUT Z
20: IF Z > 10 THEN Z = Z − 25
Z = Z + 5
IF Z >= 0 THEN 20
PRINT Z
END
```

14. Записати логічні вирази для нерівностей:
    а) $0 \le x < 10$ (відповідь: X>=0 **AND** X<10); б) $-5 < x \le 8$; в) $2 < x \le 7$;
    г) $x \le 1$ або $x > 9$; д) $x \le 2$, $x > 12$; е) $x \le 0$ і $y \ge 0$.
15. Записати логічний вираз для визначення, чи точка x належить відрізку:
    а) [0; 3) (відповідь: X>=0 **AND** X<3); б) [−5; 5); в) [10; 20];
    г) [2;14] або [20;25]; д) [4;10] і [8; 12].
16. Записати умову того, що число a є: а) парне; б) ділиться без остачі на 3; в) не ділиться без остачі на 3; г) ділиться на 3 і на 5; д) ділиться на 3 або на 5.
17. Описати три змінні та запишіть умову того, що деяка особа має зріст у межах [174; 186], вік — до 20 років і мешкає у місті Львові.
18. Уведіть два числа. Менше замініть сумою цих чисел, більше — їх різницею. Вивести результати.
19. Увести два числа. Чи належить більше число проміжку [10; 20]?
20. Увести число. Вивести повідомлення: число додатне, від'ємне чи нуль.
21. Увести два числа — значення двох кутів трикутника. Вивести усі можливі повідомлення про властивості трикутника: трикутник прямокутний, гострокутний чи тупокутний, різносторонній, рівносторонній чи рівнобічний.
22. Дано трикутник зі сторонами a, b, c. Перевірити, чи виконується умова існування трикутника.
23. Дослідити, чи вантаж з габаритними розмірами a, b, c см можна перемістити через прямокутний отвір, що має розміри e та f см.
24. Розв'язати задачу 3*а* з розділу "Задачі".
25. Розв'язати задачу 3б з розділу "Задачі".

§ 8. ВИБІР

1. Команда SELECT CASE. Конструкцію *вибір* застосовують, якщо є понад два альтернативні шляхи розгалуження. У Qbasic, VB, VB .NET для цього використовують команду SELECT CASE (багатозначний вибір):

```
SELECT CASE <вираз>
  CASE <список значень 1>
       <серія 1>
  CASE <список значень 2>
       <серія 2>
  ...
  CASE ELSE
       <серія>
END SELECT
```

Дія команди. Якщо значення виразу збігається з будь-яким значенням зі списку *n*, то виконується серія команд *n*.

Списки значень в команді CASE можна задати декількома способами. Вони наведені нижче на прикладах:

```
CASE 4, 5, 6, 7          'цілі значення від 4 до 7
CASE 4 TO 7              'дійсні значення від 4 до 7
CASE IS > 0              'значення більші від 0
CASE "A" TO "Z"          '("A", "B", "C", ..., "Z").
```

Задача 1. Нехай населені пункти позначені номерами від 1 до 4. Вартість одного квитка до конкретного пункту *k* визначається так:

$$Cina = \begin{cases} 22, k = 1, \\ 25, k = 2, \\ 30, k = 3, \\ 35, k = 4. \end{cases}$$

Скільки коштуватимуть *m* квитків до населеного пункту, номер якого вводять з клавіатури?

```
REM Програма Квитки
INPUT "Введіть номер пункту та кількість квитків", K, M
SELECT CASE K
    CASE 1
      CINA = 22
    CASE 2
      CINA = 25
    CASE 3
      CINA = 30
```

```
    CASE 4
      CINA = 35
    CASE ELSE
END SELECT
PRINT M; "квитків до пункту "; K; " коштують "; M * CINA
END
```

Якщо під час виконання програми ввести дані так: 3 5, то на екрані отримаємо: 5 квитків до пункту 3 коштують 125.

Зверніть увагу на те, що в цій програмі немає команд переходу.

Завдання. Модифікуйте програму на випадок шести населених пунктів та інших цін.

Задача 2. Скласти програму для перевірки знань з географії (назв столиць і кількості населення країн).

```
' Програма  Столиці держав з CASE
10: INPUT "Введіть назву країни"; Kr$
SELECT CASE Kr$
    CASE "Угорщина"
      St$ = "Будапешт": Nas = 11
    CASE "Італія"
      St$ = "Рим": Nas = 60
    CASE "Греція"
      St$ = "Афіни": Nas = 10
    CASE "Туреччина"
      St$ = "Анкара": Nas = 55
    CASE "Єгипет"
      St$ = "Каїр": Nas = 53
    CASE "Непал"
      St$ = "Катманду": Nas = 18
    CASE "Бельгія"
      St$ = "Брюссель": Nas = 10
    CASE ELSE
      PRINT "Цієї країни немає у списку": GOTO 10
END SELECT
PRINT "Столиця - "; St$; ", населення - "; Nas; "млн осіб"
END
```

Виконаємо програму. Якщо на запит комп'ютера ввести слово Італія, то отримаємо таку довідку про цю країну:

Столиця – Рим, населення – 60 млн осіб.

Завдання. Додайте до програми дані ще про три країни.

2*. Команда ON-GOTO. Конструкцію вибір реалізують у Qbasic також за допомогою команди вибору ON-GOTO:

ON <A> GOTO <список номерів>

Тут A – *арифметичний* вираз або змінна, а список номерів має вигляд $N_1, N_2, \ldots, N_k$.

Дія команди. Розглядається значення виразу A. Якщо A набуває дійсного значення, то воно округлюється до цілого. Якщо $A = k$, відбувається перехід до рядка з номером N_k. Якщо A виходить за межі діапазону $[1; k]$, то виконується наступна після неї команда.

Приклад 1. Розглянемо команди

```
A = 2
ON A GOTO 40, 60, 80
```

Тут відбувається перехід до рядка з номером 60.

Для розв'язування задачі 1 з пункту 1 про купівлю квитків складемо програму, де використаємо описану команду.

```
REM Програма Квитки з ON-GOTO
INPUT "Введіть номер пункту та кількість квитків", K, M
ON K GOTO 40, 50, 60, 70
40: CINA = 22 : GOTO 80
50: CINA = 25 : GOTO 80
60: CINA = 30 : GOTO 80
70: CINA = 35
80: PRINT M; "квитків до пункту "; K; " коштують "; M * CINA
END
```

Зверніть увагу на роль команд GOTO 80. Вони призначені для того, щоб обійти рядки 50, 60 та 70.

Завдання. Модифікуйте програму на випадок шести населених пунктів та інших цін.

Вправи

1. Розв'язати задачу № 4 свого варіанта з розділу "Задачі".

2*. До рядка з якою позначкою відбудеться перехід, якщо виконаються команди:

а)
```
K = 2
ON K GOTO 30, 40, 50
```

б)
```
A = 10
A = A - 8
ON A GOTO 40, 50, 60
```

в)
```
10: X = 2
X = X * X - 1
ON X GOTO 10, 40, 60, 80
```

г)
```
X = 0
Y = SIN(X) + 2
ON Y + 1 GOTO 70, 80, 90
```

д)
```
A = 3
20: A = A - 2
ON A GOTO 20, 40, 60
40: A = 5
GO TO 20 ?
```

§ 9. ЦИКЛИ (1)

1. Цикли. ***Цикл*** – це процес виконання певного набору команд деяку кількість разів. Кількість повторень повинна бути скінченною.

Розрізняють цикли, де *кількість повторень відома заздалегідь*, і цикли, де вона *заздалегідь невідома*, але її можна визначити під час виконання циклу. Обчислити суму десяти членів прогресії – це задача на використання циклу з відомою кількістю повторень. У побуті прикладом циклу, де кількість повторень дій заздалегідь невідома, є процес наповнення водою діжки невідомої місткості за допомогою літрової банки.

2. Команда циклу з параметром (FOR – NEXT). Цикл з параметром (цикл «для») призначений для організації повторень, якщо їх кількість у циклі заздалегідь відома. Цикл «для» записують так:

```
FOR I = <A1> TO <A2> STEP <A3>
        <серія команд>
NEXT I
```

Команда FOR-TO-STEP утворює *заголовок циклу*, NEXT – команда, яка фіксує *кінець* тіла циклу і змінює значення параметра I на величину A3. *Тіло циклу* – це серія команд, що є між командами FOR та NEXT.

Змінну I називають *параметром* циклу. A1, A2, A3 – арифметичні вирази, змінні або сталі. A1 задає початкове значення параметра циклу, A2 – кінцеве, A3 – значення кроку, на яке щоразу змінюємо значення параметра циклу.

Дія команди. Параметрові циклу (тут I) присвоюється значення виразу A1. Якщо це значення менше-рівне значенню A2, то виконується серія команд. Після цього значення параметра збільшується на A3 і знову порівнюється зі значення виразу A2 і т.д. Коли значення параметра стане більше, ніж значення виразу A2, то виконується наступна після NEXT команда.

Приклад 1. Виведемо таблицю квадратів чисел від 10 до 25:

```
FOR I = 10 TO 25 STEP 1
  PRINT I, I^2
NEXT I
```

Приклад 2. Складемо програму для обчислення добутку чисел від 1 до 8.

Домовимося, що поточне число зберігатимемо у змінній M, а добуток нарощуватимемо у змінній D за допомогою циклу і команди D = D * M. Перед циклом змінній D присвоюємо значення 1. Переконайтеся у правильності наступної програми:

```
' Програма  Добуток з FOR
CLS
D = 1
```

```
FOR M = 1 TO 8 STEP 1
   D = D * M
NEXT M
PRINT "D = "; D
END
```

Завдання. Модифікуйте програму для обчислення суми S чисел від 1 до 100. Крім очевидних змін, використайте такі дві команди присвоєння: S = 0, S = S + M.

Команду S = 0 можна не писати, оскільки, якщо числовій змінній не надано значення, то їй присвоюється значення 0 за замовчуванням (автоматично).

Зауваження 1. Якщо A3 = 1, то в заголовку циклу частину STEP A3 можна опустити, тобто заголовок набуде вигляду

FOR I = <A1> TO <A2>

Наприклад, у програмі про добуток можна писати так:

FOR M = 1 TO 8.

Зауваження 2. Ім'я змінної після слова NEXT можна не писати.

Якщо тіло циклу коротке, то всю конструкцію циклу можна розмістити в одному програмному рядку, завдяки чому програма стає більш компактною (не забувайте лише ставити двокрапки):

FOR I = <A1> TO <A2> STEP <A3> : <серія команд> : NEXT I

Приклад 3. Цикли можна описати так:

```
FOR N = 1 TO 100 : S = S + N : NEXT N
FOR I = 1 TO 100 : PRINT I, I * I : NEXT.
```

Задача 1. Розглянемо послідовність чисел, де перші два числа – це одиниці. Кожне наступне число є сумою двох попередніх. Такі числа називають числами Фібоначчі. Визначити десять наступних чисел послідовності.

```
' Програма Числа Фібоначчі
PRINT "Числа Фібоначчі: 1 1 ";
A = 1 : B = 1
FOR I = 1 TO 10
   C = A + B
   A = B : B = C
   PRINT C;
NEXT I
END
```

Числа Фібоначчі: 1 1 2 3 5 8 13 21 34 55 89 144.

Задача 2. Побудувати таблицю відповідності між унціями та грамами, якщо 1 унція = 28,353495 г. Початкове значення кількості унцій (Un), крок зміни (Kr) цього значення та кількість рядків (N) у таблиці задати самостійно в режимі діалогу.

```
' Програма  Таблиця мір
INPUT ""; Un, Kr, N
PRINT "Унції          Грами"
FOR M = 1 TO N
   Gr = 28.353495 * Un
   PRINT Un, Gr
   Un = Un + Kh
NEXT M
END
```

Задача 3. Протабулювати дві функції $y = \sin x$, $z = \cos x$ на проміжку [0;1], змінюючи значення аргументу на 0,1. Обчислити суму, добуток і кількість додатних значень функції y.

Табулювання функції – це побудова таблиці значень функції для різних значень аргументу.

```
' Програма  Табулювання функцій
D = 1
FOR X = 0 TO 1 STEP 0.1
      Y = SIN(X)
      Z = COS(X)
      PRINT "X="; X, "Y="; Y, "Z="; Z
      IF Y > 0 THEN S = S + Y: D = D * Y: N = N + 1
NEXT X
       PRINT "S="; S, "D="; D, "N="; N
END
```

Скільки разів тут виконується тіло циклу? Переконайтеся, що 11. Кількість повторень у циклі визначають за формулою

$$n = \left[\frac{A2 - A1}{A3}\right] + 1,$$

де $[a]$ – ціла частина від числа a.

Зауваження 3. Під час виконання арифметичних операцій з дійсними числами результати округлюються. Це може призвести до того, що в циклі з дійсним параметром циклу (див. програму 12) останнє повторення не відбудеться. Тому рекомендують використовувати параметр цілого типу.

Задача 4. Протабулювати функцію $y = 3\sin(x + 2{,}6)$ на проміжку [0; 1] з кроком $h = 0{,}1$. Знайти серед обчислених значень функції найбільше та вказати аргумент, для якого це значення досягається.

```
' Програма Табулювання функцій з пошуком даних
DIM N AS INTEGER
X = 0 : XMAX = 0
MAX = 3 * SIN(X + 2.6)
FOR N = 0 TO 10
      X = 0.1 * N
      Y = 3 * SIN(X + 2.6)
      IF MAX < Y THEN MAX = Y : XMAX =X
```

```
        PRINT "X = "; X, "Y = "; Y
NEXT N
PRINT "XMAX = ";  XMAX, "MAX = "; MAX
END
```

3. Команда циклу WHILE. Розглянемо команду WHILE (поки), призначену, зокрема, для реалізації циклів з невідомою кількістю повторень. Цей цикл записують так:

```
WHILE <логічний вираз>
      <серія команд>
WEND
```

Дія команди. Серія команд виконується в циклі, поки значення логічного виразу є істинне. Істинний логічний вираз описує умову продовження циклу.

Приклад 4. Виведемо на екран таблицю квадратів чисел від 10 до 25 за допомогою команди WHILE:

```
I = 10
WHILE I <= 25
   PRINT I, I^2
   I = I + 1
WEND
```

Як і у випадку команди FOR, команда WHILE дає змогу записувати програми без використання команд переходу, що є ознакою доброго стилю програмування. На відміну від циклу «для», цикл WHILE є універсальним: його можна використати і тоді, коли кількість повторень у циклі заздалегідь невідома. Але для WHILE потрібно організовувати зміну значення параметра в тілі циклу, тоді як у циклі «для» це виконується автоматично.

Приклад 5. Розглянемо програму обчислення добутку чисел з використанням команди циклу WHILE.

```
' Програма Добуток з WHILE
CLS
N = 1
D = 1
WHILE N <= 8
   D = D * N
   N = N + 1
WEND
PRINT "D = "; D
END
```

Задача 5. Англійські слова та їхні відповідники розмістити у вигляді словника попарно в блоці даних. Скласти програму-словник, яка у відповідь на введене англійське слово зі словника виводить його переклад.

```
' Програма Англо-український словник
CLS
DATA application, прикладна програма
DATA access, доступ, append, доповнювати
DATA accessories, додатки
DATA adapter, узгоджувач, advance, удосконалювати
DATA alignment, вирівнювання, aids, засоби
DATA allocation, розміщення, arbitrary, довільний
DATA archive, архів, array, масив, assign, призначати
DATA no, Немає слова у словнику
INPUT "Введіть англійське слово"; Word$
READ Angl$, Ukr$
WHILE Word$ <> Angl$
    READ Angl$, Ukr$
    IF Angl$ = "no" THEN PRINT Ukr$: END
WEND
PRINT "Переклад – "; Ukr$
END
```

Виконаємо програму двічі. Діалог виглядатиме так:

```
Введіть слово? aids
Переклад - засоби
Введіть слово? artificial
Немає слова у словнику
```

Програма працює так: вводимо англійське слово. Зчитуємо пару слів зі словника. Порівнюємо введене слово зі словом із словника. Якщо вони різні, то зчитуємо наступну пару. І так доти, доки не прочитаємо слово «no», яке сигналізує про кінець словника; якщо одинакові, то цикл завершуємо і виводимо слово-переклад.

Задача 6. Використовуючи команду циклу WHILE, протабулювати функцію $y = \mathrm{tg}x$ на проміжку $[0; \pi]$ з кроком $h = 0{,}1$ і обчислити середнє арифметичне значень функції, більших, ніж 0,1, та менших, ніж 0,6.

```
REM Програма Табулювання функцій з пошуком даних
CLS
DIM N AS INTEGER
X = 0
S = 0: K = O
WHILE X <= 3.1415
   Y = 3 * SIN(X + 2.6)
   PRINT "X = "; X, "Y = "; Y
   N = N + 1
   X = 0.1 * N
   IF Y > 0.1 AND Y < 0.6 THEN S = S + Y : K = K + 1
WEND
IF K <> 0 THEN
```

```
      S = S / K : PRINT "S = ",S
    ELSE
       PRINT "Функція не набуває значень з інтервала"
    END IF
    END
```

Переконайтеся, що S = 0,274051.

Вправи

1. Записати результати виконання програм:

а)
```
S = 0
FOR X = 1 TO 5 STEP 2
   S = S + X
NEXT X
PRINT S
```

б)
```
D = 1
FOR A = 1 TO 5
  D = D * A
NEXT A
PRINT D
```

в)
```
S = 0
FOR X = 10 TO 0 STEP -2
   S = S + X
NEXT X
PRINT S
```

г)
```
D = 1
M = 6
FOR X = 1 TO M STEP M / 3
   D = D * X
NEXT X  : PRINT D
```

2. Записати результати виконання програм:

а)
```
S = 0 : X = 1
WHILE X <= 5
   S = S + X
   X = X + 2
WEND
PRINT S
```

б)
```
D = 1
M = 6 : X = 1
WHILE X <= M
   D = D * X : X = X + M / 2
WEND
PRINT D
```

в)
```
D = 1 : A = 2
WHILE A < 6
   D = D * A : A = A + 1
WEND
PRINT D
```

г)
```
S = 0 : X = 10
WHILE X > 0
   S = S + X : X = X - 3
WEND
PRINT S
```

3. Розглянути і виконати програму побудови графіка функції. Описати всі команди програми.

```
CLS
' Програма  Побудова графіка функції sin(x)
PRINT TAB(10);" ГРАФІК ФУНКЦІЇ SIN(X)"
PRINT TAB(6); -1; TAB(18); 0; TAB(30); 1
PRINT " - - - - - - - - - - - - - - - - - - - - - - - >Y"
FOR X=0 TO 6.3 STEP .33
    Y=SIN(X)
    PRINT USING "X=#.#|"; X;
    PRINT TAB(7+12*(1+Y)); "*"
NEXT X
END
```

4. Побудувати графіки функцій на проміжку [0; 6] з кроком 0,3:

а) $y = \cos 2x$; б) $y = e^{-x}$; в) $y = \log(x + 1)$; г) $y = \sin^2 2x$; д) $y = 2\cos x$.

5. Розв'язати задачі № 5-6 свого варіанта з розділу "Задачі".

§ 10. ЦИКЛИ (2)

У Qbasic, VB, VB .NET застосовують ще чотири різновиди багаторядкових команд циклів DO – LOOP.

1. Цикл DO WHILE – LOOP. Ця команда має такий вигляд:

```
DO WHILE <логічний вираз>
        <тіло циклу>
LOOP
```

Команду називають командою циклу з передумовою і логічним виразом, який описує умову продовження виконання тіла циклу. Вона еквівалентна команді WHILE–WEND.

Приклад 1. Добуток перших восьми чисел обчислюють так:

```
N = 1: D = 1
DO WHILE N <= 8
   D = D * N : N = N + 1
LOOP
PRINT "D = "; D
```

2*. Цикл DO UNTIL – LOOP. Ця команда має такий вигляд:

```
DO UNTIL <логічний вираз>
        <тіло циклу>
LOOP
```

Її називають командою циклу з передумовою і логічним виразом, який описує умову закінчення виконання тіла циклу.

Приклад 2. Суму перших ста чисел обчислюють так:

```
N = 1: S = 0
DO UNTIL N > 100
   S = S + N : N = N + 1
LOOP
PRINT "S = "; S
```

Зверніть увагу на те, що умови продовження виконання тіла циклу і закінчення його виконання є взаємно протилежними.

3*. Цикл DO – LOOP WHILE. Ця команда має такий вигляд:

```
DO
        <тіло циклу>
LOOP WHILE <логічний вираз>
```

Команду називають командою циклу з післяумовою і логічним виразом, який описує умову продовження виконання тіла циклу.

Приклад 3. Добуток перших восьми чисел обчислюють так:

```
N = 1: D = 1
DO
   D = D * N : N = N + 1
LOOP WHILE N <= 8
PRINT "D = "; D
```

4. Цикл DO – LOOP UNTIL. Ця команда має такий вигляд:

```
DO
      <тіло циклу>
LOOP UNTIL <логічний вираз>
```

Команду називають командою циклу з післяумовою і логічним виразом, який описує умову закінчення виконання тіла циклу.

Приклад 4. Суму перших ста чисел обчислюють так:

```
N = 1: S = 0
DO
   S = S + N : N = N + 1
LOOP UNTIL N > 100
PRINT "S = "; S
```

5. Використання команд 1–4. Практично всі задачі можна розв'язати, застосувавши перший різновид циклу, тобто цикл DO WHILE – LOOP. Він є універсальним. Зверніть увагу на відмінність між циклами з передумовою та післяумовою. У першому випадку логічний вираз може бути складений так, що тіло циклу не виконається жодного разу, а в другому, яким би не був логічний вираз, тіло циклу виконається хоча би один раз.

Деколи виникає необхідність достроково припинити виконання циклу. Для цього використовують команду EXIT <назва циклу>, наприклад EXIT FOR, EXIT WHILE, EXIT DO.

Задача 1. Серед натуральних чисел знайти перші M прості числа. Простими є числа, більші від одиниці, що діляться лише на одиницю та на себе: 2, 3, 5, 7... Число N буде простим, якщо воно не ділиться націло на всі числа від 2 до N/2.

```
' Програма Прості числа
INPUT "Скільки знайти простих чисел"; M
PRINT "Ось вони:"
N = 2                          'Це перше просте число
K = 0                          'Готуємо лічильник простих чисел
DO WHILE K < M
   pr = 1                      'Це ознака простого числа
   FOR Z = 2 TO N/2
      IF N MOD Z = 0 THEN pr = 0 : EXIT FOR
   NEXT Z
   IF pr THEN                  'Можна писати IF pr = 1 THEN
      K = K + 1
      PRINT USING "###    "; N
```

```
        IF K MOD 10 = 0 THEN PRINT 'Друкуємо по 10 у рядку
    END IF
    N = N + 1 'Розглядаємо наступне число
LOOP
END
```

Пояснення. Пошук простого числа виконується M разів за допомогою зовнішнього циклу DO. Припустимо, що число N просте (pr = 1). У циклі FOR реалізовано метод перебору, де перевіряємо, чи число N ділиться на числа Z від 2 до N/2 з нульовою остачею. Якщо так (IF N MOD Z = 0), то припущення вважаємо хибним (pr = 0) і цикл завершуємо достроково (EXIT FOR). Знайшовши просте число, виводимо його на екран у рядок так, щоб у рядку було не більше 10 чисел.

Завдання. Удоскональте програму, щоб не перевіряти парних чисел, оскільки вони завідомо не можуть бути простими.

Вправи

1. Обчислити суму парних чисел від 2 до 100, використовуючи цикл DO WHILE – LOOP.
2. Обчислити добуток непарних чисел від 1 до 15, використовуючи цикл DO – LOOP UNTIL.
3. Серед перших ста цілих чисел визначити кількість парних, які є кратні числу 3.
4. Переписати дві на вибір програми з вправи 1 попереднього параграфа, використовуючи цикл DO WHILE – LOOP.
5. Переписати дві на вибір програми з вправи 1 попереднього параграфа, використовуючи цикл DO – LOOP WHILE або DO – LOOP UNTIL.
6. Вивести всі тризначні числа, сума цифр в яких дорівнює 15.
7. Скільки є «щасливих» чисел серед перших 1000 чисел?
8. Розв'язати задачі № 7-8 свого варіанта з розділу "Задачі".

§ 11. ТЕКСТОВІ ДАНІ

1. Дії з текстовими даними. Задачі числового характеру, де об'єктами є числові дані, не вичерпують сфер застосування комп'ютерів. Актуальними є задачі опрацювання текстових даних. Нагадаємо, що текстову сталу записують в лапках, наприклад, "Високий Замок". Її довжина не може перевищувати 255 символів. Сталу нульової довжини зображають так: "".

Текстові змінні описують за допомогою команди DEFSTR або до її імені дописують суфікс $, або оголошують за допомогою команди DIM. Наприклад, текстові змінні t, s, v можна оголосити та надати їм значення так:

```
DEFSTR t
DIM v AS STRING*10
```

```
t = "Київ"
s$ = ""
v = "Інформатика"
```

Запис STRING*10 означає, що змінна v може мати не більше 10 символів.

Над текстовими змінними визначені такі операції: з'єднання (конкатенації, символ + або &) та порівняння (=, <>, <, <=, >, >=). Два тексти порівнюються зліва направо до перших різних символів. "Більшим" вважається той символ, який в алфавіті розташований далі (він має більший номер у таблиці кодів ASCII). Наприклад, числові коди наступних символів зростають зліва направо: ..., 0, 1, 2, ..., 9, ..., A, B, C, ..., X, Y, Z, ..., a, b, c, .., y, z, ..., А, Б, В, ..., а, б, в, ... Числовий код символу можна отримати за допомогою функції ASC(символ). Зворотну дію виконує функція CHR$(число). Порівняння текстових даних використовують у задачах упорядкування інформації.

Приклад 1. Нехай T1$ = "Новий", T2$ = "Рік". Тоді з'єднанням цих рядків буде S$ = T1$ + " " + T2$. У результаті змінна S$ набуде значення "Новий Рік".

Приклад 2. Істинними є такі логічні вирази: "7" < "9", "A1" < "A2", "AB" > "AA", "ГАЛЯ" < "РОМАН", а хибними – "A" > "B", "B" = "Б", "СШ 15" > "СШ 91".

Довідка. Таблицю кодів ASCII «American Standard Code for Information Interchange» можна знайти в довідниках.

Введення–виведення текстів. Надати значення текстовим змінним (увести дані) можна трьома способами за допомогою таких команд: 1) присвоєння; 2) INPUT або LINE INPUT; 3) READ.

Тексти можна виводити на екран дисплея, на друкарський пристрій або у файл даних. Виведення текстових даних виконують командою PRINT.

Приклад 3. Розглянемо команди:

```
A$ = "САДОК ВИШНЕВИЙ"
DATA "КОЛО ХАТИ"
READ B$
```

Тут змінна B$ отримає значення "КОЛО ХАТИ".

Для введення текстового рядка, який містить кому, слід використати команду LINE INPUT замість INPUT.

Задача 1. Скласти програму для переведення арабських чисел у римські.

У блоці даних опишемо відповідність арабських і римських чисел, тобто алфавіт римської системи числення. Арабське число, наприклад 2325, розглядатимемо як таку суму:

1000+1000+0*900 +0*500+0*400+100+100+100+...+10+10+5,

кожному ненульовому доданку ставитимемо у відповідність римське зображення.

```
' Програма Переведення арабських чисел у римські
DATA 1000, M, 900, CM, 500, D, 400, CD
DATA 100, C, 90, XC, 50, L, 40, XL,
DATA 10, X, 9, IX, 5, V, 4, IV, 1, I
Rez$ = ""
INPUT "Введіть арабське число"; Number
50: READ Arab, Rym$  'Зчитування пари чисел з блоку
WHILE Number >= Arab
      Rez$ = Rez$ + Rym$
      Number = Number - Arab
WEND
IF Number > 0 THEN GOTO 50
PRINT "Римське число "; Rez$
END
```

Виконаємо програму. Знайдемо римське значення числа 1996. На екрані матимемо такий діалог:

```
Введіть арабське число? 1996
Римське число MCMXCVI
```

2. Текстові функції. Під час розв'язування багатьох задач потрібно вміти виділяти з текстових даних певну частину символів. Для цього, а також для інших перетворень над цими даними, використовують стандартні текстові функції. Розглянемо базовий набір функцій.

LEFT$(A$, K)	– виділяє K символів з A$, починаючи з першого
RIGHT$(A$, K)	– виділяє з A$ праворуч частину довжиною K символів
MID(A$, L, M)	– виділяє з A$ частину довжиною M символів, починаючи від символу з номером L
LEN(A$)	– визначає кількість символів тексту A$
VAL(A$)	– дане типу текст перетворює у числове
STR$(X)	– перетворює числове дане в текст
INSTR(A$, B$)	– дає номер позиції, з якої починається входження тексту B$ в текст A$
CHR$(номер)	– дає символ з таблиці ASCII з певним номером
ASC(символ)	– дає номер символу в таблиці кодів ASCII

Приклад 4. Нехай змінна A$ має значення "Юрій Новосад". Тоді наведені нижче функції дадуть такі результати:

Функція	Результат
LEFT$(A$,4)	"Юрій"
RIGHT$(A$,7)	"Новосад"

MID$(A$,6,1)	"Н"
LEN(A$)	12
VAL("124")	124
STR$(124)	"124"
INSTR(A$," ")	5
CHR$(74)	J
ASC("J")	74.

Задача 2. Визначити кількість літер "Д" у тексті A$.

```
' Програма Кількість літер Д у тексті
DEFINT K, I, S
A$ = "ЮРІЙ НОВОСАД"
K = LEN(A$)
S = 0
FOR I = 1 TO K
   IF MID$(A$, I, 1) = "Д" THEN S = S + 1
NEXT I
PRINT "S = "; S
END
```

Виконаємо програму. На екрані отримаємо S = 1.

Завдання. Модифікуйте програму так, щоб вона підраховувала в тексті кількість літер: 1) "Ю" і "О", 2) "Ю" або "О" .

Задача 3. Скласти програму для перетворення тексту за таким правилом: вилучити з деякого тексту, що буде введений з клавіатури, пропуски, коми і крапки, інші символи продублювати.

```
' Програма Перетворення тексту
DEFSTR A, B, C   'або DIM A AS STRING, B AS STRING,...
INPUT "Уведіть текст"; A
B = ""
FOR I = 1 TO LEN(A)
        C = MID$(A, I, 1)
        IF C<>"," AND C<>"." AND C<>" " THEN B=B+C+C
NEXT I
PRINT B
END
```

Завдання. У деякому тексті замініть: 1) усі букви "А" на "Б", 2*) усі букви "А" на "Б", а "Б" на "В".

Задача 4*. Перевести римські числа в арабські.

У змінній Rez формуємо результат. Приймемо спочатку Rez = 0. Розглянемо перший символ римського числа. Якщо це M, то Rez = 1000 й аналізуємо решту римських цифр. Якщо це не M, то перевіряємо, чи римське число починається з CM. Якщо так, то Rez = 900, якщо ні, то перевіряємо, чи римське число починається з D, і так далі, доки не вичерпаються усі символи римських чисел.

```
' Програма Переведення римських чисел в арабські
```

```
DATA 1000, M, 900, CM, 500, D, 400, CD
DATA 100, C, 90, XC, 50, L, 40, XL
DATA 10, X, 9, IX, 5, V, 4, IV, 1, I
Rez = 0
INPUT "Введіть римське число"; Number$
50: READ Arab, Rym$  'Зчитування пари чисел з блоку
WHILE Rym$ = LEFT$(Number$, LEN(Rym$))
    Rez = Rez + Arab
    Number$ = RIGHT$(Number$, LEN(Number$) - LEN(Rym$))
WEND
IF Number$ > " " THEN GOTO 50
PRINT "Арабське число "; Rez
END
```

Виконаємо програму. Переведемо римське число MCMXCVI в арабське. На екрані матимемо такий діалог:

Введіть римське число? MCMXCVI
Арабське число 1996

Довідка 1. Конструкція MID$(A$,L,M) призначена також для зміни поточного значення текстової величини за допомогою такої команди присвоєння:

```
MID$(A$, L, M) = C$
```

Дія команди. M перших символів текстового даного C$ заміщують M символів у тексті A$, починаючи з позиції L. Наприклад, якщо значенням D$ є "САДОК", то команда MID$(D$, 2, 2) = "ОР" змінює значення змінної D$ з "САДОК" на "СОРОК".

Довідка 2. Кількість і способи запису стандартних текстових функцій у різних версіях мови Бейсик різні (див. довідники).

Довідка 3. Якщо потрібної стандартної функції немає, то користувач може створити текстову функцію за допомогою команди

```
DEF FNім'я$ (<аргументи>) = <текстовий вираз>
```

3. Криптографія. Історія людства — це історія становлення інформатики. Найбільш поширеною формою подання інформації є текстова. Вона передається від джерела до споживача в закодованій формі. Алфавіти української чи англійської мови — це відкриті (відомі усім) коди. Однак з давніх часів люди намагалися захищати важливу інформацію від несанкціонованого використання, створюючи закриті коди. Так виникла ***криптографія*** — наука про способи перетворення інформації (шифрування) з метою її захисту. У Давньому Римі застосовували шифр Цезаря: кожну літеру тексту замінювали на третю після неї літеру алфавіту, записаного по колу. Слово «Цезар» писали так: «Щзйгу». Такий шифр тепер називають простим шифром зі зміщенням три.

Розкриття шифрів називають ***криптоаналізом.*** Значним досягненням англійських учених під час війни і першим ефективним за-

стосуванням електронних машин було розшифрування у 1943 р. німецьких стратегічних шифрів за допомогою машини «Колоссус».

Різні способи захисту інформації використовують і в наш час, зокрема, в банківській справі, де через комп'ютерні мережі пересилають контракти, платіжні документи, підписи тощо.

Задача 5. Деяке повідомлення закодуйте простим шифром зі зміщенням 4.

```
' Програма Шифр Цезаря
A$ = "Юстас – Цетру і т.д."
B$ =""
FOR I = 1 TO LEN$(A$)
    B$ = B$ +CHR$(ASC(MID$(A$,I,1)+4))
NEXT I
PRINT B$
```

Завдання. 1. Доповніть програму блоком розшифрування повідомлення B$. 2. Переробіть програму, застосувавши замість конструкції B$ = B$ +... команду MID$.

4. Час і дата. Стандартна змінна TIME$ містить поточний час у вигляді "година:хвилина:секунда", наприклад,

"15:35:26" або "00:00:00".

Задати новий час можна за допомогою команди присвоєння

```
T$ = "12:10:00"
TIME$ = T$.
```

Стандартна змінна DATE$ містить поточну дату у вигляді "місяць/день/рік": "03/06/2007" – 6 березня 2007 року.

Дату можна змінити командою присвоєння: DATE$ = "12/31/2007".

Будь-яку компоненту дати або часу можна визначити за допомогою стандартних текстових функцій.

Приклад 5. Числовій змінній M надамо значення поточної хвилини так: M = VAL(MID$(TIME$,4,2)).

Довідка 4. Стандартна числова змінна TIMER містить кількість секунд, які минули після опівночі. Вона дає змогу вимірювати тривалість виконання різних фрагментів програми.

Вправи

1. Дано значення текстової змінної A$ = "ЛІТЕРАТУРА". Визначити значення текстових функцій:
   а) LEFT$(A$, 6); б) RIGHT$(A$, 3); в) MID$(A$, 3, 5);
   г) LEN(A$); д) MID(A$, 3, 3) + MID(A$, 8, 1).
2. Виконати попередню вправу, якщо A$ = "СУСПІЛЬСТВО".
3. Дано дві текстові величини: C$ = "ГАЛЯ 1986 СШ 75" та D$ = "РОМАН 1985 СШ 91". Обчислити значення функцій:
   а) LEFT$(C$, 5); б) VAL(MID$(C$, 7, 4)); в) MID$(D$, 7, 4);
   г) LEN(D$); д) RIGHT$(C$, 2).

4. Визначити текстове значення: а) поточної секунди; б) поточної години; в) поточного місяця; г) поточного дня; д) поточного року.
5. Обчислити числове значення:
а) поточної секунди; б) поточної години; в) поточного місяця; г) поточного дня; д) поточного року.
6. Розв'язати задачі № 9-10 свого варіанта з розділу "Задачі".

§ 12. МАСИВИ

1. Масиви. ***Масив*** – це впорядкований скінчений набір даних одного типу, які зберігаються в послідовно розташованих комірках оперативної пам'яті та мають спільну назву, яку дає користувач.

Масив складається з *елементів*. Кожний елемент має індекс, за яким його можна знайти в масиві. Кількість індексів елемента визначає розмірність масиву. Ми вивчатимемо *одновимірні* (з одним індексом) та *двовимірні* (з двома індексами) масиви. У математиці поняттю масив відповідають поняття вектора та матриці.

Важливою характеристикою масиву є його *розмір* – загальна кількість елементів у масиві.

2. Одновимірні масиви. Елемент масиву позначають іменем масиву, а *у круглих дужках* пишуть значення індекса елемента. Наприклад, k елементів одновимірного масиву А у програмах позначають так:

A(1) A(2) ... A(K).

За допомогою одновимірних масивів в оперативній пам'яті можна зберігати такі дані: 1) список учнів (масив з текстовими елементами); 2) оцінки у класному журналі за контрольну роботу (масив з числовими елементами); 3) координати вектора (масив з елементами числового дійсного типу).

Приклад 1. Опишемо у блоці даних шість чисел (виграшні номери деякої гри) і занесемо їх у масив з іменем А:

```
DATA 23, 17, 35, 6, 41, 22
READ A(1), A(2), A(3), A(4), A(5), A(6)
```

3. Двовимірні масиви. Деякі дані зручно наводити у вигляді прямокутної таблиці, якій у Бейсику відповідає поняття двовимірного масиву. Розглянемо прямокутну таблицю В:

$$\begin{matrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{2n} \\ \dots & & & \\ b_{m1} & b_{m2} & \dots & b_{mn} \end{matrix}$$

Елементи двовимірних масивів мають два індекси. Перший індекс вказує на номер рядка, а другий – на номер стовпця, на перетині яких розміщений елемент. Індекси записують у круглих дужках і відокремлюють комою,

Елементи двовимірного масиву В позначають так:

B(1, 1) B(1, 2) ... B(1, N)

B(2, 1) B(2, 2) ... B(2, N)

...

B(M, 1) B(M, 2) ... B(M, N),

де B(1, 1) – елемент, розташований на перетині 1-го рядка та 1-го стовпця; B(5, 10) – елемент на перетині 5-го рядка і 10-го стовпця; B(K, L) – елемент на перетині K-го рядка і L-го стовпця.

За допомогою двовимірних масивів в оперативній пам'яті можна зберігати такі дані: 1) текст на сторінці підручника, де кожне слово (символ) розглядаємо як один елемент (масив з елементами текстового типу); 2) оцінки учнів у класному журналі з одного предмета за певний час навчання; 3) прямокутні таблиці з тільки числовими або тільки текстовими даними, які стосуються роботи підприємств чи організацій.

Використання масивів забезпечує наочність у роботі з даними та економію імен. Головна перевага – це прямий доступ до будь-якого елемента масиву, якщо відомий його індекс.

4. Команда опису масивів. Описати масив можна на початку програми. Назви масивів та їхні розміри визначає користувач. Таким чином резервується пам'ять, де розміщуватимуться елементи масивів.

Загальний вигляд команди опису масивів такий:

DIM <список масивів>

У списку масивів зазначають імена масивів і максимальні значення відповідних індексів. Масиви перелічують через кому.

Приклад 2. Якщо в програмі використано одновимірний масив A з п'ятьма елементами і двовимірний масив B, який складається з шести рядків і десяти стовпців, то команда матиме такий вигляд (якщо відлік значень індексів починається з одиниці):

```
DIM A(5), B(6, 10).
```

Розміри заздалегідь невідомих масивів зручно задавати за допомогою команди INPUT. Наприклад:

```
INPUT "K = "; K : INPUT "M, N = "; M, N
DIM A(K), B(M, N).
```

Зауваження 1. У мові Бейсик за замовчуванням мінімальне значення індексу дорівнює нулю. Тому насправді у прикладі 2 описано одновимірний масив A з шістьма елементами та двовимірний масив B із сімома рядками та одинадцятьма стовпцями. Якщо це зумовляє плутанину, не беріть до уваги елементи з нульовим номером.

Довідка 1. Команда OPTION BASE. Можна задати потрібне мінімальне значення індексу на початку програми за допомогою команди

OPTION BASE <число>

Число задає мінімальне значення індексу, тобто 1 або 0.

Надалі вважатимемо, що мінімальне значення індексу дорівнює одиниці. Елемент масиву з нульовим індексом не братимемо до уваги.

Масиви можна оголосити із зазначенням типу елементів за допомогою повної команди DIM так:

DIM A(K) AS INTEGER, B(M,N) AS INTEGER, D(K) AS SINGLE.

Ще один спосіб оголошення призначений для явного зазначення меж зміни індекса:

DIM A(2001 TO 2007) AS INTEGER – оголошують сім елементів масиву, індекс якого має значення років від 2001 до 2007.

5. Використання одновимірних масивів. Розглянемо способи введення-виведення масивів. Для введення чи виведення масивів потрібно виконати певні дії з усіма елементами масивів за допомогою команд присвоєння, команд READ, INPUT або PRINT, застосовуючи команду циклу.

Наприклад, увести значення п'яти елементів масиву A можна двома способами:

```
1) DIM A(5)                    2) DIM A(5)
   FOR I = 1 TO 5                 FOR I = 1 TO 5
      READ A(I)                      INPUT A(I)
   NEXT I                         NEXT I
   DATA список даних
```

Задача 1. Вісім суддів виставили команді спортсменів такі оцінки: 9.2, 9.4, 9.6, 9.3, 9.5, 9.4, 9.1, 9.3. Вивести середню оцінку з точністю до трьох знаків після десяткової крапки. Скільки було оцінок, вищих від середньої?

```
' Програма Оцінки спортсменів
K = 8
DIM A(K)
DATA 9.2, 9.4, 9.6, 9.3, 9.5, 9.4, 9.1, 9.3
S = 0: K1 = 0
FOR I = 1 TO K
   READ A(I)
   S = S + A(I)
NEXT I
S1 = S / K
PRINT USING "Середня оцінка = #.###"; S1
FOR I = 1 TO K
   IF A(I) > S1 THEN K1 = K1 + 1
NEXT I
PRINT "Вищих від середньої було "; K1; " оцінок/оцінки"
END
```

Завдання. Модифікуйте програму для двох команд спортсменів. В якої з команд середня оцінка вища?

Якщо елементи масиву описані формулою, то, щоб задати (створити) масив, потрібно використати команду присвоєння.

Задача 2. У розчинах А та В є по десять компонентів. Кількість кожного компонента в кожному розчині визначається формулами

$$a_i = 1 + \sin i, \quad \text{де } i = 1, 2, \ldots, 10;$$
$$b_i = 1 + \cos i, \quad \text{де } i = 1, 2, \ldots, 10.$$

Розчини змішали й отримали суміш С. Визначити маси всіх компонентів у суміші та масу суміші С.

Алгоритм. Кількісні характеристики компонентів розчинів занесемо в масиви А та В. Створимо масив С, який є сумою масивів А і В, й обчислимо суму всіх елементів масиву С. Елементи масиву С визначають формулою $c_i = a_i + b_i$, де $i = 1, 2, \ldots, 10$.

```
' Програма Розчини
N = 10
DIM A(N), B(N), C(N)
S = 0
FOR I=1 TO N
    A(I) = 1 + SIN(I) : B(I) = 1 + COS(I)
    C(I) = A(I) + B(I)
    PRINT "C(";I;")="; C(I)
    S = S + C(I)
NEXT I
PRINT "Маса = "; S
END
```

Завдання. Модифікуйте програму так, щоб можна було вивести на екран маси кожного компонента для кожного з розчинів.

Задача 3. Створити масив у, елементи якого обчислюють за формулою $y_k = \ln k - 5$, де $k = 1, 2, \ldots, 15$. Побудувати масив g, який складається з від'ємних елементів масиву у. Вивести повідомлення у разі відсутності шуканих величин.

```
' Програма Створення нового масиву
N = 15 : K=0
DIM Y(N), G(N)
S = 0
FOR I=1 TO N
    Y(I) = LOG(I) - 5
    IF Y(I) < 0 THEN K =K + 1 : G(K) = Y(I)
    PRINT "Y("; I; ")="; Y(I)
NEXT I
FOR I = 1 TO K
    PRINT "G("; I; ")="; G(I)
NEXT I
END
```

Завдання. Модифікуйте програму так, щоб масив g складався з елементів масиву у зі значеннями з проміжку від 0,6 до 4,8.

6. Використання двовимірних масивів. Розглянемо алгоритмічну конструкцію "вкладені цикли"

```
FOR I = <I1> TO <I2>
  FOR J = <J1> TO <J2>
    <серія команд>
  NEXT J
NEXT I
```

Її використовують для введення-виведення двовимірних масивів і виконання інших операцій з його елементами.

Задача 4. На десятьох станціях (i=1, 2,..., 10) є бензин чотирьох видів (j=1, 2, 3, 4). Кількості бензину d_{ij} відомі й задані формулою

$$d_{ij} = 1 + \cos(i + j), \quad \text{де } i = 1, 2,..., 10, \ j = 1, 2,..., 4.$$

Визначити, на якій станції та якого саме бензину є найбільше.

Це задача визначення найбільшого елемента масиву та його номера.

Алгоритм. Числові дані занесемо в двовимірний масив, який складатиметься з десяти рядків і чотирьох стовпців. Щоб знайти найбільший елемент чи елемент, який задовольняє іншу умову пошуку даних в масиві, потрібно переглянути і проаналізувати всі елементи масиву. Для цієї задачі це роблять так. Найбільший елемент позначимо іменем P. Надамо йому значення першого елемента масиву P=d_{11} і порівняємо його з наступним елементом. Якщо наступний елемент більший, то його значення присвоюємо P і так далі, інакше P не змінюємо.

Використаємо вкладені цикли, де всі елементи порівнюємо з P. Значення індексу "i" змінюватимемо в *зовнішньому* циклі, а індексу "j" – у *внутрішньому*. Якщо значення елемента є більшим від P, то його приймаємо за найбільше значення за допомогою операції присвоєння P = d_{ij}. У протилежному випадку найбільше значення залишаємо попереднім. Виводимо масив у вигляді таблиці, для чого між двома командами NEXT ставимо командуPRINT.

```
' Програма Про бензин
M = 10      'Кількість рядків
N = 4       'Кількість стовпців
DIM D(M, N)
P = 1 + COS(2) : Stan = 1 : Ben = 1
FOR I = 1 TO M
  FOR J = 1 TO N
    D(I, J) = 1 + COS(I + J)
    IF D(I,J) > P THEN P = D(I, J): Ben = J : Stan = I
    PRINT USING "###.##"; D(I, J);
  NEXT J
  PRINT
NEXT I
PRINT "Найбільше - "; P; " умовних одиниць є ";
PRINT "бензину марки "; Ben; " на станції "; Stan
END
```

Завдання. Модифікуйте програму так, щоб визначити найменший елемент масиву.

Задача 5. Задано масив B, елементи якого обчислюють за формулою $b_{ij} = i + j^2$, де $i, j = 1, 2, 3, 4, 5$. Створити і вивести на екран масив y, який складається з тих елементів масиву B, значення яких більші, ніж 20. Обчислити кількість елементів масиву y. Масив B вивести у вигляді матриці 5x5. Якщо шуканих даних немає, то вивести про це повідомлення.

```
REM Програма  Пошук у двовимірному масиві
N = 5
DIM B(N, N), Y(N * N)
K = 0
FOR I = 1 TO N
    FOR J = 1 TO N
       B(I, J) = I + J * J
       IF B(I,J) > 20 THEN K = K+ 1: Y(K) = B(I, J)
       PRINT B(I, J);
    NEXT J
    PRINT
NEXT I
IF K = 0 THEN
    PRINT "У масиві B немає елементів > 20"
ELSE
    PRINT "Виведемо створений масив y"
    FOR I = 1 TO K
       PRINT "Y("; I; ")="; Y(I)
    NEXT I
END IF
END
```

Завдання. Модифікуйте програму так, щоб масив Y складався з від'ємних елементів масиву B, елементи якого обчислюють за формулою $b_{ij} = \sin(i + j)$, де $i, j = 1, 2, 3, 4$,

7. Упорядкування масиву. Задачі впорядкування (сортування) елементів масиву за деякою ознакою мають важливе практичне значення.

Задача 6. Є список імен (прізвищ тощо), який треба видрукувати за алфавітом.

Розглянемо програму Упорядкування. Нехай є сім імен, які описані в блоці даних. Занесемо імена в текстовий масив і впорядкуємо його за зростанням значень елементів. Результати виведемо на екран, зокрема, виводитимемо проміжні результати упорядкування масиву.

```
' Програма Упорядкування масиву імен
CLS
DATA  Наталка, Ростик, Тарас, Роман, Галя, Юля, Андрій
```

```
K = 7: DIM A$(K)
FOR I = 1 TO K
  READ A$(I)
NEXT I
FOR J = 1 TO K - 1
  FOR I = 1 TO K - J
    IF A$(I) > A$(I+1) THEN c$ = A$(I): A$(I) = A$(I+1): A$(I+1)=c$
  NEXT I
  FOR I = 1 TO K
    PRINT A$(I); " "; 'Проміжні результати
  NEXT I
  PRINT
NEXT J
FOR I = 1 TO K
  PRINT A$(I); " ";
NEXT I
END
```

Після виконання програми отримаємо список імен за алфавітом:
Андрій Галя Наталка Роман Ростик Тарас Юля.

Запитання та вправи

1. Чому розміри масивів рекомендують задавати за допомогою команди INPUT?
2. Скільки елементів мають нижчеописані масиви та до якого типу вони належать, якщо мінімальне значення індексу 1:
   а) DIM A%(5), B(60), C1$(2, 3);
   б) DIM A(8), B!(2, 40), C1%(5) ;
   в) DIM A#(2, 5), B%(150), C1(30);
   г) OPTION BASE 0
   DIM A(19), B%(20), C1$(21);
   д) OPTION BASE 0
   DIM A(13), B%(15), D$(17)?
3. Який масив буде створено командою FOR i=1 TO 5 : a(i)=i : NEXT?
4. Який масив буде створено командою FOR i=1 TO 6 : d(i)=i*i : NEXT?
5. Який масив буде створено командою FOR i:=1 TO 6 : b(i)=1+2*a(i), де a — масив з вправи 3?
6. Записати команду для занесення перших 10 елементів послідовності $\sin 2i$, $i=1,2,\ldots,10$ до масиву c. До якого типу належить цей масив?
7. Описати тип масиву, що міститиме список прізвищ учнів вашого класу.
8. Оголосити масив, що міститиме назви чотирьох фірм.
9. Оголосити масив, що міститиме назви днів тижня.
10. Що виконують наступні команди: FOR i=1 TO 5 : read a(i) : NEXT?
11. Що виконують наступні команди: FOR i=1 TO 5 : print a(i) : NEXT?
12. Скласти програму для занесення елементів послідовності $\cos 3i$, $i=1,2,\ldots,12$ до масиву й обчислення суми всіх елементів.
13. Скласти програму для занесення елементів послідовності $\sin 3i$, $i=1,2,\ldots,15$ до масиву й обчислення кількості додатних елементів.

14. Створити масив з кубів перших десяти чисел. Створити другий масив, елементи якого в двічі менші від першого. Який тип другого масиву? Вивести елементи цих масивів на екран у вигляді таблиці.
15. Записати команду циклу для створення такого масиву: 3, 5, 7, 9, 11.
16. Занести до масиву парні числа: 2, 4,...,24.
17. Розв'язати задачі № 11–14 свого варіанта з розділу "Задачі".

§ 13. ФУНКЦІЇ КОРИСТУВАЧА І ПІДПРОГРАМИ

1. Поняття про структурне програмування. Задачі та програми, які ми розглядаємо в цій книжці, мають навчальний характер і не є складними та великими. На практиці ж задачі, відповідні алгоритми та програми зазвичай є або складними, або громіздкими. У таких випадках варто пам'ятати про головні принципи структурного програмування.

Структурнеяпрограмування – це концепція сучасного стилю програмування, яка передбачає наступне:

1. Попередній аналіз складної задачі чи громіздкого алгоритму з метою поділу її (його) на окремі прості частини.
2. Послідовну (зверху донизу) деталізацію частин та складання підпрограм.
3. Використання трьох базових конструкцій мови (простої, розгалуження, циклу) під час складання кожної підпрограми.
4. Написання програм, зрозумілих для людей, які їх читатимуть.
5. Намагання уникати команди переходу.
6. Систему заходів перевірки правильності програми: логічний аналіз програми до її виконання, перехресна перевірка програм, колективна робота над створенням складних програм тощо.

Пояснимо деякі з цих принципів.

Деталізація зверху донизу. Щоб розв'язати складну задачу, алгоритм поділяють на частини, виокремлюють основну частину та частини нижчого рівня. Кожну частину розробляють окремо: спочатку деталізують частини верхнього рівня, а згодом – нижчого. Тому кажуть, що відбувається аналіз зверху донизу і покрокова деталізація алгоритму. Наприкінці готові частини поєднують між собою.

Метод покрокової деталізації має важливе значення не лише в інформатиці. Одна людина може керувати розробкою великого алгоритму (створенням бюджету держави, будівництвом космічної станції тощо), доручаючи багатьом виконавцям деталізацію допоміжних алгоритмів (виконання окремих робіт).

Принципом покрокової деталізації варто керуватися під час розв'язування задач з інформатики, математики та фізики, з інших предметів, а також у побуті. Про цей принцип не варто забувати в майбутньому.

Принцип покрокової деталізації відомий віддавна. Його з успіхом застосовували у воєнній справі. Проголошене стародавніми римлянами гасло "Divide and conquer" ("Поділяй і перемагай") добре засвоїли Олександр Македонський і Наполеон Бонапарт. У 1805 р. під Аустерліцом Наполеон зумів поділити могутню австрійську армію на окремі частини й по черзі розгромити їх. І хоча Наполеон був завойовником, він увійшов в історію як геніальний полководець. Сьогодні, щоб підкорити світ, не треба воювати. Це з успіхом довели великі комп'ютерні фірми, такі як Microsoft Corporation, IBM Corporation та інші, де над окремими проектами працюють сотні, а то й тисячі спеціалістів, злагоджену діяльність яких забезпечують керівники проектів, що добре засвоїли цей принцип.

Базові конструкції. Доцільно з'ясувати, які конструкції та елементи мови можна використати під час програмування кожної частини. Нагадаємо, що у розпорядженні користувача є три базові (основні) алгоритмічні конструкції: 1) проста; 2) розгалуження; 3) цикл.

Потрібно уникати команд переходу. Безсистемне використання команд переходу (особливо до попередньої частини програми), які ускладнюють читання програми, потрібно зводити до мінімуму. Цього можна досягти за допомогою таких конструкцій мови, як умовна команда IF-THEN-ELSE, команда багатозначного вибору SELECT CASE, команда циклу WHILE та підпрограм з параметрами.

Варто знати, що у сучасному Бейсику є можливість достроково вийти з деякої конструкції за допомогою таких команд виходу: EXIT IF, EXIT SELECT, EXIT WHILE, EXIT FOR, EXIT LOOP, EXIT DEF тощо.

Треба перевіряти правильність програми до її виконання.

2. Підпрограми. Підпрограма – це функціонально завершена частина програми, яку можна використати один чи багато разів для розв'язування підзадач основної задачі. Підпрограми дають змогу реалізувати концепцію структурного програмування, суть якої полягає в поділі складної задачі на послідовність простих і створенні для кожної підзадачі відповідної програми. Кожна підпрограма проектується відповідно до правил написання структурованих програм. Сукупність підпрограм утворює програму розв'язування основної задачі.

Самостійно підпрограми не виконуються. Спочатку виконується частина програми, яку називають *головною (основною) програмою.* Підпрограми є *допоміжними,* вони викликаються у процесі виконання головної програми. Після виконання чергової підпрограми знову продовжує виконуватися головна програма.

Підпрограми можуть викликати інші підпрограми. У програмі може бути будь-яка кількість підпрограм, які обмінюються значеннями не тільки з головною програмою, але й між собою.

У мові Бейсик вирізняють два види підпрограм: нестандартні функції (функції користувача) та процедури.

3. Нестандартні функції. Функції поділяють на стандартні та нестандартні. Стандартні математичні функції описані вище. Кількість стандартних функцій обмежена, але користувач має змогу створити свої власні функції. Їх називають ***нестандартними функціями*** (або ***функціями користувача***). Нестандартну функцію використовують так само, як і стандартну. Функція може повертати в точку виклику лише один результат простого типу.

Отже, *нестандартні функції* – це функції, уведені користувачем, наприклад, для обчислювання виразів.

Нестандартну функцію треба описати. Для опису нестандартної арифметичної функції є однорядкова *команда опису функції:*

DEF FN<назва>(<список формальних параметрів>) =
<арифметичний вираз>

Назва – це зазвичай одна буква, якою користувач називає функцію, і необов'язковий символ (%, #, !, $), який зазначає тип функції. Список формальних параметрів складається з однієї або декількох простих змінних, розмежованих комами. В арифметичному виразі можуть бути змінні, сталі, інші функції. Команда DEF є *описовою*. Найчастіше функції описують на початку програми.

Приклад 1. Розглянемо дві команди опису функцій:

```
DEF FNA(X) = 2 * X
DEF FNC(A, B) = SQR(A^2 + B^2).
```

Обчислення функції. Функція обчислюється в момент звертання до неї за допомогою вказівника функції, а саме:

FN<назва>(<список фактичних параметрів>)

Фактичним параметром може бути стала або змінна, арифметичний вираз або вказівник іншої функції. Вказівники нестандартних функцій використовують в арифметичних виразах як операнди.

Дія вказівника. Якщо в конкретному арифметичному виразі трапляється вказівник нестандартної функції, то система виконує такі дії:

1) обчислює значення фактичного параметра;
2) звертається до команди опису заданої функції;
3) надає формальному параметру в арифметичному виразі значення фактичного та обчислює весь вираз;
4) отримане значення присвоює вказівнику;
5) продовжує обчислення конкретного арифметичного виразу.

Приклад 2. Нехай у програмі є такі команди:

```
DEF FNA(X) = 2 * X
DEF FNC(A, B) = SQR(A^2 + B^2)
```

```
X = 0 : A = 3 : B = 4
Z = FNA(X) + FNA(A) + FNA(2)
W = FNC(A, B)
```

Взаємодія команд дасть такі результати:

$$Z = 0 + 6 + 4 = 10,$$

$$W = SQR(9 + 16) = \sqrt{25} = 5.$$

Висновок. Завдяки використанню нестандартних функцій програма користувача є більш універсальною. Зміни в арифметичних виразах можна виконати, не змінюючи головної програми. Функція дає змогу обчислити тільки одне значення, яке присвоюється вказівнику функції.

Команда DEF дає змогу описувати функції за допомогою тільки одного програмного рядка. Якщо опис функції має складатися з декількох команд і рядків, то використовують таку блочну структуру:

```
DEF FN<назва>(<список формальних параметрів>)
    <серія команд>
    FN<назва> = <арифметичний вираз>
END DEF
```

Приклад 3. Записати функцію, яка визначає більше серед двох чисел A і B

```
DEF FNA1(A, B)
    IF A > B THEN
        FNA1 = A
    ELSE
        FNA1 = B
    END IF
END DEF
```

Функцію можна описати за допомогою такої конструкції:

```
FUNCTION <назва>(<список формальних параметрів>)
    ...
    <назва> = <вираз>
    ...
END FUNCTION
```

Така функція (на відміну від попередніх) буде відокремлена від основної програми.

4. Підпрограми-процедури. Команда CALL. ***Процедура*** – це підпрограма, де розв'язується допоміжна задача. Її можна викликати один або багато разів з різних місць головної програми і використати для складання інших програм. На відміну від функції, підпрограма-процедура дає змогу обчислити та передати в головну програму не одне, а декілька значень.

Загальний опис підпрограми-процедури такий:

```
SUB <назва>(<список формальних параметрів>)
    <серія команд>
END SUB
```

У списку формальних параметрів перелічують змінні. Підпрограму викликають з головної програми за допомогою команди виклику

```
CALL < назва>(<список фактичних параметрів>)
```

Обидва списки повинні збігатися за кількістю параметрів та їхніми типами.

Дія команди. Під час виконання команди CALL значення фактичних параметрів надаються відповідним формальним параметрам. Виконується серія команд. Результати обчислень через параметри передаються у зворотному напрямку. Виконується наступна після CALL команда.

Приклад 4. Розглянемо програму Оплата, яка, використовуючи підпрограму-процедуру, визначає вартість телефонної розмови з похвилинною оплатою 0,22 грн + 20% податку.

```
' Програма Оплата за телефонну розмову
CLS
INPUT "Уведіть кількість хвилин розмови"; K
CALL CINA(K, C)
PRINT "Вам треба заплатити "; C; "грн"
END
SUB CINA(N, C)
  C = 0.22 * N
  C = C + 0.2 * C
END SUB
```

Виконайте програму. Які результати ви отримали, якщо розмова тривала 8, 13, 15 хвилин, а похвилинна оплата 55 копійок?

Завдання. Модифікуйте програму так, щоб можна було визначити вартість телефонної розмови для різних значень вартості 1 хв розмови і податку.

Задача 1. Оформити у вигляді головної програми та підпрограм програму про розчини. В одній підпрограмі (з іменем P1) виконати додавання значень елементів двох масивів, а в іншій (P2) – обчислити суму значень елементів кожного масиву. Кількість компонент у суміші – 50.

Наступна програма розв'язує цю задачу за допомогою підпрограм. Зверніть увагу на використання дужок, якщо в списках параметрів є масиви.

```
' Програма Про розчини з SUB
CLS
N = 50 'Це головна програма
DIM A(N), B(N), C(N)
FOR I = 1 TO N
   A(I) = 1 + SIN(I)
   B(I) = 1 + COS(I)
NEXT
CALL P1(N, A(),B(),C())
CALL P2(N, C(), S)
PRINT "S = "; S
END
' А це дві підпрограми
' QBasic занесе їх в окремі вікна
' Щоб їх побачити, скористайтеся пунктом
' меню VIEW...SUB
SUB P1(N, A(), B(), C())
  FOR I = 1 TO N
     C(I) = A(I) + B(I)
  NEXT
END SUB
SUB P2(N, C(), S)
  S = 0
  FOR I = 1 TO N
     S = S + C(I)
  NEXT
END SUB
```

Результат виконання програми буде такий: S = 99.64323.

Тепер підпрограми P1 та P2 можна використати для розв'язування інших задач.

Довідка 1. Стандартні функції LBOUND(A) і UBOUND(A) визначають найменше та найбільше значення індексів масиву A. Вони дають змогу не передавати розміри масивів через списки параметрів і опрацьовувати масиви з різною кількістю елементів (так звані динамічні масиви). Наприклад, підпрограму P2 оформимо так:

```
SUB P2(C(), S)
S = 0
FOR I = LBOUND(C) TO UBOUND(C)
   S = S + C(I)
NEXT I
END SUB
```

Довідка 2. Допоміжні змінні в підпрограмі мають локальний характер. Для них автоматично резервується пам'ять і їхні значення в головну програму не повертаються. Команда

SHARED <список змінних>

надає зазначеним у списку змінним підпрограми ті ж поля пам'яті, які їм надала система після "розгляду" основної програми. Тому значення таких змінних передаються в головну програму.

5. Інший спосіб виклику підпрограм. Інший спосіб виклику підпрограм полягає у використанні замість команди CALL такої конструкції виклику:

<назва> <список фактичних параметрів>

Наприклад, у програмі про розчини можна писати так:
P1 N, A(), B(), C()
P2 N, C(), S.
Опускаючи слово CALL, не пишуть також круглих дужок.

Запитання та вправи

1. Яка відмінність між стандартною та нестандартною функціями?
2. Яка відмінність між функціями та підпрограмами-процедурами?
3. Для чого використовують формальні та фактичні параметри?
4. Оформити за вказівкою вчителя раніше складені програми у вигляді головної програми та підпрограм.
5. Що буде виведене на екран монітора в результаті виконання команд:

а)
```
DEF FNA(X) = 3 * X
PRINT FNA(1)
```

б)
```
DEF FNA(X) = 2 * X + 5
PRINT FNA(1) + FNA(2)
```

в)
```
DEF FNA(X) = X + 3
A1 = FNA(0)
PRINT A1
```

г)
```
DEF FNA(B) = 3 * B - 5
Y = FNA(5)
PRINT Y, FNA(0)
```

д)
```
DEF FNA(X) = 2 * A + 6
DEF FNB(X) = 5 * X - 3
Y = FNA(5) + FNB(5)
PRINT Y, FNA(FNB(1))?
```

6. Якими будуть результати виконання команд:

а)
```
DEF FNA(X) = X * X - 3 * X
X = 4
Y = FNA(X)
PRINT X, Y, FNA(1)
```

б)
```
DEF FNX(A) = 5 * A - C
C = 1
Y = FNX(FNX(0))
PRINT Y, FNX(C)
```

в)
```
DEF FN B(X, Y)=COS(X)+5*Y
PRINT FNB(0, 1)
```

г)
```
DEF FNA(X, Z)=6*X*Z-5*Z+1
PRINT FNA(0, 1)+FNA(2,0)
```

д)
```
DEF FN A(X) = 5 * X + 6
A1 = FNA(0) : A2 = FNA(1) : A3 = FNA(2)
PRINT A1, A2, A3 ?
```

7. Скласти функцію для обчислення площі круга за заданим радіусом.
8. Скласти функцію для переведення гривень у долари за курсом НБУ.
9. Користуючись даними про відповідність мір для рідин, скласти функції для визначення одної міри через іншу, якщо: а) 1 пайп = 477,33 л; б) 1 барель (брит.) = 163,6 л; в) 1 барель (США) = 119,2 л; г) 1 галон (брит.) = 4,546 л; д) 1 галон (США) = 3,785 л; е) 1 чарка = 0,0568 л.

10. Скласти програму для побудови таблиці мір, використовуючи підпрограму, якщо: а) 1 морська миля = 1,852 км; б) 1 дюйм = 2,54 см; в) 1 миля = 1,609344 км; г) 1 кабельт = 0,183 км; д) 1 ярд = 0,9144 м; е) 1 фут = 0,3048 м.
11. Скласти підпрограму-процедуру, яка отримує три цілі числа, а повертає їхню суму, добуток і середнє арифметичне. Скласти відповідну головну програму.
12. Скласти програму з використанням підпрограм для обчислення периметра і площі двох прямокутників.
13. Розв'язати задачі № 15–17 свого варіанта з розділу "Задачі".

§ 14. ФАЙЛИ ДАНИХ

1. Основні поняття про файли. Часто виникає потреба опрацьовувати інформацію, яка зберігається на зовнішніх носіях (дисках). Прикладами такої інформації можуть бути розклад потягів, анкетні дані учнів, інформація про успішність студентів тощо. Така інформація зберігається у файлах.

Файл – це сукупність даних, що є на зовнішньому носію, яка зберігається під деяким іменем. За призначенням файли поділяють на файли даних і файли програм. Розглянемо файли даних.

Файл даних складається із записів. Записи містять різні дані, але є однорідними у файлі. Наприклад, один запис містить дані про один комп'ютер: марку комп'ютера, обсяг вінчестера, обсяг оперативної пам'яті та швидкодію. Інший запис повинен містити в такому ж порядку аналогічну інформацію про інший комп'ютер.

Приклад 1. Розглянемо три записи:

"Pentium V", 120, 1024, 3600
"Celebris", 80, 256, 1300
"Macintosh", 60, 512, 3000

Тут кожний запис складається з чотирьох *інформаційних полів*. Кожне поле містить або числове, або текстове дане. Структура записів нагадує організацію даних в команді DATA, але у файлі записи зберігаються у зовнішній пам'яті, тобто постійно. Крім цього, VB і VB .NET не підтримують конструкції вводу даних READ – DATA. Щоб програми, наведені у попередніх параграфах, працювали у VB, дані потрібно вводити з файлів, а не з блоку DATA.

Сукупність записів утворює файл. *Ім'я файлу* дають за правилами операційної системи. Воно складається з імені пристрою (диска), де є чи повинен бути файл, основного імені файлу та позначення типу (розширення) файлу. Складові відокремлюють крапкою. Ім'я пристрою чи тип можна опускати. Назвемо наш файл KOMP.DAT. У тексті програми ім'я записують у лапках: "KOMP.DAT".

Для зручності імені файлу в програмі ставиться у відповідність деяке число n – номер файлу з діапазону, наприклад, від 1 до 12.

За способом доступу до записів файли поділяють на файли даних послідовного доступу та файли даних прямого доступу.

Розглянемо головні принципи роботи з файлами даних послідовного доступу.

Щоб узяти з ящика потрібний документ, його відкривають і послідовно переглядають вміст. Знайдений документ читають. Аналогічно чинять з файлами даних. Для того, щоб опрацювати файл, його відкривають, зчитують послідовно запис за записом в оперативну пам'ять, опрацьовують записи, закривають файл. Під час створення файл відкривають, записують або дописують у нього інформацію (дані) та закривають.

2. Створення файлів. *Відкривання файлу.* Щоб відкрити файл для занесення в нього даних (з оперативної пам'яті на зовнішній носій), використовують команду

OPEN <назва файлу> FOR OUTPUT AS #n

Наприклад,

OPEN "KOMP.DAT" FOR OUTPUT AS # 1.

Тут символ #n читають як номер файлу *n*.

Для занесення даних у файл (тобто для формування запису на зовнішньому носії) використовують команду PRINT:

PRINT #n, <список даних>

Зверніть особливу увагу на оформлення списку даних в команді PRINT у програмі Створення файлу.

Після закінчення роботи з файлами з номерами n1, n2, n3 їх потрібно закрити за допомогою команди закривання, яка має вигляд

CLOSE #n1, #n2, #n3

Команда END також закриває всі відкриті файли.

Задача 1. Створити файл, який містить інформацію про комп'ютери: марку, обсяг вінчестера, обсяг оперативної пам'яті та швидкодію.

Розглянемо програму.

```
' Програма Створення файлу
OPEN "KOMP.DAT" FOR OUTPUT AS # 1
20: READ A$, B, C, D
IF A$="КІНЕЦЬ" THEN 60
PRINT #1, A$; ","; B; ","; C; ","; D
GOTO 20
60: CLOSE #1
DATA "Pentium V", 120, 1024, 3600
DATA "Celebris", 80, 256, 1300
DATA "Macintosh", 60, 512, 3000
DATA "КІНЕЦЬ", 0, 0, 0
```

Завдання. Модифікуйте програму так, щоб у файл заносилась інформація ще про два комп'ютери.

Зверніть увагу на використання ком (",") в команді PRINT #. Коми потрібні, щоб поділити записи на дрібніші частини (інформаційні поля) і забезпечити доступ до цих частин (у майбутньому) команді читання. Остання команда призначена для формування ознаки закінчення створення файлу. Замість команд READ-DATA в таких програмах у VB рекомендують використовувати аналог команди INPUT.

Відповідний файл даних можна також створити не програмним шляхом, а за допомогою текстового редактора середовища QBasic, вимкнувши контроль за правильністю рядків, або редактора Note-Pad для VB-програм. Утворений файл записують на носій під відповідним іменем.

Зауваження 1. Інший спосіб роботи із записами (без використання ком у записах) розглянемо нижче.

Довідка 1. Для виведення даних у файл з автоматичним занесенням ком між полями та для економного розташування даних на носії є команда WRITE:

WRITE #n, <список даних>

Довідка 2. Для занесення нових записів у наявний файл використовують *команду відкриття файлу для доповнення* даними (APPEND):

OPEN <назва файлу> FOR APPEND AS #n

Дані доповнюють за допомогою команд PRINT #n чи WRITE #n.

3. Використання файлів. Припустимо, що на диску вже є файл даних, який був створений раніше. Відкриємо його.

Для відкривання файлу з метою читання даних з файлу в пам'ять є команда

OPEN <назва файлу> FOR INPUT AS #n

Наприклад:

OPEN "OLD.DAT" FOR INPUT AS #2.

Тут для читання відкриваємо файл "OLD.DAT" та присвоюємо йому номер 2.

Якщо файл відкритий, то його можна опрацьовувати. Для читання даних з файлу в пам'ять використовують команду

INPUT #n, <список змінних>

Дія команди. Змінні зі списку будуть набуватимуть відповідних значень з полів запису файлу.

Вводячи дані з файлу в пам'ять, потрібно стежити, чи не вичерпалися всі записи. Щоб визначити кінець файлу, використовують команду

IF EOF(n) THEN <номер>

Функція EOF(n) набуває значення «істина», якщо досягнуто кінця файлу з номером *n*.

Задача 2. Нехай на диску є файл KOMP.DAT. Вивести вміст файлу на екран.

Розглянемо програму.

```
REM Програма  Робота з файлом
OPEN "KOMP.DAT" FOR INPUT AS #2
120: IF EOF(2) THEN 160
INPUT #2, A$, B, C, D
PRINT A$, B, C, D
GOTO 120
160: PRINT "КІНЕЦЬ ФАЙЛУ"
CLOSE #2
END
```

У результаті виконання програми на екрані дисплея отримаємо таку інформацію:

```
Pentium V     120      1024      3600
Celebris      80       256       1300
Macintosh     60       512       3000
КІНЕЦЬ ФАЙЛУ
```

Зауваження 2. Дві наведені вище програми можна розглядати як одну.

4. Пошук інформації. Розглянемо, як реалізується пошук потрібної інформації у файлі чи текстовому масиві.

Задача 3. На магнітному носії є файл послідовного доступу RIKY.DAT, створений, наприклад, за допомогою текстового редактора. Файл складається із дев'яти записів. Кожний запис – це один текстовий рядок довжиною 22 символи. Один запис містить інформацію про одну річку: перші 10 позицій відводено на назву, наступні 5 – на довжину річки у кілометрах, останні 7 – на розміри їхніх басейнів у квадратних кілометрах. Отже, запис складається з трьох полів. Поля комами не розмежовані.

Наприклад, маємо такі записи:

```
НІЛ          6671   2870000
МІССІСІПІ    6420   3238000
АМАЗОНКА     6280   6915000
ОБ           5410   2990000
АМУР         4416   1855000
ЛЄНА         4400   2490000
КОНГО        4320   3690000
НІГЕР        4160   2092000
ВОЛГА        3531   1360000
```

Завдання. Скласти програму для читання (введення) у масив з іменем A$ (в оперативну пам'ять) даних з файлу RIKY.DAT. Вивести на екран назви річок, довжина яких більша, ніж 5500 км.

Розглянемо програму.

```
REM Програма Пошук інформації
N = 9
DIM A$(N)
OPEN "RIKY.DAT" FOR INPUT AS #1
FOR I = 1 TO N
    IF EOF(1) THEN 90
    INPUT #1, A$(I)
    IF MID$(A$(I),11,4) > "5500" THEN PRINT A$(I)
NEXT I
90: END
```

Завдання. Модифікуйте програму так, щоб отримати інформації про річку з найбільшим басейном.

У результаті виконання програми на екрані з'явиться перший запис та інші, якщо такі є.

Висновки. Використання файлів має певні переваги:

1) файли дають змогу зберігати дані на зовнішніх носіях тривалий час, а не тільки під час виконання програми;

2) кількість даних може бути великою, заздалегідь невідомою, тобто під час роботи з файлами не потрібно знати кількість записів у ньому;

3) дані в записі можуть бути різного типу;

4) дані зберігаються окремо від програм, які їх опрацьовують, і можуть бути використані різними програмами та користувачами;

5) між даними різних файлів може бути певна відповідність (релятивістські зв'язки), що дасть змогу не дублювати одні й ті ж дані у різних файлах даних.

Вправи

1. Переробити програму про файли, щоб на екрані монітора отримати інформацію про обсяг оперативної пам'яті комп'ютера Celebris.
2. Переробити програму, щоб на екрані отримати назви комп'ютерів, у яких обсяг оперативної пам'яті 512 Мб. Використати сучасні дані про характеристики комп'ютерів.
3. Доповнити інформацію про комп'ютери (приклад 1) до семи найменувань із сучасними характеристиками.
4. Придумати інший спосіб контролю за закінченням створення файлу.
5. Як змінити програму, щоб інформація із записів виводилася на екран у зворотному порядку?
6. Скласти програму об'єднання двох файлів у новий файл.
7. Розв'язати задачі № 18–20 з розділу "Задачі".

§ 15. ФАЙЛИ ДАНИХ ПРЯМОГО ДОСТУПУ

1. Основні поняття. Якщо файл даних є на диску і складається із записів однакової довжини, то, знаючи номер запису, можна внести зміни в будь-який запис чи отримати дані безпосередньо з будь-якого запису. У цьому полягає суть *прямого доступу до записів* файлу. Робота з файлами даних прямого доступу складається з таких етапів:

1) відкривання файлу та описування полів запису;
2) занесення даних з оперативної пам'яті в запис файлу;
3) зчитування даних із запису файлу в оперативну пам'ять;
4) закривання файлу.

У файлі прямого доступу всі дані є текстові. Числові дані перетворюють у текстові чи навпаки за допомогою функцій STR$ чи VAL, або спеціальних функцій, які тут не розглядатимемо.

2. Відкривання файлу та описування полів запису. Нехай на диску потрібно створити файл даних прямого доступу з такою інформацією:

НІЛ	6671	2870000
МІССІСІПІ	6420	3238000
АМАЗОНКА	6280	6915000
ОБ	5410	2990000
АМУР	4416	1855000
ЛЄНА	4400	2490000
КОНГО	4320	3690000
НІГЕР	4160	2092000
ВОЛГА	3531	1360000

Опишемо структуру запису. Доступ до його полів здійснюватиметься через імена. Надамо першому полю назву F1$. Для нього достатньо зарезервувати 10 байтів (бо імена мають не більше 10 символів). Для другого поля F2$ (довжина річки) достатньо 4 байти, для поля F3$ (розмір басейну) відведемо 8 байтів. Отже, довжина запису буде 10 + 4 + 8 = 22 байти (бо всього є 22 символи). Файл можна створити за допомогою редактора. Для цього достатньо ввести з клавіатури наведений вище текст і записати файл під деяким іменем на диск.

Розглянемо, як файл створюють програмним шляхом. Надамо файлу ім'я, наприклад, RIKY.DAT і номер, нехай, 5.

Файл прямого доступу відкриваємо командою OPEN, зазначаючи довжину запису (LEN = 22), так:

```
OPEN "RIKY.DAT" AS #5 LEN = 22
```

Після цього описуємо поля командою FIELD:

```
FIELD #5, 10 AS F1$, 4 AS F2$, 8 AS F3$
```

Отже, запис файлу з номером 5 матиме потрібну (описану вище) структуру.

3. Занесення даних у запис. Для занесення даних у запис використовують команди LSET, RSET і PUT. Команда LSET вирівнює текстове дане до лівого краю поля, а команда RSET – до правого.

Приклад 1. Команди LSET і RSET заносять дані в буферні змінні F1$, F2$, F3:

```
LSET F1$ = "НІЛ"
LSET F2$ = "6671"
RSET F3$ = "2870000"
```

З буфера змінні заносять у перший запис п'ятого файлу командою PUT:

```
PUT #5, 1
```

Занесемо дані у наступний (другий) запис:

```
LSET F1$ = "МІССІСІПІ"
LSET F2$ = "6420"
RSET F3$ = "3238000"
PUT #5, 2
```

Отже, принцип такий: заповнюємо буфер і заносимо дані з буфера в запис.

Буферним змінним не можна надавати значень звичайною командою присвоєння чи командою INPUT.

4. Занесення даних із запису в оперативну пам'ять. Розглянемо файл з номером *n*. Щоб занести в буферні змінні дані із запису *m*, використовують команду GET:

```
GET #n, m
```

Приклад 2. Вивести на екран перший запис п'ятого файлу можна так:

```
GET #5, 1
PRINT F1$, F2$, F3$
```

На екрані отримаємо:

```
НІЛ       6671        2870000
```

Файли прямого доступу закриваємо командою CLOSE:

```
CLOSE #n
```

Наприклад,

```
CLOSE #5
```

Усі вищенаведені рядки утворюють програму, що ілюструє принципи роботи з файлами прямого доступу:

```
Програма  Файл даних прямого доступу
OPEN "RIKY.DAT" AS #5 LEN=22
FIELD #5, 10 AS F1$, 4 AS F2$, 8 AS F3$
LSET F1$ = "НІЛ"
LSET F2$ = "6671"
RSET F3$ = "2870000"
PUT #5,1
LSET F1$ = "MICCICIIII"
LSET F2$ = "6420"
RSET F3$ = "3238000"
PUT #5, 2
GET #5, 1
PRINT F1$, F2$, F3$
CLOSE #5
END
```

Тут відкриваємо файл прямого доступу RIKY.DAT, присвоюємо йому номер 5, заносимо у файл два записи; читаємо з файлу перший запис, виводимо його на екран монітора.

Завдання. Розв'яжіть задачі № 18, 19, 20 свого варіанта з розділу "Задачі", користуючись файлами даних прямого доступу.

§ 16. ЗАПИСИ

Запис – це структурований тип даних, призначений для зберігання в оперативній пам'яті та опрацювання даних, що складаються з даних різних типів *(полів)*. Іноді записи ще називають структурою. Запис описують за допомогою такої конструкції:

```
TYPE <назва запису>
        <назва поля 1> AS <тип поля 1>;
        ...
        <назва поля n> AS <тип поля n>;
END TYPE
```

Назви записам та полям надає користувач. Типом поля може бути будь-який зі стандартних типів (INTEGER, LONG, SINGLE, DOUBLE, STRING*n) або інший запис. Типом поля не може бути текстовий тип невідомої довжини, тобто STRING.

Приклад. Запис про анкетні дані студентів (прізвище, ім'я, дату народження та середній бал) можна описати так:

```
TYPE grupa
        name1 AS STRING*20
        surname AS STRING*15
        birthday AS TYPE
                year AS INTEGER
                month AS INTEGER
```

```
            day AS INTEGER
            END TYPE
        sball AS SINGLE
END TYPE
```

Змінну типу запис оголошують за допомогою команди DIM так:

DIM <назва змінної> AS <назва запису>

Із записів можна створити масив. Наприклад, оголосити змінну a та масив записів stud з 20 елементів типу grupa можна так:

```
DIM a AS grupa.
DIM stud(20) AS grupa.
```

Доступ до конкретного поля запису дає складене ім'я вигляду

<назва змінної>.<назва поля>

У програмі змінним a та stud(1) можна надати, наприклад, таких значень:

```
a.name1 = "Іванець"
stud(1).surname := 'Володя': stud(1).birtday.month := 5.
```

Задача. Використовуючи тип даних масив записів, скласти програму, яка б отримувала дані про наявність на складі автомашин і надавала інформацію про марки та рік випуску машин, ціна яких менша, ніж $3000. Нехай запис містить такі поля: марка, рік випуску та ціна машини. Вивести на екран інформацію про всі машини і додатково про ті, ціна яких менша, ніж 3000.

```
' Програма Автопрайс
CLS
N = 10;
TYPE avto
   marka AS STRING*15
   year AS INTEGER
   price AS INTEGER
   END TYPE
DIM a1(N) AS avto
FOR I =1 TO N
   INPUT " Введіть марку машини"; a1(I).marka
   INPUT "її рік випуску:"; a1(I).year
   INPUT "та ціну:"; a1(I). price
NEXT
PRINT
PRINT " Фірма пропонує такі машини:"
FOR I = 1 TO N
   PRINT a1(I).marka; a1(I).year; ' $'; a1(I).price
NEXT
PRINT
```

```
PRINT "Роздрукуємо інформацію про машини"
PRINT "ціна яких менша, ніж $3000"
FOR I = 1 TO N
IF a1(I).price<3000 THEN PRINT a1(I).marka; a1(I).year
NEXT
END
```

Завдання. Модифікуйте програму Автопрайс так, щоб одержати інформацію про машини, які були випущені в 90-х роках.

Запитання та вправи

1. Які типи полів мають такі записи:

а)
```
TYPE BOOKS
  NAZVA AS STRING*50
  RIK AS INTEGER
END TYPE
```

б)
```
TYPE COMPUTER
  FIRMA AS STRING*30
  RAM AS INTEGER
END TYPE
```

в)
```
TYPE anketa
imia AS STRING*15
riknar AS INTEGER
stat AS TYPE st
    m AS STRING*3
    w AS STRING*5
    END TYPE
vaga AS SINGLE
END TYPE
```

г)
```
TYPE fabryka
nasva AS STRING*45
misto AS STRIG*17
status AS TYPE
        derjav AS STRING*1
        pryvat AS STRING*1
        END TYPE
vyrib AS LONG
END TYPE ?
```

2. Як оголосити змінну типу запис?
3. Розв'язати задачу № 21 свого варіанта з розділу "Задачі".

§ 17. ГРАФІКА (1)

1. Текстовий та графічний екрани. Досі ми працювали з текстовим екраном (формою у VB), де зображалися символи і тексти.

Розміри текстового екрана у Qbasic визначають кількістю символів у *горизонтальному* та *вертикальному* напрямках. Найчастіше це 80 символів у рядку і 25 символів у стовпці.

Зображення символу виникає шляхом зміни кольору точок на ділянці екрана, де зображається один символ. Ділянка може мати різні розміри, що залежить від типу монітора. Нехай ділянка має розміри 8 на 8 точок. Кількість точок на екрані буде такою: 640 точок у рядку та 200 точок у стовпці. Екран функціонує так: у кожну точку можна спрямувати сигнал, який змінить її колір. Якщо колір точки збігається з кольором навколишніх точок, тобто з тлом, то зображення точки на екрані буде невидиме.

Екран, який дає змогу працювати безпосередньо з точками, називають ***графічним.***

Перехід від текстового (n=0) до графічного екрана виконують за допомогою команди SCREEN, а саме:

SCREEN n

Тут *n* – номер екрана, який визначає роздільну здатність. Наприклад, 2 – роздільна здатність 640x200 графічних точок, 16 кольорів, 1 відео-сторінка, 7 – 320x200 графічних точок, 2 відео-сторінки, 8 – 640x200, 9 – 640x350, 11 або 12 – 640x480 тощо.

Розташування точки на екрані визначається її координатами – парою чисел (*X*, *Y*). На рис. 1 показано розташування трьох точок. *Нумерація точок на графічному екрані починається з верхнього лівого кута* від 0 до 639 в горизонтальному напрямку і від 0 до 199 у вертикальному (для відповідного монітора). Точки на графічному екрані називають *пікселями*.

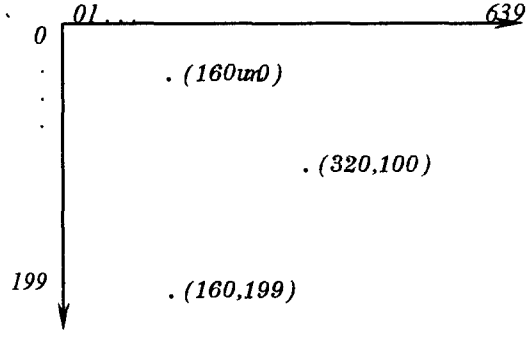


Рис. 1. Координати точок на графічному екрані

2. Кольори. Кольори в програмі *задають* командою COLOR:

COLOR A, B, C	для	$n = 0$;
COLOR A, B	для	$n = 7–10$;
COLOR A	для	$n = 12$

Тут A, B, C – числа або змінні. Значення A задає колір зображення, B – колір тла, C – колір межі.

Коди кольорів такі:

0 чорний	8 темно-сірий
1 синій	9 яскраво-синій
2 зелений	10 яскраво-зелений
3 блакитний	11 яскраво-блакитний
4 червоний	12 яскраво-червоний
5 рожевий	13 яскраво-рожевий
6 коричневий	14 жовтий
7 світло-сірий	15 білий

Приклад 1. Команда COLOR 0, 14, 3 забезпечить чорне зображення на жовтому тлі з блакитною межею *текстового* екрана.

Змінити колір точки можна за допомогою команди PSET

```
PSET (X, Y), D
```

Тут X – горизонтальна координата точки; Y – вертикальна; D – колір; X, Y, D – сталі, змінні або вирази.

Якщо кольори задані командою COLOR A, B, то засвітити точку (X, Y) кольором A (на тлі B) можна так:

```
PSET (X, Y)
```

Приклад 2. Засвітити зеленим кольором точку з координатами (50,100) можна так: PSET (50, 100), 2.

Загасити точку можна командою PSET (X, Y), B або командою

```
PRESET (X,Y)
```

Приклад 3. Загасити засвічену раніше точку (див. приклад 2) можна так: PSET (50,100), B або PRESET (50,100).

Команду PRESET використовують також для розміщення графічного курсора в стартове положення (X, Y) під час побудови рисунків.

3. Зображення прямих і прямокутників. *Для зображення (рисування) прямої* кольором D використовують команду LINE:

```
LINE (X1, Y1) – (X2, Y2), D
```

У дужках зазначають координати початку і кінця прямої. Координати початку можна опускати, якщо користувача влаштовують координати графічної точки, де є курсор. Отже, коротка форма команди має такий вигляд:

```
LINE – (X2, Y2)
```

У цьому випадку рисується (кольором A) пряма лінія від деякої поточної точки до точки з координатами (X2, Y2).

Приклад 4. З'єднати точки з координатами (50,100) та (50,150) червоною лінією можна так:

LINE (50, 100) – (50, 150),4 або LINE – (50, 150), 4.

Для зображення прямокутника використовують команду

```
LINE (X1, Y1) – (X2, Y2), D, B
```

Тут у дужках задано координати діагонально протилежних вершин прямокутника.

Щоб розмалювати весь прямокутник кольором D, замість символу B пишуть два символи BF.

Приклад 5. Нарисуємо квадрат і розмалюємо його яскраво-синім кольором: LINE (50, 100) – (100, 150), 9, BF.

Для розмальовування замкнутих ділянок (не лише прямокутників) використовують команду PAINT:

```
PAINT (X, Y), D, C
```

Тут X, Y – координати будь-якої точки, що лежить у середині замкнутої ділянки, яку розмальовують; D – колір розмальовування; C – колір межі ділянки.

Дія команди PAINT. Нехай межа ділянки (фігури) має колір C. У результаті виконання команди замкнута ділянка буде розмальована кольором D.

Приклад 6. Сонце можна намалювати так:

```
' Програма Сонце
SCREEN 7
COLOR 5, 3
CIRCLE (160, 100), 70, 4
PAINT (160, 100), 14, 4
END
```

Команда CIRCLE зображає коло. З нею ознайомимося в наступному параграфі.

Вправи

1. Показати розміщення на графічному екрані точок з координатами:
   а) (0, 0); б) (0, 125); в) (100, 125); г) (100, 125); д) (255, 191).
2. Описати дію таких команд:
   а) COLOR 5, 11; б) COLOR 1, 15; в) COLOR 3, 7;
   г) COLOR 6, 1; д) COLOR 8, 1, 8.
3. Що буде нарисовано в результаті виконання команд:

а)
```
SCREEN 7
COLOR 15, 1
FOR X=12 TO 200 STEP 4
  PSET (X, 50)
NEXT X
```

б)
```
SCREEN 8
COLOR 15, 1
LINE (10, 10) - (100, 100), 11
LINE - (100, 150), 11
LINE - (10, 10), 11
```

в)
```
SCREEN 7
COLOR 1, 11
FOR X=0 TO 300 STEP 4
 FOR Y=0 TO 199 STEP 4
  PSET(X, Y), RND(1)*16
 NEXT Y
NEXT X
```

г)
```
SCREEN 8
COLOR 7, 3
30: FOR X=10 TO 200
 PSET(X, 20)
NEXT X
FOR X=10 TO 200
 PRESET(X, 20)
NEXT X: GOTO 30 ?
```

§ 18. ГРАФІКА (2)

1. Зображення кола, дуги, сектора й еліпса. *Для зображення кола* використовують команду CIRCLE:

```
CIRCLE (X, Y), R, D
```

Тут у дужках зазначено координати центра кола; R – радіус; D – колір.

Приклад 1. Команда CIRCLE (125, 100), 50, 4 нарисує червоне коло з радіусом 50 точок і центром у точці (125, 100).

Для зображення дуги кола використовують цю ж команду з додатковими параметрами:

```
CIRCLE (X,Y), R, D, A1, A2
```

Тут A1, A2 – кути початку та кінця дуги в радіанах (1 радіан – 57 градусів, а 90 градусів – це 1.57 радіан). Відлік ведуть проти годинникової стрілки.

Приклад 2. Чверть кола можна зобразити на екрані командою

CIRCLE (125, 100), 50, 4, 0, 1.57.

Якщо A1 < 0 і A2 < 0, то попередня команда зображає на екрані комп'ютера круговий сектор, який використовують для побудови кругових діаграм.

Приклад 3. Чверть круга зображуємо так:

CIRCLE (125, 100), 50, 4, –1.57, –3.14.

Для зображення еліпса команду CIRCLE використовують з такими параметрами:

```
CIRCLE (X,Y), R, D,,, Z
```

Тут Z означає розтяг, якщо Z > 1, чи стиск, якщо Z < 1, еліпса у вертикальному напрямку. Розглянемо програму Еліпси.

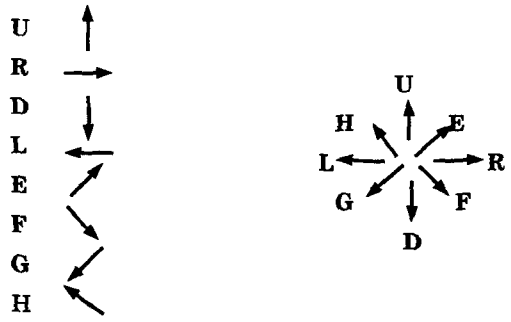
```
  Програма Еліпси
SCREEN 7      ' або 8
COLOR 3, 1
FOR I=80 TO 20 STEP –4
   D=RND(1)*16
   CIRCLE (128,96),96,D,,,10/I
   CIRCLE (128,96),96,D,,,I/10
NEXT I
END
```

У результаті виконання програми отримаємо декілька еліпсів різних кольорів. Частина їх буде розтягнута, а частина стиснута.

2*. Команда DRAW. Команду DRAW використовують (лише у Qbasic) для рисування ліній різного кольору різної довжини у восьми допустимих напрямках. Вона має вигляд

DRAW текстова стала

Тут текстова стала – це послідовність параметрів-кодів, які позначають напрямок руху. Після кожного параметра стоїть число, яке вказує довжину лінії. Параметрів, що задають напрямок руху, є вісім. Вони замальовані двома способами на рис. 2. Стрілка пояснює, в якому напрямку комп'ютер рисує лінію на екрані.



Спосіб 1 *Спосіб* 2

Рис. 2. Параметри команди DRAW

Приклад 4. За допомогою команди 30 DRAW "U30R40D30L40" можна нарисувати прямокутник, виходячи з деякої точки, заданої командою PSET, або з точки, де закінчила рисувати попередня команда, інакше рисування почнеться у центрі екрана.

Довідка. Корисними є також такі три параметри: N, B, C.

Якщо на початку рядка символів зазначити параметр N, то після рисування лінії невидимий графічний курсор повернеться у стартову точку.

Якщо на початку рядка написати B, то графічний курсор переміститься у зазначеному іншими параметрами напрямку, нічого не рисуючи.

Якщо в середині рядка символів поставити параметр C з числовим значенням кольору, то колір ліній зміниться на заданий. Цей параметр діє тільки в межах команди DRAW.

3. Імітація руху об'єкта на екрані. Для імітації руху зображення об'єкта на екрані потрібно виконати такий алгоритм:

1. Нарисувати об'єкт у заданій точці.
2. Знищити об'єкт, замалювавши його кольором тла.
3. Змінити координати об'єкта.
4. Перейти до пункту 1.

Продемонструємо цей алгоритм на прикладі програми Рух кола. На екрані зелене коло на чорному тлі рухатиметься вверх-вниз.

```
' Програма Рух кола
SCREEN 8
COLOR 2,1
Y = 10 : SY = 4
start: CIRCLE (120, Y), 40, 2    'Рисування кола
CIRCLE (120, Y), 40, 1    'Стирання кола
Y = Y + SY                'Зміна y-координати центра
IF Y < 10 OR Y > 190 THEN SY = –SY
GOTO start                'Рисування кола
```

Перервати виконання цієї програми можна акордом **Ctrl+Break**.

Зауваження. Для комп'ютерів з високою швидкодією процесора слід робити штучні паузи між рисуванням та стиранням об'єктів, щоб не втрачати їхнє правильне зображення на екрані.

Вправи

1. Описати дію команди:
   а) CIRCLE (100, 100), 40, 7; б) CIRCLE (150, 100), 40, 7, 1, 2;
   в) CIRCLE (150, 100), 40, 7, -1, -2; г) CIRCLE (100, 100), 80, 6,,, 2;
   д) CIRCLE (120, 120), 60, 5,,, 0.5.
2. Зобразити схематично слід, який рисує на екрані кожна з команд:
   а) DRAW "U20E20D20G20"; б) DRAW "L10D40R20U40L10";
   в) DRAW "U40R20D20L20"; г) DRAW "U20R10D20";
   д) DRAW "BU40C5E15C6H25C8D40".
3. Нарисувати: а) ромашку; б) робота; в) олімпійські кільця; г) східці.
4. Нарисувати годинник та імітувати рух стрілок.
5. Побудувати рухоме графічне зображення за власною уявою.
6. Розв'язати задачі № 22, 23 свого варіанта з розділу "Задачі".

§ 19. МУЗИКА

1. Команда PLAY та позначення нот. Щоб отримати музичні ефекти, використовують (у Qbasic) команду PLAY:

PLAY параметри

Вона призначена для відтворення мелодій. Ноти, висоту тону (октаву), тривалість звучання, темп звучання, рівень звуку задають кодами-параметрами, які беруть у лапки.

Ноти позначають (кодують) так:

C – до, D – ре, E – мі, F – фа, G – соль, A – ля, B – сі

Символами «+» або «#» позначають дієз, а символом «-» – бемоль. Сі-дієз, до-бемоль та мі-дієз у музичній нотації не вживають, тому комбінації B+, C- та E+ заборонені. Після назви ноти може

бути число, яке задає тривалість звучання ноти. Наприклад, C4 позначає ноту «до» четвертої октави тривалості 1/4. Якщо числове значення не задане, то діє «принцип замовчування», коли вважають, що тривалість ноти 1/4.

Для організації паузи використовують параметр P<тривалість>. Для позначення нот з крапкою пишуть крапку після відповідного параметра, що позначає ноту.

Приклад 1. Конкретні команди мають такий вигляд:

```
PLAY "CDEFGAB"  'Це є гама
PLAY "C+CP1CD-D#"
PLAY "CDEFGA.B"
```

Зауваження 1. Перед першим виконанням команди PLAY у деяких ранніх системах слід виконати команду BEEP. Вона вмикає звукову мікросхему комп'ютера та подає звуковий сигнал.

2. Октави і тривалість звучання ноти. *Октав* є сім. Їм відповідають номери n=0,1,2,...,6. Щоб задати потрібну октаву, використовують параметр O (це літера) перед кодами нот так:

On

Тут n – номер октави. Якщо октаву не зазначити, то автоматично (принцип замовчування) задається четверта октава, тобто O4.

Приклад 2. Щоб відтворити гаму на різних октавах (2, 4, 6), слід виконати команди

```
PLAY "O2CDEFGAB"
PLAY "CDEFGAB"
PLAY "O6CDEFGAB"
```

Тривалість звучання ноти задають параметром L, а саме:

Ln

Замість n слід писати такі значення для нот різної тривалості:

1 – ціла нота –
2 – половина –
4 – чверть –
8 – 1/8 –
16 – 1/16 –
32 – 1/32 –
64 – 1/64 –

За замовчуванням приймається L4, що відповідає чверті.

Приклад 3. Послідовність нот (до-мі-ре) відтворимо для трьох різних тривалостей (L4, L8, L16) двома способами:

```
PLAY "CEDL8CEDL16CED"
PLAY "C4E4D4C8E8D8C16E16D16"
```

3. Темп звучання і рівень звуку. *Темп звучання* можна задати параметром T так:

Tn

Тут *n* – *кількість* чвертей, які звучать протягом хвилини. За замовчуванням *n* = 120. Допустимий діапазон для *n*: від 32 (повільно) до 255 (швидко).

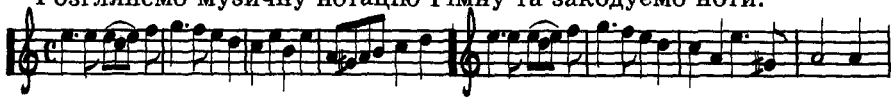
Рівень звуку регулюють параметром V (є не у всіх системах):

Vn

Тут n набуває значень від 0 (тихо) до 15 (дуже голосно). За замовчуванням приймається рівень V8.

Задача. Скласти програму для відтворення фрагмента мелодії Гімну України.

Розглянемо музичну нотацію Гімну та закодуємо ноти.



```
' Програма Фрагмент Державного Гімну України
PLAY "O5E.L8EEDEFL4G.L8FL4EDCE"
PLAY "O4BO5EO4L8AG+ABL4O5CD"
PLAY "O5E.L8EEDEFL4G.L8FL4EDC"
PLAY "O4AO5E.O4L8G+L2AL4A"
END
```

Зауваження 2. Музичні коди можна задавати малими буквами.

Вправи

1. Яка музична нотація відповідає кодам: а) "EDCE"; б) "EEDEF"; в) "O5E.L8EED"; г) "L8G+L2AL4A"; д) "BO5EO4L8AG+"?
2. Виконати і відгадати мелодії:

   а) PLAY "T150 B- D D B- L2 B- L4 D D B- B- D E- D"
   PLAY "C C C A L2 A L4 C C A A C D"

   б) PLAY "o3 a16 a16 a4 d4. a8 a#8 a8 g4. p16"
   PLAY "a#16 a#16 a#4 d4. f8 e8 d8 e4. p16"

   в) PLAY "t100 l8 a b o5 c c c c d c o4 b a l4 a e a g"
   PLAY "t100 l8 f a g+ a b a g a l4 e. p8"

   г) PLAY "l8 b. o5 c d4 d8."
   PLAY "o4 b a. b g4. l8 b. b a. g e4 l8 e. e b. b a2."

   д) PLAY "g8 f8 f2. p8 l8 a b o5 c+ d e f e. d d. p8"
   PLAY "o5 l8 ddc o4 bag b4 l8 aa a4g4f4a8g8 l4 gdfa8a8a2"
3. Запрограмувати фрагмент улюбленої мелодії. Використати його в кінці своїх програм для повідомлення про завершення їх виконання.

§ 20. СЕРЕДОВИЩЕ ПРОГРАМУВАННЯ QBASIC ТА VISUAL BASIC 6

1. Середовище програмування QBasic. Щоб запустити середовище, слід виконати команду qbasic.exe (або qbasic), що є в складі *MS-DOS* 5.0 і вище (190 Кб дискового простору). Ця програма може працювати в операційній системі Windows після налаштування її ярлика на взаємодію з Windows. На екрані з'явиться *вікно редагування* (**Untitled**) та *віконце негайного виконання* (**Immediate**), а також заставка-вітання, яку знімаємо натисканням клавіші **Esc**. Головне меню середовища складається з таких пунктів:

File (Файл) – для роботи з файлами програм;
Edit (Редагувати) – для редагування програми чи тексту;
View (Переглянути) – для перегляду текстів модулів;
Search (Шукати) – для контекстного пошуку й заміни;
Run (Виконати) – для виконання програми;
Debug (Налагодити) – для налагодження програми;
Options (Режими) – для задання додаткових режимів;
Help (Допомога) – для надання допомоги.

Щоб активізувати меню, слід натиснути клавішу **Alt**. Щоб активізувати потрібний пункт меню, його треба вибрати за допомогою стрілок переміщення курсора й натиснути клавішу вводу або відповідну висвітлену букву **F**, **E** чи **R** тощо.

Програму можна вводити зразу ж після зняття заставки, про що свідчить наявність курсора у вікні редагування. Щоб записати програму на диск, слід активізувати пункт **File** і виконати підпункт **Save** (Зберегти) або **Save as...** (Зберегти під новим іменем). Для введення нової програми треба виконати підпункт **New**, а для занесення раніше створеної програми з диска у вікно редагування – підпункт **Open...** Для роздрукування файлу користуємося підпунктом **Print...**, а для виходу із середовища – підпунктом **Exit**.

Програму можна виконати за допомогою пункту **Run**, або акорду **Shift+F5**. Припинити виконання програми можна акордом **Ctrl+Break**. Після перегляду результатів повертаємося у вікно редагування натисканням будь-якої клавіші. Щоб ще раз переглянути результати, користуємося пунктом **View** або клавішею **F4**. Для перемикання вікон використовують клавішу **F6**. Натиснувши її, можемо вводити окремі команди у вікні **Immediate**.

Система повідомляє про помилки. Помилки слід виправити, користуючись клавішами **Delete**, **Backspace**, **Insert** тощо, а також засобами пункту **Edit**. Якщо службові слова набрано малими буквами, система сама зробить їх великими. Система вигідно розташує операнди у виразах, відокремивши їх пропусками, тобто виявлятиме дружнє ставлення до користувача.

Корисними є такі засоби редагування з використанням буфера – проміжної пам'яті:

Shift+стрілки – позначення рядка чи блоку кольором;
Del – вилучення позначеного тексту (блоку);
Esc – зняття позначення;
Shift+Del – перенесення позначеного тексту в буфер;
Ctrl+Ins – копіювання позначеного тексту в буфер;
Shift+Ins – вставляння блоку з буфера в текст.

Щоб занести раніше створену програму з диска у вікно редагування, користуємося підпунктом **Open...** З'являється нове вікно. У ньому слід написати повний шлях до файлу або, скориставшись клавішею Tab, перейти в нижню частину вікна, вибрати потрібний диск, назву файлу й натиснути клавішу вводу. Не забувайте користуватися клавішею Tab для вибору потрібної дії з тих, які будуть пропонуватися у додаткових віконцях у кутових дужках <...>; а також клавішами **Esc** чи **N** для скасування дії; клавішами вводу чи **Y** для підтвердження дії; повторним виконанням пункту для ліквідації наслідків першого виконання.

Програма має зручний редактор, який можна використати для написання повідомлень, листів і особливо для створення файлів даних.

Програма має розвинуту службу допомоги. Її викликають акордом **Shift+F1** або клавішею вводу відразу після запуску програми. Допомога організована за змістом (Content), де можна знайти інформацію щодо таблиці кодів ASCII чи помилок періоду виконання програми, а також багато іншої корисної інформації. Зручною є організація допомоги за ключовими словами, згрупованими за алфавітом (Index). Тут можна ознайомитися з правилами написання і використання команд у загальному випадку, а також з прикладами.

Середовище Qbasic – це інтерпретатор, що є частковим випадком компілятора Quick Basic 4.5. Все, що було написано вище для Qbasic, справедливо для Quick Basic 4.5. Середовище Quick Basic дає змогу створювати exe-файли програм і виконувати їх незалежно від нього. Крім служби допомоги, до середовища додається велика колекція програм, зокрема, для роботи з графікою, корисних для вивчення різноманітних можливостей мови Бейсик.

Якщо на комп'ютері встановлена якась екзотична давня версія мови Бейсик, то більшість наведених у цій книжці програм нормально функціонуватимуть, якщо рядки програм пронумерувати.

2. Виконання програм у середовищі Visual Basic 6. У першому розділі описано виконання програм у середовищі Qbasic. Розглянемо, як виконати Бейсик-програму у середовищі VB. Тепер програми називатимемо проектами. Є два шляхи. Перший – короткий (без створення графічного інтерфейсу), другий – довший (зі створенням графічного інтерфейсу). Перший спосіб ми описали в § 1 і зараз нагадаємо. Другий спосіб детально розглядатимемо в розділі 3.

Після запуску середовища VB отримаєте форму, де відображатимуться результати роботи програми, і декілька супровідних вікон. Серед них провідник проектів (Project), вікно властивостей (Properties) поточного об'єкта (тепер форми), вікно Form Layout, в якому задають розташування форми з майбутніми результатами на екрані методом перетягування.

У вікні властивостей для форми можна задати потрібні розміри методом перетягування маркерів, колір фону (BackColor), властивість AutoRedraw як True (!), підпис форми (Caption) до вподоби або відповідно до змісту задачі, видимість (Visible) як True, шрифт тексту (Font) тощо. Більшість властивостей вже задані, їх змінювати не варто.

Двічі клацніть на формі. Отримаєте вікно із заготовкою процедури, яка виконуватиметься під час запуску проекту на виконання. Вертикальний курсор позначає точку вводу тексту програми. Вводьте сюди текст, дописавши метод Show на початку, вилучивши команду END і не використовуючи деяких заборонених у VB команд, наприклад, DEF-описів типів, PRINT USING, PLAY, DRAW.

Коли програму введено, виконайте її, натиснувши **F5**. Отримаєте результати або повідомлення про помилку. Виправте помилку. Але спочатку закрийте вікно очікуваних результатів (Form1), піктограма якого з'явиться на панелі задач робочого столу Windows. Тільки тепер можна вносити зміну до тексту програми.

Якщо змінні треба описати, то застосовуйте команду DIM, принцип замовчування або, в крайньому випадку, символи-суфікси.

Якщо дані потрібно таки форматувати (як командою PRINT USING), то застосуйте до них функцію форматування FORMAT, наприклад, так: c = FORMAT(<число або вираз>, <формат, що був у USING>), а тепер застосовуйте метод PRINT c.

Перепишіть результати у зошит і закрийте вікно форми. Після цього можна зберегти свою роботу (проект) засобами меню (**File, Save Project**) у двох файлах (файлі форми і файлі проекту), причому обов'язково у власній, заздалегідь підготовленій або новоствореній папці.

Можна також створити exe-файл (**File, Make Project.exe...**) у власній папці та виконати проект незалежно від середовища VB.

Другий спосіб створення VB-програми полягає у підготовці графічного інтерфейсу з використанням таких елементів керування: кнопок, після натискання яких виконуватимуться певні дії (уся програма або її частини); полів для введення даних, що замінять команди READ та INPUT; полів для виведення даних, що замінять метод PRINT; перемикачів і радіокнопок, що асоціюватимуться з командами IF та CASE тощо. Про все це дізнаєтеся у розділі 3.

У четвертому розділі описано етапи створення проектів і розв'язків у середовищі Visual Basic .NET.

Розділ 2. ЗАДАЧІ

Різних за темами задач є 23. Більшість задач мають по 25 індивідуальних завдань. Номер завдання, яке має розв'язати учень чи студент, у конкретній задачі визначається числом i – номером варіанта. Номер варіанта – це номер студента (учня) в журналі або число, утворене з останніх двох цифр номера залікової книжки. Оскільки варіантів завдань є лише 25 і, якщо $i>25$, то від числа i потрібно відняти 25 або 50, або 75. Якщо в умові задачі немає конкретних даних, то їх треба задати на свій розсуд, керуючись її змістом. Скрізь вимагається скласти програму для розв'язування задачі. У кінці кожної програми зазначити своє прізвище.

В умовах багатьох задач є посилання на функції. Потрібні функції слід вибирати з наведеної нижче таблиці відповідно до значення числа n, яке залежить від номера конкретного варіанта i та умови задачі. Наприклад, розглянемо варіант i=21. Якщо в умові деякої задачі зазначена функція $y=f_{i+19}(x)$, то i+19=21+19=40. Оскільки це число більше, ніж 25, то від нього віднімаємо 25 і отримуємо номер індивідуального завдання n=40–25=15. Отже, вибираємо з таблиці функцію з номером n=15.

n	Функція $f_n(x)$
1	$9{,}2\cos x^2-\lvert\sin x/1{,}1\rvert$
2	$12{,}4\sin\lvert x/2{,}1\rvert)-8{,}3\cos 1{,}2x$
3	$\lvert\cos x/2{,}7\rvert+9{,}1\sin(1{,}2x+1)$
4	$\lvert\sin x/3{,}12+\cos x^2\rvert-8{,}3\sin 3x$
5	$\cos\lvert 2x\rvert/1{,}12-\cos(3x-2)+6{,}15$
6	$\sin x\cos x^2\sin(x+1{,}4)+5{,}14$
7	$\lvert\sin(2x-1{,}5)+3\sin x^2\rvert+2{,}38$
8	$\cos x^2\sin(2x-1)+4{,}29$
9	$\cos(x^2+1)-\lvert\sin 2x-5{,}76\rvert$
10	$\sin x-\cos x^3\sin(x^2-4{,}2)+4{,}27$
11	$\lvert\sin 12x\cos\lvert 2x\rvert/3\rvert+4{,}21$
12	$\cos x^3/2{,}1+\cos x^2/1{,}1-8.3\sin(3x+1)$
13	$\sin x^2\cos x^3-\sin x+5{,}2$
14	$2\sin x\sin(2x-1{,}5)\cos(2x+1{,}5)-6$
15	$\lvert\cos x^2-0{,}51\rvert\sin(3x-4)-4{,}44$
16	$\cos 2{,}1x\sin\lvert x\rvert/0{,}15-5{,}8$
17	$\lvert\cos 2x^3+2\sin(x/1{,}2\text{-}3{,}4)\rvert+10{,}51\cos\lvert 3x\rvert$

18	$\lvert\sin(x^2/1{,}5\text{-}2)\rvert+11{,}73\cos(1{,}6x-1)$
19	$13{,}4\cos\lvert x\rvert\sin(x^2-2{,}25)$
20	$\cos(x^2-3{,}8)\rvert/4{,}5\text{-}9{,}7\sin(x\text{-}3{,}1)$
21	$13{,}4\sin(\text{-}1{,}26)\cos\lvert x/7{,}5\rvert$
22	$2\sin\lvert 2x\rvert\cos 2x-11{,}6\sin(x/0{,}4\text{-}1)$
23	$\sin\lvert x\rvert/0{,}1+ 9{,}4\sin(3x\text{-}2{,}5)$
24	$10{,}8\lvert\cos(x^2/1{,}13)\rvert\sin(x+1{,}4)$
25	$11{,}2\cos(2x-1)+\lvert\sin 1{,}5x\rvert/1{,}7$

Задача 1. Дуже проста програма. Скласти і виконати програму, задавши вхідні дані самостійно.

1. Квіткова клумба має форму круга. Обчислити її периметр і площу за заданим радіусом.
2. Обчислити периметр і площу прямокутного трикутника за заданим катетом та гострим кутом.
3. Обчислити довжину кола і площу круга за заданим діаметром.
4. Ділянка лісу має форму рівнобічної трапеції. Обчислити її периметр і площу за заданими сторонами.
5. Ресторан закуповує щодня масло m_1 кг по 8.50 грн. за кілограм, сметану m_2 кг по 2.40 грн., вершки m_3 кг по 4.10 грн. Визначити суми, потрібні для купівлі окремих продуктів, і загальну суму.
6. Скільки секунд мають доба, тиждень, рік?
7. Обчислити кінетичну ($E=mv^2/2$) та потенціальну ($P=mgh$) енергії тіла заданої маси m, яке рухається на висоті h зі швидкістю v.
8. Ціни на два види товарів зросли на p відсотків. Вивести старі та нові ціни.
9. Обчислити площу поверхні $S=4\pi r^2$ та об'єм $V=4\pi r^3/3$ сфери за заданим радіусом r.
10. Швидкість світла 299792 км/с. Яку відстань долає світло за годину, добу?
11. Увести врожайність трьох сортів пшениці (36, 40, 44 т/га) і розміри трьох відповідних полів (у га). Скільки зібрали пшениці з кожного поля і з трьох полів разом?
12. Радіус Місяця 1740 км. Обчислити площу поверхні ($S=4\pi r^2$) та об'єм планети ($V=(4/3)\pi r^3$).
13. Обчислити довжину гіпотенузи та площу прямокутного трикутника за заданими двома катетами.
14. Обчислити об'єм та площу бічної поверхні куба, якщо відоме ребро.
15. Увести продуктивності роботи трьох труб, які наповнюють басейн, і час їхньої роботи. Скільки води набрано в басейні?
16. Яку площу і периметр матиме квадрат, описаний навколо круга заданої площі S.
17. Тіло падає з прискоренням g. Визначити пройдений тілом шлях $h=gt^2/2$ після першої та другої секунд падіння.

18. Обчислити периметр і площу прямокутного трикутника за заданими катетами.
19. Телефонні розмови з трьома населеними пунктами коштують c_1, c_2, c_3 коп/хв. Розмови тривали t_1, t_2, t_3 хв відповідно. Яку суму нарахує комп'ютер до оплати за кожну і всі розмови?
20. Обчислити площу бічної поверхні $S=2\pi rh$ та об'єм $V=\pi r^2h$ діжки за заданою висотою h та радіусом основи r.
21. Квіткова клумба має форму квадрата. Обчислити її периметр і площу за заданою стороною.
22. Обчислити катет та площу прямокутного трикутника за заданими гіпотенузою та другим катетом.
23. Обчислити сторону та площу $S=d^2/2$ квадрата, якщо відома його діагональ d.
24. Обчислити площу бічної поверхні $S=\pi rl$ та об'єм $V=\pi r^2h/3$ конуса за заданою висотою h, твірною l та радіусом основи r.
25. Поїзд їхав t_1 год зі швидкістю v_1км/год, t_2 год зі швидкість v_2 і t_3 год зі швидкістю v_3. Визначити пройдені шляхи з різною швидкістю і повний шлях.

Задача 2 (про трикутник). Трикутник задано координатами вершин A(0; 0), B(i; $i-1$) та C($-i$; $i+1$), де i – номер варіанта.

1. Обчислити висоту h_a та бісектрису W_c.
2. Обчислити медіану m_a та бісектрису W_b.
3. Обчислити бісектрису W_a та радіус вписаного кола r.
4. Обчислити висоту h_a та медіану m_b.
5. Обчислити медіану m_b та бісектрису W_c.
6. Обчислити бісектрису W_a та радіус описаного кола R.
7. Обчислити висоту h_b та бісектрису W_a.
8. Обчислити висоту h_b та медіану m_c.
9. Обчислити висоту h_a та радіус вписаного кола r.
10. Обчислити медіану m_c та бісектрису W_a.
11. Обчислити висоту h_b та бісектрису W_c.
12. Обчислити медіану m_c та радіус вписаного кола r.
13. Обчислити висоту h_b та медіану m_a.
14. Обчислити медіану m_a та радіус описаного кола R.
15. Обчислити медіану m_a та бісектрису W_c.
16. Обчислити висоту h_c та бісектрису W_a.
17. Обчислити медіану m_b та радіус вписаного кола r.
18. Обчислити висоту h_c та медіану m_a.
19. Обчислити медіану m_b та бісектрису W_a.
20. Обчислити медіану m_c та радіус описаного кола R.
21. Обчислити висоту h_c та бісектрису W_b.
22. Обчислити висоту h_c та медіану m_b.
23. Обчислити висоту h_a та радіус описаного кола R.

24. Обчислити висоту h_a та бісектрису W_b.
25. Обчислити висоту h_a та медіану m_c.

Задача 3а. Розгалуження. Увести довільне значення x та обчислити й вивести значення складеної функції

$$y=\begin{cases} f_{i+2}(\varphi), & \text{якщо } |x|<10, \\ f_{i+3}(\omega), & \text{якщо } |x|\ge 10, \end{cases}$$

де $\varphi=\mathrm{tg}(x+a)-\log_i|b+7|$, $\omega=c\sqrt[5]{x^2+de^{1.3}}$, i – номер варіанта.

Скласти дві програми, використовуючи: 1) повну команду розгалуження **if**; 2) коротку команду розгалуження **if**. Вирази для функцій f_{i+2} і f_{i+3} вибрати з таблиці на стр. 84. Вхідні дані x, a, b, c, d ввести з клавіатури довільні.

Задача 3б. Розгалуження. Нехай оплата робіт залежить від типу виконаної роботи чи виду підприємницької діяльності (А, Б, В) і нараховується за формулою

$$y=\begin{cases} 100\,|f_{i+2}(i)+50| & \text{для робіт типу А}, \\ 150\,|f_{i+3}(i)+100| & \text{для робіт типу Б}, \\ 200\,|f_{i+4}(i)+135| & \text{для робіт типу В}, \end{cases}$$

де i – номер варіанта. Для робіт типу А податок складає 10%, для Б – 15%, для В –20%. Увести тип робіт. Вивести нараховану суму, суми податку і суму до видачі. Розв'язати задачу трьома способами, використовуючи: 1) повну команду розгалуження **if**; 2) коротку команду розгалуження **if**; 3) команду вибору **case**.

Задача 4. Вибір. Скласти програму для розв'язування наведеного нижче завдання двома способами, використовуючи: 1) команду **case**; 2) команду **if**. Придумати і задати вхідні дані так, щоб вибір був з 4–7 альтернатив.

1. Ввести номер студента зі списку. Вивести його прізвище та ім'я.
2. Є дані про автомобілі чотирьох моделей. Як вхідне дане ввести номер моделі і отримати характеристики: рік випуску і ціну.
3. Ввести номер поїзда. Вивести назву пункту призначення.
4. Ввести назву країни. Вивести назву її столиці.
5. Ввести номер дня тижня. Вивести його назву.
6. Ввести номер трамвая. Вивести назви його кінцевих зупинок.
7. Ввести назву країни. Вивести назву континента.
8. Ввести номер місяця. Вивести назву пори року.
9. Ввести номер студента у списку. Вивести його ім'я.
10. Ввести назву міста. Вивести кількість населення і площу міста.
11. Ввести номер місяця. Вивести назву місяця і номер кварталу.
12. Ввести номер автобуса. Вивести кількість зупинок його маршруту.

13. Ввести першу букву назви країни. Вивести кількість населення і міст цієї країни.
14. Ввести телефонний код райцентру. Вивести його назву.
15. Ввести номер дня тижня. Вивести кількість пар (уроків) у цей день.
16. Є дані про шість товарів. Ввести числовий код одного з них, отримати довідку про ціну і кількість товару на складі.
17. Ввести номер місяця. Вивести кількість днів у ньому.
18. Ввести числовий код групи. Вивести кількість студентів.
19. Ввести число з діапазону 0..5. Вивести його значення двома мовами.
20. Ввести номер поїзда. Вивести довідку про час відправлення.
21. Ввести першу букву назви річки. Вивести довідку про її довжину.
22. Ввести числовий код сузір'я. Вивести кількість зірок у ньому.
23. Ввести номер дня тижня. Вивести його назву і кількість пар.
24. Ввести номер квартири. Вивести кількість кімнат і мешканців у ній.
25. Увести число з діапазону 5..9. Вивести його значення двома мовами.

Задача 5. Цикли. Таблиця мір. Побудувати таблицю відповідностей між мірами. Початкове значення міри, крок зміни цього значення та кількість рядків у таблиці (10–15) задати самостійно у режимі діалогу. Оформити таблицю якнайкраще, використовуючи формати виведення.

1. 1 унція = 28.353495 г = 142 карати;
2. 1 драхм = 1.77185 г = 0.06249 унцій;
3. 1 карат = 0.2 г = 2.9412 гран;
4. 1 гран = 0.068 г = 0.038378 драхм;
5. 1 пайп = 54.18 пек = 477.33 л;
6. 1 галон (брит.) = 1.2 галон (США) = 4.546 л;
7. 1 галон (США) = 0.0347 сак = 3.785 л;
8. 1 чарка = 0.0568 л = 0.00012 пайпа;
9. 1 квартет = 291 л = 5123.24 чарок;
10. 1 страйк = 72.73 л =1280.46 чарок;
11. 1 челдрон = 1.309 л = 0.149 пека;
12. 1 сак = 109 л =1.499 страйка;
13. 1 пек = 8.81 л = 0.07929 сака;
14. 1 корд малий = 3.624 куб. м = 128 куб. футів;
15. 1 стандарт = 4.672 куб. м = 0.165 рода;
16. 1 род = 28.3 куб. м = 1000 куб. футів;
17. 1 чейн будівельний = 30.48 м = 100 футів;
18. 1 фінгер =11.4 см = 4.5 дюймів;
19. 1 нейл = 5.7 см = 2.25 дюймів;
20. 1 фут = 0.3048 м = 12 дюймів;
21. 1 ярд = 0.9144 м = 3 фути;
22. 1 кабельт Брит. =0.183 км = 680 футів;
23. 1 кабельт США =219.5 м = 720 футів;
24. 1 дюйм = 2.54 см = 12 ліній;
25. 1 морська миля = 1.852 км = 6076 футів.

Задача 6. Цикли. Обчислення скінченних сум і добутків. Обчислити значення виразу z для свого варіанта:

1) $z=a+b$;
2) $z=ab$;
3) $z=\mathrm{tg}(b)-a$;
4) $z=(a+b)^2$;
5) $z=5ab-4$;
6) $z=\sin(a)+b$;
7) $z=a^b$;
8) $z=a^2+3b$;
9) $z=(ab)^{1/4}$;
10) $z=ab-\pi$;
11) $z=a-2b$;
12) $z=b^a$;
13) $z=\cos(ab)$;
14) $z=|a-b|$;
15) $z=\mathrm{ctg}(2a)-b$;
16) $z=\exp(3ab)$;
17) $z=4ba-b$;
18) $z=2a-b$,
19) $z=|12a-\cos(b)|$;
20) $z=2a-b$;
21) $z=\mathrm{tg}(a+b)$;
22) $z=\ln|a+4b|$;
23) $z=3ab-\cos(b)$;
24) $z=4a+\exp(b)$;
25) $z=5a-2b$,

якщо

$$a=\sum_{x=i}^{i+8} f_{i+5}(x), \quad b=\prod_{x=i}^{i+5} f_{i+6}(x),$$

де i – номер варіанта, x – ціле число. Вирази для функцій f_{i+5} і f_{i+6} вибрати з таблиці на стр. 84. Вивести значення i, a, b, z і відформатувати результати.

Задача 7а-7б. Цикли. Обчислення нескінченних сум. Утворити нескінченно спадну числову послідовність:

7а) $a_k=f_{i+7}(k)/k$, де i – номер варіанта, $k=1,2,\ldots$;

7б) $a_k=(-1)^k f_{i+7}(k)x^k/(k!)$, де i – номер варіанта, x – довільне дане з проміжку $(0;1)$, $k=1,2,\ldots$

Обчислити суму цієї послідовності з точністю $e=0.001$. Скільки потрібно доданків для досягнення заданої точності? Виконайте програму тричі для різних значень точності: 0,001; 0,0001; 0,00001.

Задача 8. Цикли. Табулювання функції і пошук даних. Протабулювати функцію $y=f_{i+8}(x)$ на проміжку $[0; i]$ з кроком $h=0.1i$, де i – номер варіанта. Результати обчислень вивести на екран у вигляді таблиці пар чисел x, y. Виконати завдання пошуку даних. Якщо шуканих даних немає, вивести про це повідомлення.

1. Обчислити суми першого та останнього значень функції.
2. Обчислити суму та добуток всіх значень функції y, для яких виконуються нерівності $y<-3,2$ або $y>0$.
3. Обчислити добуток та кількість усіх значень функції y, для яких виконуються нерівності $y<-3$ або $y>0,4$.
4. Обчислити добуток усіх від'ємних значень функції y та визначити кількість додатних.
5. Обчислити добуток значень аргумента (x), для яких досягаються мінімальне та максимальне значеннння функції y.

6. Скільки було від'ємних значень? Визначити максимальне значення
7. Визначити суму додатних значень функції та кількість від'ємних.
8. Скільки від'ємних та додатних значень має функція y?
9. Обчислити суму та кількість додатних значень функції y.
10. Обчислити суму квадратів усіх додатних значень функцій y. Визначити, для якого x функція набуває мінімального значення.
11. Обчислити модуль різниці максимального та першого значень y.
12. Обчислити суму усіх значень функції y, для яких виконуються нерівності $y<1{,}2$ або $y>4$. Визначити максимальне значення функції.
13. Обчислити добуток додатних значень та кількість від'ємних.
14. Обчислити добуток від'ємних значень функції y. У якій точці (x) функція набуває максимального значення.
15. Обчислити добуток усіх значень функції y, для яких справджується нерівність $1<y<3{,}1$. Визначити, для якого x функція набуває мінімального значення.
16. Обчислити кількість та добуток усіх від'ємних значень y.
17. Обчислити суму квадратів та добуток усіх значень функції y, для яких справджується нерівність $-2{,}41<y<5$.
18. Обчислити модуль добутку максимального та мінімального значень.
19. Обчислити середнє арифметичне всіх від'ємних значень функції.
20. Обчислити суму кубів всіх додатних значень та їхню кількість.
21. Знайти середнє арифметичне тих значень функції y, для яких виконуються нерівності $y<0$ або $y>1$.
22. Знайти мінімальне значення функції, а також визначити значення аргумента, для якого воно досягається.
23. Обчислити суму та кількість тих значень функції y, для яких виконується нерівність $0<y<1$.
24. Обчислити кількість та добуток тих значень функції y, для яких виконуються нерівності $1{,}3<y<5$.
25. Яких значень функції більше: додатних чи від'ємних?

Задача 9. Текстові дані. Ввести прізвище, ім'я та по батькові як одне текстове дане. Визначити довжину даного і кількість букв "а" у ньому. Виконати додатково завдання свого варіанта:

1. Вивести ім'я та кількість букв у третьому слові.
2. Визначити скільки букв 'а' є у прізвищі.
3. Вивести три букви – свої ініціали з крапками.
4. Вивести довжини прізвища та імені.
5. Вивести прізвище та ініціали.
6. Вивести ім'я та кількість букв у прізвищі.
7. Визначити скільки букв 'о' є в імені.
8. Вивести найдовше слово.
9. Вилучити усі букви 'а' та 'о' з прізвища.
10. Вивести ім'я у стовпчик.
11. Чи починається хоч би одне слово з букви 'М'?
12. Усі букви 'і' в імені продублювати.

13. Вивести прізвище та кількість букв у імені.
14. Вивести ім'я у зворотному порядку.
15. Вивести прізвище у стовпчик.
16. Вивести ім'я та по батькові та кількість букв у імені.
17. Вивести найкоротше слово.
18. Вивести дане без пропусків. Скільки букв є в імені?
19. Вивести довжини трьох слів.
20. Вивести ім'я та кількість букв у прізвищі.
21. Вивести ім'я, прізвище.
22. Кожну букву імені продублювати.
23. Вивести прізвище у зворотному порядку.
24. Визначити скільки букв 'а' та 'б' є у прізвищі.
25. Вивести третє слово та кількість букв у прізвищі.

Задача 10. Текстові дані. Криптографія. Придумати та описати словесно власний спосіб шифрування тексту. Скласти програму для введення тексту як даного типу string (до 255 символів), його шифрування і виведення результату.

Задача 11. Одновимірні масиви. Розглянемо фінансову діяльність фірми протягом десяти років. Нехай дохід фірми за k-ий рік обчислюється за формулою $y_k=100f_{i+9}(k)$ умовних одиниць, де k=1997, 1992, ..., 2006; i – номер варіанта. Якщо $y_k>0$, то вважатимемо, що фірма у відповідний рік мала прибуток, а у випадку $y_k<0$ – збитки. Вивести на екран таблицю: номер року, величина доходу.

Пошук даних. Виконати додатково завдання свого варіанта, наведене нижче. Вивести повідомлення, якщо шуканих даних немає, наприклад, якщо збитків чи прибутків не було взагалі.

1. Обчислити суму прибутків фірми. Визначити максимальний збиток (якщо збитки були).
2. Обчислити суму збитків. У якому році збиток був максимальний?
3. Обчислити суми прибутків та збитків фірми та їх різницю. Коли прибуток був максимальний?
4. Скільки років поспіль прибутків було менше, ніж 1000, але більше, ніж 500 у.о? Коли фірма зазнала найбільших збитків?
5. Обчислити суму збитків. У якому році прибуток був найбільший?
6. Обчислити суму прибутків у межах $0<y_k<710$ (в у.о.). У якому році фірма зазнала найбільших збитків?
7. Скільки років прибутки були в межах від 200 до 700 у.о? Які це були роки?
8. Обчислити суму всіх збитків. У якому році збиток був найбільший? Який це був збиток?
9. Обчислити суму тих збитків, для яких справджуються умови $y_k<-650$ або $y_k>-150$ (в у.о.). Визначити найбільший прибуток.
10. Визначити суми прибутків та збитків. Скільки років фірма була прибутковою?

11. Обчислити суму прибутків, що були у межах $230<y_k<8500$ (в у.о.). Скільки років фірма мала такі прибутки?
12. Обчислити суму збитків, що були у межах $750<y_k<-200$ (в у.о.). Коли дохід був мінімальний?
13. Обчислити суму прибутків та збитків за перші сім років роботи та їх різницю. Визначити максимальний прибуток за цей період.
14. Обчислити суми прибутків, що були в межах $y_k<170$ або $y_k>620$ (в у.о.). Скільки років фірма мала такі прибутки?
15. Обчислити суму збитків і визначити скільки років фірма була збитковою? У якому році збиток був максимальний?
16. Визначити найбільший збиток. У якому році фірма мала найбільший прибуток?
17. У які роки фірма мала найбільші прибуток та збиток?
18. Обчислити суму збитків. Чи був хоч раз нульовий баланс?
19. Обчислити суми прибутків і збитків фірми та їх різницю. Визначити максимальний збиток фірми.
20. Обчислити суму збитів, для яких справджується умова $y_k<-590$ або $y_k>-330$ (в у.о.). Визначитити найбільший прибуток і в якому році він був отриманий?
21. Обчислити суму збитків фірми. У якому році прибуток був найменший? Визначити його величину.
22. Обчислити середні арифметичні всіх прибутків та збитків.
23. Обчислити суми прибутків і збитків за перші п'ять років роботи. Скільки років на протязі цього періоду фірма мала прибутки?
24. Обчислити суму прибутків, які були в межах $315<y_k<958$ (в у.о.). У якому році збитки були найбільші?
25. Коли прибутки були більші, ніж 580 та меньші, ніж 100 у.о? Коли був максимальний прибуток?

Задача 12. Одновимірні масиви та складний пошук. Утворити і вивести масив y з елементами $y_k=f_{i+10}(k)$, де i – номер варіанта, $k=\overline{1,7}$. Виконати завдання свого варіанта. У разі відсутності шуканих даних, вивести повідомлення про це.

1. Утворити (і вивести) новий масив, що складається з додатних елементів масиву y.
2. Знайти суму третього та шостого додатних елементів.
3. Другий від'ємний елемент замінити мінімальним.
4. Скільки є елементів з мінімальним значенням серед додатних?
5. Ненульові елементи масиву занести в інший масив.
6. Обчислити суму перших чотирьох від'ємних елементів.
7. Вивести номер передостаннього додатного елемента.
8. Утворити новий масив з від'ємних елементів масиву y.
9. Знайти добуток другого та четвертого елементів більших, ніж 3.
10. Максимальний елемент поміняти місцями з другим нульовим.
11. Останній від'ємний елемент замінити найбільшим.

12. Обчислити добуток другого від'ємного та п'ятого елементів.
13. Елементи масиву більші, ніж 1 занести в інший масив.
14. Вивести номери двох найбільших елементів. Обчислити їх суму.
15. Чи є два елементи серед від'ємних з максимальним значенням?
16. Максимальний елемент поміняти місцями з четвертим, що задовольняє умові $y_k>1$.
17. Третій додатний елемент замінити максимальним.
18. Визначити номер п'ятого від'ємного елемента.
19. Обчислити добуток перших трьох додатних елементів та визначити їхні номери.
20. Обчислити суму другого додатного та третього елементів.
21. Елементи масиву менші, ніж 4 занести в новий масив.
22. Утворити масив, значення якого знаходяться між значенням третього елемента заданого масиву та максимальним значенням.
23. Вивести добуток номерів двох найменших елементів серед додатних.
24. Визначити суму номерів другого та третього від'ємного елементів.
25. Вивести номери другого та четвертого додатних елементів.

Задача 13. Двовимірні масиви. Простий пошук. Утворити масив з елементами $a_{kn}=nf_{i+11}(k)+\sin(k)f_{i+12}(n)$, де i – номер варіанта, $k,n=\overline{1,4}$. Вивести його на екран у вигляді таблиці (матриці). Виконати додатково завдання свого варіанта.

1. Визначити індекси мінімального елемента масиву. Обчислити добуток його від'ємних елементів.
2. Обчислити кількість елементів масиву, для яких виконується нерівність $1<a_{kn}<6$.
3. Обчислити добуток значень тих елементів, для яких справджуються нерівності $a_{kn}<-1$ або $a_{kn}>1$.
4. Обчислити кількість додатних елементів та їхній добуток.
5. Обчислити суму квадратів елементів, значення яких більші, ніж 1.
6. Обчислити добуток квадратів тих елементів масиву, для яких виконується нерівність $|a_{kn}|<3$.
7. Обчислити кількість тих елементів масиву, для яких виконується нерівність $a_{kn}>3$ та суму елементів менших, ніж 9.
8. Обчислити добуток від'ємних елементів. Визначити індекси максимального елемента.
9. Обчислити суму діагональних елементів масиву та кількість від'ємних елементів.
10. Обчислити добуток тих елементів масиву, для яких виконується нерівність $2<a_{kn}<10$.
11. Визначити індекси максимального елемента масиву. Обчислити добуток елементів над головною діагоналлю.
12. Обчислити добуток елементів перших двох рядків.
13. Обчислити суму елементів масиву над головною діагоналю. Визначити індекси мінімального елемента.

14. Обчислити суму від'ємних елементів. Знайти максимальний.
15. Обчислити добуток мінімального і максимального елементів масиву.
16. Визначити індекси мінімального і максимального елементів масиву.
17. Елементи масиву, що дорівнюють нулю, замінити на 1. Знайти суму елементів під головною діагоналлю.
18. Визначити кількість від'ємних та суму додатних елементів.
19. Обчислити добуток тих елементів, для яких виконуються нерівності $a_{kn}<-5$ або $a_{kn}>3$. Визначити індекси мінімального елемента.
20. Визначити індекси максимального та мінімального елементів масиву. Обчислити їхній добуток.
21. Обчислити добуток елементів над головною діагоналлю матриці та визначити їхню кількість.
22. Обчислити середнє арифметичне додатних елементів масиву.
23. Обчислити суму тих елементів масиву, для яких виконується нерівність $1<a_{kn}<5$. Знайти максимальний елемент.
24. Обчислити суму діагональних елементів матриці та кількість елементів, значення яких менші, ніж 3.
25. Обчислити добуток елементів під головною діагоналлю на суму елементів на головною діагоналлю.

Задача 14. Двовимірні масиви. Задача про вибори. Нехай шість населених пунктів позначені номерами від 1 до 6 (величина k), а п'ять кандидатів – номерами від 1 до 5 (величина n). Кількість голосів, набраних кандидатами у кожному пункті визначається формулою a_{kn}=INT(10*RND(i)+50), де i – номер варіанта. Вивести на екран таблицю результатів голосування, де у рядках є дані з населених пунктів, а у стовпцях – дані щодо конкретних кандидатів. Визначити і вивести значення величин з додаткового завдання. Створити одновимірний масив з шуканими даними.

1. Які підсумкові результати кожного кандидата? (*Підказка*: утворити одновимірний масив з сум значень усіх стовпців таблиці).
2. Які номери населених пунктів, де кількість поданих голосів перевищила 150 (утворити одновимірний масив з цих номерів)?
3. Хто з кандидатів набрав максимальну, а хто - мінімальну кількість голосів у четвертому населенному пункті?
4. Яка кількість голосів була подана за першого і третього кандидатів у всіх населених пунктах?
5. В яких населених пунктах другий і четвертий кандидати набрали максимальну кількість голосів?
6. Скільки виборців взяли участь у голосуванні у кожному населеному пункті?
7. Хто з кандидатів має максимальний рейтинг?
8. Хто з кандидатів набрав максимальну кількість голосів у другому населеному пункті?
9. В яких населених пунктах кількість опитаних більша деякого заданого числа n?

10. За кого з кандидатів подано кількість голосів меншу деякого заданого числа n?
11. В яких населених пунктах перший кандидат набрав максимальну кількість голосів?
12. В якому населеному пункті проголосувало найбільше людей?
13. Хто з кандидатів набрав найбільше голосів у другому і третьому населених пунктах?
14. В якому населеному пункті перший кандидат набрав мінімальну кількість голосів, а в якому максимальну?
15. У кого з-поміж другого, четвертого і п'ятого кандидатів найвищий рейтинг?
16. Хто набрав максимальну, а хто - мінімальну кількість голосів у першому населеному пункті?
17. У яких населених пунктах перший і п'ятий кандидат набрали більше, ніж 100 голосів?
18. Які номери населених пунктів, де кількість учасників виборів не перевищила 450?
19. У кого з кандидатів рейтинг більший деякого заданого числа n?
20. В яких містах кількість виборців менша деякого заданого числа?
21. Які кандидати набрали мінімальну кількість голосів в кожному із населених пунктів?
22. Які кандидати набрали максимальну і мінімальну кількість голосів в другому і п'ятому населених пунктах?
23. У кого з кандидатів найменший рейтинг?
24. У скількох кандидатів рейтинг перевищує деяке задане число n?
25. В яких населених пунктах третій кандидадат набрав максимальну кількість голосів?

Задача 15. Підпрограми. Використовуючи підпрограми, утворити і вивести масив y з елементами $y_k=f_{i+15}(k)$, $k=\overline{1,12}$, i – номер варіанта. Скласти ще одну підпрограму для пошуку даних у цьому масиві. Критерій пошуку даних взяти з задачі № 8.

Задача 16. Підпрограми. Використовуючи підпрограми, утворити і вивести масив **A**, елементи якого описані формулою $a_{m,n}=2mf_{i+16}(n)$, $m,n=\overline{1,4}$, i – номер варіанта. Скласти ще одну підпрограму для пошуку даних у цьому масиві. Критерій пошуку взяти з умови задачі № 13.

Задача 17. Підпрограми для масивів з різною кількістю елементів. У підрозділі Y є 15 співробітників, а в G – 20. Протягом місяця вони відпрацювали деяку кількість днів, яка задана як випадкова величина зі значеннями y_n= INT(31*RND(N)), $n=\overline{1,8}$, g_k= INT(31*RND(K)), $k=\overline{1,10}$. Денна оплата праці d у.о. Податкова ставка 20%. Використовуючи підпрограми, утворити масиви y, g,

вивести значення їх елементів на екран і виконати завдання пошуку даних для кожного підрозділу. Вивести повідомлення, якщо шуканих даних немає.

1. Скільки осіб працювали у кожному підрозділі більше 15 днів?
2. Хто найменше заробив у кожному підрозділі?
3. Кому у кожному підрозділі нараховано більше, ніж 100 у.о.?
4. Скільки людино-днів було відпрацьовано у кожному підрозділі?
5. Який середній заробіток у кожному підрозділі?
6. Скільки осіб отримали більше, ніж 50 і менше, ніж 120 у.о.?
7. Скільки осіб працювали менше, ніж 10 днів?
8. Яка сума податку була сплачена у кожному підрозділі?
9. Хто сплатив найбільший податок у кожному підрозділі?
10. У скількох осіб податок перевищив 20 у.о.?
11. Який середній податок був у кожному підрозділі?
12. У якому підрозділі більший середній заробіток?
13. Хто сплатив найменший податок у кожному підрозділі?
14. Скільки осіб працювали лише один день у кожному підрозділі?
15. У скількох осіб заробіток вищий за середній?
16. У якому підрозділі менший середній заробіток?
17. У скількох осіб заробіток відхиляється від середнього менше, ніж на 10%?
18. У якому підрозділі був зафіксований найбільший заробіток?
19. Скільки осіб працювали більше, ніж 5 і менше, ніж 12 днів?
20. Який середній заробіток перших п'яти осіб?
21. У скількох осіб заробіток був менший за середній?
22. Який середній заробіток останніх чотирьох осіб?
23. У якому підрозділі було відпрацьовано більшу кількість людино-днів?
24. Хто заробив більше, ніж 100 і менше, ніж 200 у.о.?
25. Скільки осіб працювали 2, 3 або 4 днів?
26. Завдання підвищеної складності. Яка кількість відпрацьованих днів найчастіше була зафіксована у кожному підрозділі?

Задача 18. Створення файлу. Створити файл програмним шляхом, який містить, наприклад, інформацію про машини, літаки, птахів, квіти, країни тощо. Вивести вмістиме файлу на екран для контролю.

Задача 19. Використання файлу. Використовуючи файл, створений під час виконання завдання № 18, придумати критерій пошуку деяких даних і здійснити пошук даних у цьому файлі.

Задача 20. Файли. Для деякої предметної області створити файл засобами редактора. Скласти програму для виведення вмісту файлу на екран та пошуку даних у файлі за деяким критерієм. Сюжет для наповнення файлу та критерій пошуку придумати самостійно.

Задача 21. Масив записів. Придумати і описати структуру запису та скласти програму для створення масиву з шести-семи записів і опрацювання відповідних даних згідно з деяким сюжетом. У сюжеті задати і описати критерій пошуку деякої інформації.

Прикладами структур даних можуть бути: дані про студентів (прізвище, ім'я, оцінки з певних предметів), адреси та телефони друзів, характеристики комп'ютерів, автомобілів, аудіотехніки, довідки про країни (назва, кількість населення, площа) тощо.

Приклад сюжету: створити і вивести на екран масив записів про автомобілі на складі (назва моделі, рік випуску, ціна, колір), а також знайти у масиві і вивести на екран назви моделей червоного кольору, які випускалися в 2001 році.

Задача 22. Графіка. Моя емблема. У заданій частині графічного екрана нарисувати фігуру 1, у середині фігури 1 – фігуру 2, а у середині фігури 2 – коло. Усі елементи рисунка виконати різними кольорами. Замкнені області залити кольорами.

Частини екрана, типи фігур визначають згідно з варіантом *i* так:

n:=i **mod 9** – частина екрана; b:=i **mod** 5 – фігура 1, фігура 2.

n, b	**Частина екрана**	**Фігура 1**	**Фігура 2**
0	Верхня половина	Коло	Квадрат
1	Нижня половина	Прямокутник	Коло
2	Ліва половина	Трикутних	Еліпс
3	Права половина	Еліпс	Прямокутник
4	Верхня права чверть	Квадрат	Трикутник
5	Нижня ліва чверть	Коло	Еліпс
6	Нижня права чверть	Прямокутник	Коло
7	Верхня ліва чверть	Трикутних	Прямокутник
8	Весь екран	Еліпс	Трикутник

Наприклад, якщо *i*=15, то n=15 **mod** 9=6, b=15 **mod** 5 =0. Отже; у нижній правій чверті графічного екрана потрібно нарисувати коло, в ньому – квадрат, а в квадраті – знову нарисувати коло.

Задача 23. Рисуємо прапор. Нарисувати дво- або триколірний прапор деякої держави на свій вибір, наприклад, України, Росії, Франції, Данії, Швеції тощо.

Розділ 3. VISUAL BASIC 6 ТА VBA

§ 1. ВСТУП ДО ВІЗУАЛЬНОГО ПРОГРАМУВАННЯ

1. Основні поняття. Технологія роботи у середовищі Visual Basic базується на ідеях об'єктно-орієнтованого та візуального програмування. 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 керування у вікні: кнопки, списки, текстові поля тощо. Середовище візуального програмування Visual Basic – 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, який називається компонентом. Автоматизація програмування досягається завдяки можливості переносити компонент на форму (у програму) з палітри компонентів і змінювати його властивості, не вносячи вручну змін до програмного коду.

Формою називають компонент, який володіє властивостями вікна Windows і призначений для розташування інших компонентів. Компоненти служать для організації діалогу з користувачем. Це кнопки, списки, текстові поля, зображення, конструктор меню тощо. Вони відображаються на екрані під час виконання програми.

Проект — це сукупність файлів, з яких складається програма, створена в середовищі Visual Basic.

2. Інструменти середовища Visual Basic. Вікно середовища містить головне меню, панель інструментів, а також:

- палітру компонентів (ToolBox);
- вікно властивостей об'єктів (Properties Window);
- вікно форми (Form);
- редактор коду (Code).

Усі ці засоби можна відкрити у разі потреби командами головного меню View ⇒ ToolBox, View ⇒ Properties Window, View ⇒ Form та View ⇒ Code.

3. Головне меню та панель інструментів. Головне меню складається з таких команд (пунктів): File, Edit, View, Project, Format, Debug, Run, Query, Diagram, Tools, Add-Ins, Window, Help.

Меню File містить стандартні команди для роботи з файлами проекту. За допомогою цих команд можна створити новий проект (New Project), відкрити чи закрити файл проекту (Open Project,

Remove Project), додати проект (Add Project), зберегти проект чи форму (Save Project, Save Project As, Save Form.., Save Form.. As), роздрукувати вибрані файли проекту (Print) чи створити exe-модуль (Make Project.exe).

Меню Edit містить стандартні команди для пошуку та заміни фрагмента тексту (Find, Replace, Find Next), команди роботи з буфером (Copy, Cut, Paste тощо) тощо. У меню View знаходяться команди візуалізації елементів середовища. Меню Project містить команди керування проектом, зокрема команди додавання файлів до проекту (Add Form, Add Module тощо).

За допомогою команд меню Format можна вирівнювати компоненти відносно сітки та між собою (Align), задавати порядок відображення компонентів, які перетинаються (Order ⇒ Bring to Front, Order ⇒ Send to Back), змінювати розміри вибраного компонента (Make Same Size) і т.і.

Меню Debug призначене для налагодження проекту та пошуку помилок. Меню Run містить команди керування роботою програми (Run, Break, End, тощо). У меню Tools знаходяться команди для задання параметрів середовища.

Панель інструментів служить для розташування кнопок інструментів. На ній можуть міститися кнопки всіх згаданих команд.

4. Палітра компонентів. Палітра компонентів розташована в окремому вікні (рис. 1). Щоб помістити компонент у центр вікна форми, двічі клацають на його піктограмі. Якщо потрібно розташувати компонент десь на формі, клацають один раз на його піктограмі і обводять контур у потрібному місці форми. Вибраний компонент можна переміщати на формі, а також змінювати розміри, перетягуючи його маркери.

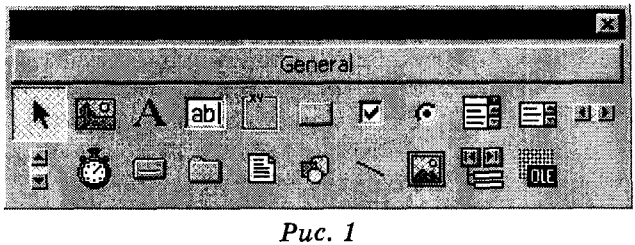


Рис. 1

5. Вікно властивостей об'єктів. За допомогою вікна властивостей можна задавати початкові значення властивостей об'єкта. Вікно інспектора об'єктів містить список компонентів поточної форми, а також дві закладки для властивостей: упорядковані за алфавітом (Alphabetic) та за призначенням (Categorized). Щоб активізувати вікно інспектора об'єктів, використовують клавішу **F4**. Список властивостей складається з двох стовпців: лівий містить назви властивостей компонентів, а правий — їхні значення (рис. 2).

Для введення значень властивостей числового та текстового типу (Width, Name тощо) використовується стандартне поле введення. Значення властивостей перерахованого типу (Alignment, MousePointer тощо) задаються комбінованим списком, звідки вибирають потрібне. Деякі комплексні властивості (Font, Picture, Icon тощо) використовують діалогові вікна, набір керуючих елементів яких залежить від конкретної властивості.

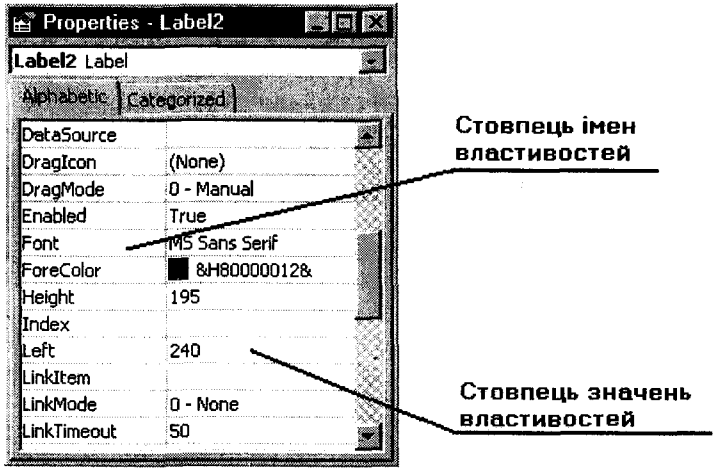


Рис. 2

6. Вікно форми. Форма — це вікно Windows, яке утворюється в одному з можливих для вікон стилів. Увесь внутрішній простір є робочою областю, яка має сітку вирівнювання для зручного розташування компонентів на формі. Для виконання групових операцій декілька компонентів можна об'єднувати. Для цього необхідно натиснути на ліву клавішу миші і переміщенням вказівника охопити всі потрібні компоненти. У групу долучають компоненти, які хоча б частково попадають в охоплену область. Можна також долучити/вилучити окремий елемент. Для цього необхідно натиснути клавішу **Shift** та, не відпускаючи її, вибрати мишею потрібний компонент на формі. Вилучення виокремлених компонентів чи групи виконується клавішею **Delete**. Переміщення виокремленого компонента в межах форми здійснюється мишею. Над компонентами та їхніми групами можна виконувати операції вирізування, копіювання в буфер обміну і вставляння з буфера.

Вирівнювати компоненти можна як відносно вікна форми, так і один відносно одного. Для цього використовується команда головного меню Format ⇒ Align. Інша можливість — безпосередньо задати властивості Left та Top компонентів. Компоненти у групі вирівнюються відносно того компонента, який попав у групу першим.

7. Структура проекту. Проектом називають сукупність файлів, з яких Visual Basic створює готову для виконання програму. До складу кожного проекту входять наступні файли:

- файл проекту *.vbp. Це невеликий файл, який містить посилання на всі файли проекту та ініціалізує програму;
- файли опису всіх форм, які входять у проект: файл модуля *.bas і файл форми *.frm. У цих файлах містяться тексти Basic – програми;
- файл ресурсів програми *.frx. У ньому описані ресурси, які не належать формі, наприклад, піктограма програми;
- файл параметрів проекту *.vbw;

Для збереження проекту необхідно задати імена форм (автоматично пропонуються імена Form1.frm, Form2.frm, ...), модулів (Module1.bas, Module2.bas, ...) та ім'я проекту (Project1.vbp). Ці імена можна змінити на власні. Для переміщення проекту на інший комп'ютер необхідно мати лише файли таких типів: *.vbp, *.frm, *.bas, *.frx. Інші файли створюються автоматично.

8. Редактор коду. Редактор коду програми знаходиться в окремому вікні. У верхній частині цього вікна розміщено два комбіновані списки компонентів форми та подій, які можуть бути до них застосовані. Застосування події до певного об'єкта веде до появи заготовки базового коду відповідної процедури (підпрограми) у вікні редактора. Заготовка (шаблон) складається з заголовка процедури та ключових слів **End Sub**. Отже, код проекту має такий загальний вигляд:

```
'Опис глобальних змінних
Private Sub <ім'я об'єкта>_<ім'я методу> ()
'Тут користувач записує тіло процедури
End Sub
'Інші процедури
```

Зазначимо, що у комбінованому списку об'єктів є засіб (General), який після вибору у правому списку елемента (Declarations) дає програмістові доступ до секції опису глобальних змінних. Заготовку власної функції можна вставити у код програми за допомогою команди головного меню Tools ⇒ Add Procedure. Доступ до такої функції здійснюється вибором у лівому верхньому списку редактора коду позиції (General), а у правому — назви цієї функції. Крім того, відкрити текст будь-якої процедури чи функції можна і безпосередньо у вікні коду за допомогою клавіш зі стрілками чи смуг прокручування.

Зауваження. Усі наведені нижче практичні роботи можна виконати, використавши середовища візуального програмування Visual Basic for Application (VBA), інтегроване в пакет офісних програм Microsoft Office. Незначні відмінності можуть стосуватися лише назв об'єктів та набору візуальних компонентів.

§ 2. ЗАДАЧА ПРО АНКЕТУ. Практична робота № 1. Програмування кнопок. Об'єкти: форма, текстове поле, зображення, кнопка

Мета роботи. Створити форму "Анкета студента" з даними про себе і двома фотографіями (портретною і художньою), які перекривають одна одну і мають з'являтися в результаті натискання на кнопки (рис. 6).

Ознайомитися з такими об'єктами: форма (**Form**), текстове поле (**Label**), зображення (**Image**), кнопка (**CommandButton**) та їхніми основними властивостями: підпис (**Caption**), колір (**ForeColor, BackColor**), шрифт (**Font**), видимість (**Visible**), ширина (**Width**), висота (**Height**) та іншими.

Теоретичні відомості. Об'єкт **Form** використовують для створення нового вікна. Розглянемо такі властивості форми:

Властивість	Опис властивості	Приклади значень
ScaleMode	Одиниці вимірювання лінійних розмірів	Twip (твіп), Point, Pixel
BorderStyle	Можливість змінювати розміри вікна	Sizeable (вікно з довільними розмірами)
Width,Height	Ширина і висота вікна	503, 224 (числове значення)
Font	Шрифт	Комплексна властивість, задається у діалоговому вікні
Icon	Задаємо піктограму, яка буде в заголовку форми під час виконання програми	(None) – стандартна піктограма, або завантажена з певного файлу *.ico
Name	Ім'я форми	Form1 (ідентифікатор)
Caption	Заголовок форми	Довільний рядок символів
BackColor	Колір фону форми	• ToolTip, Desktop (перелічуваний тип) або • &H0000C0C0& (числове значення - задається у діалоговому вікні)
Enabled	Доступність для дій під час виконання програми	True, False

Left, Top	Координати лівого верхнього кутка вікна	200, 108 (числове значення)
WindowState	Стан вікна у момент запуску програми	Normal, Maximized, Minimized

Об'єкт **Label** призначений для створення текстових полів (написів, текстів) у вікні програми. Окрім аналогічних до наведених у попередній таблиці властивостей **Width, Height, Font, BackColor, Name, Caption, Enabled, Left, Top,** він володіє ще й такими:

Властивість	Опис властивості	Приклади значень
Alignment	Вирівнювання тексту в межах поля	Center, LeftJustify, RightJustify
AutoSize	Приведення меж поля до границь тексту	True, False
Visible	Видимість об'єкта	True, False
WordWrap	Перенесення слів тексту у новий рядок	True, False
ForeColor	Колір тексту	&H0000C0C0&

Об'єкт **Image** призначений для вставляння графічних об'єктів з файлів типу *.bmp, *.emf, *.ico, *.wmf у форму. Окрім відомих властивостей **Width, Height, Name, Enabled, Left, Top, Visible,** використовують такі:

Властивість	Опис властивості	Приклади значень
Center	Вирівнювання малюнка до центру відносно поля, що його містить	True, False
Picture	Ім'я графічного файлу	Задається у діалоговому вікні
Stretch	Приведення розміру зображення до заданих розмірів об'єкта	True, False

Об'єкт **CommandButton** використовують для створення кнопок на формі. Кнопки мають такі властивості: **Visible, Width, Height, Font, BackColor, Name, Caption, Enabled, Left, Top** та інші.

Зауваження. У цій та наступних роботах зірочками (*) позначені пункти необов'язкові для виконання учнями. Їх рекомендуємо виконувати студентам, оскільки вони важливі для повноти розуміння матеріалу.

Хід роботи

1. **Завантажте середовище візуального програмування Visual Basic.** Запуск Visual Basic виконують клацанням на піктограмі Microsoft Visual Basic або за допомогою каскадного меню Start (Пуск) ⇒ Programs (Програми) ⇒ Microsoft Visual Studio x.0 ⇒ Microsoft Visual Basic x.0, де x — версія програми. У вікні New Project виберемо Standard EXE ⇒ Відкрити. Отримаємо декілька вікон, зокрема:
   - **головне вікно Project1-Microsoft Visual Basic [design],** де розміщені панель інструментів та головне меню;
   - **вікно форми Project1-Form1(Form),** в якому будуть розташовані результати роботи майбутньої програми;
   - **палітра компонентів** (вікно без назви) з піктограмами візуальних об'єктів.

   Зауваження. Якщо на екрані немає вікна форми чи палітри компонентів, то їх можна відкрити за допомогою команд головного меню View ⇒ Object та View ⇒ Toolbox відповідно.
2. **Активізуйте ще два вікна Visual Basic:**
   - **вікно властивостей Properties Window** зі значеннями властивостей активного об'єкта.
   - **вікно тексту програми Project1-Form1(Code).**

   Якщо цих вікон немає, виконайте команди головного меню View ⇒ Properties Window та View ⇒ Code.
3*. **Запустіть програму Project1 на виконання і розгляньте вікно порожньої поки що форми. Поекспериментуйте з вікном форми.** Запустити програму можна декількома способами:
   - Виконати команду Run ⇒ Start головного меню;
   - Клацнути на кнопці Start панелі інструментів;
   - Натиснути на функціональну клавішу **F5**.

   Виконайте такі дії: максимізуйте вікно, відновіть його попередній розмір, мінімізуйте та знову розгорніть вікно, пересуньте на робочому столі та змініть його розміри, викличте системне меню (**Alt** + пропуск). Виконайте ті ж дії за допомогою команд Move, Size та інших і клавіатури.

   Висновок: вікно форми володіє усіма властивостями стандартного вікна операційної системи Windows.
4. **Закрийте вікно програми Form1, мінімізуйте головне вікно Visual Basic і створіть на робочому диску папку з іменем групи, а у ній власну папку, названу вашим прізвищем. Знову активізуйте вікно Visual Basic.**
5. **Збережіть створену програму у своїй папці.**

   Для цього виберіть команду головного меню File ⇒ Save Project as (Зберегти Проект) або натисніть на кнопку Save Project на панелі інструментів. У першому рядку вікна, яке з'явиться

("Save File As") під заголовком "Save in:" (Папка:), за допомогою випадаючого меню виберіть ім'я робочого диска, після чого знайдіть і відкрийте свою папку. Задайте назву для файлу форми, попередньо стерши запропоновану комп'ютером назву Form1 ⇒ Save. У наступному вікні "Save Project As" дайте назву файлові проекту, стерши запропоновану комп'ютером назву Project1 ⇒ Save.

6*.Візуально ознайомтеся з властивостями форми Width та Height. Змініть за допомогою миші розміри форми. Переконайтесь, що зміна розмірів форми веде до зміни її властивостей Width та Height (ширини і висоти форми) у вікні властивостей Properties-Form1.

Зауваження. Переглядати чи змінювати значення властивостей об'єктів зручно на закладці Categorized вікна Properties, де вони згруповані за своїм призначенням. На закладці Alphabetic цього вікна властивості об'єктів впорядковані за алфавітом за винятком Name – імені об'єкта.

7*.Дослідіть, як зміна значень властивостей Width чи Height форми у вікні Properties веде до зміни розміру форми. Введіть відповідне значення у твіпах і натисніть на клавішу **Enter**.

8*.Змініть колір фону форми. Для цього у вікні властивостей форми у рядку BackColor за допомогою *випадаючого меню* задайте значення кольору фону одним із способів:

- на закладці System виберіть один із системних кольорів;
- на закладці Palette безпосередньо виберіть колір фону.

Поекспериментуйте з властивістю BackColor і задайте її значення на власний розсуд.

9*.Виконайте програму ще раз (див. пункт 3).

10.Вставте у форму текстове поле (об'єкт типу Label) з текстом "Анкета студента". Два рази клацніть мишею на піктограмі Label палітри компонентів. Розташуйте вставлений об'єкт, наприклад, так, як показано на рис. 3, перетягуючи його мишею. Якщо об'єкт Label1 невиокремлений, активізуйте його і у вікні Properties змініть значення властивості Caption з Label1 на текст "Анкета студента" без лапок. Задайте значення властивості AutoSize цього об'єкта як True. Змініть значення властивості Font (шрифт) цього поля на такі:

Font : Times New Roman Cyr (чи MS Sans Serif);
Font style: Bold;
Size : 14;

Задайте колір підпису, вказавши значення властивості ForeColor.

Зауваження. У вікні Properties відображається список властивостей лише активного на даний момент об'єкта.

11*.Аналогічно вставте у форму ще декілька текстових полів з вашими біографічними даними.

Один із варіантів розташування текстових полів показаний на рис. 4.

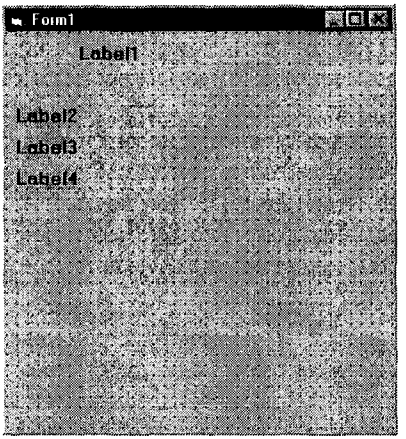


Рис. 3

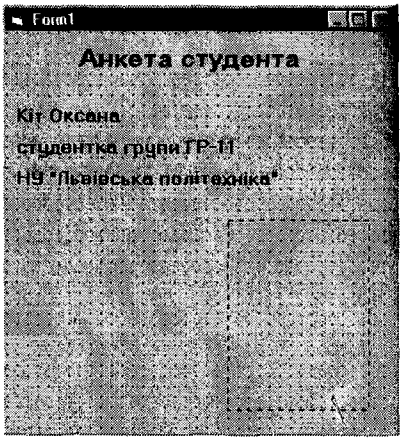


Рис. 4

12.Вставте у форму об'єкт типу Image (зображення).

Для цього клацніть один раз лівою клавішею миші на піктограмі Image палітри компонентів і, наприклад, у нижньому правому куті форми обведіть контур для майбутнього зображення (фотографії). Якщо потрібно, змініть розмір форми чи вставленого об'єкта та добийтеся якнайкращого розташування на ній створених раніше об'єктів. Змінювати розміри об'єкта можна методом їх "розтягування" за маркери (чорні габаритні квадратики). Запам'ятайте назву, яку Visual Basic присвоїть цьому об'єкту (значення властивості Name) або замініть її на свій розсуд. За замовчуванням цей об'єкт матиме стандартну назву Image1.

13.Вставте свою портретну фотографію за допомогою властивості Picture (ілюстрація) об'єкта Image1.

Для цього спочатку виокремте цей об'єкт і задайте значення True його властивості Stretch. Активізуйте рядок Picture у вікні Properties. Клацнувши на кнопці , викличте діалогове вікно вибору малюнка Load picture, де зазначте шлях до файлу з фотографією. Якщо такого файлу немає, скористайтесь будь-яким малюнком з бібліотеки Microsoft Clipart, яка за замовчуванням знаходиться у папці C:\ Program Files \ Microsoft Office \ Clipart \ Popular. Виберіть будь-який файл ⇒ Open.

14.Вставте свою художню фотографію у форму поверх існуючої, скориставшись ще одним об'єктом типу Image.

Один із варіантів розташування фотографії показаний на рис. 5. Вважатимемо, що цей об'єкт має назву Image2.

Зауваження. Під час накладання об'єктів може виникнути потре-

ба використати команди Send To Back (переслати назад) чи Bring To Front (перенести наперед), які є в їхніх контекстових меню.

15*.Поекспериментуйте з властивістю Visible (видимість) обох зображень, кожного разу виконуючи програму (див. пункт 3).
Встановіть значення властивості Visible у False для обох зображень.

16.Вставте у форму кнопки для засвічування фотографій — два об'єкти типу CommandButton з назвами Command1 і Command2.
Піктограма об'єкта типу CommandButton (кнопка) знаходиться на палітрі компонентів Visual Basic. Поміняйте підписи на кнопках (змініть властивість Caption) на "Портретна фотографія" та "Художня фотографія" відповідно. Виберіть найкращий, на ваш розсуд, кирилізований шрифт для підписів. Якщо використано картинки із стандартної бібліотеки Clipart, виберіть для кнопок цікаві підписи. Один із варіантів розташування кнопок показано на рис. 6.

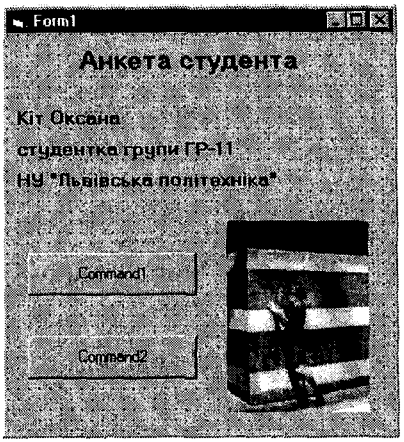


Рис. 5

Рис. 6

17.Запрограмуйте кнопку "Портретна фотографія" так, щоб після її натискання у формі з'являлась портретна фотографія.
Для програмування кнопки Command1 необхідно два рази клацнути на ній лівою клавішею миші. В результаті активізується вікно тексту програми з заготовкою процедури Command1_Click, яка опрацьовуватиме подію клацання на кнопці Command1:

```
Private Sub Command1_Click()
End Sub
```

У заготовку необхідно вставити текст програми реакції на цю подію. Процедура матиме такий вигляд:

```
Private Sub Command1_Click()
Image1.Visible = True        'Портретна фотографія стає видимою
Image2.Visible = False       'Художня фотографія стає невидимою
End Sub
```

За допомогою даної процедури властивість видимості об'єкта Image1 вмикаємо, і цю ж властивість об'єкта Image2 вимикаємо. Для кнопки "Художня фотографія" дії будуть протилежні. Зверніть увагу на використання складених імен типу Image1.Visible, в яких назва об'єкта від його властивості відокремлюється крапкою. Такі складені імена дають доступ до значення конкретної властивості об'єкта. Після введення з клавіатури крапки Visual Basic пропонує програмісту список властивостей, методів та подій для даного об'єкта. Вибір потрібної властивості здійснюється клавішами зі стрілками, а підтвердження — пропуском. Крім того, ім'я потрібної властивості чи методу можна безпосередньо набрати на клавіатурі.

18. Запрограмуйте кнопку "Художня фотографія" відповідно до її призначення (див. пункт 17).

Текст процедури для цієї кнопки матиме вигляд:

```
Private Sub Command2_Click()
Image1.Visible = False      'Портретна фотографія стає невидимою
Image2.Visible = True       'Художня фотографія стає видимою
End Sub
```

Щоб створити таку процедуру швидко, можна скопіювати дві команди присвоєння з попередньої процедури у нову і поміняти вирази справа.

19. Запустіть програму і впевніться, що кнопки виконують свої функції. Закрийте вікно програми "Анкета студента".

20. Збережіть створену програму у своїй папці.

Виберіть елемент головного меню File ⇒ Save Project або натисніть кнопку Save Project на панелі інструментів.

21*. Створіть exe–файл програми.

Виконайте команду головного меню File ⇒ Make <ім'я проекту.exe...>. У вікні, що відкриється, вкажіть особисту папку та ім'я exe–файлу ⇒ Ok.

22*. Закрийте Visual Basic, виконайте створену програму і поекспериментуйте з кнопками.

Запустіть exe–файл з іменем проекту і піктограмою зі своєї папки.

23. Продемонструйте створену форму викладачеві. Закінчіть роботу.

Задачі

Задача 1.1. Вставте у форму третю фотографію (фото вашого будинку чи машини) і ще одну кнопку з відповідним підписом, яка її висвітлюватиме. Якщо файлу з такою фотографією немає, скористайтесь будь-яким файлом з бібліотеки Clipart (див. п. 13).

Задача 1.2. Поміняйте підписи до кнопок на такі: "Змінити фотографію" та "Забрати фотографію", перепрограмувавши

кнопки відповідно до нового призначення. Запишіть фрагменти зміненого програмного коду у звіт. Виконайте програму і переконайтесь у правильності її роботи.

Підказка. У тексті процедур, що описують роботу кнопок, можна скористатися командами вигляду:

```
If Image1.Visible = True Then . . .   'Якщо видимість = True
'Або рівносильною командою
If Image1.Visible Then                'Тут умова також істинна,
                                      'якщо видимість увімкнена
```

Задача 1.3. Поміняйте сценарій роботи програми для задачі 1.2 на наступний:

- *відразу після запуску програми фотографій не видно, є дві кнопки "Портретна фотографія" і "Забрати фотографію" , доступною є лише перша кнопка;*
- *після клацання на кнопці "Портретна фотографія" у формі з'являється портретне фото, підпис на першій кнопці змінюється на "Художня фотографія", стає доступною кнопка "Забрати фотографію";*
- *після клацання на кнопці "Художня фотографія" фотографія у формі змінюється на художню, а підпис на цій кнопці змінюється на "Третя фотографія";*
- *після клацання на кнопці "Третя фотографія" фотографія у формі змінюється на третю, а підпис на цій кнопці змінюється на "Портретна фотографія";*
- *після клацання на кнопці "Забрати фотографію" фотографія зникає і ця кнопка стає недоступною.*

Запишіть фрагменти програмного коду у звіт. Виконайте програму і переконайтесь у правильності її роботи.

Підказка. У тексті процедур, які описують роботу кнопок, можна скористатися командами, що змінюють властивості кнопок Caption (підпис), Visible (видимість), Enabled (доступність).

Задача 1.4. Окрім вимог, поданих в умовах задачі 1.3, після клацання на кнопці "Забрати фотографію" ця кнопка стає не лише недоступною, але і невидимою.

Задача 1.5. Змініть програмний код розв'язування задачі 1.4 так, щоб після вимкнення фотографій напис на першій кнопці завжди відповідав фотографії, яка повинна з'явитися після її натискання.

Задача 1.6. Виходячи з умови задачі 1.5, добийтеся того, щоб послідовність перемикання фотографій не порушувалася внаслідок їх вимкнення, а також додайте текстовий підпис з назвою фотографії, видимою у поточний момент.

§ 3. ЗАДАЧА ПРО ОБМІН ВАЛЮТИ. Практична робота № 2. Програмування розгалужень. Об'єкти: поля редагування, перемикачі

Мета роботи. Створити форму з назвою "Обмін валюти", на якій можна змоделювати операції обміну валюти в обмінному пункті. Застосувати поля редагування (**TextBox**) та перемикачі (**OptionButton**, дослівно кнопка вибору), а також кнопки для виконання обчислень та закінчення роботи програми (див. рис. 8).

Теоретичні відомості. Об'єкти типу **TextBox** використовують для введення рядка символів з клавіатури. Окрім відомих уже властивостей, поля редагування **TextBox** володіють такими:

Властивість	Опис властивості	Приклади значень
PasswordChar	Символ для введення пароля	Порожній рядок (пряме відображення тексту), * (текст відображатиметься зірочками)
ToolTipText	Текст підказки, яка висвітлюється, якщо навести курсор миші	«Введіть суму» (довільний рядок символів)
Text	Текст у полі редагування	«0,0001» (довільний рядок символів)

Об'єкти типу **OptionButton** призначені для створення у формі засобу для вибирання однієї альтернативної можливості серед декількох. Розглянемо такі властивості перемикачів:

Властивість	Опис властивості	Приклади значень
Value	Стан перемикача	True (вибраний), False (не вибраний)
TabIndex	Порядок вибору об'єкта клавішею **Tab**	0 (перший), 4 (п'ятий)
TabStop	Доступ до даного об'єкта табулятором	True (буде доступним), False (не буде)

Хід роботи

1. **Завантажте середовище Visual Basic.**
2. ***.Відмовтесь від можливості змінювати розміри вікна програми, надавши властивості форми BorderStyle значення Fixed Dialog.** Задавши це значення, виконайте програму і переконайтеся, що не можна змінити розмір форми. Зверніть увагу на відсутність у вікні кнопок мінімізації та максимізації. Завершіть роботу програми.

8. **Вставте у форму два об'єкти типу OptionButton (перемикачі) як це показано на рис. 7.**
   Для цього двічі клацніть на піктограмі об'єкта типу OptionButton (перемикач) на палітрі компонентів і розмістіть його у потрібному місці форми. Повторіть ці дії, щоб вставити другий перемикач.
4. **Задайте початкове значення правого перемикача як активне.**
   Для цього клацніть на правому перемикачі і значення його властивості Value (контроль вибору) задайте True.
5. **Вставте у форму два поля редагування — об'єкти Text1 та Text2.**
   Для цього клацніть на піктограмі об'єкта типу TextBox (поле редагування) палітри компонентів, а потім обведіть контур цього об'єкта на формі. Вставте другий об'єкт (рис. 7).
   *Запустіть програму і поекспериментуйте зі вставленими об'єктами: клацніть у полі редагування, введіть деяке число, вилучіть його. Закрийте вікно програми.
6. **Розташуйте у формі два текстові поля — об'єкти Label1 та Label2 (рис. 7).**
7. **Вставте у форму два поля редагування — об'єкти Text3 та Text4 (рис. 7).**
8. **Вставте у форму дві кнопки — об'єкти типу CommandButton (рис. 7).**
9. **Вставте у форму ще два текстові поля — об'єкти Label3 та Label4 (рис. 7).**

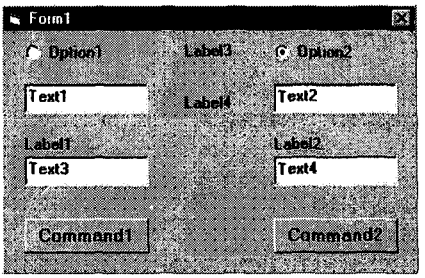


Рис. 7

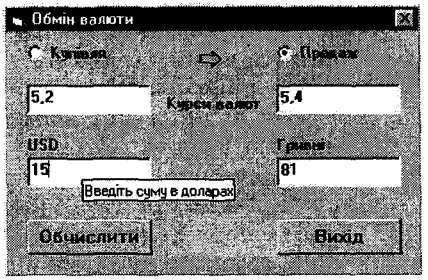


Рис. 8

10. **Збережіть створену на даний момент форму у своїй папці.**
    File ⇒ Save Project. Імена файлів форми та проекту занотуйте у звіт. У подальшому періодично, зокрема перед черговими запусками проекту на виконання, зберігайте файли програми (File ⇒ Save Project, вводити імена файлів вже не потрібно).
11. ***.Змініть назву форми з "Form1" на "Обмін валюти".**
    Для цього змініть значення властивості Caption форми. Клацніть на формі і на рядку Caption у вікні Properties. Введіть назву форми без лапок. Зверніть увагу на те, що для об'єктів деяких

типів (зокрема, Label, Form, CheckBox та інших) значення властивостей Caption та Name збігаються.

12. **Змініть підписи Caption на об'єктах типу OptionButton, Label та CommandButton так, як показано на рис. 8.**
Для цього по черзі вибирайте об'єкти (клацайте на них) і змінюйте значення властивості Caption.

13*. **Задайте однакові розміри для всіх текстових полів, полів редагування та кнопок і вирівняйте їх на формі.**
Для цього одночасно виокремте п'ять об'єктів лівого стовпця одним із способів:

- тримаючи натиснутою клавішу Shift, почергово активізуйте об'єкти, клацаючи на них лівою клавішею миші;
- обведіть навколо цих об'єктів контур, утримуючи натиснутою ліву клавішу миші.

У вікні властивостей задайте спільні для цих об'єктів значення властивостей Width (ширина), Height (висота) та Left (відступ від лівої межі вікна) у твіпах. Зверніть увагу, що подвійне клацання по назві спільної властивості приводить до надання їй значення властивості першого виокремленого об'єкта створеної групи. Можете змінити стиль, колір чи розмір шрифту одночасно для усіх виокремлених об'єктів (властивості Font, ForeColor, BackColor). Зніміть виокремлення, клацнувши на вільному місці форми. Аналогічно виконайте вирівнювання правого стовпця об'єктів. Вирівняйте вставлені поля попарно у горизонтальному напрямку. Для цього змініть властивість Top (відступ від верхньої межі вікна у пікселях) для відповідних груп об'єктів. Збережіть роботу (Save All).

14. **Задайте значення курсів купівлі та продажу валюти, а також кількість валюти, яку кантор купує чи продає.**
Для цього введіть потрібне число, наприклад 5,20, як значення властивості Text об'єкта Text1. Повторіть це для об'єкта Text2 (значення 5,40) та Text3 (наприклад, 15).

15. **Очистіть поле редагування Text4.**
Для цього вилучіть значення властивості Text для об'єкта Text4. Не сплутайте значення властивостей Name та Text цих об'єктів.

16*. **Заблокуйте можливість введення даних для поля Text4, задавши його властивість Enabled (доступність) як False, оскільки це поле міститиме результат.**
Змінювати значення певної властивості можна подвійним клацанням на ній лівою клавішею миші. Збережіть роботу (Save Project). Виконайте програму і переконайтеся, що не можна ввести чи редагувати дані у полі об'єкта Text4.

17. **Запрограмуйте перемикачі так, щоб напрямок стрілки показував на вид операції: купівля чи продаж. Зробіть активним поле Text3.**
Клацніть двічі на правому перемикачі Option2 (Продаж). Отримаєте заготовку процедури Option2_Click. У тілі цієї процедури

опишіть дії, які мають відбутися у результаті клацання на правому перемикачі Option2:

```
Private Sub Option2_Click()
Label3.Caption = "=>"          ' Змінюємо напрямок стрілки
Text3.SetFocus                 ' Активізуємо поле Text3
End Sub
```

Аналогічно запрограмуйте подію Click клацанням на лівому перемикачі Option1, врахувавши, що стрілка має показувати на ліве поле ('<=').

**Фрагмент програмного коду створеної процедури запишіть у звіт.*

18*.Запустіть програму і переконайтесь, що перемикач виконує свої функції згідно пункту 17.

19.Запрограмуйте кнопку "Вихід".

Скористайтесь процедурою закінчення роботи програми End:

```
Private Sub Command2_Click()
End              'Закінчуємо роботу програми
End Sub
```

20.Запрограмуйте кнопку "Обчислити".

Текст процедури цієї кнопки передбачатиме перевірку стану одного із перемикачів (увімкнений чи ні). Перемикачі створені так, що другий завжди перебуватиме у протилежному стані.

```
Private Sub Command1_Click()
If Option1.Value = True Then
Text4.Text = Text3.Text * Text1.Text    'USD * курс купівлі
Else
Text4.Text = Text3.Text * Text2.Text    'USD * курс продажу
End If
End Sub
```

21.Збережіть роботу (Save Project).

22.Виконайте програму і поексперементуйте з різними грошовими сумами і операціями купівлі чи продажу. Закрийте вікно програми "Обмін валюти".

Для переривання роботи програми у випадку неправильного введення вхідних даних виконайте пункт головного меню Run ⇒ End чи клацніть на кнопці ■ End панелі інструментів головного вікна Visual Basic.

Зауваження. Зверніть увагу на використання коми чи крапки у вхідних даних. У числах, які стосуються курсу валюти, гривневої чи доларової сум для десяткової крапки використайте символ, передбачений операційною системою вашого комп'ютера (див. Start (Пуск) ⇒ Settings (Налаштовування) ⇒ Control Panel (Панель керування) ⇒ Regional Settings (Місцеві параметри) ⇒

закладка Number (Числа), рядок Decimal symbol (Символ десяткової крапки)).

23*.Змініть розміри та кольори символів (зокрема об'єктів Label3 і Text3), розташування об'єктів, фон форми (властивість BackColor) так, щоб форма виглядала якнайкраще.

24*.Забезпечте появу підказки "Введіть суму в доларах" після переміщення вказівника миші до поля Text3.

Виберіть об'єкт Text3 і як значення властивості ToolTipText уведіть текст підказки. Збережіть роботу, запустіть програму і переконайтеся, що підказка з'являється.

25*.Поміняйте вигляд стрілки з => на ⇨, а <= на ⇦.

Для цього виберіть об'єкт Label3 і як значення властивості Caption введіть українську букву р, після чого, активізувавши властивість Font, виберіть назву шрифта Wingdings. Двічі клацніть на правому перемикачі і в його процедурі введіть українську букву р замість =>. У процедурі для лівого перемикача символи <= замініть буквою п. Збережіть роботу, запустіть програму і переконайтеся, що стрілка змінила свій вигляд.

26.Продемонструйте створену форму викладачеві. Закінчіть роботу.

Задачі

Задача 2.1. Передбачте у створеній програмі ще одну кнопку для очистки полів грошових сум. Виконайте програму і переконайтесь у правильності її роботи.

Підказка. Для об'єктів Text3, Text4 у процедурі опрацювання події натискання на цю кнопку використайте команду присвоєння їхнім властивостям Text порожнього рядка ("") .

Задача 2.2. Забезпечте появу підказки "Введіть курс купівлі" та "Введіть курс продажу" після переміщення вказівника миші до полів Text1 та Text2 відповідно (див п. 24).

Задача 2.3. У процедурі для кнопки "Обчислити" передбачте 1% збору у пенсійний фонд від операції купівлі–продажу.

Задача 2.4. Модифікуйте програму, передбачивши додаткову можливість зміни типу операцій (купівля, продаж) внаслідок клацання мишею на стрілці. Запишіть у звіт фрагмент програмного коду, який реалізує цю можливість. Виконайте програму.

Підказка. Для цього двічі клацніть на текстовому полі стрілки. Відкриється вікно програмного коду із заготовкою процедури Label3_Click (опис дій у випадку клацання на об'єкті Label3). В тілі цієї процедури можна скористатися командами вигляду:

```
If Label3.Caption = "=>" Then
• • •   'Встановлюємо перемикач у ліве положення, змінюючи
        'значення властивості Value об'єкта Option1
Else
• • •   'Встановлюємо перемикач у праве положення, змінюючи
        'значення властивості Value об'єкта Option2
End If
```

Задача 2.5. Спростіть форму (вилучіть зайві об'єкти) та змініть код кнопки "Обчислити" так, щоб її можна було використати для переведення миль у кілометри чи навпаки в залежності від положення перемикача (1 миля = 1,609344 кілометрів).

Задача 2.6. У створену для задачі 2.5. форму вставте рамку (об'єкт Frame), а в неї — два перемикачі для вибору типу милі з двох можливих значень: морської чи звичайної (1 морська миля = 1,852 кілометрів).

§ 4. ЗАДАЧА ТАБУЛЮВАННЯ ФУНКЦІЇ. Практична робота № 3. Програмування циклів. Об'єкти: CheckBox, Frame. Робота з меню

Мета роботи. Створити форму для розв'язування задачі табулювання функції. Побудувати у ній головне меню з командами: закінчити роботу програми, табулювати функцію, очистити поля виведення результатів. Результати табулювання вивести у багаторядкове поле редагування (об'єкт типу **Text**). Передбачити можливість виведення результатів на екран, у файл, у масив. Напрямок виведення задати за допомогою трьох прапорців (об'єктів типу **CheckBox**), розташованих у рамці (**Frame**) (див. рис. 9).

Об'єкт **CheckBox** використовують для створення незалежного дво- чи трипозиційного прапорця: увімкнено/вимкнено(/недоступний). Для цього об'єкта визначені такі дві нові властивості:

Властивість	Опис властивості	Приклади значень
Value	Стан прапорця	Grayed (недоступний), Unchecked (вимкнений), Checked (увімкнений)
MousePointer	Вигляд вказівника миші на об'єкті	Arrow (стрілка), Cross (хрест)

Рамка **Frame** призначена для розміщення у ній групи із кількох об'єктів. Рамку використовують для покращення дизайну вікна програми. Властивості цього об'єкта аналогічні до описаних вище.

Хід роботи

1. Завантажте середовище Visual Basic.

2*.Змініть заголовок (Caption) форми з "Form1" на "Табулювання функції" (без лапок) і збільшіть розміри форми у вертикальному напрямку.

3*.Змініть піктограму у лівому верхньому куті форми, задавши конкретний файл з рисунком піктограми як значення властивості Icon (піктограма) форми.

Клацніть у рядку Icon на ▪, щоб отримати вікно Load Icone. Відкрийте папку C: \ Program Files \ Microsoft Visual Studio \ Common \ MSDev98 \ Template \ ALT, виберіть графічний файл з будь-якою піктограмою ⇒ Open ⇒ Ok.

4*.Збережіть виконану на даний момент форму у своїй власній папці (File ⇒ Save Project).

5. Вставте у форму поле редагування Text1.

Збільшіть розміри поля. Властивість ScrollBars (наявність смуг прокручування) цього об'єкта задайте Both (будуть обидві смуги — вертикальна і горизонтальна). Властивість MultiLine задайте True – дозвіл об'єкту працювати більш ніж з одним рядком.

6. Розташуйте у формі поля редагування Text2, Text3, Text4 і відповідні їм текстові поля "Ліва межа", "Права межа", "Крок", а також текстове поле для вигляду даної функції y=sin(x) +1.

Розмір, стиль і колір шрифтів виберіть на власний розсуд так, щоб форма виглядала якнайкраще ⇒ Save Project.

7. Задайте початкові значення для полів редагування лівої і правої меж аргумента функції та для кроку зміни цього аргумента, наприклад, такі, як на рис. 9.

Для цього змініть властивість Text цих обєктів. Для набору символа десяткової крапки чи коми використайте символ, передбачений операційною системою комп'ютера.

8*.Вирівняйте вставлені поля редагування до лівого краю першого об'єкта та надайте їм однакові розміри.

Виокремте ці об'єкти і скористайтесь потрібними командами головного меню Format ⇒ Align ⇒ ... та Format ⇒ Make Same Size ⇒ ...

9. Вставте у форму рамку (об'єкт типу Frame).

Для цього використайте компоненту Frame ▪ палітри компонентів. Змініть значення властивості Caption (підпис) цього об'єкта на слово "Вивести" (без лапок). Розмір, стиль і колір шрифту виберіть на власний розсуд. Збільшіть рамку.

10.Вставте у рамку три прапорці (об'єкти типу CheckBox).

Для цього використайте компоненту CheckBox ☑ із палітри компонентів. Вирівняйте прапорці і змініть значення їх властивостей Caption на такі, які показані на рис. 9. Стиль і колір шрифтів виберіть на власний розсуд.

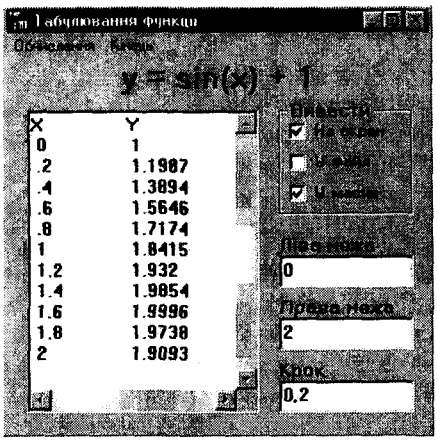


Рис. 9

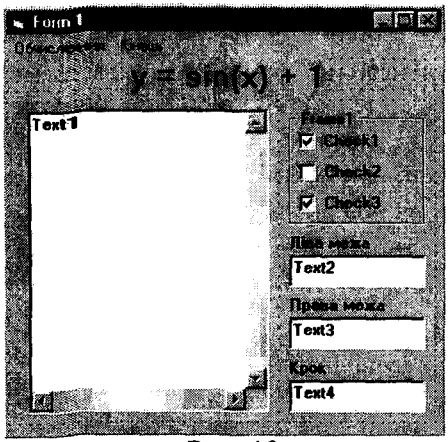


Рис. 10

11.Встановіть прапорці "На екран" та "У масив" у положення "увімкнено".

Для цього виокремте ці об'єкти та змініть значення їхніх властивостей Value на Checked.

12.Введіть назви команд головного меню форми (рис. 11-13).

Команди головного меню, як і інші компоненти Visual Basic, є об'єктами. Для створення команд виберіть елемент головного меню Tools ⇒ Menu Editor. У вікні, що з'явиться, введіть назви команд меню (властивість Caption) і їхні імена (властивість Name), щоразу натискаючи на кнопку "Next":

Caption	Name
Обчислення	mnuCalc
Протабулювати	mnuTabul
Очистити поле виведення	mnuClear
Кінець	mnuFinish
Про програму	mnuAbout
Кінець	mnuEnd

Задайте ієрархію команд за допомогою стрілок: ← (підпорядкувати) та → (вивести із підпорядкування). Змінити послідовність команд можна за допомогою стрілок: ↑ та ↓.

13*.Запрограмуйте команду "Очистити поле виведення" головного меню.

Клацніть на команді головного меню форми "Очистити поле виведення", не запускаючи програму на виконання. З'явиться заготовка процедури реакції на подію виклику цієї команди. У ній запишіть команду присвоєння порожнього рядка для очистки поля редагування Text1:

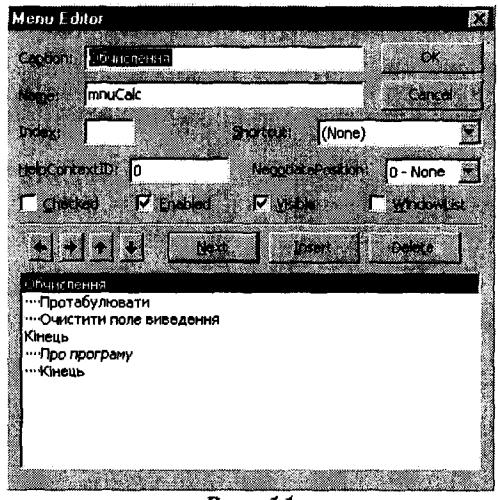


Рис. 11

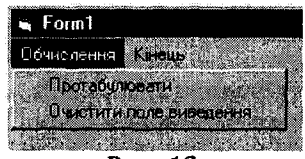


Рис. 12

Рис.13

```
Private Sub mnuClear_Click()
Text1.Text = ""
End Sub
```

14*.Запрограмуйте команду "Кінець" головного меню, скориставшись стандартною процедурою End:

```
Private Sub mnuEnd_Click()
End
End Sub
```

Збережіть виконану на даний момент форму у своїй папці (File⇒Save Project).

15*.Запустіть створену програму та дослідіть її роботу.

Поекспериментуйте з багаторядковим полем редагування Text1, вводячи та коректуючи у ньому будь-який текст. Зверніть увагу на те, що в цьому вікні можна виконувати такі ж дії з текстом, як і в текстовому редакторі: виокремлювати фрагмент тексту, копіювати, переносити чи вилучати фрагмент. Витріть текст за допомогою команди головного меню "Очистити поле виведення". Закінчіть роботу програми, клацнувши на команді "Кінець".

16.Запрограмуйте команду "Протабулювати".

Виконайте команду "Протабулювати" з головного меню форми, клацнувши на ній один раз. З'явиться заготовка процедури, яку заповніть так:

```
Private Sub mnuTabul_Click()
Dim x, y As Double   'x – ордината, y – значення ф-ії
Dim NewLine, Space As String

NewLine = Chr(13) + Chr(10)          'Символ "Enter"
Space = Chr(9)                       'Символ "Tab"
Text1.Text = "X" + Space + "Y" + NewLine  'Шапка таблиці

For x = Text2.Text To Text3.Text Step Text4.Text
   y = Sin(x) + 1
   If Check1.Value = Checked Then 'Якщо встановлений
                                  'прапорець "На екран"
      'Друкуємо рядок таблиці
      Text1.Text = Text1.Text + Str(x) + _
      Space + Str(Format$(y, "0.0000")) + NewLine
      End If 'Кінець Якщо
   Next       'Кінець циклу
End Sub       'Кінець процедури
```

17. **Виконайте програму і поекспериментуйте з різними значеннями лівої, правої межі та кроку зміни аргумента. Закрийте вікно програми "Табулювання функції".**
18. **Збережіть створену програму у своїй папці.**
19. **Продемонструйте створену форму викладачеві. Закінчіть роботу.**

Задачі

Задача 3.1. Модифікуйте реалізацію програми, передбачивши можливість табулювання функції і її похідної. Вибір варіанта табулювання (з похідною чи без неї) здійснити за допомогою додаткового прапорця.

Підказка. Виконайте такі дії:

- *вставте у форму об'єкт типу CheckBox (прапорець), надайте його властивості Caption значення "Похідна", виберіть для підпису один із кирилізованих шрифтів 12-го розміру, вирівняйте вставлений об'єкт.*
- *змініть програмний код кнопки "Протабулювати", використавши в тексті її процедури такі команди:*

```
If CheckBoxN.Value=Checked Then
'Якщо прапорець похідної встановлений
      Text1.Text = "X" + Space + "Y" + Space + "Y'" +_
       NewLine
      'Рядок  містить підписи для стовпців аргумента,
      'значення функції та її похідної
Else  'Якщо прапорець не встановлений
      Text1.Text = "X" + Space + "Y" +  NewLine
      'Рядок  містить лише підписи для стовпців
      'аргумента та значення функції
End If
• • •
'У циклі табулювання:
y = Sin(x) + 1              'Обчислюємо значення функції
y1 = Cos(x)                 'Обчислюємо значення похідної
If CheckBoxN.Checked Then   'Якщо прапорець N
                            'встановлений
      Text1.Text = Text1.Text + Str(x) + _
      Space + Str(Format$(y, "0.0000")) + _
      Space + Str(Format$(y1, "0.0000")) + NewLine
      'Формуємо символьний рядок з аргумента, значення
      'функції і її похідної, між якими є декілька пропусків
Else  'Якщо прапорець не встановлений
      • • • 'Фрагмент процедури із п. 16
```

Задача 3.2. Визначити кількість значень функції більших, ніж 0,5 і менших, ніж 1.

Задача 3.3. Передбачте у створеній програмі додаткову можливість для визначення максимального та мінімального значень функції.

Підказка. У тілі процедури кнопки "Протабулювати", скористайтеся такими командами:

```
'На початку процедури:
max = Sin(a) + 1
• • •
'У циклі табулювання:
If max<y Then
        max = y
        End If
• • •
'Після циклу табулювання:
Text1.Text = Text1.Text + Str(max)
```

Задача 3.4. Змініть процедуру команди "Протабулювати" так, щоб для увімкненого прапорця "У масив" виведення результатів виконувалось в одновимірний масив.

Задача 3.5 (побудова графіка). Виконайте команду "Components" контекстового меню панелі компонентів, і в одержаному вікні на закладці Controls встановіть прапорець у позиції ChartFX 2.0 OLE Custom Control ⇒ Ok. Вставте у форму об'єкт ChartFX (діаграма) для побудови графіка функції. Виберіть контекстове меню Properties (Властивості) цього об'єкта і на закладці Appearance у списку Gallery Type виберіть піктограму потрібного графіка. На закладці 3DView заберіть прапорець 3D. На закладці DataValues задайте очікувані (орієнтовно) максимальне і мінімальне значення функції ⇒ Ok. Створіть додатковий пункт меню або вставте кнопку "Намалювати графік" для отримання графіка і запрограмуйте (наприклад, кнопку) так:

```
Private Sub Command1_Click()
Dim x As Double
Dim n, i, cod As Long
x = 1          'Початковий x
n = 20         'Кількість точок на графіку
cod = ChartFX1.OpenDataEx(COD_VALUES, 1, n)
For i = 0 To n - 1
   ChartFX1.Value(i) = Sin(x) + 1      'Будуємо графік
   ChartFX1.Legend(i) = x              'Формуємо легенду
   x = x + 0.2                         'Наступне значення x
   Next
ChartFX1.CloseData (COD_VALUES)
End Sub
```

Задача 3.6. В умовах задачі 3.5 використайте задані на формі значення лівої і правої меж графіка та кроку розбиття.

§ 5. ЗАДАЧА ПРО АДРЕСНУ КНИЖКУ. Практична робота № 4. Файли записів

Мета роботи. Створити програму, яка працюватиме з базою даних — адресною (записною) книжкою, сформованою у вигляді файлу записів (рис. 16). Полями кожного запису є: прізвище і ім'я (рядок символів довжиною до 50 символів) та особисті дані (рядок довжиною до 500 символів). Необхідно реалізувати наступні операції: створення та вилучення запису, збереження та зчитування з диска файлу записів, редагування та пошук потрібних даних, перегляд записів та навігацію по них.

Теоретичні відомості. Опис глобальних типів у Visual Basic здійснюється в окремому модулі. Вікно модуля можна відкрити за допомогою команди головного меню Project ⇒ Add Module ⇒ Open (Відкрити). Заготовку власної функції чи процедури можна вставити у код програми за допомогою команди головного меню Tools ⇒ Add Procedure. Перехід по потрібних модулів і форм проекту здійснюється за допомогою вікна, яке відкривається командою головного меню View ⇒ Project Explorer (рис. 14).

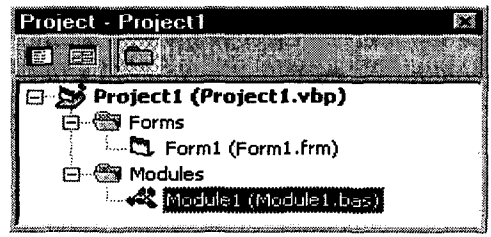


Рис. 14

Відкрити потрібний компонент проекту можна, двічі клацнувши по ньому мишею у вікні проекту.

Хід роботи

1. **Завантажте середовище Visual Basic.**
2. **Заповніть форму візуальними об'єктами (рис. 15).**
   Назви кнопок показані на рис. 16. Властивість MultiLine об'єкта Text2 задайте True.
3*. **Значення властивості ScrollBars об'єкта Text2 задайте Both. Властивість BorderStyle форми задайте Fixed Single.**
4. **Збережіть проект у своїй папці.**
5. **Введіть опис типу запису TPerson і опис глобальних змінних програми.**
   Для цього відкрийте вікно нового модуля за допомогою команди головного меню Project ⇒ Add Module ⇒ Open (Відкрити) і введіть у нього опис типу TPerson:

```
Type TPerson
  Name      As String * 50
  Comment   As String * 500
  End Type
```

Поверніться у вікно коду форми і введіть у нього опис глобальних змінних програми:

```
Dim Person   As TPerson   'Змінна типу запису TPerson
Dim N, i     As Integer   'Номери останнього і активного
записів
```

Зауваження. Одержати доступ до секції опису глобальних змінних можна, вибравши у лівому верхньому комбінованому списку редактора коду елемент (General), а у правому списку — елемент (Declarations).

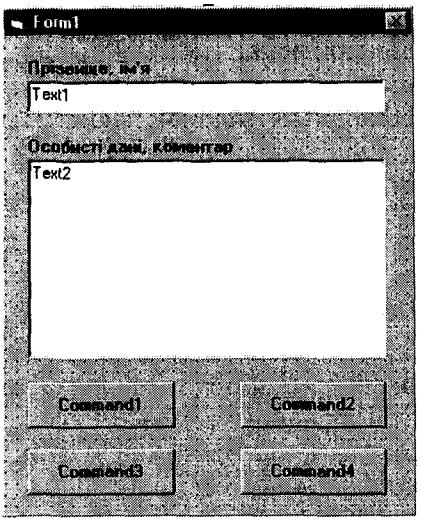


Рис. 15

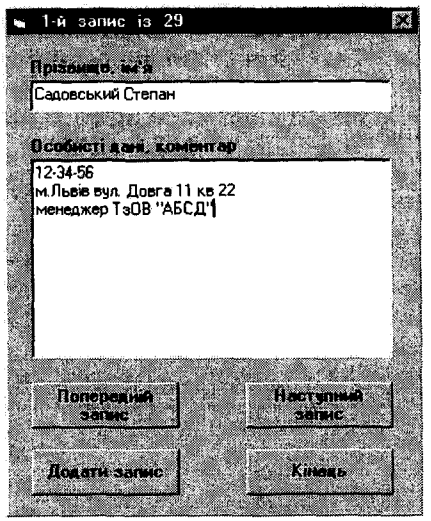


Рис. 16

6. **Запрограмуйте процедуру створення форми Form_Load():** Двічі клацніть на вільному місці форми і введіть текст процедури:

```
Private Sub Form_Load()
  'Відкриваємо файл "data.dat", перший вільний номер файла = 1
Open "data.dat" For Random As 1 Len = Len(Person)
  'К-сть записів у файлі = к-сть байтів у файлі / к-сть байтів у записі
N = FileLen("e:\data.dat") / Len(Person)
  'Друкуємо номер запису і їхню кількість у заголовку форми
Form1.Caption = Str(i) + "-й  запис  із " + Str(N)
If N > 0 Then                'Якщо файл не порожній –
    i = 1
    ReadPerson               'зчитуємо перший запис,
Else: Command3_Click         'інакше – додаємо перший запис
End If
End Sub
```

7. **Створіть власні процедури зчитування запису з файлу у форму (ReadPerson) і збереження активного запису у файлі (WritePerson).** Для цього клацніть на формі і виконайте команду головного меню Tools ⇒ Add Procedure. У рядку Name введіть ім'я процедури ReadPerson. Встановіть перемикачі групи Type у позицію Sub

(процедура), а групи Scope у позицію Public (загальнодоступн
⇒ Ok. Заповніть одержану заготовку кодом:

```
Public Sub ReadPerson()
Get #1, i, Person              'Зчитуємо і - тий запис у змінну Person
Text1.Text = Person.Name       'Відображаємо поля запису на формі
Text2.Text = Person.Comment
  'Друкуємо номер запису і їх кількість у заголовку форми
Form1.Caption = Str(i) + "-й  запис  із " + Str(N)
End Sub
```

Аналогічно створіть процедуру WritePerson:

```
Public Sub WritePerson()
Person.Name = Text1.Text          'Записуємо дані з текстових
Person.Comment = Text2.Text       'полів у змінну Person
Put #1, i, Person                 'Зберігаємо запис Person у файлі
End Sub
```

8. **Запрограмуйте командні кнопки.**

```
Private Sub Command1_Click()      'Попередній запис
WritePerson                       'Зберігаємо зміни в активному записі
If i > 1 Then                     'Якщо запис не перший,
     i = i - 1                    'зменшуємо його номер
     ReadPerson                   'і зчитуємо з файлу
     End If
End Sub
```

```
Private Sub Command2_Click()      'Наступний запис
WritePerson                       'Зберігаємо зміни в активному записі
If i < N Then                     'Якщо запис не останній,
     i = i + 1                    'збільшуємо його номер
     ReadPerson                   'і зчитуємо з файлу
     End If
End Sub
```

```
Private Sub Command3_Click()      'Додати запис
If i > 0 Then                     'Якцо є запис на формі
     WritePerson                  'Зберігаємо його
     End If
N = N + 1                         'Збільшуємо кількість записів
i = N                             'Активний запис на останній позиції
  'Друкуємо номер запису і їх кількість у заголовку форми
Form1.Caption = Str(i) + "-й  запис  із " + Str(N)
Text1.Text = ""                   'Очищаємо поля форми для введення
Text2.Text = ""                   'нового запису
End Sub
```

```
Private Sub Command4_Click()      'Кінець
WritePerson                       'Зберігаємо активний запис
Close #1                          'Закриваємо файл
End                               'Закінчуємо роботу
End Sub
```

9*.Збережіть роботу і запустіть проект на виконання.
Введіть декілька записів і переконайтеся у правильності роботи командних кнопок. Закінчіть роботу кнопкою Кінець. Запустіть програму ще раз і переконайтеся у тому, що записи правильно зчитано з файлу. Додайте ще один запис, введіть у нього дані і відразу закрийте вікно системною кнопкою ⊠. Запустіть програму на виконання і перевірте останній запис. Чому запис не зберігся?

10*.Забезпечте збереження останнього запису і всього файлу після закриття вікна програми системною кнопкою ⊠ чи командами системного меню.
Для цього у лівому верхньому комбінованому списку редактора коду виберіть об'єкт Form, а у правому — подію Unload (вивантажити) для цього об'єкта. У вікні коду з'явиться заготовка процедури Form_Unload, яка виконуватиметься у момент закриття вікна. Введіть команду виклику процедури Command4_Click кнопки "Кінець":

```
Private Sub Form_Unload(Cancel As Integer)
Command4_Click
End Sub
```

11.Збережіть роботу, запустіть програму і переконайтеся у правильності її роботи.

Задачі

Задача 4.1. Вставте і запрограмуйте кнопку "Перейти на останній запис".

Задача 4.2. Вставте і запрограмуйте кнопку "Перейти на перший запис".

Задача 4.3. Вставте і запрограмуйте кнопку "Знайти наступний" так, щоб після натискання на неї здійснювався перехід вперед у файлі на запис, значення поля Name якого збігається з рядком, уведеним у додаткове поле редагування.

Задача 4.4. Вставте і запрограмуйте кнопку "Знайти попередній" так, щоб після натискання на неї здійснювався перехід назад у файлі на запис, значення поля Name якого збігається з рядком, уведеним у додаткове поле редагування.

Задача 4.5. Вставте і запрограмуйте кнопку "Вилучити запис" так, щоб після натискання на неї вилучався активний запис.

Підказка. У процедуру цієї кнопки введіть цикл перезапису всіх записів файлу, що містяться за вилученим, на одну позицію назад:

```
Dim j As Integer
Get #1, i + 1, Person                 'У форму
Text1.Text = Person.Name              ' заноситься наступний
Text2.Text = Person.Comment           ' запис
For j = i + 1 To N
   Get #1, j, Person                  'Перезапис на одну
   Put #1, j - 1, Person              'позицію назад
   Next
Person.Name = ""                      'Очищаємо
Person.Comment = ""                   ' останній
Put #1, N, Person                     ' запис
N = N - 1                             'Коригуємо напис у заголовку
Form1.Caption = Str(i) + "-й  запис  із " + Str(N)
```

§ 6. СТВОРЮЄМО НАВЧАЛЬНУ ПРОГРАМУ. Практична робота № 5. Мультимедійні ефекти. Об'єкти: фігура, таймер, мультимедійний програвач та індикатор стану

Мета роботи. Розробити програму для перевірки знань англійських слів шляхом тестування. Користувач повинен за обмежений час методом перетягування розташувати три малюнки під відповідними англійськими словами (рис. 18). Застосувати індикатор часу виконання завдання, звукові ефекти та вивести вікно-повідомлення з підсумками тестування.

Отримати навички роботи з такими об'єктами: геометрична фігура (**Shape**), таймер (**Timer**), мультимедійний програвач (**MediaPlayer**), індикатор стану (**ProgressBar**).

Теоретичні відомості. Розглянемо призначення і властивості деяких нових об'єктів. Геометрична фігура (**Shape**, піктограма ) призначена для зображення елементарних геометричних фігур і має, зокрема, такі властивості:

Властивість	Опис властивості	Приклади значень
Shape	Форма фігури	Rounded Rectangle (прямокутник з округленими краями)
BorderWidth	Товщина границі фігури	1, 5 (ціле число)

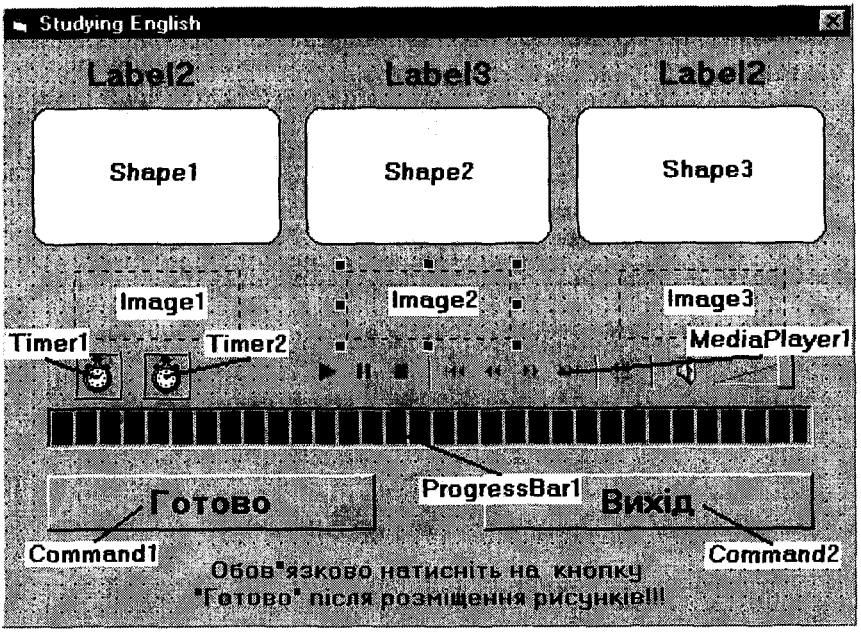


Рис. 17

Таймер (Timer, піктограма) використовують для повторення фрагмента коду програми з певною періодичністю. Відповідний фрагмент розташовують у тілі процедури опрацювання події Timer таймера. Періодичність вмикання таймера у мілісекундах задають властивістю **Interval.**

Мультимедійний програвач (**MediaPlayer,** піктограма) призначений для програвання відео- та аудіофайлів. Встановити піктограму програвача на палітру компонентів можна, виконавши команду контекстового меню палітри ToolBox: Components... ⇒ закладка Controls ⇒ прапорець Windows Media Player ⇒ Ok. Керування програвачем може здійснюватися як за допомогою традиційних кнопок Play, Pause, Stop, Next тощо на етапі виконання програми, так і з програмного коду шляхом виконання методів цього об'єкта, наприклад:

MediaPlayer1.FileName:= "повне ім'я відео- чи аудіофайлу";

Індикатор стану (**ProgressBar,** піктограма) використовують для наочної демонстрації стану виконання деякого процесу. Встановити піктограму індикатора стану на палітру компонентів можна, виконавши команду контекстового меню палітри ToolBox: Components... ⇒ закладка Controls ⇒ прапорець Microsoft Windows Common Controls 6.0 ⇒ Ok. Розглянемо три властивості індикатора:

Властивість	Опис властивості	Приклади значень
Orientation	Орієнтація індикатора	ccOrientationalHorizontal (горизонтальний рядок)
Value	Відображає стан індикатора	Ціле число між **Max** і **Min**
Scrolling	Вигляд біжучого рядка індикатора	ccScrollingStandard (стандартний), ccScrollingSmooth (суцільний)

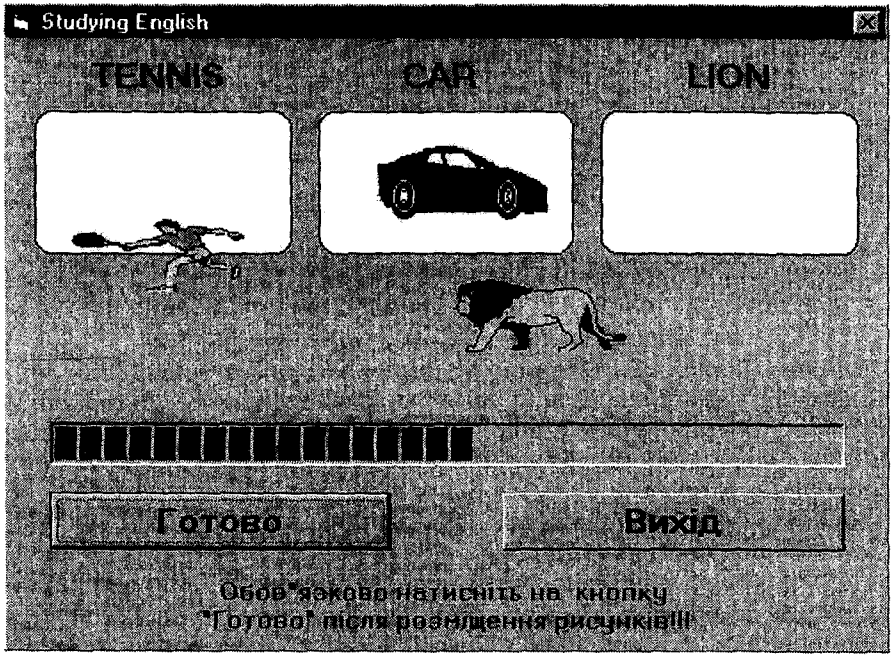


Рис. 18

Хід роботи

1. **Завантажте середовище Visual Basic.**
2*. **Змініть заголовок (Caption) з "Form1" на "Studying English".**
3*. **Відмовтесь від системних кнопок згортання та максимізації форми, задавши значення False для її властивостей MaxButton і MinButton.**

   Роботу з програмою будемо завершувати натисканням на кнопку Вихід.
4. **Розташуйте на формі об'єкти так, як показано на рис. 17. Задайте такі властивості об'єктів:**

Об'єкт	Властивість	Значення
Timer1	Interval	500
Timer2	Interval	200
MediaPlayer1	Visible	False

Зауваження. Розміри фігур повинні бути більшими, ніж розміри малюнків, оскільки останні слід розташовувати строго у середині фігур. Якщо під час накладання рисунка на фігуру рисунок зникає, перемістіть його на передній план командою його контекстового меню Bring To Front.

5. **Уведіть опис глобальних змінних програми:**

```
Dim shiftX, shiftY, Press As Integer
```

6*. **Двічі клацніть на першому таймері і запрограмуйте блимання повідомлення "Обов'язково натисніть на кнопку "Готово" після розміщення рисунків!!" так:**

```
Private Sub Timer1_Timer()
If Label1.Visible = True Then      'Якщо поле світиться,
   Label1.Visible = False          'його гасимо,
   Else: Label1.Visible = True     'інакше – засвічуємо
   End If
End Sub
```

7. **Запрограмуйте кнопку Command2 як кнопку завершення роботи програми командою End.**
8. **Вставте малюнки car.wmf, lion.wmf і tennis.wmf з папки C:\Program Files\ Microsoft Office\ Clipart \ Popular в об'єкти Image1, Image2 і Image3 відповідно.**
9. **Властивостям Caption текстових полів Label2, Label3 і Label4 надайте значення "TENNIS", "CAR" і "LION" відповідно.**
10. **Запрограмуйте процедуру створення форми Form_Load:**

```
Private Sub Form_Load()
ProgressBar1.Value = 100    'Встановлюємо рядок індикації
                            'часу виконання на 100%
End Sub
```

11. **Забезпечте можливість перетягування першого малюнка на формі, запрограмувавши опрацювання таких трьох подій для об'єкта Image1: MouseDown (НатисканняМиші), OnMouseMove (ПереміщенняМиші) та OnMouseUp (ВідпусканняМиші).**

Для вставляння у програму заготовок згаданих процедур скористайтесь лівим та правим комбінованими списками, розміщеними у верхній частині вікна редактора коду. У лівому списку виберіть компонент Image1, а у правому — потрібну подію для цього компонента.

```
Private Sub Image1_MouseDown(Button As Integer, _
          Shift As Integer, x As Single, y As Single)
Press = 1          'Запам'ятовуємо, що клавіша миші
                   'натиснута на об'єкті
shiftX = x         'Запам'ятовуємо координату (x;y) точки
shiftY = y         'клацання мишею в середині малюнка
End Sub
```

```
Private Sub Image1_MouseMove(Button As Integer, _
          Shift As Integer, x As Single, y As Single)
If Press = 1 Then      'Якщо натиснута клавіша миші,
     'змінюємо координати малюнка на величину зміни
     'координати вказівника миші (x;y) з урахуванням
     'його зміщень в середині малюнка shiftX, shiftY
     Image1.Top = Image1.Top + y - shiftY
     Image1.Left = Image1.Left + x - shiftX
     End If
End Sub
```

```
Private Sub Image1_MouseUp(Button As Integer, _
          Shift As Integer, x As Single, y As Single)
Press = 0      'Запам'ятовуємо, що клавіша миші
               'відпущена на зображенні Image1
End Sub
```

Програма не перевіряє, на яку саме клавішу миші натиснув користувач, і тому перетягування можна здійснювати довільною клавішею. Перевірити клавіші миші можна, проаналізувавши аргумент Button (типу Integer) наведених процедур:

Значення Button	Натиснуті кнопки миші
1	ліва
2	права
3	ліва і права
4	середня
5	ліва і середня
6	права і середня
7	ліва, права і середня

Координати вказівника миші передаються у процедуру за допомогою аргументів x і y типу Integer.

12. Аналогічно запрограмуйте відповідні події для перетягування двох інших малюнків.

13. Створіть власну функцію ImageInShape для перевірки розташування малюнка (об'єкта Image) в середині деякої геометричної фігури (об'єкта Shape).

Послідовність створення власної процедури описана у пункті 7 попередньої роботи.

```
Public Function ImageInShape(I As Image, S As Shape)
If I.Left >= S.Left And _
   I.Left + I.Width <= S.Left + S.Width And _
   I.Top >= S.Top And _
   I.Top + I.Height <= S.Top + S.Height Then
          ImageInShape = True     'Малюнок є в середині фігури
Else
          ImageInShape = False    'Малюнок є поза фігурою
End If
End Function
```

14. Запрограмуйте кнопку "Готово", яка перевіряє правильність розташування малюнків в середині фігур і висвітлює стандартне інформаційне вікно Windows з повідомленням "Правильно!" чи "Не правильно!".

```
Private Sub Command1_Click()
Timer1.Enabled = False        'Зупиняємо блимання Label1
Timer2.Enabled = False        'Зупиняємо індикатор часу
If ImageInShape(Image1, Shape2) And _
   ImageInShape(Image2, Shape3) And _
   ImageInShape(Image3, Shape1) Then
          'Зчитуємо файл аплодування
          MediaPlayer1.FileName = "applause.wav"
          'Відкриваємо вікно повідомлення "Правильно!"
          MsgBox "Правильно!"
Else:     MediaPlayer1.FileName = "notify.wav"
          'Відкриваємо вікно повідомлення "Неправильно!"
          MsgBox "Неправильно!"
          'Вмикаємо обидва таймери та індикатор часу
          Timer1.Enabled = True
          Timer2.Enabled = True
          ProgressBar1.Value = 100
End If
End Sub
```

15. Запрограмуйте другий таймер, який забезпечує індикацію часу виконання від 100% до 0.

Якщо користувач не встиг розташувати малюнки у прямокутниках і натиснути на кнопку "Готово" у відведений час, подається відповідний звук і відкривається стандартне інформаційне вікно Windows з повідомленням "Не встигли. Спробуйте ще раз!". Після того, як користувач натисне на кнопку Ok цього вікна, знову запускаються таймери блимання підпису в нижній частині вікна і руху індикатора часу.

```
Private Sub Timer2_Timer()
'Працює індикатор часу:
ProgressBar1.Value = ProgressBar1.Value - 1
If ProgressBar1.Value = 0 Then        'Якщо час вичерпано
    Timer1.Enabled = False
    Timer2.Enabled = False
    MediaPlayer1.FileName = "chimes.wav"
    MsgBox "Не встигли. Спробуйте ще раз!"
    Timer1.Enabled = True
    Timer2.Enabled = True
    ProgressBar1.Value = 100
    End If
End Sub
```

16. Збережіть програму та проект у робочій папці.

17. Скопіюйте музичні файли chimes.wav, applause.wav та notify.wav у свою робочу папку та папку Visual Basic C:\ Program Files \ Microsoft Visual Studio \ VB98 з папок C:\ WINDOWS \ MEDIA \ Office97 та C: \ WINDOWS \ MEDIA.

18. Запустіть програму. Перетягуйте мишею малюнки в середину відповідних фігур.

Задачі

Задача 5.1. Забезпечте можливість перетягування малюнків лише лівою клавішею миші (див. пункт 11).

Задача 5.2. Збільшіть удвічі частоту блимання тексту "Обов'язково натисніть на кнопку "Готово" після розміщення рисунків!!".

Задача 5.3. Змініть тип (властивість Orientation) індикатора стану на вертикальний, скоригувавши форму і розташування об'єктів.

Задача 5.4. Підберіть найвдалішу швидкість руху індикатора часу.

Задача 5.5. Вставте у форму три додаткові рисунки, фігури та підписи до них. Змініть текст програми, передбачивши у ній опрацювання доданих об'єктів.

§ 7. ПОНЯТТЯ ПРО СЕРЕДОВИЩЕ ПРОГРАМУВАННЯ VISUAL BASIC FOR APPLICATIONS (VBA).

Практична робота № 6. Програмування в середовищі VBA для програми Microsoft Word

Мета роботи. Створити для програми Microsoft Word макрос мовою VBA для перетворення тексту, набраного на клавіатурі в помилково заданій іншомовній розкладці, у текст потрібною мовою.

Теоретичні відомості. Середовище програмування Visual Basic for Application (VBA) створене фірмою Microsoft на базі середовища Microsoft Visual Basic 3.0 для Windows. Уперше використавши VBA у MS Excel і MS Project, фахівці Microsoft застосували його у популярних програмах, таких як Word, Mail, PowerPoint. VBA входить до складу й інших не менш популярних програм, наприклад, 1С.

Для освоєння і використання VBA достатньо знайомства з одною з програм з Microsoft Office і знань мови Бейсик у рамках даної книжки. Мова Бейсик, починаючи від версії Qbasic, завжди була під пильною увагою особисто Білла Гейтса – засновника фірми Microsoft. На його думку, знань цієї мови досить для розв'язування прикладних задач з різних областей моделювання і програмування.

Процес створення програми мовою VBA автоматизований настільки, що користувач може навіть не здогадуватися про те, що він займається програмуванням. Досить у середовищі офісної програми відкрити панель для записування макросу і виконати потрібні дії за допомогою маніпулятора-миші та клавіатури і буде створена нова програма. Код програми генерується автоматично. Згодом достатньо просто викликати створений макрос (створену програму), який повторить дії користувача.

Іноді розв'язки деяких задач у середовищах офісних програм одержати традиційним способом, тобто шляхом записування макроса, або складно або в принципі неможливо. Тоді на допомогу знову приходить VBA, але тепер уже користувачеві потрібні знання про синтаксис мови, елементи керування і властивості об'єктів, до яких будуть застосовані значні можливості мови VBA.

Нарешті вищий рівень використання VBA – це розв'язування задач інтеграції програм, тобто задач обміну даними між програмами Microsoft Office і створення для них додаткових можливостей.

Отже, VBA – це середовище програмування, що є інтегрованою з офісними програмами оболонкою, яка призначена для створення невеликих програм, які називаються макросами.

Головне вікно VBA (вікно створення проекта) стандартно складається з трьох вікон: вікна редактора Edit Window, вікна властивостей Properties Window і вікна оглядача проекта Projecr Explorer.

У головному вікні є основні засоби для роботи з програмою: рядок меню, панель інструментів Standard, панель налагодження Debug, панель редагування Edit і панель налаштування форми UserForm.

У вікні властивостей Properties Window зібрані основні властивості вибраного (виокремленого) об'єкта або елемента керування. Це вікно потрібне тільки під час роботи з формою в режимі конструктора. Такий режим для середовища VB описувався у попередніх параграфах.

У вікні редактора коду Code Editor формується програмний код для розв'язування деякої задачі.

Вікно оглядача проекта Project Explorer є аналогом провідника у Windows і призначене для перегляду структури відкритих проектів і навігації між їхніми компонентами.

Основи мови складають типи даних, команди, візуальні компоненти керування та об'єкти, до яких усе це застосовується.

У VBA використовується 12 стандартних (базових) типів змінних:

- Byte – однобайтова змінна з цілочисельним значенням від 0 до 255;
- Integer – двобайтова цілочисельна змінна, що отримує значення від –32 768 до +32 767;
- Long – чотирибайтова цілочисельна змінна зі значеннями від – 2 147 483 648 до + 2 147 483 647;
- Decimal – змінна для цілих десяткових чисел із тридцятьма значущими цифрами і для дійсних чисел зі значеннями з діапазону від –7,9... до +7,9... з двадцятьма дев'ятьма значущими цифрами після коми;
- Single – змінна для чисел з "плаваючою комою" зі значеннями з діапазонів від $-3{,}402\ 823 \cdot 10^{38}$ до $-1{,}401\ 298 \cdot 10^{-45}$ для від'ємних чисел і від $+1{,}401\ 298 \cdot 10^{-45}$ до $+3{,}402\ 823 \cdot 10^{38}$ для додатних;
- Double – змінна для чисел з "плаваючою комою" подвійної точності зі значеннями з діапазонів від $-1{,}797\ 693\ 134\ 862\ 32 \cdot 10^{308}$ до $-4{,}940\ 656\ 412\ 412\ 47 \cdot 10^{-324}$ для від'ємних чисел і від $+4{,}940\ 656\ 412\ 412\ 47 \cdot 10^{-324}$ до $1{,}797\ 693\ 134\ 862\ 32 \cdot 10^{308}$ для додатних;
- Boolean – змінна логічного типу, що отримує значення True (істина) або False (хибність);

- String – рядкова змінна, що може містити до 65000 символів у 16-розрядних операційних системах або до 2^{32} символів у 32-розрядних операційних системах, наприклад, у Windows 98, XP;
- Date – змінна для дати/часу зі значеннями від 1 січня 100 року до 31 грудня 9999 року;
- Currency – змінна для грошових величин з можливими значеннями з діапазону від –922 337 203 685 477,5808 до +922 337 203 685 477,5807;
- Object – змінна об'єктного типу, що містить посилання на об'єкт;
- Variant – змінна будь-якого типу.

Інші структури даних: записи, масиви, файли і т. п. розглядають як похідні типи, які створюють користувачі для розв'язування тієї або іншої задачі.

Команди у мові VBA практично ті ж, що й у звичайному VB.

Складання програм (процедур) принципово не відрізняється від програмування в середовищах VB і Qbasic.

Інтерфейс користувача у проектах VBA реалізується за допомогою візуальних компонентів, які ще називаються ***елементами керування*** (controls). Ці елементи дають змогу реалізувати практично будь-які задуми. З їхньою допомогою створюють більшість елементів інтерфейсу: кнопки, списки, поля введення та інші, детально розглянуті в попередніх параграфах.

Хід роботи

1. **Запустіть програму Microsoft Word, увімкніть англійську розкладку клавіатури і наберіть деякий текст, вважаючи, що текст треба ввести українською мовою.**
2. **Створіть макрос під назвою Convertor (перетворювач).**
   Для цього виконайте команду з головного меню Tools ⇒ Macro ⇒ Macros (або натисніть на комбінацію клавіш Alt-F8) ⇒ у рядку Macro name введіть ім'я Convertor і натисніть на Create (Створити). Відкриється ще одне вікно "Microsoft Visual Basic – Normal", у якому створюватиметься макрос.
3. **Відкрийте вікна, необхідні для роботи в VBA:**
   - вікно проекта Project-Normal командою View ⇒ Project Explorer;
   - вікно тексту програми Normal – NewMacros (Code) командою View ⇒ Code (це вікно вже може бути відкрите);
   - вікно властивостей Properties-NewMacros командою View ⇒ Properties;
4. **Створіть вікно "Перетворення тексту" з кнопками "Ok (Уперед)" та "Cancel (Відмінити)" (див. рис. 19).**

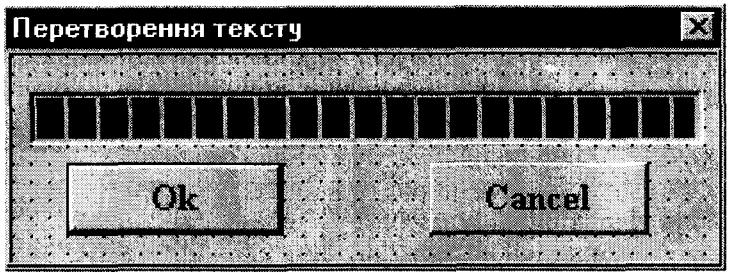


Рис.19

Для цього виконайте команду Insert ⇒ UserForm. З'явиться форма UserForm1, швидкий доступ до якої здійснюється подвійним клацанням на пункті UserForm1 розділу Forms вікна проекта Project-Normal. Змінити колір і заголовок форми можна звичайним способом за допомогою вікна властивостей форми. Кнопки вибираються з вікна Toolbox за допомогою піктограми CommandButton.

Зауваження. Після активізації форми з'явиться вікно інструментів Toolbox, закрити чи знову відкрити яке можна за допомогою команди головного меню View ⇒ Toolbox.

5. **Вставте у форму індикатор стану виконання процесу перетворення тексту Pogress Bar.**
   Для цього виконайте команду контекстового меню вікна інструментів Additional Controls... У вікні, що з'явиться, виберіть пункт Microsoft ProgressBar Control ⇒ Version x.0 ⇒ Ok. Розташуйте індикатор стану у формі так, як показано на рис. 19, скориставшись піктограмою ProgressBar вікна Toolbox. Вибрати вигляд індикатора стану можна, змінивши його властивість Scrolling.
6. **Активізуйте вікно форми і запустіть програму (форму) на виконання, клацнувши на кнопці (Run Sub/UserForm).**
   Закрийте форму. Активізуйте будь-яке інше вікно VBA і знову спробуйте запустити форму. Нічого не відбудеться, оскільки тепер уже запускається макрос, а в ньому не передбачена процедура відкривання створеної форми.
   Зауваження. Знову відкрити форму можна, двічі клацнувши по її імені у вікні проекта.
7. **Запрограмуйте відкривання вікна "Перетворення тексту" у момент запуску макроса.**

Для цього у вікні з текстом програми Normal–NewMacros (Code) у процедурі Convertor викличте метод Show (показати) для форми UserForm1:

```
Sub Convertor()
UserForm1.Show
' Вставляємо цей рядок
'Convertor Macro
'Macro created 07.05.02 by user1
'
End Sub
```

Тепер запуск макроса в будь-якому випадку приведе до відкривання вікна форми UserForm1. Переконайтеся у цьому.

8. **Поставте у відповідность створеному макросу комбінацію "гарячих" клавіш Ctrl+Shift+C.**

Для цього поверніться у вікно Microsoft Word командою View ⇒ Microsoft Word або через панель задач і виконайте команду головного меню View ⇒ Toolbars ⇒ Customize... На закладці Commands вікна Customize клацніть на кнопці Keyboard... У вікні, що відкриється, у списку Categories виберіть Macros, а у вікні Commands – наш макрос Convertor. Клацніть у рядку Press new shortcut key ⇒ натисніть комбінацію клавіш Ctrl + Shift + C і присвойте її макросові кнопкою Assign ⇒ Close. Переконайтесь, що в Microsoft Word вікно форми відкривається комбінацією клавіш Ctrl+Shift+C.

Зауваження. Присвоєння макросу комбінації клавіш може не відбутися, якщо ця комбінація вже використовується для іншого макроса.

9. **Запрограмуйте кнопку Ok, двічі клацнувши по ній.**

Відкриється вікно з текстом програми форми Normal– UserForm1 (Code), де запрограмуйте процедуру CommandButton1_Click():

```
Private Sub CommandButton1_Click()
Dim i As Long          'Номер символу
Dim b As String
Dim newword As String      'Перетворений рядок
Word = Selection
' Вихідний рядок=виокремлений фрагмент.
```

```
' Тут і нижче word є глобальною змінною
' типу string, яку опишемо пізніше.
' Ця змінна міститиме виокремлений
' фрагмент тексту, який перетворюється.

For i = 1 To Len(word)
        Select Case Mid(word, i, 1)
                Case "q":  b = "й": GoTo fin
                Case "w":  b = "ц": GoTo fin
                Case "e":  b = "у": GoTo fin
                Case "r":  b = "до": GoTo fin
                '... і т. і. див. клавіатуру.
                'Переприсвоюємо інші символи:
                Case "m":  b = "ь": GoTo fin
        Case Else:
        'В інших випадках символи не міняємо:
                b = Mid(word, i, 1)
        End Select

        fin:
        'Пересуваємо смужку індикатора стану:
        ProgressBar1.Value = Int(100 * i / Len(word))
        Newword = newword + b
'Приєднуємо перетворений символ
        Next I
'Розглядаємо наступний символ
WordBasic.Insert newword
End
'Закриваємо форму
End Sub
```

10. Опишіть глобальну змінну word.

Для цього у лівому випадаючому списку вікна Normal – UserForm1 (Code) виберіть (General), а у правому, – (Declaration). Введіть опис змінної:

```
Dim word As String
```

11. Закрийте вікно VBA, виокремте неправильно набраний фрагмент тексту і викличте макрос перетворення тексту комбінацією клавіш Ctrl+Shift+C.

Кнопка Ok здійснює перетворення і закриває вікно макросу, кнопка Cancel – закриває вікно без перетворення тексту.

Задачі

Задача 6.1. Запрограмуйте кнопку "Cancel (Відмінити)" так, щоб після її натискання вікно форми закривалося без перетворення виокремленого фрагмента тексту.
Підказка. Скористайтеся командою закривання форми END.

*Задача 6.2. Забезпечте двосторонню заміну фрагментів тексту для двох мов Українська⇒Англійська і Англійська⇒Українська так: будь-яку українську букву виокремленого фрагмента тексту макрос має замінити відповідною англійською і навпаки. Це доцільно на практиці, оскільки **весь** фрагмент тексту користувач може набрати у невідповідній розкладці клавіатури: англійській чи українській.*
Підказка. Для цього продовжіть список замін у команді CASE процедури CommandButton1_Click( ) так:

```
...
Case "й": b = "q": GoTo fin
Case "ц": b = "w": GoTo fin
Case "у": b = "e": GoTo fin
Case "до": b = "r": GoTo fin
...
```

Задача 6.3. Вставте у форму ще одну кнопку "Append (Вставити)" і об'єкт Label (підпис). Після перетворення тексту одержаний фрагмент має висвітлюватися у вікні як підпис, і тільки після клацання на кнопці "Вставити" вставлятися в основний текст Microsoft Word.

Задача 6.4. В умові задачі 6.3 замініть об'єкт Label (підпис) на TextBox (поле редагування).

Задача 6.5. В умові задачі 6.4 вставте у форму кнопку "Clear (Очистити)" яка замінятиме перетворений фрагмент тексту порожнім рядком. Користувач матиме змогу вибрати, замінити фрагмент тексту перетвореним чи знищити його.

Задача 6.6. В умові задачі 6.5 вставте у форму кнопку "UnDo (Повернути назад)", за допомогою якої відміняється виконане перед цим перетворення тексту.

§ 8. ЗАДАЧА ПРО ТАБЛИЦЮ КОДІВ ASCII.

Практична робота № 7. Програмування в середовищі VBA для програми Microsoft Excel

Мета роботи. Створити макрос мовою VBA для виведення кодової таблиці символів ASCII (рис. 20) у діапазон клітинок електронної таблиці (скор. ЕТ) Microsoft Excel. Навчитись вставляти та програмувати кнопки.

Теоретичні відомості. Передбачається, що читачі знайомі з основними прийомами роботи з ЕТ. Усі візуальні елементи інтерфейсу користувача (перемикачі, кнопки та ін.), а також такі елементи ЕТ, як **Range** (діапазон), **Cells** (клітинки), **WorkSheets** (робочі сторінки) тощо, описуються засобами VBA як об'єкти зі своїми властивостями (атрибутами) та методами.

Розглянемо елемент **Cells**. Атрибут **Cells** з індексами рядка і стовпця, які записуються в круглих дужках, дає доступ до відповідної клітинки ЕТ, наприклад, Cells(1,2) – це клітинка B1 активної сторінки. Цей атрибут використовується об'єктами **Application** (програма), **Range** та **Worksheets**, наприклад так:

```
Application.Cells(1,3).Value=12.3              'C1 = 12.3
Range(Cells(1, 1), Cells(5, 3)).Font.Name = "Arial"
            'У діапазоні A1:C5 задано шрифт Arial
Worksheets(„Sheet1").Cells(5, 3). Formula ="=if(a1=1;a2;a3)"
            'На сторінці 1 у C5 є формула =if(a1=1;a2;a3)
```

Нагадаємо, що функція **Chr(**<код символу>) повертає відповідний символ таблиці кодів ASCII – дане типу рядок.

Хід роботи

1. **Запустіть програму Microsoft Excel.**
2. **Створіть макрос для виведення таблиці кодів ASCII в діапазон ЕТ.**

   Для цього виконайте команди Сервіс ⇒ Макрос ⇒ Макроси з головного меню Excel. У вікні, що з'явиться, у рядку „Ім'я макросу" введіть назву макросу (наприклад, Ascii) і натисніть кнопку „Створити". Відкриється вікно "Microsoft Visual Basic" з заготовкою процедури Ascii. Запрограмуйте її так:

```
Sub Ascii() : Dim i, j As Integer
For i = 0 To 25  'у рядках – перші дві цифри коду символу
For j = 0 To 9   'у стовпцях – остання цифра коду символу
If i * 10 + j < 256 Then  'коди 256–259 не заповнюємо
  Cells(i + 2, j + 2) = Chr(i * 10 + j) 'розміщаємо символ
End If
Next j : Next i : End Sub
```

3. **Розграфіть діапазон клітинок A1:K27. Задайте необхідні формати клітинок та ширину стовпців (рис. 19). Під таблицею розташуйте кнопку „Заповнити таблицю".**
Для вставляння кнопки увімкніть панель інструментів „Форми" (Forms) і виберіть у ній елемент „Кнопка". Обведіть контур для кнопки у потрібному місці робочої сторінки ET. З'явиться вікно „Призначити макрос об'єкту". Виберіть макрос Ascii ⇒ Ok. Змініть напис на кнопці, скориставшись її контекстовим меню.

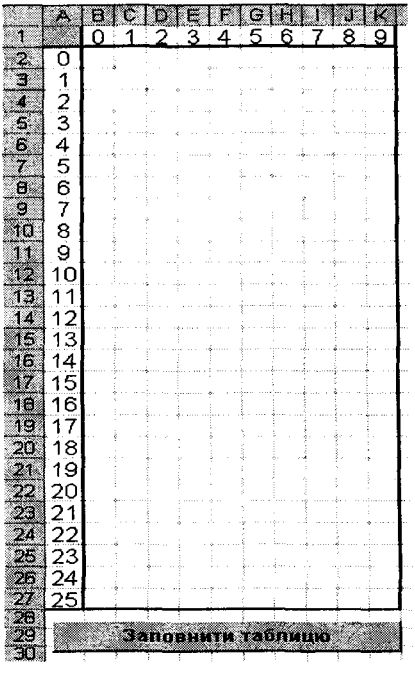


Рис. 19

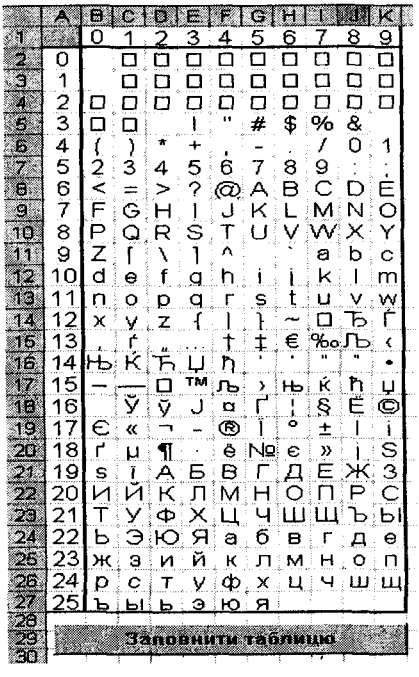


Рис. 20

4. **Клацніть на кнопці „Заповнити таблицю" – отримаєте таблицю кодів ASCII.**
5. **Користуючись таблицею, знайдіть діапазон кодів ASCII, які відповідають символам латинського та кириличного алфавітів. Збережіть відповідний файл ET на диску.**

Задачі

Задача 7.1. Змініть програму так, щоб в таблицю виводилися після натискання кнопок лише малі (або великі) латинські символи.

Підказка. Знайдіть перше входження, наприклад, латинського символу "a" в таблицю ASCII і виведіть лише наступні 26 сим-

*волів. Для цього в подвійному циклі For макросу Ascii використайте оператор If Chr(i * 10 + j)="a" Then . . .*

Задача 7.2. Змініть код макросу так, щоб в таблицю виводилися лише малі (або великі) символи кирилиці.

§ 9. РОЗВ'ЯЗУВАННЯ НЕЛІНІЙНОГО РІВНЯННЯ.

Практична робота № 8. Реалізація методу простих ітерацій засобами VBA у програмі Microsoft Excel.

Мета роботи. Створити макрос розв'язування нелінійного рівняння методом простих ітерацій шляхом побудови таблиці. Ознайомитися з властивостями **Formula**, **ActiveSheet**, **Value**, **Columns**, **Font** та **Color**, а також з методом **Copy**. Отримати навички роботи з об'єктами **WorksheetFunction** та **Application**.

Теоретичні відомості. Щоб розв'язати нелінійне рівняння $f(x)=0$ методом простих ітерацій, його потрібно звести до вигляду $x=z(x)$ так, щоб виконувалась умова $|z'(x)|<1$, яка називається умовою збіжності ітераційного процесу. Наприклад, рівняння $x-\cos x=0$ перетворюють до вигляду $x=\cos x$. Метод реалізують за допомогою рекурентної формули так:

$$x_{i+1}=z(x_i),$$

де x_0 – довільне початкове наближення до розв'язку, $i = 0, 1, 2, 3...$ Обчислення припиняють, коли різниця (яка називається похибкою) $|x_{i+1} - x_i|$ стане меншою від деякого наперед заданого малого числа. Нехай тут це число є 0,001. Якщо така умова виконається, то вважають, що розв'язок знайдено з точністю 0,001.

Формули, які реалізують описаний метод засобами ЕТ, показані на рис. 22. Вручну заповнюємо лише перший рядок таблиці A3:C3. Інші рядки заповнить VBA-програма. Результат роботи програми показані на рис. 21. Зауважимо, що в клітинку A4 програма заносить не формулу, а число – результат обчислень з клітинки B3, в клітинку A5 – результат обчислень з клітинки B4 і т. д.

Для реалізації програми використаємо функцію **CountA (СЧЁТЗ)** MS Excel, яка стосовно деякого діапазону повертає ціле число – кількість не порожніх клітинок у цьому діапазоні. Наприклад, результатом обчислення функції **=СЧЁТЗ**(A1:A21) для таблиці, зображеної на рис.21 буде число 20, адже в цьому діапазоні є 20 не порожніх клітинок. Зазначимо, що у мові програмування VBA доступ до функцій MS Excel здійснюється через об'єкт Worksheet-Function. За замовчуванням він є атрибутом об'єкта Application. У більшості випадків немає необхідності явно зазначати об'єкт WorksheetFunction для доступу до функцій MS Excel, оскільки він за замовчуванням є активним для об'єкту Application тощо. Наприклад, розглянемо три коректні вирази:

```
Application.WorksheetFunction.Max(Columns(1))  '=1, див. рис.21
WorksheetFunction.Pi                           '=3.14159265
Application.Sum(Range("a3:c3"))                '=2
```

Значення **Value** та **Formula** використовують з об'єктами типу **Range** з метою виконання типових для ЕТ дій – надання значень клітинкам та занесення формул. Значенням **Value** є числове значення заданої клітинки. Значенням **Formula** є вираз для формули у вигляді рядка символів, наприклад:

```
Application.Range("A4").Value = 3          'a4=3
Application.Range("a10").Value = 5.8       'a10=5.8
Application.Range("a5").Formula = "=$A$4+$A$10"
'в а5 є формула =$A$4+$A$10, що дає результат  8.8
```

	A	B	C
1	Розв"язування рівняння y=cos(x) методом простої ітерації		Розв'язати рівняння
2	x	cos(x)	| x-cos(x) |
3	0,0000000	1,0000000	1,0000000
4	1,0000000	0,5403023	0,4596977
5	0,5403023	0,8575532	0,3172509
6	0,8575532	0,6542898	0,2032634
7	0,6542898	0,7934804	0,1391906
8	0,7934804	0,7013688	0,0921116
9	0,7013688	0,7639597	0,0625909
10	0,7639597	0,7221024	0,0418573
11	0,7221024	0,7504178	0,0283153
12	0,7504178	0,7314040	0,0190137
13	0,7314040	0,7442374	0,0128333
14	0,7442374	0,7356047	0,0086326
15	0,7356047	0,7414251	0,0058203
16	0,741425	0,7375069	0,003918
17	0,737507	0,7401473	0,002640
18	0,740147	0,7383692	0,001778
19	0,738369	0,7395672	0,001198
20	0,739567	0,7387603	0,000807
21			

Рис. 21

	A	B	C
1	Розв"язування рівняння y=cos(x) методом простої ітерації		Розв'язати рівняння
2	x	cos(x)	| x-cos(x) |
3	0	=COS(A3)	=ABS(A3-B3)
4	1	=COS(A4)	=ABS(A4-B4)
5	0,5403023058	=COS(A5)	=ABS(A5-B5)
6	0,8575532158	=COS(A6)	=ABS(A6-B6)
7	0,6542897904	=COS(A7)	=ABS(A7-B7)
8	0,7934803587	=COS(A8)	=ABS(A8-B8)
9	0,7013687736	=COS(A9)	=ABS(A9-B9)
10	0,7639596829	=COS(A10)	=ABS(A10-B10)
11	0,7221024250	=COS(A11)	=ABS(A11-B11)
12	0,7504177617	=COS(A12)	=ABS(A12-B12)
13	0,7314040424	=COS(A13)	=ABS(A13-B13)
14	0,7442373549	=COS(A14)	=ABS(A14-B14)
15	0,7356047404	=COS(A15)	=ABS(A15-B15)
16	0,7414250866	=COS(A16)	=ABS(A16-B16)
17	0,7375068905	=COS(A17)	=ABS(A17-B17)
18	0,7401473355	=COS(A18)	=ABS(A18-B18)
19	0,7383692041	=COS(A19)	=ABS(A19-B19)
20	0,7395672022	=COS(A20)	=ABS(A20-B20)
21			

Рис. 22

Властивість **ActiveSheet** містить активну сторінку робочої книжки MS Excel. Ця властивість характерна для об'єктів **Application, Windows** (вікна), **WorkBooks** (робочі книжки). Наведені нижче вирази повертають назву ("Sheet8" або "Лист5" тощо) активної сторінки робочої книжки файлу filename.xls:

```
Windows("filename").ActiveSheet.Name
Workbooks("filename.xls").ActiveSheet.Name
Application.ActiveSheet.Name
```

Властивість **Columns** містить стовпець активної сторінки чи певного діапазону. Нею володіють об'єкти **Application, Range, WorkSheet,** наприклад:

```
Worksheets("Sheet1").Columns(1).Font.Bold = True
                ' шрифт у стовпці A буде  півжирним
Range("e5:h9").Columns(2).Value = 25
' клітинки другого стовпця f5:f9 діапазону e5:h9 містять 25
```

Функція **CStr** перетворює числа, дати і логічні вирази у рядкові змінні. Розглянемо фрагменти коду

```
Dim MyDouble, MyString
MyDouble = 437.324  'Змінна MyDouble містить число  437.324
MyString = CStr(MyDouble) ' MyString містить рядок "437.324"
```

Метод **Copy** застосовують до різних об'єктів, зокрема, **Range, WorkSheet** тощо. Цей метод копіює об'єкт, який його викликав, у зазначений діапазон. Зауважимо, що формули копіюються за правилами копіювання формул в ЕТ, тобто з модифікацією значень:

```
Cells(3, 2).Copy(Cells(3, 4))        ' вміст клітинки b3
'копіюється в d3, тоді d3 містить ф-лу =COS(C3)(див. рис. 22)
```

Хід роботи

1. **Запустіть програму Microsoft Excel.**
2. **Створіть макрос для розв'язування нелінійного рівняння методом простої ітерації і назвіть його Iteration.**
   Запрограмуйте його рекурсивно так:

```
Sub Iteration()
Dim n As Integer
n = WorksheetFunction.CountA(ActiveSheet.Columns(1)) + 1
'n – номер першої не порожньої клітинки стовпця A
Cells(n, 1).Value = Cells(n – 1, 2).Value
'в n-ту клітинку стовпця A копіюється число  з
'n–1-ї клітинки стовпця  B
Cells(n – 1, 2).Copy (Cells(n, 2))
'в n-ту клітинку стовпця B копіюється формула =COS(An)
'з n–1-ї клітинки стовпця  B
Cells(n,3).Formula = "=abs(a" + CStr(n) + "-b" + CStr(n) + ")"
'формула в стовпці C формується як рядок =abs(an-bn)
If Abs(Cells(n, 3).Value) > 0.001 Then
Iteration  'якщо точність не досягнута, продовжуємо
'ітераційний процес шляхом рекурсивного виклику функції
'Iteration, інакше результат виводимо червоним кольором
Else
Cells(n, 2).Font.Color = QBColor(12)
End If
End Sub
```

3. **Введіть рядки з заголовком, як показано на рис. 21. Розграфіть таблицю. Задайте необхідні формати клітинок та ширину стовпців. Введіть початкове наближення в клітинку А3 та формули методу в клітинки В3 і С3 (див. рис. 22).**
4. **Розташуйте на сторінці кнопку „Розв'язати рівняння" та призначте їй макрос Iteration.**
5. **Клацніть на кнопці „Розв'язати рівняння" – отримаєте таблицю (див. рис. 22). Збережіть роботу у файлі.**
   Візуально переконайтесь, що одержана похибка не перевищує задане число 0,001. Запишіть розв'язок рівняння у звіт.

Задачі

Задача 8.1. Розв'яжіть рівняння $2nx = \sin nx + n$, де n – номер варіанта. Підказка. Для цього достатньо звести рівняння до вигляду, придатного до застосування методу простої ітерації і змінити формулу у клітинці D3.

Задача 8.2. Змініть код макросу так, щоб похибка методу простих ітерацій знаходилась в одній із клітинок таблиці, наприклад, у клітинці D2. Введіть в цю клітинку число 0.0001, вручну очистіть одержану раніше таблицю (за винятком заголовка та першого рядка з формулами) і запустіть макрос на виконання. Запишіть одержаний розв'язок рівняння у звіт. Змініть похибку на ще меншу і ще раз отримайте результат. Скільки рядків має таблиця тепер?

Задача 8.3. Розташуйте на сторінці кнопку „Очистити" і складіть для неї макрос очистки клітинок таблиці за винятком заголовка та першого рядка.

Задача 8.4. Замініть команду в макросі Iteration

```
Cells(n,3).Formula = "=abs(a"+CStr(n)+ "b" + CStr(n) + ")"
```

(яка відповідає за заповнення стовпця С формулами) іншою командою, використавши у ній метод Copy.

Задача 8.5. Розмістіть кнопку „Наступна ітерація" і запрограмуйте її так, щоб після натискання на неї відбувалося обчислення лише одного наступного кроку (рядка) методу простих ітерацій.

§ 10. ЗАДАЧА ПРО ОБЛІК ЛІКІВ НА СКЛАДІ.

Практична робота № 9. Створення засобами VBA форми для введення даних про ліки в електронну таблицю

Мета роботи. Створити форму (рис. 24, 25) для введення даних у таблицю MS Excel, яка має містити інформацію про ліки на аптечному складі (рис. 23). Навчитися працювати з візуальними об'єктами VBA, зокрема, з випадаючим списком (**ComboBox**) та кнопкою корекції (**SpinButton**).

Теоретичні відомості. Об'єкт **ComboBox** використовують для створення випадаючого списку. Він містить одно- або двовимірний масив елементів. Вибір певного рядка цього масиву здійснюється клацанням на кнопці випадаючого списку.

Номер цього рядка міститься у властивості **ListIndex** об'єкта **ComboBox**. Нумерація елементів списку починається з нуля. Якщо **ListIndex** = – 1, то елемент списку не вибрано. У цьому випадку користувач може ввести власне значення списку, яке буде зберігатися у властивості **Text**. Атрибут **List** (список) цього об'єкта є масивом його елементів з текстовими даними. Ця властивість разом із записаними у круглих дужках індексами рядка і стовпця масиву дає доступ до відповідного елементу випадаючого списку. Розглянемо приклад списку з трьома елементами:

```
ComboBox1.List = Array("Елемент 1", "Елемент 2", "Елемент 3")
ComboBox1.List(1, 0)      ' вираз з результатом "Елемент 2"
ComboBox1.ListIndex = 2 ' робимо активним третій елемент.
```

База даних лікарських засобів підприємства "Найкраща аптека"

Ввести новий запис

N п/п	Код товару	Сертифікація	Продавати без рецепта	Приймати на склад	Група товару	Наявність на складі	Термін зберігання	К-сть (упак.)
1	ПРг 78-71/4	Так	Ні	Ні	Послаблюючі	Так	48	200
2	Рпо 67-77/9	Так	Так	Так	Анальгетики	Так	96	45
3	Рпо 89-19/1	Ні	Ні	Так	Інші	Ні	18	36
4	Лдо 90-09/0	Так	Так	Так	Серцеві ЛЗ	Так	36	60
6	ПРг 45-60/1	Так	Ні	Так	Вітаміни	Так	60	120
7	Рпо 89-77/1	Так	Так	Ні	Анальгетики	Ні	84	24
9	Прг 45-88/2	Ні	Ні	Так	Послаблюючі	Ні	36	120
10	РоГ 21-12/0	Так	Так	Ні	Гормони	Так	48	45

Рис. 23

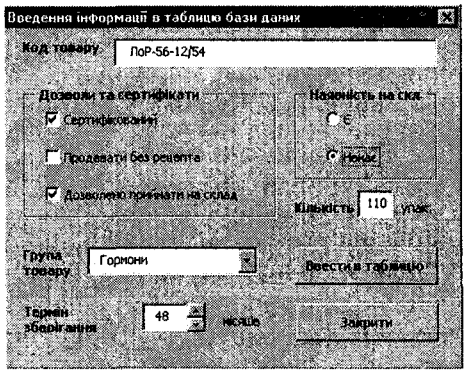


Рис. 24

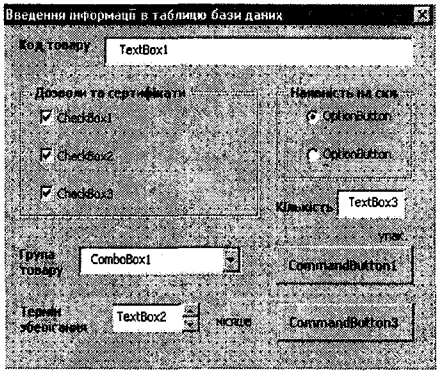


Рис. 25

Кнопка корекції **SpinButton** (піктограма  панелі Toolbox) призначена для корекції деякого цілочислового значення під час роботи програми. Властивості кнопки такі:

Властивість	Опис властивості	Приклади значень
Value	Цілочисельне значення, яке може корегуватися	деяке ціле число
Max	Максимальне значення	ціле число
Min	Мінімальне значення	ціле число
SmallChange	Крок зміни значення	ціле число
Visible	Видимість об'єкта	True, False

Хід роботи.

1. **Запустіть програму Microsoft Excel.**
2. **Створіть макрос відкривання форми для введення інформації в таблицю лікарських засобів і назвіть його ShowForm.**
   Запрограмуйте макрос так:

```
Sub ShowForm()
UserForm1.Show        'Відкриваємо форму UserForm1
End Sub
```

3. **Підготуйте заготовку таблиці Excel, як показано на рис. 23.**
4. **Розмістіть на сторінці ЕТ кнопку „Ввести новий запис" (див. рис. 23) та призначте їй макрос ShowForm.**
5. **Перейдіть у вікно VBA та відкрийте вікно форми UserForm1.**
6. **Розмістіть у цьому вікні об'єкти так, як показано на рис. 25.**
7. **Змініть заголовок форми та задайте інші початкові значення об'єктів (див. рис. 25). Збережіть таблицю.**
   Крок зміни терміну зберігання ліків задайте 6 (місяців). Для цього змініть властивість SmallChange об'єкта SpinButton1.
8. **Запрограмуйте зміну значення поля TextBox2 як реакцію на подію клацання на кнопці SpinButton1.**
   Для цього запрограмуйте кнопку SpinButton1 так:

```
Private Sub SpinButton1_Change()
TextBox2.Text = CStr(SpinButton1.Value)  ' змінене числове
'значення перетворюємо в рядок і заносимо у поле
' TextBox2
End Sub
```

9. **Задайте можливість введення чи зміни даних у полі редагування TextBox2.**
   Для цього двічі клацніть на полі TextBox2 і введіть такий код:

```
Private Sub TextBox2_Change()
SpinButton1.Value = CInt(TextBox2.Text) 'введене значення у
'поле TextBox2 записується як поточне кнопки SpinButton1
End Sub
```

10. Запрограмуйте кнопку „Закрити" так:

```
Private Sub CommandButton3_Click()
End                          'закінчуємо роботу
End Sub
```

11. Заповніть масив елементів випадаючого списку ComboBox1. Для цього двічі клацніть на формі, змініть подію Click на Initialize і введіть такий код:

```
Private Sub UserForm_Initialize()
'На етапі створення форми заповнюємо масив групами
'товарів і заносимо його до випадаючого списку:
ComboBox1.List = Array("Вітаміни", "Анальгетики",_
 "Серцеві ЛЗ", "Гормони", "Нейролептики",_
 "Послаблюючі","Інші")
End Sub
```

12. Запрограмуйте кнопку „Ввести в таблицю", яка призначена для занесення даних з форми в ЕТ.

```
Private Sub CommandButton1_Click()
Dim nazva, sert, vidp, pryjm, grupa, najav As String
Dim termin, kilk, n As Integer
'Знаходимо перший незаповнений рядок таблиці:
n = Application.WorksheetFunction._
                              CountA(ActiveSheet.Columns(1)) + 1
'Зчитуємо інформацію з форми введення:
nazva = TextBox1.Text
If CheckBox1.Value = True Then sert = "Так" Else sert = "Ні"
If CheckBox2.Value = True Then vidp = "Так" Else vidp = "Ні"
If CheckBox3.Value = True Then pryjm = "Так" Else pryjm = "Ні"
If ComboBox1.ListIndex > -1 Then
grupa = ComboBox1.List(ComboBox1.ListIndex, 0)
Else
grupa = ComboBox1.Text
End If
If OptionButton1.Value = True Then
najav = "Так"
Else
najav = "Ні"
End If
'Перетворюємо вхідні дані – кількість та термін
```

```
'зберігання – в числові:
termin = CInt(TextBox2.Value)
kilk = CInt(TextBox3.Value)
'Записуємо інформацію в таблицю Excel:
Cells(n, 1).Value = n
Cells(n, 2).Value = nazva
Cells(n, 3).Value = sert
Cells(n, 4).Value = vidp
Cells(n, 5).Value = pryjm
Cells(n, 6).Value = grupa
Cells(n, 7).Value = najav
Cells(n, 8).Value = termin
Cells(n, 9).Value = kilk
'Готуємо форму для введення наступних даних:
TextBox1.Text = ""
TextBox2.Text = "36"
TextBox3.Text = "1"
CheckBox1.Value = True
CheckBox2.Value = True
CheckBox3.Value = True
OptionButton1.Value = True
End Sub
```

13. **Поверніться в таблицю Excel, збережіть таблицю, перевірте роботу макросу.**
Для цього клацніть на кнопці „Ввести в таблицю" і введіть дані про декілька товарів, зазначаючи різні групи лікарських засобів і терміни їх зберігання. Не залишайте ці поля порожніми. Спробуйте змінювати термін зберігання товару за допомогою кнопок корекції, а також безпосереднім введенням числа у поле **TextBox2**. Введіть власну назву групи товарів у поле **ComboBox1**. Закрийте форму, клацнувши на кнопці „Закрити", і перевірте правильність введеної інформації в таблиці.

Задачі

Задача 9.1. Додайте до форми ще два поля редагування: "Виробник" (об'єкт типу TextBox) та „Постачальник" (об'єкт типу ComboBox). Підготуйте назви чотирьох постачальників і змініть коди процедур UserForm_Initialize і CommandButton1_Click відповідним чином. Змініть ЕТ, додавши до неї відповідні два стовпці.

Задача 9.2. Додайте до форми UserForm1 ще одну кнопку „Очистити", клацнувши на якій користувач витре інформацію з полів введення на формі і поверне їх до початкового стану без занесення інформації у таблицю, що корисно у випадку виявлення помилки під час введення даних.

§ 11. ЗАДАЧА ПРО ЗАХИСТ ІНФОРМАЦІЇ.

Практична робота № 10. Робота з текстами й датами засобами VBA у програмі MS Excel

Мета роботи. Створити макрос для шифрування та розшифрування деякого тексту (текстового рядка) шифром Цезаря.

Постановка задачі. Для шифрування тексту шифром з одним ключем до ASCII-коду кожного символу тексту додають фіксоване число *n*, яке називається ключем шифру. У шифрі Цезаря кожен символ тексту заміняють на символ, який знаходиться в таблиці кодів ASCII на *n* позицій правіше. Якщо в результаті такої заміни зашифрований символ виходить за межі алфавіту (деякого заданого діапазону кодів ASCII), то алфавіт розглядають записаним по колу, і відлік продовжують від його початку. Розшифрування відбувається у зворотному порядку.

	A	B	C	D	E	F	G	H
1		Неділя	Понеділок	Вівторок	Середа	Четвер	П'ятниця	Субота
2	1	29	10	24	14	7	9	37
3	2	47	16	24	42	11	25	27
4	3	43	14	48	47	30	19	44
5	4	23	42	2	7	17	18	9
6	5	11	39	20	36	17	22	15
7	6	7	31	34	46	48	25	45
8	7	29	31	1	25	20	49	29
9	8	8	2	38	35	44	28	45
10	9	24	46	48	1	45	19	38
11	10	5	18	21	33	17	5	3
12	11	34	27	21	5	18	41	9
13	12	18	35	14	37	42	37	49
14	13	24	4	8	21	35	44	20
15	14	40	1	16	5	1	50	48
16	15	36	43	48	1	37	4	39
17	16	16	18	28	2	23	7	2
18	17	12	32	29	47	40	42	4
19	18	44	15	28	38	30	20	15
20	19	9	4	47	18	45	28	19
21	20	13	13	7	33	40	13	30
22	21	39	27	18	17	9	46	2
23	22	11	42	18	32	48	16	21
24	23	49	28	48	39	6	28	20
25	24	12	1	27	35	27	22	46
26	25	46	32	15	34	47	17	20
27	26	9	33	41	39	32	25	48
28	27	43	27	46	2	12	43	20
29	28	25	32	35	23	18	23	24
30	29	18	46	40	18	26	26	45
31	30	34	40	9	8	39	34	15
32	31	10	39	31	22	16	3	34
33								
34		Генерувати таблицю випадкових чисел			Шифр Цезаря			
35								

Рис. 26

Зауважимо, що значення n не може тут перевищувати кількість літер вибраного алфавіту, оскільки в нашому макросі з метою його спрощення передбачена можливість лише одноразового переходу від кінця алфавіту до початку чи навпаки. Припускатимемо, що алфавіт для побудови повідомлень знаходиться між 32 і 255 символами кодів ASCII включно. Цей діапазон можна оглянути на рис. 20 практичної роботи № 7. Ключ для цього алфавіту не може перевищувати числа 224 (255–32+1).

Ключ шифру для певного дня вибирають з таблиці випадкових чисел (рис. 26), яка є фіксованою і відомою лише відправнику та одержувачу повідомлення.

Спочатку слід створити макрос з метою генерування таблиці випадкових цілих чисел з діапазону, нехай [1; 50], і відображення цієї таблиці на сторінці ET. Розмірність таблиці така: 31 рядок та 7 стовпців. Ключ залежатиме від введеної у форму (рис. 27) дати і розміщуватиметься у побудованій таблиці на перехресті рядка з днем дати та стовпця з назвою дня тижня цієї дати. Макрос запускатиметься кнопкою „Генерувати таблицю випадкових чисел".

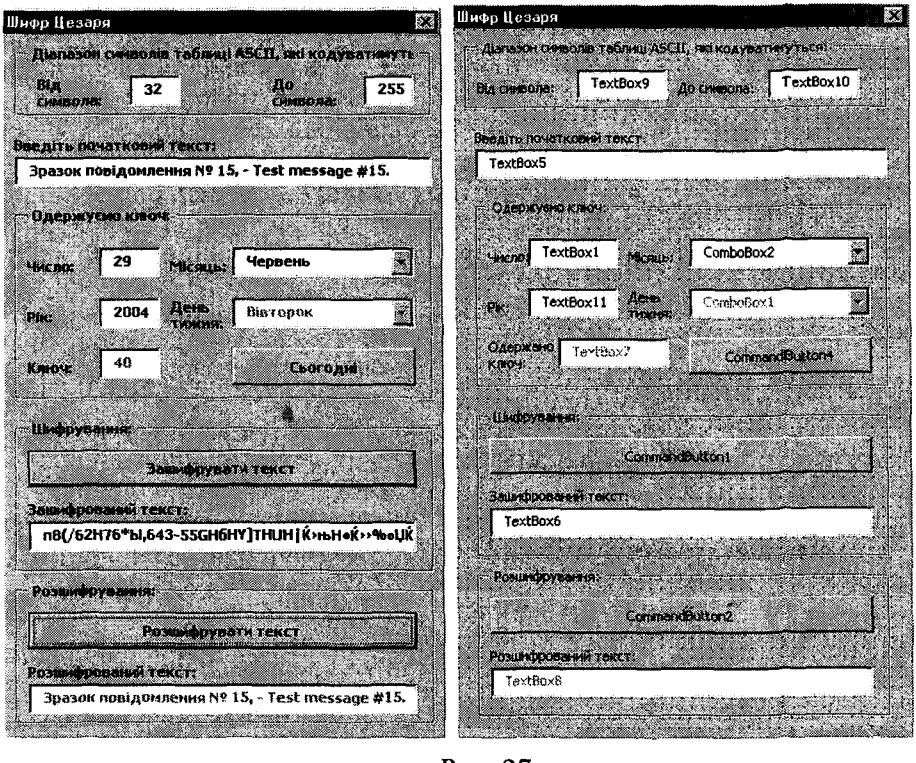


Рис. 27

Інший макрос (кнопка „Шифр Цезаря") здійснюватиме автоматичний пошук ключа з таблиці випадкових чисел, а також шифрування і розшифрування повідомлення знайденим ключем.

Зазначимо, що створені макроси можуть використовуватись для шифрування повідомлень, які відсилаються засобами поштової програми. Для цього відправник шифрує повідомлення і копіює його у вікно своєї поштової програми через буфер обміну. Одержувач копіює отримане повідомлення у поле „Зашифрований текст" форми цієї програми і розшифровує його, натиснувши відповідну кнопку. Розшифрування відбудеться коректно, якщо відправник та одержувач мають однакові таблиці ключів. Таблиці ключів пересилаються спеціальними способами. Такі і більш складні прийоми шифрування використовуються у банківській, військовій справі тощо.

Теоретичні відомості. Функція Asc(<рядок>) повертає ціле число – ASCII-код першого символу тексту, наприклад,

```
MyNumber = Asc("A")          ' MyNumber = 65
MyNumber = Asc("a")          ' MyNumber = 97
MyNumber = Asc("Apple")      ' MyNumber = 65
```

Змінні типу Date (дата) можуть містити значення дат, які присвоюються їм за допомогою літеральних змінних виду #<рядок символів>#, наприклад, Dim MyDate as Date: MyDate = #25/10/2004#.

Зображення даних типу Date залежить від налаштування дати в ОС Windows (Панель Керування ⇒ Мови і Стандарти ⇒ закладка „Дата"). Системна дата знаходиться у змінній з іменем Date.

Функція day(<дата>) дає ціле число з діапазону [1; 31] – день місяця введеної дати. Функція Month(<дата>) дає ціле число з діапазону [1; 12] – місяць року введеної дати. Функція Weekday(<дата>) дає ціле число з діапазону [1; 7] – день тижня. Нумерація відбувається в американському форматі: 1 – неділя, 2 – понеділок та ін. Функція Year(<дата>) дає ціле число з діапазону [100; 9999] – рік дати. Приклади:

```
Dim MyDay, MyMounth, MyWeekday, MyYear as Integer
MyDate = #February 12, 1969#
MyDay = day(MyDate)              ' MyDay = 12
MyMonth = Month(MyDate)          ' MyMonth = 2
MyWeekday = Weekday(MyDate)      ' MyWeekday = 4 (середа)
MyYear = Year (MyDate)           ' MyYear = 1969
```

Хід роботи.

1. **Запустіть програму Microsoft Excel.**
2. **Створіть макрос Fill генерування випадкових чисел (таблиці ключів), які заноситимуться в діапазон клітинок b2:h32.** Запрограмуйте його так:

```
Sub Fill()
'Вводимо в діапазон b2:h32 формули:
Application.Range("b2:h32").Formula =_
                    "=Round(RAND()*49+1,0)"
'Перезапишемо в цей же діапазон замість формул числа,
'отримані за допомогою цих же формул так:
Application.Range("b2:h32").Value =_
                      Application.Range("b2:h32").Value
End Sub
```

3. **Розграфіть таблицю, задайте формати клітинок та ширину стовпців, як на рис. 26. Збережіть роботу у файлі.**
4. **Розмістіть на сторінці ЕТ кнопку „Генерувати таблицю випадкових чисел" та призначте їй макрос Fill. Виконайте макрос.** Переконайтесь, що таблиця заповнилася випадковими числами.
5. **Відкрийте вікно форми UserForm1. Розмістіть на формі об'єкти так, як показано на рис. 27. Збережіть роботу.** Змініть заголовок форми та задайте інші початкові значення візуальних компонентів так, як показано на рис. 27 (за винятком полів TextBox6 та TextBox8, оскільки вони повинні залишатися порожніми). Об'єкти ComboBox1, TextBox7 та TextBox8 зробіть недоступними.
6. **Запрограмуйте процедуру KeyCalc для обчислення ключа для заданої дати.** Перейдіть у вікно тексту програми форми UserForm1. Для цього виберіть відповідний пункт контекстового меню цього об'єкта у вікні VBA Project. У розділі General ⇒ Declaration цього вікна запрограмуйте процедуру для обчислення ключа:

```
Private Sub KeyCalc ()
Dim DataStr As String
If TextBox1.Value > 0 And TextBox1.Value < 32 And_
TextBox11.Value <> 0 Then 'перевірка коректності дати
If TextBox11.Text = "" Then TextBox11.Value = 0
                'ці поля введення не можуть бути порожніми
If TextBox1.Text = "" Then
TextBox1.Value = 0
DataStr = TextBox1.Text + "." + CStr(ComboBox2.ListIndex_
+ 1) + "." + TextBox11.Text
                'формуємо текстовий рядок з датою
ComboBox1.ListIndex = Weekday(CDate(Str)) - 1 ' програма
'обчислює день тижня і заносить його у випадаючий список
cod=Cells(TextBox1.Value+1, ComboBox1.ListIndex+2).Value
TextBox7.Value=cod        'переписуємо ключ у поле „Ключ"
End If
End Sub
```

7. **Забезпечте обчислення нового ключа після зміни дати.**
Для цього по черзі двічі клацайте на формі на об'єктах, які забезпечують введення дати (ComboBox2, TextBox1 та TextBox11), і програмуйте їх реакцію на подію зміни дати:

```
Private Sub ComboBox2_Change()  'якщо змінюється місяць,
KeyCalc                         'то переобчислюємо ключ
End Sub
```

```
Private Sub TextBox1_Change()   'якщо змінюється число,
KeyCalc                         'то переобчислюємо ключ
End Sub
```

```
Private Sub TextBox11_Change()  'якщо змінюється рік,
KeyCalc                         'то переобчислюємо ключ
End Sub
```

8. **Запрограмуйте кнопку „Сьогодні" так:**

```
Private Sub CommandButton4_Click()
TextBox1.Text = day(Date)                    'заносимо в поля
ComboBox2.ListIndex = Month(Date) - 1        'введення дати
ComboBox1.ListIndex = Weekday(Date) - 1 'системну дату
TextBox11.Text = Year(Date)
End Sub
```

9. **Двічі клацніть на формі і запрограмуйте ініціалізацію списку елементів об'єктів ComboBox1 та ComboBox2:**

```
Private Sub UserForm_Initialize()
ComboBox1.List = Array("Неділя", "Понеділок", "Вівторок",_
"Середа", "Четвер", "П'ятниця", "Субота")
ComboBox2.List = Array("Січень", "Лютий", "Березень",_
"Квітень", "Травень", "Червень", "Липень", "Серпень",_
"Вересень", "Жовтень", "Листопад", "Грудень")
ComboBox2.ListIndex = 0      'місяць = „Січень"
CommandButton4_Click         'встановлюємо системну дату
End Sub
```

10. **Запрограмуйте кнопку „Зашифрувати текст" так:**

```
Private Sub CommandButton1_Click()
Dim cod, ascii, i, start, finish As Integer
Dim inp, out As String
start = CInt(TextBox9.Value)'початкове та кінцеве значення
finish = CInt(TextBox10.Value)      'кодів ASCII алфавіту
cod = TextBox7.Value 'одержуємо ключ шифру з поля TextBox7
inp = TextBox5.Value      'це рядок, який шифруватиметься
out = ""                  'це заготовка для зашифрованого рядка
```

```
For i = 1 To Len(inp)          'перебираємо усі символи
ascii = Asc(Mid(inp, i, 1))  'одержуємо ASCII-код символу
If ascii + cod > finish Then  'якщо символ виходить за межі
ascii = (ascii + cod) - (finish - start + 1)   'додаємо до коду
'значення ключа і зміщуємось вліво на довжину алфавіту
Else                          'якщо нема виходу за межі алфавіту,
ascii = (ascii + cod)          'то лише додаємо до коду ключ
End If
out = out + Chr(ascii)  'формуємо рядок зашифрованого тексту
Next i
TextBox6.Value = out        'відображаємо його на формі
TextBox8.Value = ""  'очищаємо поле  розшифрування тексту
End Sub
```

11. Аналогічно запрограмуйте кнопку „Розшифрувати текст":

```
Private Sub CommandButton2_Click()
Dim cod, ascii, i, start, finish As Integer
Dim inp, out As String
start = CInt(TextBox9.Value)     'початкове та кінцеве значення
finish = CInt(TextBox10.Value)       'ASCII-кодів алфавіту
cod = TextBox7.Value       'одержуємо ключ шифру з поля TextBox7
inp = TextBox6.Value        'рядок, який розшифровуватиметься
out = ""                   'заготовка для розшифрованого рядка
For i = 1 To Len(inp)         'перебираємо усі символи
ascii = Asc(Mid(inp, i, 1))  'одержуємо ASCII-код символу
If ascii - cod < start Then  'якщо символ виходить за межі
ascii = (ascii - cod) + (finish - start + 1) 'віднімаємо від коду
'ключ і зміщуємось вправо на довжину алфавіту
Else                         'якщо нема виходу за межі алфавіту,
ascii = (ascii - cod)         'то віднімаємо від коду ключ
End If
out = out + Chr(ascii)       'формуємо рядок розшифрованого тексту
Next i
TextBox8.Value = out         'і відображаємо його на формі
End Sub
```

12. Поверніться в таблицю Excel, збережіть роботу, перевірте функціонування макросу.

Для цього введіть текст деякого повідомлення, клацніть на кнопках „Зашифрувати текст" і „Розшифрувати текст". Переконайтесь, що початковий і розшифрований тексти збігаються. Спробуйте змінювати дату, спостерігайте за зміною значення ключа і візуально переконайтесь, що воно вибирається з таблиці випадкових чисел правильно. Перевірте правильність кодування декількох перших символів вручну, використовуючи рис. 20.

Розділ 4. VISUAL BASIC .NET

§ 1. ОСНОВНІ ПОНЯТТЯ

1. Призначення середовища і мови VB .NET. Мова Visual Basic .NET створена Microsoft 1991 року для застосування в інтегрованому середовищі проектування програм IDE Visual Studio .NET. Тут, окрім неї, можна використовувати мови C++, C#, J#. Мова і середовище призначені для створення повноцінних Windows-програм засобами візуального й об'єктно-орієнтованого програмування (ООП). Якщо у мові Qbasic метою створення є програма, у мові Visual Basic – проект, то у VB .NET – розв'язок (solution). Розв'язок проблеми чи задачі у VS .NET може складатися з одного чи декількох проектів. З проектом асоціюється одна чи декілька форм, на яких відображаються шукані результати, а також елементи керування: кнопки, поля для введення-виведення даних, перемикачі, контейнери, в яких розташовують картинки чи інші об'єкти, меню тощо.

Розв'язок у VB .NET – це контейнер, де зберігаються проекти і файли, що стосуються конкретної задачі. Файл розв'язку має розширення sln, файли коду форми чи модуля – vb. У цій книжці ми обмежимося розв'язками, що складаються з одного проекту.

У VB .NET реалізована та ж концепція візуального програмування, що і у VB (див. розділ 3). Однак можливості мови щодо ООП значно більші. Сьогодні мову широко застосовують у сфері професійного програмування для баз даних, локальних мереж та інтернету.

2. Запуск середовища. Після запуску середовища на екрані отримаємо декілька вікон: основне з головним меню та панелями інструментів, стартову сторінку (Start Page), провідник проектів і файлів (Solution Explorer), панель елементів керування (Toolbox), засоби допомоги (Dynamic Help), вікно класів (Class View), вікно провідника доступних серверів (Server Explorer). З вікнами Dynamic Help, Class View, Server Explorer не працюватимемо, тож закрийте їх (рис. 1).

Стартове вікно містить список доступних проектів, засоби для роботи в інтернеті, кнопки відкриття проекту чи створення нового.

Якщо створюватимете новий проект, то натисніть відповідну кнопку (New Project), переконайтеся, що тип проекту – Visual Basic Project, шаблон – Windows Application. Задайте назву проекту, наприклад, Myproject1. За допомогою кнопки Browse зазначте диск і папку, де зберігатимете проект. Натисніть кнопку ОК – отримаєте наступне вікно. Воно міститиме елементи керування Windows Forms на панелі Toolbox та стартове вікно з активною закладкою дизайнера форми (Form1.vb[Design]), провідником розв'язку Myproject1, що матиме один однойменний проект. У разі потреби до роз-

в'язку можна буде згодом додати ще один проект. Проект є контейнером, що містить системну папку References, системний файл AssemblyInfo.vb та файл форми Form1.vb. Власне файл Form1.vb, що містить інформацію про форму та її програмний код, нам потрібно навчитися створювати. Зауважимо, що, користуючись контекстним чи головним меню, форму можна перейменувати.

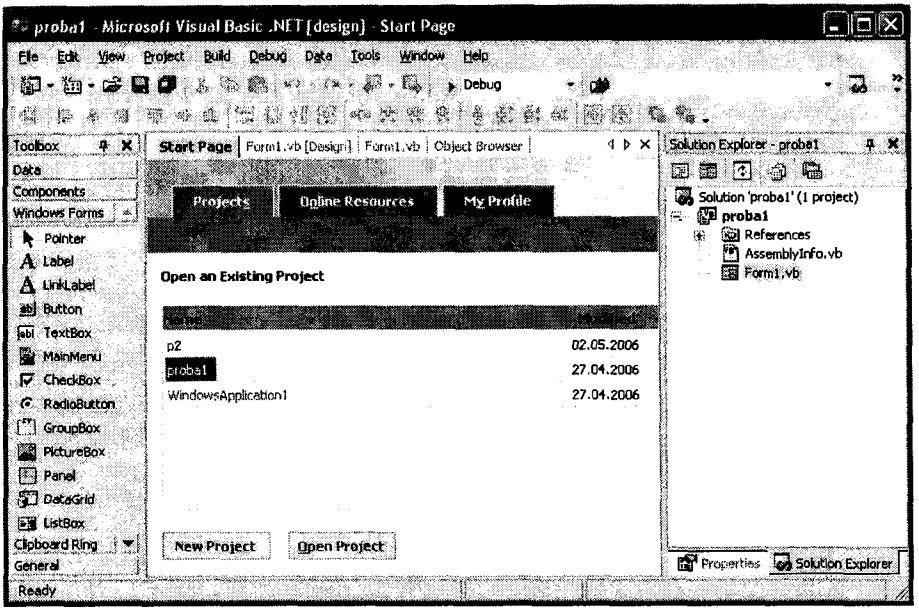


Рис. 1.

Відкрийте новий або наявний проект і активізуйте закладку дизайнера Form1.vb[Design]. Знайдіть у нижній правій частині вікна або праворуч закладку властивостей (Properties) і клацніть на ній – розгорнеться вікно властивостей поточного об'єкта (тепер файлу форми – file properties), а вікно провідника розв'язку згорнеться. Клацніть на формі – у вікні властивостей з'являться властивості форми Form1 як об'єкта класу Windows.Forms. Можна приступати до створення першого проекту (розв'язку) в середовищі VB .NET.

3. Перенесення Qbasic-програм у середовище VB .NET. Ви вже бачили, як легко переносяться Qbasic-програми в середовище Visual Basic. На жаль, перехід до VB .NET є дещо складнішим.

По-перше, форма у VB .NET не володіє методом Print, тому результати не можна виводити безпосередньо на форму тим же способом, що в VB. Їх виводять в об'єкти TextBox або ListBox чи застосовують спецприйоми.

По-друге, до назв усіх математичних функцій (а деякі з них мають навіть дещо інші назви, наприклад, корінь квадратний не SQR, а Sqrt) потрібно дописати назву класу Math., наприклад, Math.Sin, Math.Sqrt тощо.

У VB .NET практикують виведення окремих даних чи повідомлень у спеціальне вікно MessageBox за допомогою методу Show:

```
MessageBox.Show(<текстове чи числове дане>, <заголовок вікна>)
```

Таке вікно називають модальним – його слід закривати, щоб продовжити роботу, натиснувши на кнопку OK тощо.

Середовище VB .NET є компілятором. Готовий програмний код спочатку компілюють, застосувавши команди головного меню **Build, Build Solution** або **Build <проект>**, або **ReBuild** у разі повторної компіляції, а потім виконують **Debug, Start** або натискають **F5.**

Змінні у програмі потрібно оголошувати обов'язково, але тепер (на відміну від VB) змінні одного типу можна оголошувати списком:

```
Dim <список змінних> As <назва типу>
```

Для надання значень змінним використовують або команду присвоєння, або функцію InputBox.

Команду END у кінці програми не пишуть. Метод Show на початку також не зазначають.

4. Об'єкт форма (Form). *Форма* – це об'єкт класу Windows Form. Для форми у вікні Properties задають такі властивості: підпис (Text), розміри (Size), розташування (Location), колір фону (BackColor), фонову картинку (BackGroundImage), колір текстів і рисунків (ForeColor), шрифт (Font) та багато інших менш важливих властивостей. Для форми визначені методи Show() – показати і Hide() – заховати, які застосовуються під час роботи з декількома формами з метою створення шоу – ефекту зміни слайдів.

Завдання. Створіть новий проект. Розгляньте вікно властивостей форми. Змінюйте значення наведених вище властивостей і спостерігайте за змінами на формі. Закрийте проект без зберігання.

Форма – це контейнер, де розміщують елементи керування, взяті з панелі Toolbox. Щоб розмістити елемент на формі, клацають на піктограмі елемента на панелі, а потім – на формі. Елемент керування можна переміщати методом перетягування чи змінювати його розміри, перетягуючи маркери. Ці ж дії виконують способом задавання у вікні Properties потрібних значень властивостей вибраного елемента.

Якщо однотипних елементів на формі є декілька, то вони отримують номери відповідно до порядку їх вставляння, наприклад, TextBox1, TextBox2, TextBox3, TextBox4. Це є системні назви (властивість Name) елементів керування. Назви можна змінювати

на власний розсуд. Це рекомендують робити, керуючись змістом задачі. Наприклад, треба давати такі «промовисті» назви: Number1TextBox, Number2TextBox, AddresTextBox, NameTextBox. Або варто дописувати префікси з назвами типів до конкретних імен, наприклад, txtNumber1, txtNumber2, txtAddres, txtName (тут txt – префікс елемента TextBox), або не змінювати системних назв. В останньому випадку до кожного проекту варто складати таблиці з поясненням для себе і читача призначення елементів керування, змінних тощо, наприклад, так:

TextBox1 – перше число,
NameTextBox – ім'я співробітника тощо.

Рекомендуємо познайомитися у вікні Properties форми чи кнопки з виглядом кольорів, їх тризначними RGB-кодами та назвами. Для цього треба відкрити закладку Custom значення властивості BackColor, клацнувши мишею у правому полі властивості BackColor.

5. Елемент керування – текстове поле (TextBox). Елемент керування *текстове поле* – об'єкт класу TextBox – призначений для організації введення-виведення даних. Дане задають на етапі проектування як значення властивості Text поля у вікні Properties або на етапі виконання проекту за допомогою команди присвоєння, наприклад, так:

```
TextBox1.Text = 5                        'Задамо значення поля 5
TextBox3.Text = "Данило"                 'Значення поля – Данило
TextBox3.Text = TextBox3.Text & "  Галицький"
```

Операція & забезпечує злиття текстів (додавання текстових даних). У полі TextBox3 отримаємо текст "Данило Галицький".

Нехай у результаті обчислень отримали x = 10, а y = 25. Щоб подати цей результат на екран у деякому полі TextBox2, пишуть

```
TextBox2.Text = "x =" & x & "   " & "y=" & y          'або
TextBox2.Text = "x =" & Str(x) & "   " & "y=" & Str(y),
```

де Str() – необов'язкова функція перетворення даного з числового типу в текстовий.

Щоб вивести результати у полі TextBox у вигляді таблиці значень змінних x та y, використовують такий прийом:

```
TextBox2.Text = TextBox2.Text & x & "  " & y & vbNewLine.
```

Рядок пропусків " " призначений для розмежування даних декількома пропусками. Замість нього можна використовувати стандартну сталу vbTab (для виведення даних в зонах) або функцію Space$(n), де *n* – кількість пропусків. Стандартна стала vbNewLine призначена для переходу на новий рядок.

Для елемента TextBox на етапі конструювання задають властивість Multiline як True для виведення текстів у декількох рядках, смуги прокручування (ScrollBars як Both) і достатні розміри (Size).

6. Елемент керування – кнопка (Button). У Qbasic-програмі всі обчислення виконуються відразу після запуску команди Run. У VB дещо інакше. Певні серії дій можна закріпити за кнопками, натискання на які на етапі виконання проекту приведе до їх реалізації.

Кнопки (об'єкти класу Button) призначені для керування процесами, що відбуваються на формі. Кнопки реагують на подію Click (клацання на кнопці).

Запуск проекту веде до появи форми, на якій користувач на етапі конструювання розташовує потрібну кількість кнопок і програмує їх. Часто достатньо одної кнопки Button1, на якій написано «Виконати». «Виконати» є значенням властивості Text кнопки Button1. Можна використати ще одну кнопку Button2 зі змістом «Кінець» для закриття проекту (хоча проект можна завершити, закривши його вікно). Префікс для назв кнопок – btn.

Щоб запрограмувати кнопку, потрібно в режимі дизайнера двічі клацнути на ній. Відкриється *стандартна заготовка коду* з вертикальним курсором, що позначає точку введення тексту програми

```
Private Sub Button1_Click(ByVal sender As System.Object,
        ByVal e As System.EventArgs) Handles Button1.Click
|
End Sub
```

Заготовку заповнюємо текстом програми. Простежимо, як просто програмується кнопка «Кінець». Двічі клацнувши на ній, наберіть команду End – кнопку запрограмовано.

7. Мій перший VB .NET розв'язок (проект). Розглянемо задачу.

Задача про трикутник. Задано довжини двох катетів прямокутного трикутника. Обчислити його периметр і площу. Реалізувати два способи виведення результатів: у текстове поле на формі і у два вікна MessageBox із заголовками "Це периметр", "Це площа".

Метод і алгоритм розв'язування задачі. Нехай a і b – змінні, де знаходяться значення довжин катетів, а p та s – змінні, де будуть міститися результати. Нехай a = 5, b = 7. Метод і алгоритм розв'язування такий:

```
a = 5
b = 7
c = √(a² + b²)          'Знаходимо гіпотенузу
p = a + b + c
s = a * b / 2
Вивести p та s.
```

Інтерфейс проекту. На формі розмістіть текстове поле TextBox1, куди виводитимемо результати в двох рядках так:

```
Це розв'язок задачі:
p= ...        s = ...
```

Задайте властивість Multiline як True і розтягніть поле.

Внизу форми розмістіть кнопку Button1 і задайте її властивість Text як «Виконати» (рис. 2).

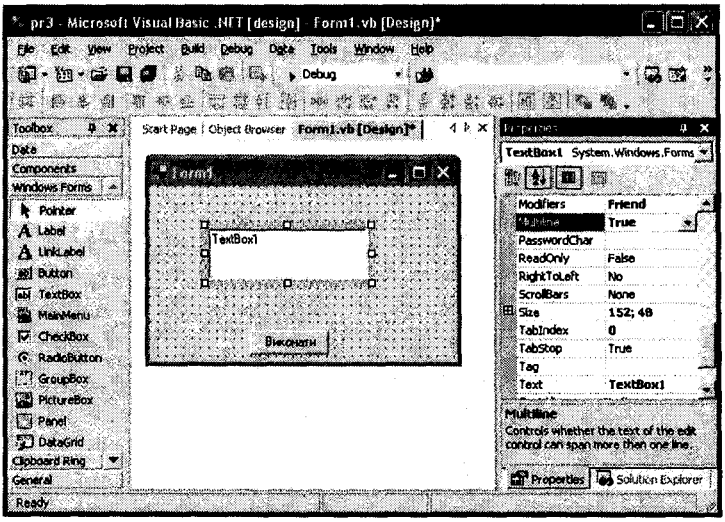


Рис. 2.

Сценарій. Під час запуску проекту (після виправлення помилок та описок і отримання у вікні Output повідомлення про успішну компіляцію) на екран має з'явитися форма з двома елементами керування: порожнім елементом TextBox1 і кнопкою «Виконати». Після натискання на кнопку спочатку мають з'явитися повідомлення з розв'язками в двох модальних вікнах, які потрібно буде закрити, а потім у полі TextBox1 з'явиться розв'язок задачі. Проект має завершити роботу після клацання на кнопці закривання вікна форми (тобто кнопку «Кінець» використовувати не потрібно). Зберегти проект.

Програмування. Двічі клацніть на кнопці «Виконати» і введіть

```
Dim a, b, c, p, s As Single
a = 5 : b = 7
c = Math.Sqrt(a ^ 2 + b ^ 2)
p = a + b + c
s = 1 / 2 * a * b
MessageBox.Show(p, «Це периметр»)
MessageBox.Show(s, «Це площа»)
TextBox1.Text = «Це результати:» & vbNewLine
TextBox1.Text = TextBox1.Text & «p=» & p & _
                vbTab & «c = « & c
```

Символ «_» використовують для перенесення тексту команди у наступний рядок.

Виконайте команди **Build Solution**, а потім **Debug, Start**. Не забувайте закривати модальні діалогові вікна. Переконайтеся, що

Це результати:
p=20.60233 c = 8.602325.

Висновок. Наведена програма відрізняється від Qbasic-програми головно способами виведення результатів на екран.

8. Вікно Output. Qbasic-програму можна перенести в код завантаження форми, замінивши команди PRINT на команди Debug.Writeline(<рядок>), де рядок формується за такими ж принципами, як і для елемента TextBox, але без вказівки переведення рядка vbNewLine. Наприклад, так: Debug.Writeline("p=" & p & vbTab & "s=" & s). Результати роботи програми виводитимуться у вікні Outlook, а на формі ніяких дій не відбуватиметься. Прийом для виведення даних безпосередньо на форму вивчатимемо згодом.

Задачі

Задача 1.1. Поекспериментуйте з програмою про трикутник, вводячи різні дані. Відімкніть режим виведення даних у діалогові вікна, закоментувавши відповідні два рядки коду. Для цього перед ними потрібно поставити символ-апостроф «'».

Задача 1.2. Дослідіть, як можна змінювати властивості форми програмним шляхом. Розташуйте на формі чотири кнопки і підпишіть їх так: «Червоний» – для зміни кольору тла форми на червоний, «Блакитний» – для зміни кольору тла форми на небесно-блакитний, «Ширина» – для збільшення ширини форми (праворуч) на 50 пікселів, «Висота» – для збільшення висоти форми (вниз) на 50 пікселів.

Підказка. Запрограмуйте кнопки. Використайте такі команди присвоєння значень властивостям форми в кодах кнопок: Me.BackColor = System.Drawing.Color.Red, Me.BackColor = System.Drawing.Color.SkyBlue, Me.Width = Me.Width + 50, Me.Hight = Me.Hight + 50. Слово Me – це власна назва головної форми.

Можете вибрати інші кольори. Увівши крапку після слова Color, отримаєте підказку з можливими значеннями кольорів. На потрібному кольорі треба клацнути двічі. Завжди після введення крапки у складених іменах отримуватимете підказку про те, що можна ввести далі. Продовжіть введення «вручну» або виберіть потрібне значення із запропонованого списку.

Задача 1.3. Розташуйте на формі чотири кнопки. Після натискання першої кнопки форма змінює колір і кнопка стає недоступною (Button1.Enabled = False). Якщо натиснути на другу кнопку, то вона змінює колір, а перша кнопка зникає (Button1.Visible = False – стає невидимою). Після натискання третьої кнопки напис на ній змінюється на «Привіт!» (Button3.Text = "Привіт!"). Клацання на четверту кнопку має зробити першу кнопку видимою і доступною, а інші невидимими.

§ 2. ВВЕДЕННЯ-ВИВЕДЕННЯ ДАНИХ

1. Елемент керування напис (Label). *Напис* – об'єкт класу Label – призначений для оздоблення форми, виведення на форму текстів, найчастіше для підписування текстових полів та інших елементів керування. Він замінює команду PRINT <текстове дане > в програмі Qbasic. Префікс назви елемента – lbl.

Напис має певні властивості, які користувач програмує або задає у вікні Properties: BackColor, BorderStyle – вигляд рамки (None – рамка невидима), Enabled – доступність (значення True або False), Font, ForeColor, Location, Text, TextAlign – вирівнювання тексту в рамці, Visible – видимість усього елемента (True або False).

2. Застосування поля TextBox для введення даних.

Задача про планети. Є три планети: Земля, Марс, Венера. Знаючи радіус планети, обчислити довжину екватора і площу поверхні кожної планети.

Математична модель. Моделлю задачі є куля, де, знаючи радіус r, потрібно знайти довжину кола l і площу поверхні кулі s.

Метод розв'язування. З курсу математики відомо, що

$l = 2\pi r$, $s = 4/3\pi r^2$.

Алгоритм. Алгоритм розв'язування задачі такий:

Ввести r
l = 2*pi*r
s = 4/3*pi*r^2
Вивести l,s
Повторити попередні пункти ще двічі.

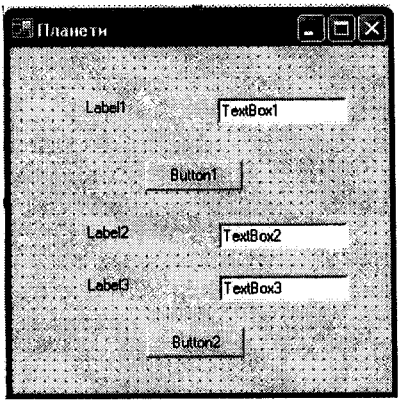


Рис. 3.

Інтерфейс проекту (рис. 3). Увійдемо в середовище і створимо новий проект з назвою Planets. На формі Form1 «Планети» зверху і праворуч від середини розташуємо поле TextBox1 для введення значення радіуса планети. Ліворуч від поля помістимо напис Label1 «Радіус:». Під ним розташуємо кнопку Button1 «Обчислити», а під

кнопкою – два поля TextBox2 та TextBox3 для отримання результатів. Над полями розташуємо написи Label2 «Екватор:» та Label3 «Площа поверхні:», а під ними – кнопку Button2 «Кінець». По черзі вибиратимемо елементи і, користуючись клавішами переміщення курсору, вирівняємо їх, як на рис. 3.

Сценарій. Після запуску проекту, текстові поля порожні. Фокус розміщений у першому текстовому полі. Сюди вводитимемо значення радіуса першої планети. Після клацання на кнопці «Обчислити» результати мають з'явитися у відповідних полях. Закриваємо проект. Виконаємо проект ще двічі, вводячи щоразу радіуси двох інших планет.

Програмування. Запрограмуємо кнопку «Кінець», набравши одну команду End. Кнопку «Обчислити» програмуємо так:

```
Dim r, l, s As Single
r = TextBox1.Text
l = 2* Math.pi* r
s = 4 / 3 * Math.pi * r ^ 2
TextBox2.Text = l
TextBox3.Text = s.
```

Розгляньте інший спосіб програмування цієї кнопки без застосування допоміжних змінних r, l, s:

```
TextBox2.Text = 2 * Math.pi * TextBox1.Text
TextBox3.Text = 4 / 3 * Math.pi * TextBox1.Text ^ 2.
```

Виконання проекту. Відкомпілюйте проект (**Build** Planets) і виконайте його. Введіть значення радіуса і натисніть кнопку «Обчислити». Результати перепишіть у зошит. Закрийте вікно проекту. Виконайте проект ще двічі (для цього натискайте на **F5**) для дослідження інших планет.

3. Модифікації проекту. Не зручно проект виконувати тричі. Модифікуємо проект, щоб отримати всі результати без його перезапусків. Для цього поряд з кнопкою «Обчислити» помістимо на форму кнопку Button3 «Очистити», яка очищуватиме всі текстові поля і надаватиме фокус першому полю за допомогою методу Focus(). Після отримання результатів заборонимо натискати на кнопку «Обчислити», зробивши її недоступною відразу після першого натискання, але зробимо її доступною після натискання кнопки «Очистити». Тепер можна буде натиснути на кнопку «Обчислити» ще раз, ввести радіус і отримати дані щодо другої планети. Після цього знову натискаємо кнопки «Очистити» і «Обчислити».

Код кнопки «Обчислити» треба доповнити командою

```
Button1.Enable = False.
```

А код кнопки «Очистити» треба створити так:

```
TextBox1.Text = ""
TextBox2.Text = ""
```

```
TextBox3.Text = ""
TextBox1.Focus()
Button1.Enabled = True
```

Зауважимо, що швидко переходити від вікна форми до коду можна за допомогою клавіші **F7**, а назад – **Shift+F7**.

4. Використання функції InputBox для введення даних. Функція InputBox замінює команду INPUT у Qbasic-програмі. Ця функція описана у § 4 розділу 1. Проект про планети модифікуємо так: після запуску проекту треба натиснути на кнопку «Обчислити», має з'явитися діалогове вікно з заголовком «Радіус планети» і підказкою «Введіть радіус». Після введення радіуса і натискання на кнопку ОК значення радіуса має з'явитися в полі TextBox1. Відбудуться обчислення. В інших двох полях отримаємо результати.

Для цього замість команди r = TextBox1.Text у коді кнопки Button2 запишіть команди

```
r = InputBox("Введіть радіус", "Радіус планети")
TextBox1.Text = r
```

або одну команду

```
TextBox1.Text  = InputBox("Введіть радіус", "Радіус планети").
```

Висновок. Тепер у форму вводити дані (значення радіуса) «вручну» не потрібно, це зроблено майже автоматично. Використовуючи функцію InputBox, створюють програми, які задовольняють властивість масовості, тобто придатності для розв'язування не одної конкретної задачі, а певного класу задач. Тепер можна обчислити площу поверхні футбольного м'яча будь-якого радіуса, повітряної кулі тощо.

5. Використання елемента керування ListBox (список) для виведення даних. Постійно переписувати результати з текстових полів у зошит нецікаво. Було би ліпше отримати на екрані всі результати, а потім вирішувати, що з ними робити. Ви вже знаєте, що є спосіб виведення результатів у текстове поле в декілька рядків, але тоді команда виведення записується досить складно (див. задачу 1).

Розглянемо простіший спосіб виведення багаторядкових даних. Для цього використаємо елемент керування *список* ListBox – контейнер, що містить список даних і у разі потреби смуги прокручування. Цей елемент зручно використовувати, коли треба створити таблицю даних. Елемент володіє такими корисними методами: Items.Clear – очистити список, Items.Add(<дане>) – дописати дане до списку.

Інтерфейс і сценарій. Змініть інтерфейс і сценарій проекту. Вилучіть кнопку «Очистити» і текстові поля TextBox2 і TextBox3. Вставте два елементи-списки ListBox1 і ListBox2. Щоб очистити поля списків від слів ListBox1 та ListBox2, активізуйте у властивості Items значення Collection... і натисніть ОК у вікні String Collection Editor. Задайте розміри списків так, щоб у кожному елементі

поміщалось по три рядки результатів (бо є три планети). Розташуйте над списками підписи «Екватори:» і «Площі:».

Змініть код кнопки «Обчислити», застосувавши такі методи:

```
ListBox1.Items.Add(l)
ListBox2.Items.Add(s)
```

або

```
ListBox1. Items.Add(2 * Math.pi * TextBox1.Text)
ListBox2. Items.Add(4 / 3 * Math.pi * TextBox1.Text ^ 2).
```

Вилучіть команду Button1.Enable = False.

Обов'язково вилучіть код процедури Button3_Click.

Тепер кнопка «Обчислити» буде доступною. У діалогове вікно введіть значення радіуса і після натискання на кнопку ОК отримаєте результати для першої планети. Натискайте кнопку –результати отримуватимете у вигляді списку (одностовпцевої таблиці). Одержавши всі результати, перепишіть їх у зошит.

Висновок. Ви навчилися створювати списки програмним шляхом. Список володіє багатьма корисними можливостями: властивостями і методами, які ми вивчатимемо згодом.

Задачі

Задача 2.1. Розв'яжіть задачу 1 свого варіанта з розділу «Задачі».

Задача 2.2. Розв'яжіть задачу 2 свого варіанта з розділу «Задачі».

Задача 2.3. Модифікуйте проект про планети так: після запуску проекту значення радіуса у текстове поле TextBox1 ввести неможливо (воно недоступне, властивість Enabled задайте як False). Після заповнення діалогового вікна воно стає доступним.

Задача 2.4. Змоделюйте гру «хрестики-нулики». Розташуйте на формі у вигляді квадрата 3 на 3 дев'ять текстових полів без підписів (витріть значення властивості Text у вікні Properties) і зменшіть розміри полів, щоб в них вмістився один символ. Розташуйте внизу кнопку «Очистити». Перший гравець має клацати на полі один раз – у полі з'явиться символ «X», другий гравець має клацати двічі – в полі з'явиться символ «0». Кнопка «Очистити» очищає всі поля для наступної гри.

Підказка. Вставте дев'ять полів, користуючись панеллю Toolbar. Двічі клацніть на полі, щоб запрограмувати його, або перейдіть у вікно коду (F7). Не програмуйте подію TextChanged, шаблон якої з'являється за замовчуванням у першому випадку. Над кодом у полі Class Name виберіть елемент TextBox1, а в полі Method Name – подію Click. З'явиться заготовка процедури, куди введіть команду TextBox1.Text = "X". Тепер виберіть в Class Name об'єкт TextBox1 і подію DoubleClick. У запропонований шаблон процедури введіть команду TextBox1.Text ="0". Повторіть ці дії для дев'яти полів. Запрограмуйте кнопку «Очистити», ввівши команди TextBox1. Text='''', TextBox2.Text='''',..., TextBox9.Text=''''. Зіграйте у створену гру. Вивчаючи наступні параграфи, удоскональте проект, доповнивши його засобами перевірки виграшних ситуацій тощо.

§ 3. ВИБІР СПОСОБІВ ОПРАЦЮВАННЯ ДАНИХ

1. Елемент керування радіокнопка (RadioButton). Елемент керування *радіокнопка* – об'єкт класу RadioButton. Його використовують для вибору одного і тільки одного з декількох можливих варіантів опрацювання даних. Наприклад, дані, що стосуються умови задачі, можна вводити або «вручну», заповнюючи текстові поля на формі, або в режимі «діалогу», використовуючи функцію InputBox. Вибір залежить від уподобань користувача, тому в проектах варто передбачати обидва (всі) варіанти, а користуватися одним.

2. Елемент керування прапорець (CheckBox). *Прапорець* – об'єкт класу CheckBox. Його використовують, коли потрібно вибрати декілька варіантів опрацювання даних серед багатьох можливих. Наприклад, дані, що є результатами роботи програми, можна виводити або лише у текстові поля, або лише у вікна MessageBox, або ще кудись, або у всі вищезгадані елементи одночасно. На етапі конструювання розв'язку треба передбачити різні можливості опрацювання даних чи подій, а користувач на етапі експлуатації проекту вибере потрібне.

3. Група елементів керування (GroupBox). Елементи керування зі спільною темою чи застосуванням об'єднують у *групу* – об'єкт класу GroupBox. Ознакою групи є рамка, що охоплює елементи. Рамка може мати зверху заголовок. Щоб вставити у форму рамку групи, слід в ToolBar клацнути на піктограмі елемента керування GroupBox, а потім обвести мишею прямокутну рамку на формі. Після цього в рамку треба перемістити елементи керування, що утворюватимуть групу, або елементи керування створити в рамці. Тепер під час переміщення рамки переміщатимуться і всі елементи, які в ній є. У результаті вилучення рамки її елементи також вилучаються. Групи найчастіше створюють з радіокнопок і прапорців. Наприклад, варто створити групу з підписом «Ввід» (а це властивість Text елемента GroupBox) для радіокнопок, описаних у першому пункті, а також групу з підписом «Вивід» для прапорців, описаних у пункті 2.

Для створення групи призначений також елемент керування Panel, який (на відміну від елемента GroupBox) може мати смуги прокручування, але не має підпису.

4. Задача «Довідка». Нехай потрібно визначити, скільки кілометрів (додому) і скільки футів (під кормою) мають п'ять чи інша кількість (k) морських миль, якщо відомо, що 1 морська миля = 1,852 км = 6076 футів.

Інтерфейс проекту. Вхідним даним у задачі є k, для введення якого створимо на формі текстове поле TextBox1 з підписом «Милі:» в елементі Label1 (див. рис. 4). Передбачимо два способи введення значення k. Для цього помістимо на форму дві радіокнопки RadioButton1, RadioButton2 і підпишемо їх «Діалог» та «Вручну»

(це властивість Text радіокнопок). Зробимо активною першу радіокнопку (властивість Checked задайте як True). Це означає, що за замовчуванням користувач планує скористатися діалоговим методом. А якщо захоче вибрати інший метод, то йому треба буде клацнути на другій радіокнопці.

Розташуємо кнопки Button1 «О&бчислити» і Button2 з підписом «&Кінець». Символ «&» недаремно стоїть перед символом «б». Реально підпис матиме вигляд «Обчислити». Це означає, що є ще один спосіб розпочати обчислення: натиснути комбінацію гарячих клавіш **Alt + б** (однак така можливість функціонує поки що лише для англійських букв).

Розташуємо два текстові поля для результатів (підписані «Км:» та «Фути:» за допомогою елементів Label2 і Label3) і два прапорці з підписами «У поля» (CkeckBox1) і «У вікно» (CheckBox2). Очистимо всі текстові поля. Серед кнопок надамо фокус (асоціацію з клавішею Enter на клавіатурі) кнопці «Обчислити», надавши властивості форми AcceptButton значення Button1, а властивості CancelButton (асоціація з клавішею Esc) – значення Button2.

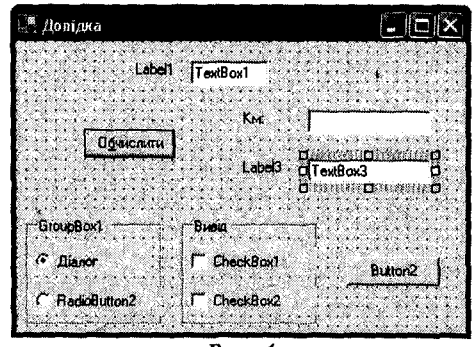


Рис. 4

Сценарій. Після завантаження проекту користувач повинен вибрати способи введення даних і виведення результатів. Усі текстові поля порожні. Коли вибрано режим ручного вводу даних, у текстове поле «Милі:» потрібно ввести число. Фокус встановлено на кнопці «Обчислити». Це означає, що для початку виконання алгоритму достатньо буде натиснути на клавішу вводу.

У розпорядженні користувача є чотири способи, щоб закрити проект: 1) клацнути на системній кнопці у правому верхньому куті форми, 2) клацнути на кнопці «Кінець», 3) скористатись комбінацією «гарячих» клавіш **Alt + К** (наразі не функціонує, вихід – можна зробити букву «К» англійською), 4) натиснути на клавіатурі **Esc**.

Програмування. Запрограмуємо кнопку «Обчислити».

```
Dim k, km, foots As Single
If RadioButton1.Checked = True Then
```

```
    TextBox1.Text = InputBox("Введіть милі", "Довідка")
End If
If TextBox1.Text = "" Then
    MessageBox.Show("Не задано милі", "Аварія")
    TextBox1.Text = InputBox("Введіть милі", "Довідка")
End If
k = TextBox1.Text
km = 1.852 * k
foots = 6076 * k
If CheckBox1.Checked = True Then
    TextBox2.Text = km
    TextBox3.Text = foots
End If
If CheckBox2.Checked = True Then
    MessageBox.Show(km, "Це кілометри")
    MessageBox.Show(foots, "Це фути")
End If
```

Виконайте проект і поекспериментуйте з різними способами введення і виведення даних.

Задачі

Задача 3.1. У проекті «Довідка» використайте кнопку «&Очистити», за допомогою якої можна очистити текстові поля і повторити обчислення.

Задача 3.2. «Аварійна» ситуація виникає, якщо користувач забуває вибрати спосіб виведення даних, оскільки за замовчуванням не вибраного жодного способу. Запропонуйте шляхи уникнення такої ситуації програмним шляхом. Розташуйте на формі напис «Не забувайте задати режими!»

Задача 3.3. Переробіть проект, щоб можна було визначити, скільки гривень треба заплатити в пункті обміну валюти, щоб придбати 100 доларів, 100 євро, 100 фунтів стерлінгів.

Задача 3.4. Розв'яжіть задачу 3 свого варіанта з розділу «Задачі».

§ 4. ЦИКЛИ І МАСИВИ

1. Цикли в задачах опрацювання даних. Цикли і масиви детально описано у першому розділі. Ви можете реалізувати будь-яку задачу обчислювального характеру у VB .NET, використовуючи команди циклів і відомі елементи керування. Нагадаємо, що задачі на побудову таблиць реалізуються за допомогою багаторядкового текстового поля TextBox з увімкненою властивістю MultiLine або елемента ListBox і його методу Items.Add(<значення>).

Задача про побудову таблиці. Побудувати таблицю для переведення доларів у гривні від 1 до $k = 10$ доларів з кроком 1. Вважати, що 1 долар = 5,05 грн.

Інтерфейс проекту. На формі розташуємо елемент ListBox1 для створення таблиці у вигляді списку і кнопку Button1 «Обчислити».

Програмування. Запрограмуємо кнопку «Обчислити».

```
Dim i, k As Integer, Gr As Single
k = InputBox("Введіть кінцеве значення","Перевід валюти")
For i = 1 To k
    Gr = 5.05 * i
    ListBox1.Items.Add(i &"дол. =" & Gr &"грн")
Next i
```

Запропонуйте інший (ліпший) метод оформлення результатів.

2. Цикли в задачах опрацювання елементів керування. Елемент керування ListBox містить список числових чи текстових даних. Досі ми використовували його для виведення результатів, тобто для створення списку. Цей список може бути використаний як джерело даних для розв'язування задач. Список можна створити програмним шляхом за допомогою команди циклу або «вручну» на етапі конструювання.

Розглянемо способи створення списку.

Задача про список. Створити список, що міститиме роки від 1980 до 2010.

Програмування. Програмуємо кнопку «Створити список» або програмуємо форму, якщо список має бути створений в момент завантаження проекту.

```
Dim i As Integer
For i = 1980 To 2010
    ListBox1.Items.Add(i)
Next i
```

Задача про масив і список. Масив чисел {156, 161, 174, 183, 190} занести у список.

Програмування. Програмуємо кнопку «Занести у список».

```
Dim A( ) As Integer = {156, 161, 174, 183, 190}
Dim i As Integer
For i = 0 To 4 : ListBox1.Items.Add(A(i))
Next i
```

Завдання. Розв'яжіть зворотну задачу: дано список, занести його елементи у масив. Масив вивести у зворотному порядку в інший список.

Зверніть увагу на створення безрозмірного масиву під час його оголошення. Продемонструємо, як можна створити двовимірний масив чисел B з двох рядків по три елементи в кожному

```
Dim B ( , ) As Integer = { {1, 2, 3}, {4, 5, 6}}.
```

Задача про країни. Створити програмним шляхом список з назвами чотирьох країн. Для кожної країни отримати довідку про назву столиці, кількість населення і площу території (рис.5).

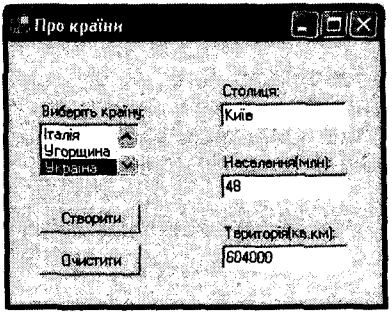


Рис. 5

Інтерфейс проекту. Використаємо ListBox1 для створення списку зі значеннями Україна, Угорщина, Італія, Греція. Список створюватиметься після натискання кнопки «Створити». Довідкова інформація має з'являтися у трьох текстових полях в момент вибору значення зі списку. Кнопка «Очистити» призначена для очищення текстових полів.

Теоретичні відомості. Масив елементів керування. Елементи списку можна вибирати. Номер вибраного елемента зберігається у властивості SelectedIndex, а значення – в SelectedItem. Це використовують для створення алгоритмів опрацювання даних. Зауважимо, що нумерація елементів у списку починається від 0. Вибір мишею елемента списку – це подія SelectedIndexChanged, визначена для списку. Реакцією на цю подію буде виведення даних у текстові поля. Зауважимо, що аналогічна подія визначена для радіокнопки, прапорця, текстового поля тощо.

Доступ до і-го елемента списку дає вираз ListBox1.Items.Item(i).

3. Клас Controls. Усі елементи керування на формі утворюють масив елементів керування класу Controls. Доступ до елементів масиву дає вираз Controls.Item(<Індекс>). Останній доданий у форму елемент є першим у масиві Controls і має індекс 0. Перший доданий елемент буде останнім у масиві й матиме індекс Controls.Count – 1 (всього елементів є Count).

Щоб переглянути всі елементи і визначити до якого типу вони належать (наприклад, до типу текстових полів), використовують таку алгоритмічну конструкцію:

```
For i = 0 To Controls.Count – 1
    If TypeOf Controls.Item(i) Is TextBox Then <дії>
Next i
```

Діями можуть бути команди очищення текстового поля, занесення в нього деякого значення тощо. Такий прийом застосовують, зокрема, для очищення великої кількості полів.

Програмування. Запрограмуємо кнопку «Створити».

```
Dim A() As String={ "Україна", "Угорщина", _
"Італія", "Греція" }
Dim i As Integer
For i = 0 To 3
    ListBox1.Items.Add(A(i))
Next i
```

Запрограмуємо подію SelectedIndexChanged вибору зі списку нового значення. Двічі клацнемо на списку і введемо такий код:

```
Select Case ListBox1.SelectedItem
Case "Україна"
    TextBox1.Text = "Київ"
    TextBox2.Text = 48
    TextBox3.Text = 604000
Case "Угорщина"
    TextBox1.Text = "Будапешт"
    TextBox2.Text = 11
    TextBox3.Text = 204000
Case "Італія"
    TextBox1.Text = "Рим"
    TextBox2.Text = 60
    TextBox3.Text = 304000
Case "Греція"
    TextBox1.Text = "Афіни"
    TextBox2.Text = 10
    TextBox3.Text = 64000
End Select
```

Запрограмуємо кнопку «Очистити».

```
Dim i As Integer
For i = 0 To Controls.Count - 1
    If TypeOf Controls.Item(i) Is TextBox Then
        Controls.Item(i).Text =""
    End If
Next i
```

Запустіть проект на виконання і запропонуйте шляхи його удосконалення.

Задачі

Задача 4.1. Модифікуйте код проекту про країни, використавши замість команди Select Case ListBox1.SelectedItem команду Select Case ListBox1.SelectedIndex.

Задача 4.2. Дані про країни збережіть у трьох явно заданих масивах (подібно до назв країн) у процедурі, де вони використовуються, і виводьте у текстові поля значення з відповідних масивів.

Задача 4.3. Усі чотири масиви створіть з використанням функції InputBox, тобто в діалоговому режимі.

Увага! У цьому випадку масиви можна створювати будь-де, але описати їх потрібно зі службовими словами Dim чи Private як глобальні, тобто перед усіма процедурами.

Виконайте задачі 4–14 з розділу «Задачі».

§ 5. ПІДПРОГРАМИ

1. Застосування підпрограм. Над великими чи громіздкими розв'язками зазвичай працює група програмістів, розподіливши роботу між собою. Кожний реалізує частину роботи або власний проект. Коли всі проекти готові, їх можуть використовувати всі учасники групи, а також інші користувачі. Розв'язок частини деякої складної чи громіздкої задачі подають у вигляді підпрограми користувача. Підпрограми поділяють на процедури і функції.

Важливою властивістю підпрограм є наявність засобів для обміну даними між собою, що можна використати, наприклад, коли результати роботи деякої підпрограми є вхідними даними для іншої підпрограми.

2. Процедури опрацювання подій. Вирізняють процедури опрацювання подій (Click, DoubleClick і багато інших), на які реагують елементи керування, і процедури користувача. Процедури опрацювання подій ми складали, починаючи від першого параграфа, а процедури користувача розглядали у розділі 1.

Часто для різних елементів керування потрібно складати одну і ту ж процедуру. Наприклад, така ситуація виникає, якщо потрібно, щоб у результаті клацання на деяких елементах чи зміни тексту в деяких елементах очищувалися текстові поля TextBox1 і TextBox2 тощо. Тоді складають одну процедуру, де відповідні елементи перелічують через кому після службового слова Handles в заголовку процедури, наприклад,

```
Private Sub MyProc_Click(<стандартний список _
        параметрів>) Handles <список потрібних елементів>
TextBox1.Text = ""
TextBox2.Text = ""
End Sub
```

3. Процедури користувача. Процедури користувача ми розглядали для мови Qbasic у розділі 1. У VB .NET є незначні відмінності.

Задача про метеостанцію. Дано набір з 24 значень температур, заміряних щогодини протягом доби, починаючи з 00 год {8, 7, 7, 6, 6, 5, 5, 6, 7, 8, 10, 12, 14, 15, 17, 17, 17, 16, 16, 15, 14, 12, 11,10}.

Визначити: 1) середньодобову температуру; 2) максимальну температуру; 3) години, коли була максимальна температура; 4) мінімальну температуру; 5) години, коли була мінімальна температура; 6) моду спостережуваних значень – значення, що найчастіше трапляються; 7) медіану спостережуваних значень – значення, що розміщене посередині числового набору; 8) середньоквадратичне відхилення; 9) кількість різних значень температур; 10) таблицю частот у вигляді таблиці значень температури і кількостей повторень кожного різного значення; 11) розкид температур – різницю між максимальним і мінімальним значенням; 12) яка температура утримувалась найбільшу кількість годин підряд.

Зауважимо, що описаний набір експериментальних даних називають вибіркою. Для його зберігання використаємо масив.

Розв'язок задачі складатиметься з 12 підпрограм опрацювання даних, які доручимо розробляти 12 програмістам. Інтерфейси проектів тут не розглядатимемо, оскільки, маючи підпрограми, можна створити відповідний інтерфейс до вподоби користувача.

Проаналізуємо, в яких випадках результати роботи підпрограми можуть давати вхідні дані для іншої підпрограми, і виокремимо їх в окрему послідовність створення. Наприклад, 11-й проект суттєво залежить від 2-го і 4-го, а 6-й проект може використати на вході результати 10-го.

Розглянемо реалізацію 2, 4 і 11-го проектів. Вважатимемо, що вибірка зберігається в цілочисловому масиві t, що містить 24 елементи від 0-го до 23-го, які відповідають годинам спостережень. Нумерація елементів у масиві починається з 0.

Спрощений інтерфейс задачі. Розташуємо на формі кнопку «Обчислити» і текстове поле TextBox1 «Годин:», куди вводитимемо кількість перших годин спостережень після 24.00, наприклад, 8, 12 чи всі 24. Розташуємо на формі три текстові поля «Максимум:», «Мінімум:», «Розкид:» для виведення результатів роботи трьох процедур. Поля мають очищуватися в результаті клацання мишею в будь-якому з них.

Програмування. Запрограмуємо кнопку «Обчислити».

```
Dim t() As Integer = {8, 7, 7, 6, 6, 5, 5, 6, 7, 8, 10, 12, _
14, 15, 17, 17, 17, 16, 16, 15, 14, 12, 11, 10}
n = TextBox1.Text
Dim Maxt, Mint, Rozkyd As Integer
Call Pr2(t, n, Maxt) :  Call Pr4(t, n, Mint)
Call Pr11(Maxt, Mint, Rozkyd)
TextBox2.Text = Maxt
TextBox3.Text = Mint
TextBox4.Text = Rozkyd
```

Розглянемо інший спосіб створення масиву – введення елементів масиву в режимі діалогу:

```
Dim t( ) As Integer, i As Integer
For i = 0 To n
    t(i) = Input("Введіть "& i &"-й елемент", "Метеостанція")
Next i
```

Складемо процедуру Pr2, яка у змінній Maxt повертає найбільше значення температури.

```
Sub Pr2(ByVal t() As Integer, ByVal n As Integer, _
                ByRef Maxt _As Integer)
        Dim i As Integer
        Maxt = t(0)
        For i = 1 To n
            If t(i) > Maxt Then Maxt = t(i)
        Next i
End Sub
```

Складемо процедуру Pr4, яка у змінній Mint повертає найменше значення температури.

```
Sub Pr4(ByVal t() As Integer, ByVal n As Integer, _
                ByRef Mint _As Integer)
        Dim i As Integer
        Mint = t(0)
        For i = 1 To n
            If t(i) < Mint Then Mint = t(i)
        Next i
End Sub
```

Складемо процедуру Pr11, яка опирається на результати роботи процедур Pr2 I Pr4.

```
Sub Pr11(ByVal Maxt As Integer, ByVal Mint As Integer, _
                ByRef Rozkyd As Integer)
        Rozkyd = Maxt – Mint
End Sub
```

Створимо процедура MyProc_Click так: двічі клацнемо на елементі TextBox1 – з'явиться заготовка процедури, яку перетворимо до такого вигляду:

```
Sub MyProc_Click(ByVal sender As Object, ByVal e As _
  System.EventArgs) Handles TextBox1.Click, _
                TextBox2.Click, TextBox3.Click
        TextBox1.Text = ""
        TextBox2.Text = ""
        TextBox3.Text = ""
End Sub
```

Інші процедури цього проекту складіть самостійно.

Реалізуйте проект та поекспериментуйте з ним, вводячи різну кількість годин і очищаючи текстові поля.

4. ByVal і ByRef. Службове слово ByVal позначає параметри, які передають у процедуру значення вхідних даних. Ці параметри не можуть змінювати значень у тілі процедури. Такий режим діє за замовчуванням. Слово ByRef слід зазначати для тих параметрів, які повертають з процедури адреси результатів. Значення таких параметрів можна (і треба) змінювати в тілі процедури.

5. Доступність процедур і функцій. Якщо перед словом Sub чи Function написано слово Private, то процедура чи функція доступна в тому контейнері, де вона описана (в коді форми, коді елемента керування тощо). Щоб процедура чи функція була загальнодоступною, використовують службове слово Public або не зазначають жодного. Такі процедури і функції вважаються визначеними глобально. Доступ до таких процедур чи функцій з віддалених місць проекту здійснюється із зазначенням назви форми так:

```
<назва форми>.<назва процедури>
```

Інколи процедури треба захистити від стороннього доступу, наприклад, через локальну мережу. Тоді пишуть слово Protected.

Оголошення Static (на відміну від VB 6) не можна застосовувати до процедури. Його слід застосовувати лише до відповідних локальних змінних процедури. Значення статичних локальних змінних не втрачаються після виходу з процедури чи функції, як це характерно для локальних параметрів.

6. Функції користувача. Функція користувача у VB .NET має такий загальний вигляд:

```
Function <назва>(<список формальних параметрів>)
...
Return <вираз>
End Function
```

Дія функції. Обчислюється вираз і його значення надається назві функції.

Задача про середнє арифметичне. Обчислити середнє арифметичне трьох чисел, використовуючи функцію.

Складемо для цього функцію під назвою Sa:

```
Function Sa(ByVal a As Single, ByVal b As Single, _
                ByVal c As Single)
Return (a + b + c) / 3
End Function
```

Переконайтеся, що Sa(10, 15, 17) дає значення 14.

7. Модулі. Процедури і функції, які мають практичне значення, користувачі можуть використовувати багаторазово, причому в різних проектах. Для цього їх організовують у модулі, що зберігаються в окремих файлах.

Створення модуля. Досі ми працювали в стартовому модулі класу головної форми. Для створення власного модуля потрібно в меню **Project** поточного проекту виконати команду **Add New Item**. У діалоговому вікні цієї команди виберіть шаблон Module і надайте модулю у полі Name власне ім'я, наприклад MyModule1 тощо. Клацніть на кнопці Open – модуль долучиться до структури проекту, а користувач отримає заготовку для тіла модуля

```
Module MyModul1
...
End Modul
```

Занесіть сюди текст одної чи декількох підпрограм (процедур чи функцій) і збережіть модуль командою **File, Save** MyModyle1.vb у папці поточного проекту або командою **Save As...** в іншій папці.

Використання модуля. Створіть новий проект, наповніть його потрібним змістом і у меню **Project** виконайте команду **Add Existing Item**. У вікні цієї команди виберіть потрібний модуль, відшукавши його у файловій системі, і натисніть на кнопку **Open** – модуль буде доданий до поточного проекту. Можете використовувати його підпрограми як свої власні.

Завдання. З побудованих у цьому параграфі підпрограм створіть власний модуль.

Задачі

Задача 5.1. Переробіть процедури Pr2, Pr4, Pr11 у функції.

Задача 5.2. Модифікуйте проект так: розглядається довільний проміжок доби між двома заданими годинами.

Задача 5.3. Створіть розв'язок задачі як сукупність проектів, які розробляє група програмістів у локальній мережі. Кожний учасник групи може використати процедуру, створену на іншому комп'ютері. Кожний учасник може зібрати розв'язок задачі з доступних йому через мережу компонент.

Задача 5.4. Розв'яжіть задачі 15–17 з розділу «Задачі», використовуючи процедури й обмін даними через локальну мережу.

§ 6. СТРУКТУРИ І ФАЙЛИ

1. Структури і команда With. Структура у VB .NET – це те ж саме, що тип даних запис у мові Qbasic. Опис структури починається зі слова Structure, а саме:

```
Structure <назва структури>
    <назва поля 1> As <тип поля 1>
    ...
    <назва поля n> As <тип поля n>
End Structure
```

Перед назвою кожного поля зазначають службове слово Dim, Private чи Public. Наприклад,

```
Structure MyStructure
    Dim Prizv As String
    Dim Rik As Integer
    Dim Rist As Integer
    Dim Vaga As Single
    Dim Khar As String
End Structure
```

Після опису структури оголошують потрібну кількість змінних типу (класу) структура чи масив структур. Наприклад, оголосимо змінну Ms типу (класу) MyStructure: Dim Ms As MyStructure.

До конкретного поля структури можна звертатися так: Ms.Prizv, Ms.Rik тощо. Щоб постійно не писати назву змінної типу структури, застосовують команду With–End With, наприклад, так:

```
With Ms
    .Prizv = "Вірт": .Rik = 1981: .Rist = 182 'і т.д.
End With.
```

2. Файли даних послідовного доступу. У файли даних послідовного доступу можна заносити будь-які дані, наприклад, результати роботи попередніх програм чи дані типу структури.

Для відкриття файлу даних послідовного доступу призначена команда

```
FileOpen(<номер файлу>, "назва файлу зі шляхом", _
        <тип доступу>)
```

Номер файлу задають явно числом 1, 2, 3 тощо, або неявно деякою змінною, значення якої визначають функцією FreeFile().

Тип доступу – один з можливих: OpenMode.Output – для створення файлу, OpenMode.Input – для введення даних з файлу, OpenMode.Append – для дописування даних в наявний файл.

Щоб занести дані у файл, використовують такі команди:

```
Print (<номер файлу>, <список даних>)
Write(<номер файлу>, <список даних>)
WriteLine(<номер файлу>, <список даних>)
```

Команда Print заносить дані у рядок файлу підряд, якщо елементи списку даних розмежовані крапкою з комою, і у зони, якщо вони розмежовані комами.

Команда Write заносить дані у поточний рядок і ставить кому після кожного даного.

Команда WriteLine формує рядок з даними, розмежовуючи їх комами, і ставить позначку «кінець рядка».

Зчитувати дані з файлу можна лише по одному полю за допомогою команди

```
Input(<номер файлу>, <змінна відповідного типу>)
```

Можна зчитати один рядок з файлу як дане типу String так:

```
LineInput(<номер файлу>,<змінна типу String>)
```

Під час зчитування даних з файлу за допомогою функції EOF(<номер файлу>) слід перевіряти, чи є ще у ньому дані.

Після опрацювання файли закривають командою

```
FileClose(<список номерів>)
```

3. Задача на тему створення файлу. Створити файл послідовного доступу з даними про декількох осіб (учнів класу, студентів, співробітників тощо). Для кожної особи, крім прізвища, структура має містити такі дані: рік народження, зріст, вагу, характер тощо.

Модель, інтерфейс і сценарій. Дані про особу заноситимемо у структуру даних типу MyStructure, що має поля Priz, Rik, Rist, Vaga, Khar. Коли структура буде створена, занесемо дані зі структури у файл "d:\Myfolder\Myfile".

Інтерфейс проекту складатиметься з таких елементів: ListBox1 (список з прізвищами), чотирьох текстових полів з відповідними підписами («Рік:», «Зріст:», «Вага:», «Характер:»), куди «вручну» вводитимемо дані про кожну особу, а також кнопки Button1 («У файл») для занесення даних у структуру і зі структури у файл.

Нехай список містить п'яти елементів: Атанасов, Вірт, Гейтс, Возняк, Голеріт. Список створимо на етапі конструювання форми «вручну». Для цього у вікні Properties елемента ListBox1 виберемо у властивості Item значення Collection і клацнемо на кнопці з трьома крапками. Відкриється вікно String Collection Editor, куди у стовпець (натискаючи Enter) занесемо прізвища осіб і натиснемо клавішу ОК.

Після завантаження форми треба вибрати прізвище зі списку і ввести «вручну» заздалегідь приготовані чи видумані дані у поля. Коли натиснути кнопку «У файл», дані заносяться з елементів керування у структуру, а зі структури – у файл.

Програмування. Клацнемо двічі по формі та в код форми занесемо команду

```
FileOpen(1, "d:\Myfolder\Myfile", OpenMode.Output).
```

Опис структури MyStructure виконаємо перед кодом форми (на початку модуля) і оголосимо одну структуру Ms типу MyStructure.

Запрограмуємо кнопку «У файл».

```
With Ms
    .Prizv = ListBox1.SelectedItem
    .Rik   = TextBox1.Text
    .Rist  = TextBox2.Text
    .Vaga  = TextBox3.Text
    .Khar  = TextBox4.Text
     WriteLine(1, .Prizv, .Rik, .Rist, .Vaga, .Khar)
End With
```

Вибір прізвища зі списку має вести до очищення усіх текстових полів. Двічі клацнемо на ListBox1 і введемо текст коду:

```
Dim i As Integer
For i = 0 To Controls.Count - 1
    If TypeOf Controls.Item(i) Is TextBox Then
       Controls.Item(i).Text =""
    End If
Next i
```

Виконання. Запустіть проект на виконання. Після виправлення описок виберіть у списку прізвище і введіть відповідні дані у текстові поля. Натисніть кнопку «У файл». Повторіть ці дії тричотири рази. Закрийте проект і перегляньте створений файл за допомогою редактора NotePad.

4. Задача про використання файлу. Створити проект, який використовуватиме побудований вище файл так: переглядатиме файл і після натискання кнопки «Знайти і показати» виводитиме на форму дані про осіб, чий зріст більший, ніж 160 см.

Програмування. Розмістіть на формі п'ять текстових полів і кнопку. Відкрийте файл для читання командою

```
FileOpen(1, "d:\Myfolder\Myfile", OpenMode.Input)
```

Запрограмуйте кнопку «Знайти і показати».

```
Dim i As Integer
For i = 0 To Controls.Count - 1
      If TypeOf Controls.Item(i) Is TextBox Then
      Controls.Item(i).Text = ""
Next i
While Not EOF(1)
  With Ms
       Input(1, .Prizv)
```

```
            Input(1, .Rik)
            Input(1, .Rist)
            Input(1, .Vaga)
            Input(1, .Khar)
            If .Rist > 160 Then
               TextBox1.Text = .Prizv
               TextBox2.Text = .Rik
               TextBox3.Text = .Rist
               TextBox4.Text = .Vaga
               TextBox5.Text = .Khar
               Exit While
            End If
         End With
      End While
```

5. Файли даних прямого доступу. Файл даних прямого доступу використовують для створення файлів записів (файлів структур). Під час роботи з файлами структур прямого доступу (але не масивами структур) опис полів типу String структури має бути уточнений зазначенням їхньої конкретної довжини за допомогою виразу-декларації <VBFixedString(довжина поля)>, який записують у кутових дужках перед словом Dim чи Public в описі відповідного поля.

Файли прямого доступу відкривають командою

```
FileOpen(<номер файлу>, "назва файлу зі шляхом", _
<тип доступу>, <допустимі дії з даними>, <дозвіл на _
спільне використання>, <довжина запису>)
```

Тип доступу єдиний – OpenMode.Random. Четвертий і п'ятий параметри є необов'язкові. Зазвичай значення четвертого параметра OpenAcess.ReadWrite (можна читати чи записувати дані), а п'ятого – OpenShare.Shared (доступний всім). Останній параметр – це число, яке явно описує довжину запису в байтах, або функція Len(<назва запису>).

Записи заносять у файл командою

```
FilePut(<номер файлу>,<змінна>,<номер запису>)
```

Якщо номер запису не зазначати, то запис дописується в кінець файлу. Запис зчитують з файлу командою

```
FileGet(<номер файлу>,<змінна>,<номер запису>)
```

Завдання. Перероблiть попередні два проекти, використавши файл даних прямого доступу.

Задачі

Задача 6.1. Перероблiть будь-яку задачу на свій вибір, організувавши виведення результатів у файл даних послідовного доступу (без використання структур) і перегляньте файл у редакторі NotePad.

Задача 6.2. Створіть структуру і файл даних на тему «Успішність у навчанні».

Задача 6.3. Використайте файл даних «Успішність» і виведіть в інший файл прізвища тих осіб, які мають високі оцінки з математики і фізики.

Задача 6.4. Розв'яжіть задачі 18–21 з розділу «Задачі».

§ 7. РЯДКИ

1. Дані класу String (текстові дані, рядки). Будь-яка послідовність символів, взята в лапки, утворює дане-літерал з класу String. Змінні, які отримують такі значення, є об'єктами класу String. Для текстових даних визначена операція злиття (символ & або +, якщо після + немає числових даних), операції порівняння, а також певні властивості і функції. Інший термін для таких функцій – методи об'єкта класу String.

Нехай a, b, a1, a2,... – це дані типу String, c, c1, c2 n,m – числові дані, g – масив даних типу String. Розглянемо головні функції, визначені для цих даних. Такі функції можна застосовувати лише у виразах. Нехай a="Інформатика".

Властивість a.Length дає довжину рядка a.

Наприклад, c = a.Length. Результат c = 11.

Функція a.Substring(n) дає підрядок рядка a, починаючи з позиції n, де відлік позицій ведеться від нуля.

Наприклад, a1 = a.Substring(5). Результат a1 = "матика";

Функція a.Substring(n,m) дає підрядок рядка a довжиною m символів, починаючи з символу n.

Наприклад, a2 = a.Substring(2,5). Результат a2="форма".

Функція a.IndexOf(b, n) дає число – позицію входження рядка b у рядок a, починаючи із позиції n.

Наприклад, c1 = a.IndexOf("ф", 0). Результат c1 = 2.

Функція a.Insert(n,b) вставляє рядок b у рядок a перед позицією з номером n.

Наприклад, a3 = a.Insert(0,"К"). Результат a3 ="КІнформатика".

Функція a.ToLower утворює рядок з рядкових літер рядка a.

Наприклад, a4 = a.ToLower. Результат a4="інформатика".

Функція a.ToUpper утворює рядок з прописних літер рядка a.

Наприклад, a5 = a.ToUpper. Результат a5 = "ІНФОРМАТИКА".

Метод Int32.Parse(a) повертає числове ціле значення символьного рядка.

Наприклад, c2 = Int32.Parse("12345"). Результат c2=12345.

Метод Single.Parse повертає десяткове значення рядка.

Наприклад, c3 = Single.Parse("12.34"). Результат c3=12.34.

Метод c.ToString переводить числове дане с в дане типу рядок.

Наприклад, a = 123.ToString. Результат a= "123".

Властивість a.Chars(n) типу Char дає символ з рядка a, що є на позиції з номером n.

Наприклад, a6 = a.Chars(3). Результат a6 ="o".

Функція a.Remove(n,m) дає рядок, що утворюється шляхом вилучення m символів з рядка a, починаючи з позиції n.

Наприклад, a7 = a.Remove(2,4). Результат a7="Інатика".

Функція a.Split(b, n) повертає масив значень типу String, який утворюється шляхом поділу рядка a на частини, де b є символом-розділювачем, а n – розмір масиву.

Наприклад, g = a.Split("р",2). Результат g(0) ="Інфо", g(1) = "матика".

Функція a.Join(b, g) утворює дане типу String з масиву g, де b є символом-розділювачем.

Наприклад a = a.Join(",",g). Результат a="Інфо,матика".

Функція Chr(c) дає символ з таблиці кодів ASCII за номером c, а функція Asc(a) дає номер символу a в таблиці ASCII. Наприклад, Chr(65) дає "A", а Asc(66) – "B".

Корисними є також функції Trim, PadLeft, PadRight, StartWith, IndexOfAny (їх призначення можна дізнатися з довідкової системи).

2. Опрацювання текстових даних. Головні принципи роботи з текстовими даними розглядалися в розділі 1. У VB .NET змінилися лише назви функцій і вигляд команд. З'явились також нові функції, наприклад, Split і Join. Їх використовують під час роботи з файлами даних, де буває потрібно замінити розділювач-пропуск між даними на розділювач-кому чи навпаки.

Задача. Нехай з файлу зчитано дане-рядок, в якому окремі поля розмежовані пропусками. Створити дане-рядок, в якому ці поля будуть розмежовані комами.

Програмування. Фрагмент коду буде такий:

```
Dim a As String, g() As String
a = "Федя Коля Петя Вася"
g = a.Split(" ", 4)
a = a.Join(",", g)
```

Отримаємо a = "Федя,Коля,Петя,Вася".

Задачі

Задача 7.1. Наведіть результати застосування текстових функції з пункту 1 до даного a= "Філософія".

Задача 7.2. Розв'яжіть задачу № 9 з розділу «Задачі».

Задача 7.3. Розв'яжіть задачу № 10 з розділу «Задачі».

§ 8. ГРАФІКА

1. Виведення даних на форму. Дані безпосередньо на форму можна вивести, якщо їх розглядати як елементи графічного об'єкта, наприклад, з іменем c, що належить до класу Graphics форми. Графічних об'єктів може бути один або декілька. Даними можуть бути тексти, лінії, фігури (прямокутники, кола, еліпси, полігони тощо). Геометричні фігури рисують за допомогою пера (об'єкта p з класу Pen) певної товщини (Width) і кольору (Color). Замальовують їх за допомогою пензля (об'єкта b з класу SolidBrash чи іншого), що також характеризується товщиною і кольором. Пензель використовують, зокрема, для виведення текстів. Для цього залучають також об'єкт з класу Font, що задає атрибути шрифту або за замовчуванням (об'єкт f з властивостями DefaltFont), або як об'єкт f1 із заданими властивостями: назва шрифту, розмір символів у пунктах, стиль написання (Bold, Italic тощо).

Над графічними об'єктами визначені такі методи: виведення на принтер Print, витирання Refresh, виведення даного типу String в графічний об'єкт – DrawString(), рисування лінії – DrawLine(), прямокутника – DrawRectangle(), еліпса – DrawEllipse(), замальовування прямокутника FillRectangle(), еліпса – FillEllipse() тощо.

Метод DrawString() має такі параметри: рядок даних, шрифт і пензель як об'єкти, координати виведення.

Метод DrawLine() має параметри: перо p як об'єкт і координати (X1, Y1, X2, Y2) двох точок на площині.

Інші методи мають по п'ять параметрів: перо або пензель відповідно, чотири параметри прямокутника або прямокутник як об'єкт класу Rectangle.

Прямокутник задають чотирма параметрами: двома координатами лівої верхньої вершини, шириною і висотою. Еліпс визначений як фігура, вписана в прямокутник, тобто тими ж атрибутами, що і прямокутник. Коло визначене як фігура, вписана в квадрат. Квадрат – це прямокутник з рівними сторонами. Точку можна визначити як квадрат зі стороною 1 піксель чи круг такого ж радіуса.

Прямокутник – це об'єкт, наприклад z, класу System.Drawing. Rectangle. Об'єкт прямокутник має такі властивості: z.X, z.Y – координати лівої верхньої вершини, z.Width, z.Height – ширина і висота об'єкта в пікселях.

Координати на формі відлічують у пікселях від лівого верхнього кута. X-коодинату відлічують зліва направо, а Y-координату – зверху донизу.

2. Рисування емблеми. У верхній лівій частині форми нарисуємо квадрат зі стороною 120 пікселів. Замалюємо квадрат червоним кольором. У квадрат впишемо коло. Замалюємо круг синім кольором. У крузі напишемо великими білими буквами своє ім'я, наприклад, ФЕДЯ.

Інтерфейс і програмування. Розташуємо внизу на формі одну кнопку «Нарисувати» і запрограмуємо її.

```
Dim c As Graphics = CreateGraphics()
Dim f As Font = Form1.DefaultFont
Dim f1 As Font = New Font("Ariel", 20, FontStyle.Bold)
Dim b As New SolidBrush(Color.Red)
Dim p As New Pen(Color.Blue)
Dim z As New System.Drawing.Rectangle
z.X = 30 : z.Y = 30
z.Width = 120 : z.Height = 120
p.Width = 4
p.Color = Color.Red
c.DrawRectangle(p, z)
p.Color = Color.Green
p.Width = 2
c.DrawRectangle(p, z)
c.FillRectangle(b, z)
p.Color = Color.Blue
c.DrawEllipse(p, z)
b.Color = Color.Blue
c.FillEllipse(b, z)
b.Color = Color.White
c.DrawString("ФЕДЯ", f1, b, 43, 74)
```

Зверніть увагу на ефект розмивання контура прямокутника зеленим кольором шляхом зміни ширини і кольору пера.

Методом c.Refresh() ліквідовують нарисовану картинку чи текст.

3. Елемент керування PictureBox. Поширеним способом рисування чи виведення тексту є виведення не на форму, а в об'єкт PictureBox, що є контейнером для інших об'єктів. Для цього спочатку оголошують об'єкт, наприклад, c1 з класу Graphics, що прикріплений до об'єкта PictureBox1, так:

```
Dim c1 As Graphics = PictureBox1.CreateGraphics.
```

Якщо в програмі замінити назву об'єкта c на c1, то отримаємо емблему не безпосередньо на формі, а в контейнері PictureBox1.

4. Створення графічного редактора. Створимо графічний редактор, де можна рисувати лінії під час переміщення натиснутої клавіші миші двома кольорами: синім, якщо натиснута ліва клавіша, і червоним, якщо натиснута права. Рисунок можна зберегти у файлі або вилучити.

Інтерфейс і програмування. Розташуємо на формі елемент керування PictureBox1, який міститиме рисунок, і задамо його властивість BorderStyle як Fixed3D. Внизу ліворуч розташуємо два підписані текстові поля «Ширина пера:», «Ширина гумки:», а праворуч – дві кнопки «Очистити» і «Зберегти».

Для елемента PictureBox1 запрограмуємо подію MouseMove так:

```
Dim c As Graphics = PictureBox1.CreateGraphics
Dim p As New Pen(Color.Blue)
Dim x, y As Integer
x = MousePosition.X - ActiveForm.Location.X - 5 - _
    PictureBox1.Location.X
y = MousePosition.Y - ActiveForm.Location.Y - 30 - _
    PictureBox1.Location.Y
p.Width = TextBox1.Text
If MouseButtons = MouseButtons.Left Then
   p.Color = Color.Blue
Else
   p.Color = Color.Red
End If
If Draw Then c.DrawLine(p, x, y, x + 1, y + 1)
```

Тут значення x та y спеціальним чином обчислюємо так, щоб кінець стрілки вказівника миші показував позицію рисування.

Глобальна змінна Draw визначає початок і кінець процесу рисування, що настає після натискання чи відпускання клавіші миші.

Для елемента PictureBox1 запрограмуємо подію MouseDown так:

```
Draw = True.
```

Для елемента PictureBox1 запрограмуємо подію MouseUp так:

```
Draw = False.
```

Змінну Draw треба описати перед усіма процедурами як глобальну, а саме: Dim Draw As Boolean.

Запрограмуємо кнопку Button1 «Очистити» так:

```
PictureBox1.ForeColor = System.Drawing.Color.Gray
PictureBox1.Refresh()
```

Виконайте проект. Нарисуйте двоколірну фігуру, змінюючи товщину пера під час рисування.

5. Корисні властивості елемента керування PictureBox. Елемент керування PictureBox використовують як контейнер для готової картинки, яку поміщають в нього з графічного файлу. Важливими є такі значення властивості SizeMode цього елемента: Normal – можна вручну змінювати розміри елемента, StretchImage – автоматично розтягує чи стискує картинку до заданих користувачем розмірів елемента, AutoSize – робить розміри елемента такі, як і картинки, CentreImage – розташовує картинку посередині елемента.

Картинку вставляють з файлу за допомогою властивості Image.

Картинку можна зробити видимою командою PictureBox1.Visible = True або невидимою PictureBox1.Visible= False. Ці ж дії можна виконати, задавши властивість Visible елемента PictureBox1 у вікні Properties.

Під час роботи з картинками значення властивості BorderStyle задають як None чи FixedSingle.

Елементи керування PictureBox та Label можна використати для створення анімації. Наприклад, наступний код змусить картинку, поміщену в PictureBox1, переміщатися зліва направо (зміною значення властивості Left) з певною швидкістю, що залежить від тривалості виконання внутрішнього циклу:

```
Dim x, d As Integer
For x = 40 To 200
   PictureBox1.Left = x
   For d = 1 To 1000000 : Next d
Next x
```

Рух у вертикальному напрямку регулюється значенням властивості PictureBox1.Top.

6. Випадкові числа. Спочатку потрібно оголосити об'єкт класу системного генератора випадкових чисел

```
Dim MyGenerator = New System.Random
```

Після цього випадкове число з діапазону (a, b) утворюють так:

MyNumber=MyGenerator.Next(a, b).

Випадкові числа використовують для моделювання різних даних: значень елементів масивів, координат об'єктів під час рисування, створення ігор тощо.

Задачі

Задача 8.1. Модифікуйте графічний редактор, підібравши однакові колір фону PictureBox1 і колір рисування правою клавішею, щоб права клавіша функціонувала як гумка заданої у TextBox2. Text ширини.

*Задача 8.2**. Запрограмуйте кнопку «Зберегти». Для цього треба ознайомитися з додатковими відомостями про роботу з зовнішніми файлами.

Задача 8.3. Лінія у нашому графічному редакторі утворюється як сукупність штрихів. Створіть лінію, точки якої моделюватимуться квадратами зі стороною 1 або кругами.

*Задача 8.4**. Модифікуйте графічний редактор, щоб значення кольорів можна було передавати з форми у програму, зважаючи на те, що дане типу String не конвертується в системне значення кольору Color.Red тощо.

Задача 8.5. Реалізуйте такий проект: клацання мишею на формі веде до рисування кола в точці, куди вказує курсор, і виведення координат курсору.

Задача 8.6. Через форму досить швидко пролітає тарілка, декілька тарілок або об'єкти-мішені на зразок тарілки з'являються на мить на екрані. Потрібно їх збити клацанням мишею на них. У разі влучення лічильник влучень збільшується на одиницю чи настають певні події (з'являються повідомлення, перефарбовується тло форми, звучать фанфари тощо).

Методичні рекомендації

Вивчаючи мову Бейсик, важливо знати еволюційний шлях її розвитку. Вирізняють чотири покоління мови Бейсик: 1) «ранній Бейсик» у 60–70-х роках існував у вигляді великої кількості версій-інтерпретаторів, характеризувався неструктурованими командами і простими типами даних; його справедливо критикували професіонали; він поступався іншим мовам за можливостями, але залишався найдоступнішою і найзрозумілішою мовою програмування для початківців; 2) «швидкий Бейсик» став у 80-х роках компілятором у середовищі Quick Basic і залишився інтерпретатором у модифікованому середовищі Qbasic, отримав від Microsoft структуровану систему команд, розвинені типи даних і вже не поступався популярній у той час мові Паскаль; 3) «візуальний Бейсик» (серія Visual Basic і VBA) успадкував у 90-х роках систему команд і типи даних від швидкого Бейсика, а завдяки наявності засобів візуального програмування став поряд з лідерами у світі інструментальних засобів швидкого проектування програм, хоча засоби ООП були розвинені слабо; 4) Visual Basic. NET став потужнім інструментом візуального програмування з повноцінною концепцією ООП, претендує на лідерство серед аналогічних інструментальних засобів, успадкував від попередників усе ліпше, але наявність принципових новинок концептуально відрізняє його як від Qbasic, так і від Visual Basic 6 тощо.

Як користуватися цим посібником? Рекомендуємо обов'язково ознайомитися спочатку з розділом 1, оскільки виклад мови Qbasic у книжці ведеться відповідно до концепцій Visual Basic 6 і Visual Basic .NET. Читач почуватиметься особливо комфортно, вивчаючи розділи 3 або 4 після розділу 1.

Вивчення мови можна почати з розділів 3 або 4, але тоді треба буде регулярно заглядати до розділу 1, де описано базові поняття мови Бейсик (алгоритмічні конструкції та основні типи даних).

Можна відразу користуватися середовищем Visual Basic 6, вивчаючи розділ 1, адже програми легко переносяться у середовище Visual Basic з Qbasic чи Quick Basic 4.5 (чи Turbo Basic, чи Power Basic).

Вивчаючи лише Visual Basic, рекомендуємо адаптувати і реалізувати проекти, описані у розділі 4.

Можна розпочати навчання з розділу 4 і час від часу зазирати у розділ 1.

У посібнику описано лише найважливіші аспекти візуального програмування в середовищах Visual Basic 6, VBA і Visual Basic .NET. Але цього достатньо, щоб самостійно опанувати такі теми, як «Web-форми» і «Web-сервіси». Наполегливо рекомендуємо це зробити, якщо на вашому комп'ютері налаштований Web-сервер або встановлений IIS (Internet Information Server).

Бажаємо приємного навчання!

Список рекомендованої літератури

1. *Алексеев В. Е., Ваулин А. С., Петрова Г. Б.* Вычислительная техника и программирование. Практикум по программированию: Практ. пособие / Под ред. А. В. Петрова. М., 1991.

2. Введение в программирование на языке *Microsoft BASIC*: Учеб. пособие /*Ю. Я. Максимов, С. В. Осипов, А. В. Потемкин* и др. Под ред. В. Г. Потемкина. М., 1991.

3. *Глинський Я. М.* Інформатика: Навч. посіб. Львів, 1992.

4. *Глинский Я. Н., Анохин В. Е., Ряжская В. А.* Бейсик. Qbasic и Visual Basic. СПб: ДиаСофтЮП, 2002.

5. *Касаткин В. Н.* Информация, алгоритмы, ЭВМ: Пособие для учителя. М., 1991.

6. *Кергаль М.* Методы программирования на Бейсике. М., 1991.

7. *Кетков Ю. Л. GW-, Turbo- и Quick Basic* для IBM PC. М., 1992.

8. *Мейер Б., Боддуэн К.* Методы программирования. М., 1981.

9. *Очков В. Ф., Пухначев Ю. В.* 24 этюда на Бейсике. М., 1988.

10. *Паранчук Я. С., Глинський Я. М., Мороз В. І.* Бейсик: програмування в середовищах Qbasic, Turbo Basic, Power Basic. К., 1998.

11. *Райманс Г. Г.* Qbasic: Основы языка программирования (вводный курс). К., 1992.

12. *Светозарова Г. И., Мельников А. А., Козловский А. В.* Практикум по программированию на языке Бейсик: Учеб. пособие для вузов. М., 1988.

13. *Уолш Б.* Программирование на Бейсике. М., 1988.

14. *Чернов Б. И.* Программирование на алгоритмических языках Бейсик, Фортран, Паскаль: Кн. для внеклас. чтения. М., 1991.

15. *Ананьев Н., Федоров А.* Самоучитель Visual Basic 6.0. СПб.: БХВ-Петербург, 2002.

16. *Гарнаев А.* Самоучитель VBA. СПб.: БХВ-Петербург, 2003.

17. *Коропов Б.* VBA: специальный справочник. СПб.: Питер, 2003.

18. *Райтингер М., Мус Г.* Visual Basic 6.0. BHV. К., 2001.

19. *Сайлер Б., Спотте Д.* Использование Visual Basic 6. М., 2001.

20. *Хорев В. Д.* Самоучитель программирования на VBA в Microsoft Office. Юниор. К., 2001.

21. *Кузьменко В. Г.* VBA 2003. ООО Бином-Пресс. М., 2004.

22. *Малачівський П. С.* Програмування в середовищі Visual Basic. Бескид Біт. Львів, 2004.

23. *Есипов А. С.* Информатика и информационные технологии для учащихся школ и колледжей. СПб.: БХВ-Петербург, 2004.

24. Основи програмування. Видавнича група BHV. К., 2004.

25. *Зак Д.* Самоучитель Visual Basic .NET. BHV. К.; СПб.: Питер, 2003.

26. *Валнум К.* Visual Basic .NET. ООО «Изд. АСТ». М., 2004.

27. *Шапошников И. В.* Web-сервисы Microsoft .NET. СПб.: БХВ-Петербург, 2002.

ЗМІСТ

Розділ 1. QBASIC 3
1. Основні поняття 3
2. Типи даних 7
3. Команда присвоєння. Функції. Вирази 12
4. Введення даних 16
5. Виведення результатів 19
6. Прості програми 24
7. Розгалуження 26
8. Вибір 32
9. Цикли (1) 35
10. Цикли (2) 41
11. Текстові дані 43
12. Масиви 49
13. Функції користувача і підпрограми 56
14. Файли даних 63
15. Файли даних прямого доступу 68
16. Записи 70
17. Графіка (1) 72
18. Графіка (2) 76
19. Музика 78
20. Середовище програмування QBasic та Visual Basic 6 81

Розділ 2. ЗАДАЧІ 84

Розділ 3. VISUAL BASIC 6 ТА VBA 98
1. Вступ до візуального програмування 98
2. Задача про анкету 102
3. Задача про обмін валюти 110
4. Задача табулювання функції 114
5. Задача про адресну книжку 122
6. Створюємо навчальну програму 126
7. Поняття про Visual Basic for Applications (VBA) 133
8. VBA. Задача про таблицю кодів ASCII 140
9. VBA. Розв'язування нелінійного рівняння 142
10. VBA. Задача про облік ліків на складі 145
11. VBA. Задача про захист інформації 150

Розділ 4. VISUAL BASIC .NET 156
1. Основні поняття 156
2. Введення-виведення даних 163
3. Вибір способів опрацювання даних 167
4. Цикли і масиви 169
5. Підпрограми 173
6. Структури і файли 177
7. Рядки 182
8. Графіка 184

Методичні рекомендації 188

Список літератури 189