АКАДЕМІЯ ВНУТРІШНІХ ВІЙСЬК МВС УКРАЇНИ

С.О. Белінський, В.Є. Козлов, В.О. Середа

ІНСТРУМЕНТАЛЬНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Навчальний посібник

Харків
2011

УДК 681.3(075)
Б 43

Белінський С.О. Інструментальне програмне забезпечення: навч. посіб./ С.О. Белінський, В.Є. Козлов, В.О. Середа. – Х.: Акад. внутрішніх військ МВС України, 2011. – 55 с.

Посібник призначений для вивчення блоку змістовних модулів із аналогічною назвою навчальних дисциплін кафедри інформатики та прикладних інформаційних технологій. В ньому наведені основні відомості з технології програмування, використання засобів візуального проектування додатків на прикладі системи Visual BASIC:

Особливістю посібника є те, що наведені у прикладах і вправах програмні коди можуть бути відтворені суб'єктами навчання шляхом їх набору у відповідних шаблонах додатків, що розробляються. Головне при цьому – розуміння виконуваних дій.

Для перевірки отриманих знань, умінь і навичок посібник містить питання для самоконтролю, завдання для самостійного відпрацьовування та контрольні завдання.

Рецензенти:

О.П. Кондратенко, доктор технічних наук, професор, професор кафедри автомобільної техніки;

В.Г. Кобзєв, кандидат технічних наук, старший науковий співробітник, доцент кафедри інформаційних управляючих систем Харківського національного університету радіоелектроніки

Рекомендовано до видання в якості навчального посібника Вченою радою Академії внутрішніх військ МВС України (протокол №20 від 29 червня 2011 р.)

ЗМІСТ

1. ЗАГАЛЬНІ ВІДОМОСТІ ПРО СИСТЕМИ ПРОГРАМУВАННЯ

Широке застосування комп'ютерів, зокрема персональних, стало можливим завдяки розвинутому програмному забезпеченню. Але існує безліч конкретних завдань, для яких неможливо заздалегідь вибрати відповідний програмний додаток. Для розробки оригінальних програм і вирішення фахових завдань звичайно використовують **інструментальне програмне забезпечення** (інструментальні системи, системи програмування) – програми, що забезпечують створення та експлуатацію нових програм для комп'ютера. Їх основу складають **мови програмування** (рис. 1.1) – формальні мови для опису даних (інформації) і алгоритмів (програм) її обробки на електронних обчислювальних машинах (ЕОМ).

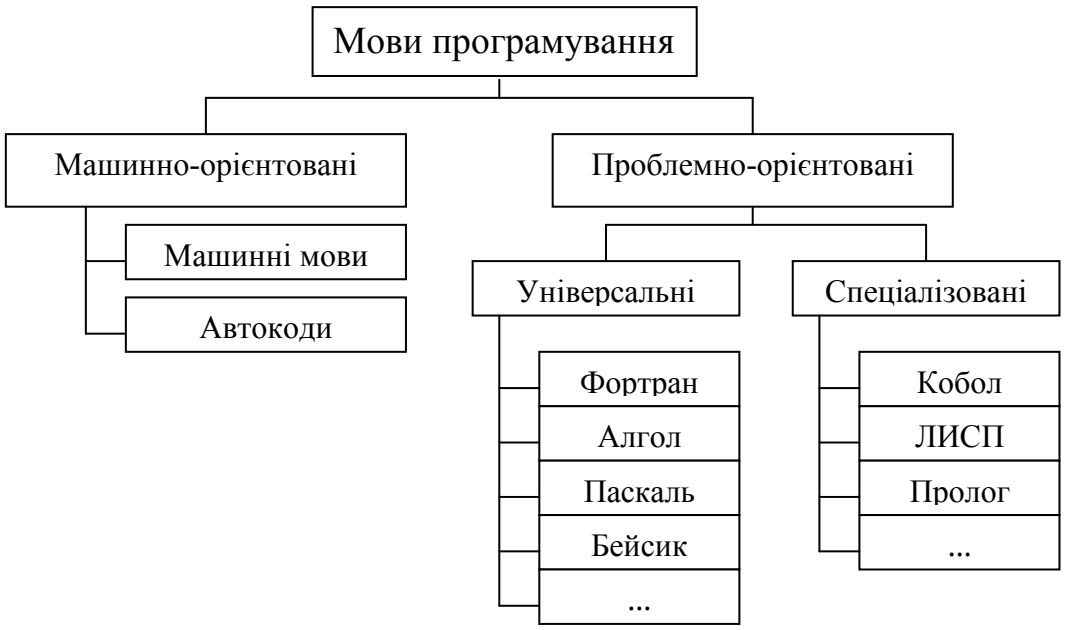


Рис. 1.1

Машинно-орієнтовані мови програмування (машинні мови та автокоди) дозволяють у повній мірі використовувати можливості ЕОМ. У теперішній час застосовуються достатньо рідко, в основному спеціалістами у галузі програмування – так званими системними програмістами.

Машинна мова – формальна мова для опису програм вирішення завдання, зміст і правила якої реалізуються апаратними засобами конкретної ЕОМ, інакше – система команд ЕОМ, в якій для виконання кожної операції визначені команди (складаються з кодів операцій та адрес (або кодів) операндів – змінних, над якими виконується операція).

Автокод – мова асемблера – машинно-залежна мова програмування, в якій коди операцій подані мнемонічними позначеннями, а адреси операндів – ідентифікаторами.

Проблемно-орієнтовані (алгоритмічні) мови - машинно-незалежні мови програмування, які дозволяють виконувати написані на них програми на будь-яких ЕОМ. Вони орієнтовані на користувачів ЕОМ, які не є фахівцями у галузі програмування. Спеціалізовані алгоритмічні мови близькі до професійної мови фахівців різних галузей науки та техніки, а універсальні – придатні для створення програм у будь-якій предметній галузі.

Взагалі, **система програмування** – сукупність мови програмування й відповідного їй мовного процесора.

Мовний процесор – програма або технічний засіб, що виконує трансляцію програми на машинну мову. Мовні процесори: асемблери, компілятори, інтерпретатори, конвертори.

Асемблер – транслятор програм з мови асемблера на машинну мову.

Компілятор – програма або технічний пристрій, який виконує компіляцію програми (лексичний аналіз, синтаксичний аналіз, генерацію машинного (об′єктного) коду, конструювання робочої програми), тобто перетворення всієї програми в машинні коди.

Інтерпретатор – програма або технічний пристрій, який виконує інтерпретацію програми (аналіз та виконання речень вихідної мови – кожного рядка програми один за одним).

Конвертор – транслятор з деякої мови на іншу мову того ж рівня.

До складу сучасних систем програмування входять також редактор текстів програм, бібліотека стандартних програм і програми для налагодження розроблюваних програм.

Редактор текстів програм - це програма, що забезпечує набір програми на певній мові програмування та її редагування.

Бібліотека стандартних програм містить готові, корисні для програміста програми (програми виконання математичних функцій, логічних функцій, операцій з текстовими даними та ін). Ці програми користувач-програміст може включати в текст розроблюваної програми при потребі.

Програми для налагоджування дозволяють довести розроблювану програму до експлуатаційної норми.

Для користувачів-початківців найбільш придатними для програмування є мови програмування високого рівня, тобто такі, синтаксис і граматика яких максимально наближені до природної людської мови.

Для IBM PC-сумісних комп’ютерів існують відповідні системи програмування, що працюють в середовищі MS DOS, Windows різних версій та інших і дозволяють створювати програми на популярних мовах програмування Фортран, Паскаль, Бейсик, Си, Си++.

Останнім часом широко застосовуються системи програмування на мові Java (Symantec Cafe, Microsoft J++ та ін.), які дозволяють створювати програми, що викликаються при перегляді Web-сторінок в глобальній мережі Internet.

Особливим класом систем програмування є системи візуального програмування (СВП). У них є засоби для створення користувацького інтерфейсу, опису процедур обробки даних, заготовки для виконання типових дій з обробки даних і т.п. Ці системи також, як правило, дають можливість працювати з різними СУБД – Oracle, Sybase, Microsoft SQL, Server та ін. Серед найбільш популярних систем такого типу можна вказати СВП Power Builder (розробка фірми Sybase), Delphi (Borland), Visual BASIC (Microsoft). Остання використовується фірмою Microsoft для створення прикладних програм (додатків).

---

2. ТЕХНОЛОГІЯ ВИРІШЕННЯ ЗАВДАНЬ ІЗ ВИКОРИСТАННЯМ СИСТЕМ ПРОГРАМУВАННЯ

Поняття **програмування** в широкому розумінні - це підготовка завдання до вирішення на комп'ютері; у вузькому розумінні – це запис алгоритму вирішення завдання мовою програмування у вигляді програми. Тому звичайно роботу по підготовці завдання до вирішення на комп'ютері називають алгоритмізацією, а програмуванням - безпосередню роботу по складанню програми.

Алгоритмізація - це робота по підготовці завдання до вирішення на комп'ютері; може виконуватись без комп'ютера. При вирішенні розрахункових задач алгоритмізація включає наступні етапи:

- отримання та вивчення завдання на розробку задачі;
- розробка технічного завдання на створення програмного виробу;
- постановка задачі;
- розробка алгоритму вирішення задачі.

Далі відпрацьовуються етапи:

- програмування;
- перевірка працездатності програмного виробу;
- розробка експлуатаційної документації;
- вирішення задачі на комп'ютері.

У загальному вигляді процес створення програми може бути поданий деякою послідовністю етапів (рис. 2.1).

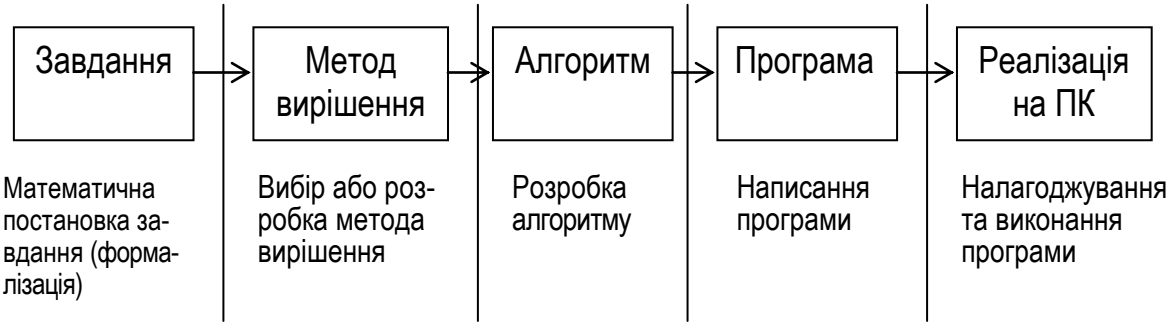


Рис. 2.1

Найбільш складним із перелічених є етап розробки алгоритму вирішення задачі відповідно до вибраного методу (методики).

Алгоритм - це скінчена послідовність вказівок або правил перетворення вхідних даних, що забезпечує отримання очікуваного результату.

Вказівки або правила однозначно визначають окремі кроки перетворень:

- операції введення-виведення;
- обчислювальні процедури;
- операції перевірки умов на передавання управління або зміну дій за алгоритмом;
- операції початку, зупинки, закінчення перетворень алгоритму;
- операції організації циклічних процедур та інші.

Кожний алгоритм має певні властивості та відноситься до певного виду та типу.

Обов'язковими властивості будь-якого алгоритму є такі:

- визначеність або детермінованість (вказані в алгоритмі дії мають виключати будь-яке довільне тлумачення);
- масовість (даний алгоритм можна застосувати для вирішення даної задачі для будь-якого набору вихідних даних, допустимого для цієї задачі);
- результативність (алгоритм має неодмінно приводити до отримання строго передбачуваного результату через скінчене число кроків перетворень).

Всі алгоритми за структурою зв'язків між діями в процесі вирішення задачі, поділяються на наступні типи:

- лінійні в яких кожна наступна операція виконується строго за попередньою від початку до кінця;
- розгалужені в котрих в залежності від певної умови може виконуватись одна з декількох можливих гілок алгоритму;
- циклічні які мають серії дій, повторюваних до тих пір, поки виконується певна поставлена умова;
- комбіновані, які мають признаки будь-яких (або усіх) перелічених вище типів.

Для подання алгоритмів використовують словесний (текстовий) або графічний способи. Перший полягає в традиційному записі детальних інструкцій-вказівок засобами звичайного письма.

Графічний спосіб полягає в зображенні алгоритму у вигляді графічної схеми (блок-схеми), яка складається зі стандартних символів, з'єднаних лініями (табл. 2.1), за визначеними правилами.

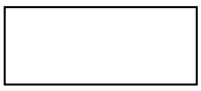
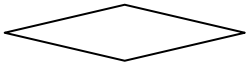
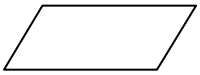
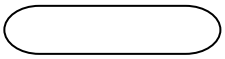
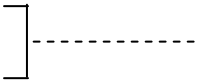
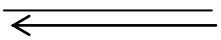
На рис. 2.2 наведена блок-схема алгоритму створення програми.

1. Блок-схема починається символом **Початок** і закінчується символом **Кінець.**

2. Символам у блок-схемі надаються порядкові номери, які проставляються зліва над зображенням символу. Номери символів вказують на послідовність виконання дій алгоритму і полегшують прочитання блок-схеми.

Таблиця 2.1

Стандартні символи блок-схем алгоритмів

Найменування символу	Графічне зображення	Функція символу
1. Процес		Виконання операції або групи операцій, завдяки яким змінюються значення, форма подання або розташування даних
2. Прийняття рішення		Вибір напрямку виконання алгоритму залежно від деяких змінних умов
3. Введення-виведення		Перетворення даних у форму, придатну для оброблення (введення) або відображення добутих результатів (виведення)
4. З'єднувач		Зазначення зв'язку між перерваними лініями потоку, що зв'язують символи
5. Початок-кінець (пуск-зупин)		Початок, кінець, переривання процесу оброблення даних
6. Коментар		Зв'язок між елементом схеми і поясненням
7. Лінія потоку		Зазначення послідовності зв'язків між символами

3. Всередині символів записуються виконувані операції або відповідна інформація, обумовлена зображенням блока.

4. На блок-схемі шляхи передавання управління між символами (лінії потоку) виконуються тільки вертикальними або горизонтальними лініями. Напрямки зверху-вниз і зліва-направо є основними і стрілками можуть не позначатися. В інших випадках напрямок ліній потоку вказується стрілками.

5. Для пояснень до окремих символів використовуються коментарі, які пояснюють стан об'єктів алгоритму в його окремих точках, пояснюють зв'язки між окремими його ланками.

6. Для зв'язку окремих фрагментів блок-схеми в одне ціле слугує з'єднувач, в якому вказують унікальне позначення, однакове для відповідних з'єднувачів.

7. В блок-схемі не може бути «обірваних» гілок.

Ступінь деталізації блок-схеми алгоритму узгоджують виконавець і замовник. Слід зазначити, що чим детальніше й коректніше розроблена блок-схема алгоритму, чим менше в ній малозрозумілих блоків і дій, тим швидше й з меншою кількістю помилок буде написана програма.

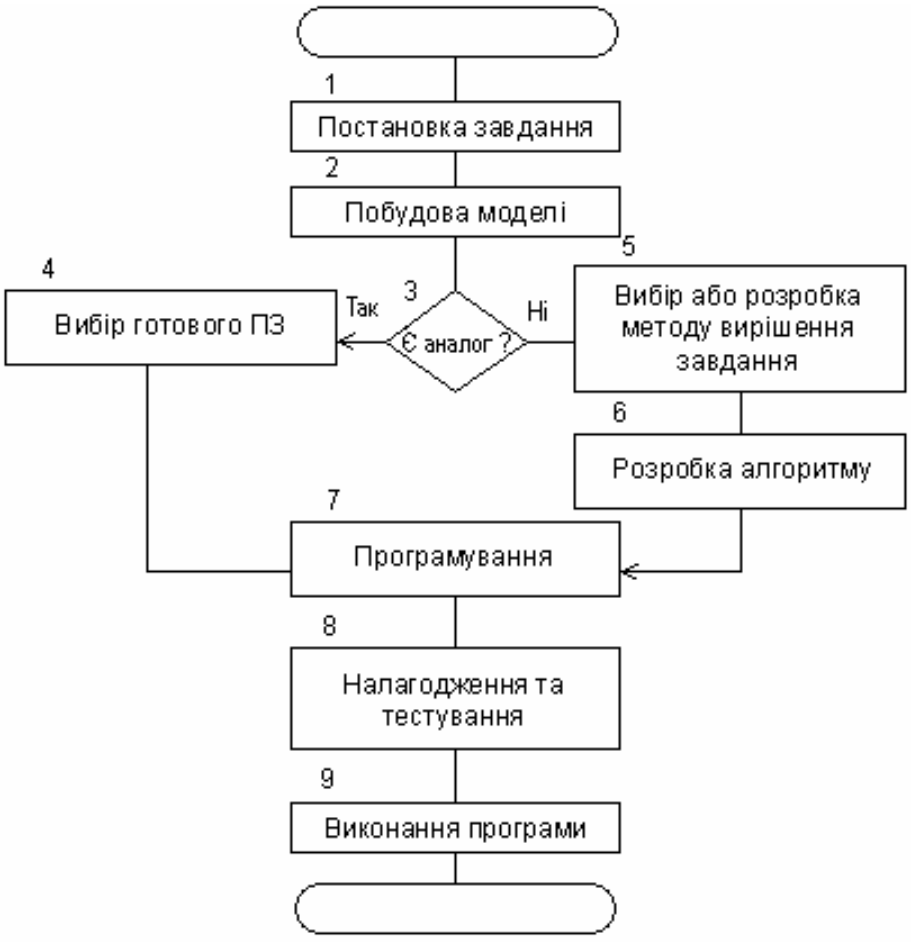


Рис. 2.2

Програма – запис алгоритму на мові програмування у вигляді інструкцій, які сприймаються і виконуються комп'ютером. Мова програмування має свій алфавіт та правила побудови мовних конструкцій. Основними компонентами мови програмування є оператори і команди-директиви.

Оператор – мовна конструкція, котра визначає певну дію (перетворення, операцію) в алгоритмі. З операторів складаються програми.

Команда-директива – мовна конструкція, котра визначає певну дію з програмою, або її частиною. Команди-директиви виступають інструментом користувача-програміста при роботі в системі програмування як при складанні програм, так і при їх експлуатації.

Приклад 1. Розробити програму розрахунків за формулою

$$y = \frac{A+B}{\sqrt{x-1}}, \qquad (2.1)$$

де A та B – константи, а x – змінна.

Відповідно з розглянутою вище технологією після вивчення завдання розробник сумісно з замовником розробляють ***технічне завдання*** обсягом 2-4 аркуші, яке може містити такі розділи:

- підстава для розробки, де вказується документ (чи документи), на підставі якого ведеться розробка; особа, що затвердила цей документ, дата затвердження ; найменування та (або) умовне позначення теми розробки;

- призначення розробки, де розкриваються функціональне й експлуатаційне призначення програми (програмного виробу);

- вимоги до програми (програмного виробу0 з розкриттям вимог до функціональних характеристик, умови експлуатації, вимоги до складу й параметрів використовуваних технічних засобів, вимоги до інформаційної та програмної сумісності (якщо розроблюваний програмний засіб є частиною комплексного проекту, або має бути реалізований різними програмними середовищами);

- техніко-економічні показники;

- вимоги до програмної документації;

- стадії та етапи розробки з термінами їх виконання.

Постановка задачі передбачає її аналіз та вибір методів вирішення. Результати роботи на цьому етапі оформляються у вигляді офіційного документу, який називається ***описом задачі*** з такими розділами.

1. Загальний опис задачі:

- детальний запис умови задачі – повторює наведену вище умову;

- характеристика розрахунку – розрахунок у межах арифметичних операцій та обчислення квадратного кореня, складність при комп'ютерному вирішенні - захист від недопустимих операцій (корінь з від'ємного числа та ділення на нуль);

- мета вирішення задачі на комп′ютері – автоматизація розрахунків.

2. Опис вихідних даних (перелік вихідних даних за їх типами):

- символьні (текстові) – відсутні;

- числові; постійні – A, B – будь-які раціональні числа без обмежень; змінна x – раціональне число; $x > 1$;

- розмірність даних – довільна, допустима для даного типу комп'ютерів.

3. Опис результатів вирішення задачі:

- кінцевий результат – значення y;

- форма подання (виведення) результатів – на екран виводяться у вигляді для x = x y = y;

- результати без збереження в пам'яті переписуються з екрану.

4. Опис методики вирішення задачі:

- математична модель задачі задана розрахунковою формулою, обмеження на значення вихідних даних – $x < 1$;

- при введенні недопустимих вихідних даних на екран має виводитись повідомлення "введене значення x НЕДОПУСТИМЕ, введіть $x > 1$";

- метод вирішення задачі на комп’ютері – прямий розрахунок.

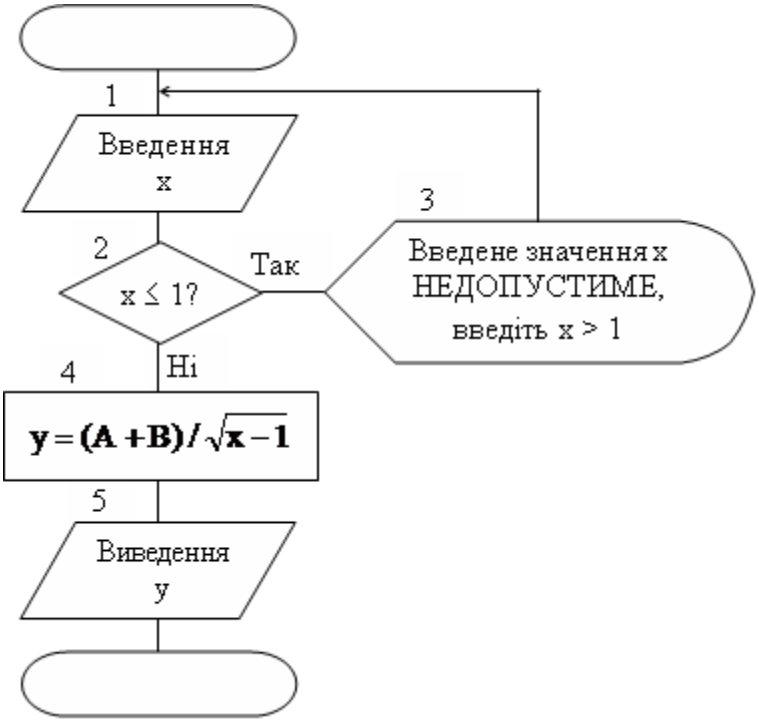


Рис. 2.3

Опис задачі є основою для її алгоритмізації. Блок-схема алгоритму розрахунку за формулою (2.1) наведена на рис. 2.3.

За розробленим алгоритмом розробляють програму у вибраній або заданій системі програмування. Для користувачів-початківців найбільш придатними для програмування є мови програмування високого рівня, тобто такі, синтаксис і граматика яких максимально наближені до природної людської мови. Система програмування Visual BASIC використовує мову BASIC – одну з найпоширеніших мов програмування високого рівня. Слово Visual означає, що у цій системі реалізовано візуальний стиль програмування, за допомогою якого програми не пишуть, а проектують.

---

3. ОСНОВИ ПРОГРАМУВАННЯ В СИСТЕМІ VISUAL BASIC

3.1. Коротка характеристика системи програмування Visual BASIC

Перед тим як починати набирати перший рядок програми, створюють інтерфейс, тобто зовнішній вигляд робочого середовища, з яким буде працювати користувач.

Visual BASIC (VB) належить до групи об’єктно-орієнтованих мов програмування, що обумовлює низку характерних особливостей процесу програмування.

Програма, створена за допомогою VB, називається проектом. Існує три основних компоненти проекту Visual BASIC: це власне проект, форма й елементи управління.

Проект (Project) – це те ж саме, що і комп'ютерна **програма (program)** або **додаток (application)**.

Форма (Form) – це компонент VB у вигляді вікна, призначений для того, щоб користувач міг взаємодіяти з комп’ютером (тут доречне таке поняття як інтерфейс користувача).

Елементи управління (Controls) – це елементи VB, що мають визначені графічні й функціональні властивості. Їх поміщають на форму як елементи, на які користувач може впливати (командні кнопки, текстові вікна, написи, смуги прокручування тощо). Сама форма теж є елементом управління. Термін елементи управління є синонімом терміна **objects (об’єкти).** Об’єкти створюються на екранній формі за допомогою засобів управління. Їх можна переміщати, настроювати. змінювати їх розміри.

Кожний об’єкт VB характеризується подіями, методами та властивостями.

Подія – це характеристика об’єкта, що описує зовнішній вплив, на який реагує об’єкт при застосуванні додатка. Подіями можуть бути, наприклад, завантаження форми, натискання на будь-яку клавішу клавіатури, клацання мишею на елементі управління, рух миші, такт системного годинника і т. ін.

Події ініціюються:

- діями користувача;
- повідомленнями, які находять від системних або інших додатків;
- додатком, що використовується.

Код додатку VB може поділятися на більш дрібні блоки, які називаються **процедурами**. Для створення процедури обробки події потрібно зі списку об’єктів вікна редактора коду вибрати елемент, з яким пов’язана процедура, що розробляється, а зі списку процедур цього вікна – елемент з ім’ям події для вибраного об’єкта.

З кожною подією пов’язана процедура її оброблення – фрагмент **програмного коду**, який виконується після здійснення певної події.

Таким чином, VB підтримує подійну модель на відміну від процедурної моделі, де виконання програми здійснюється у суворій послідовності з першого рядка коду і далі за визначеним наперед шляхом, зумовлюючи відповідні процедури.

У подійному додатку (event-driverapplication – додатку, який управляється подіями) виконання коду не відбувається за заздалегідь установленим шляхом: різні частини коду виконуються залежно від події, що відбу-

лася (event). Кожний елемент управління, виведений на екранну форму, підтримує одну або декілька подій. Якщо записано процедуру для події цього елемента, то команди програмного коду будуть виконуватись автоматично, як тільки ця подія станеться.

Крім процедур оброблення подій існує ще один тип процедур – **процедури загального призначення**. Ці процедури починають виконуватись не у відповідь на якусь подію, а після їх виклику з будь-якого місця програми.

Метод – це подія, яка може здійснюватись над об’єктом. Метод – це також процедура, що входить до складу VB. Приклади методів: **Cls** - очищення екранної форми; **Form2. Show** – показати форму, **Form2. Hide** – закрити форму.

Важливим поняттям у проектах Visual BASIC є **властивість (property)**. Кожна характерна ознака елемента управління (і самої форми) визначається їхніми властивостями. Наприклад, властивостями є ім'я, заголовок, розміри, кольори, положення на формі елемента управління тощо.

3.2. Методика створення програм в системі Visual BASIC

У Visual BASIC проект складається з форми, що містить кілька елементів управління. Він, як правило, зберігається в двох вихідних файлах: у файлі, що має розширення **vbp** (іноді **vbw**) зберігається безпосередньо проект; форма зберігається у файлі, що має розширення **frm** (іноді **frx**).

У проекті VB ніщо не відбувається, поки не відбудеться подія. Коли подія відбувається, проект знаходить серію інструкцій, записаних за допомогою операторів мови BASIC, пов'язаних з цією подією, тобто **процедуру обробки події.** Запис процедур обробки подій і є власне кажучи дійсним комп'ютерним програмуванням.

Отже, розробка додатка на VB складається з наступних етапів:

1. Формалізація та алгоритмізація завдання (структура даних, алгоритм тощо).

2. Проектування інтерфейсу, тобто розміщення (або малювання) на формі потрібних елементів управління, кнопок, списків і т.п.

3. Установка (призначення) властивостей для елементів управління.

4. Написання програми (процедури, програмного коду) обробки подій, що пов’язує поміщені на форму елементи управління.

5. Налагодження програми.

6. Остаточна компіляція і, якщо це необхідно, створення дистрибутива (тобто файлу setup.exe).

3.3. Елементи програмування в системі Visual BASIC

У системі Visual BASIC є близько 200 вбудованих операторів і функцій. Кожні оператор і функція мають чітку структуру (синтаксис), тобто правила граматики, пунктуації та орфографії, що використовуються для їх опису і розпізнавання. Для опису операторів застосовуються різні символи (букви латиниці й кирилиці, цифри, розділові та спеціальні знаки) та ключові слова.

Ключове (зарезервоване) слово призначене винятково для Visual BASIC; програміст не може застосовувати ключові слова у своїх власних цілях. Типи даних; функції, а також такі слова як **As, Const, Dim, Do, Else, End, Exit, For, If, Loop, Next, New, Print, Private, Public, ReDim, Rem, Static, Step, Sub, To, Then, Until, While - Dim**, **As**, **New**, **ReDim**, **If**, **Them**, **Else**, **Loop**, **While**, **End** – ключові слова. За замовчуванням для виділення ключових слів у вікні редагування коду використовується шрифт синього кольору.

Синтаксис – певні правила запису програм, яких треба дотримуватися, щоб транслятор системи VB «розумів» програмний код. У кожному рядку коду є оператор, який може мати додаткові параметри. Оператори можуть розташовуватися послідовно в одному рядку, тоді вони розділяються двокрапкою. Можна розділяти логічний рядок й оператор на кілька фізичних рядків. **Роздільником рядків** служить пропуск, який іде за символом підкреслення (_).

Коментарі у VB записують, починаючи з верхньої коми (‘) або команди **Rem**, які розміщують в окремому рядку.

3.3.1. Операції та функції Visual BASIC

Основні математичні операції VB наведені в табл. 3.1; вбудовані функції наведені в табл. 3.2.

Таблиця 3.1

Операція	Символ оператора
Додавання	+
Віднімання	–
Множення	*
Ділення	/
Ділення без остачі	\
Ділення за модулем (обчислення остачі)	**mod**
Зведення до міри	^

Таблиця 3.2

Функція	Дія функції
Abs(x)	Повертає абсолютне значення **x**
Atn(x)	Повертає арктангенс **x**. Кут виражений в радіанах
Cos(x)	Повертає косинус кута **x**. Кут виражений в радіанах
Exp(x)	Повертає константу **e** в степені **x**
Log(x)	Повертає натуральний логарифм аргументу **x**
Rnd(x)	Генерує випадкові числа в діапазоні **0...1**
Sgn(x)	Повертає **1**, якщо x < 0; **0**, якщо x = 0; **– 1**, якщо x > 0
Sin(x)	Повертає синус кута **x**. Кут виражений в радіанах
Sqr(x)	Повертає квадратний корінь **x**
Str(x)	Перетворює числове значення на рядок
Tan(x)	Повертає тангенс кута **x**. Кут виражений в радіанах
Val(x)	Перетворює рядок на числове значення

3.3.2. Оголошення та опис змінних

Змінна – це іменована область пам'яті, призначена для збереження даних, яка в ході виконання програми набуває різних значень. Імена змінних у Visual BASIC створюються за певними правилами:

- першим символом імені має бути літера;
- інші символи – літери та цифри;
- можна використовувати символ підкреслення, але не можна застосовувати крапку;
- кількість символів не може перевищувати 255;
- ім'я не повинно бути ключовим словом VB.

Значення змінної – це дані, які зберігаються й обробляються системою Visual BASIC. Спосіб збереження та оброблення даних залежить від того, до якого типу вони належать.

Типом даних називається спосіб збереження і подання даних у комп'ютері, який задає певний формат або розмір вмісту змінної. Залежно від вмісту розрізняють змінні різних типів (табл. 3.3).

У мові Visual BASIC існують три способи оголошення типу змінної.

1. Явне, за допомогою оператора **Dim,** наприклад:

Dim A **As Integer**;

Dim B **As Currency**;

Dim C **As String** або

Dim C **As String***30 (у цьому випадку довжина змінної обмежена тридцятьма символами).

2. Неявне, коли тип змінної визначається за замовченням як **Variant** або її вмістом. Наприклад, якщо вміст змінної – число 5, то її тип **Integer ,** якщо 1.2, то тип **Double** тощо.

3. Оголошення за допомогою ідентифікаторів типів.

Таблиця 3.3

Тип даних	Опис і діапазон даних
Boolean	Набуває тільки одного з двох значень: **True** або **False**
Byte	Позитивні числа в діапазоні **0 ... 255**
Currency	Грошові одиниці від – 9223372036854775808 до 9223372036854775807. Чотири дробових десяткових розряди забезпечують правильне округлення
Date	Значення дати (від 1 січня 100 р. до 31 грудня 9999 р.) і часу
Double	Дані з подвійною точністю від – 179769313486232 D+308 до 179769313486232 D+308
Integer	Числа без десяткової крапки від – 32768 до 32767
Long	Цілі числа від – 2147483648 до 2147483647
Objekt	Містить посилання на об′єкти (форми та елементи управління)
Single	Числа від – 3402823 E+38 до 3402823 E+38
String	Можуть бути як алфавітними, так і числовими; можуть містити символи ^, %, @
Variant	Дані будь-якого типу, що використовуються для елементів та інших додатків, для яких тип даних невідомий

Для змінних визначають області видимості:

- локальна змінна визначається всередині процедури або функції, наприклад:

Static A **As Integer**;

Dim B **As Integer**;

- змінна контейнерної області визначається у секції і доступна всім процедурам тільки всередені контейнера – модуля форми, наприклад

Private A **As Integer**;

- змінна глобальної області визначається у секції стандартного модуля й доступна в усіх модулях і процедурах проекту:

Public A **As Single**

Const = 3.1415926535897932 або

Const Pi **As Single** = 3.1415926535897932.

3.4. Робоче вікно Visual BASIC

Робоче вікно Visual BASIC відкривають або за створеним раніше додатком, або для нового проекту (або форми).

Щоб створити новий проект (сукупність файлів, що входять у додаток і зберігають інформацію про його компоненти), треба запустити програму VB (**Пуск/ Программы/ Microsoft Visual BASIC**). У діалоговому вікні **New Project (Новый проект)** за замовчуванням буде обрана вкладка

New (Новый) і виділений значок **Standard EXE** (рис. 3.1). При натисканні кнопки **Открыть**. відкриється діалогове вікно **New Project**. Курсор встановлюють на значку **Standard EXE** й натискають кнопку **ОК** або двічі клацають на цьому значку.

Можна також виконати такі дії:

- натиснути кнопку **Add Standard EXE Project** на панелі інструментів;
- натиснути кнопку **Add Form** на панелі інструментів;
- у діалоговому вікні **Add Form**, що відкрилося, встановлюють курсор на значок **Form** і натискають кнопку **Открыть** або двічі клацають на значку.

Рис. 3.1

У результаті відкривається вікно **Project1** (див. рис. 3.2), що містить нову форму, з якою можна починати працювати: змінювати встановлені за замовчуванням властивості, поміщати в неї елементи управління, використовуючи для цього **панель елементів управління**, що є основним робочим інструментом при розробці форм додатка. У рядку заголовка вікна програми VB з'являється слово **design**, що вказує, що програма перебуває в режимі розробки додатку.

У процесі роботи проект необхідно періодично зберігати. При збереженні проекту необхідно пам'ятати, що проект складається з компонентів, таких як форми, модулі коду, класи, елементи управління й т.д. Для полегшення роботи із проектом доцільно створити окрему папку, у якій потім можна зберігати всі файли проекту.

Для збереження проекту виконують такі дії.

1. У меню **File (Файл)** вибирають команду **Save Project (Сохранить проект)** або натискають кнопку **Save Project** на панелі інструментів.

2. У діалоговому вікні **Save File As список**, що відкриється, тип файлу містить значення **Form Files**, указуючи, що зберігається форма, яка входить у додаток. У списку **Папка**, що розкривається, виберають папку, в якій буде збережена форма, потім у поле **Имя файла** вводять ім'я форми й натискають кнопку **Сохранить**.

3. Так як ніяких компонентів, окрім форми, новий додаток не містить, з'являється діалогове вікно **Save Project As (Сохранить проект как)** для збереження самого проекту. За замовчуванням у списку **Папка** обрана папка, у якій зберегли форму. Уводять найменування проекту і натискають кнопку **Сохранить**.

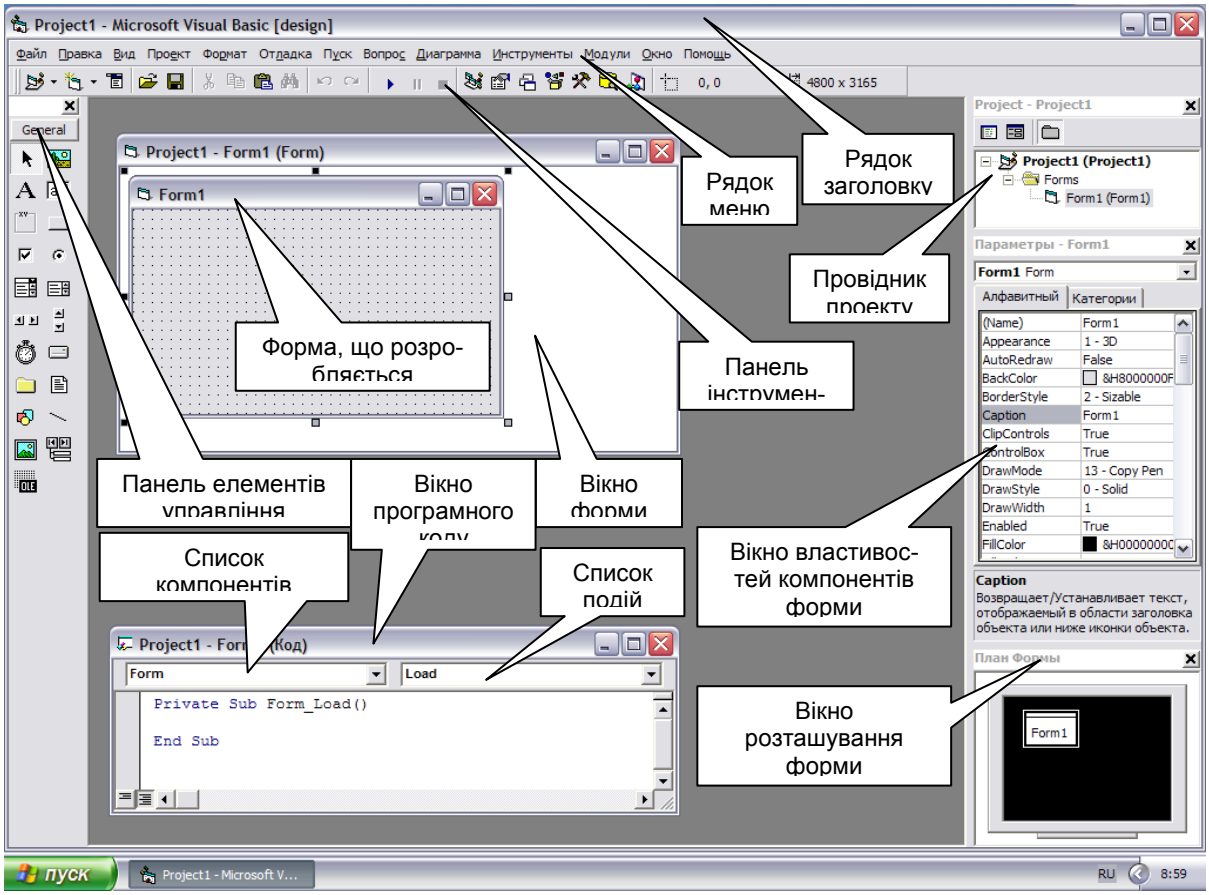


Рис. 3.2

При наступному збереженні проекту VB за замовчуванням зберігає файли під їхніми іменами. Якщо ж у проекті з моменту останнього збереження були змінені компоненти або додані нові, відкриється діалогове вікно **Save File As** для присвоєння їм імен.

Щоб змінити раніше створений проект, його треба відкрити. У меню **File (Файл)** вибирають команду **Open Project (Открыть проект)**, потім у діалоговому вікні **Open Project**, що відкрилося**,** знаходять на диску необхідний файл і натискають кнопку **Открыть**.

Для того щоб одержати доступ до компонентів, що входять у проект, виконують одну з таких дій:

- у меню **View (Вид)** вибирають команду **Project Explorer (Проводник проекта)**;

- натискають кнопку **Project Explorer** на стандартній панелі інструментів.

Відкриється **вікно провідника проекту** (рис. 3.3), що містить список усіх його компонентів. У верхній частині вікна розміщене ім'я проекту, задане при його збереженні. У групі **Forms** розташовуються компоненти-форми. Для відкриття будь-якої з форм треба двічі клацнути на її імені.

Для розміщення у формі об'єктів-елементів управління використовується **панель елементів управління** (див. рис. 3.2). Якщо її немає на екрані, виконують одну з таких дій:

- вибирають із меню **View (Вид)** команду **Toolbox (Панель элементов управления)**;

- натискають кнопку **Toolbox** на стандартній панелі інструментів (при «зависанні» над кнопкою панелі «спливає» її назва).

Вправа 1. Розміщення елемента управління у визначеному місці.

Є два способи розміщення елементів управління на формі.

1. Подвійне клацання на обраному елементі управління блоку інструментів. Елемент управління буде створений із заданими за замовчуванням (default) розмірами і поміщений у середину форми.

Знайдіть у блоці інструментів елемент управління під назвою **Command button (Командная кнопка).** Клацніть подвійним клацанням по цьому елементі управління і він з'явиться посередині форми (рис. 3.4).

2. Клацніть один раз по обраному елементу управління блоку інструментів. Тепер перемістите курсор миші на форму. Помітьте, що курсор змінив свій вид на (+). Встановіть хрестик у тому місці, де буде знаходитися лівий верхній кут вашого елемента управління. Натисніть ліву кнопку миші і, утримуючи її, перетягніть курсор до того місця, де буде знаходитися правий нижній кут цього елемента управління. При цьому буде видний прямокутний контур. Коли розміри контуру досягнуть необхідної величини, відпустіть кнопку миші. На формі з'явиться вибраний елемент управління.

Всі елементи управління мають характерні для них **властивості (параметри)**, які задаються у **вікні параметрів** (рис. 3.5).

Вправа 2. Переміщення елемента управління та зміна його розмірів.

Як тільки елемент управління поміщений на форму (незалежно від способу розміщення), його можна перемістити або змінити його розміри. Щоб **перемістити (move)** елемент управління, виберіть його, тобто клацніть лівою кнопкою миші на ньому (з'являться маніпулятори розміру). Перетягніть його на нове місце і відпустіть кнопку миші.

Для зміни **розміру (resize)** також виберіть елемент управління, а потім підведіть курсор до одного з маніпуляторів, щоб вид курсору змінився на подвійну стрілку. Після цього, натиснувши клавішу миші, ви можете перетягнути обрану сторону або кут елемента управління на необхідну відстань.

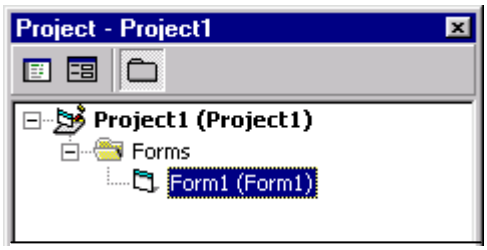


Рис. 3.3

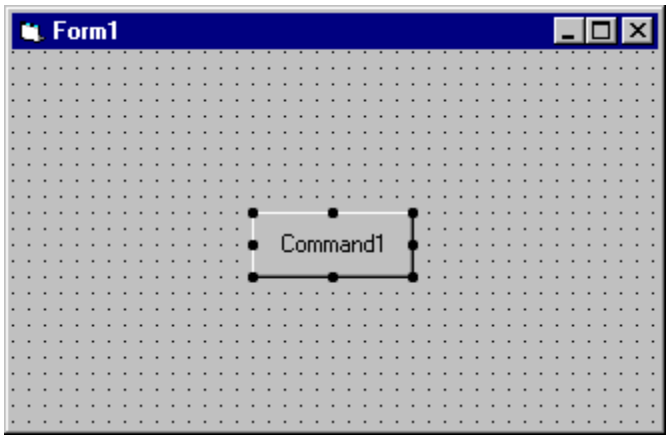


Рис. 3.4

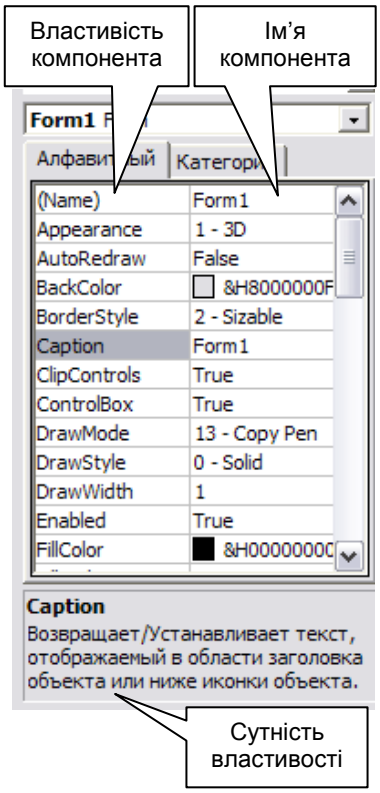


Рис. 3.5

Зверніть увагу на маніпулятори розмірів навколо елементу управління – чорні квадратики. Це означає, що елемент управління знаходиться в **активному (active)** стані. Клацніть на вільному місці форми і маніпулятори зникнуть, але з'являться навколо форми, – тепер форма знаходиться в активному стані. Клацніть на командній кнопці знову, щоб зробити її активною. Перемістіть командну кнопку в різні частини форми. Тепер змініть її розмір, наприклад зробіть „велику" або „маленьку", „високу" або „широку" кнопку. Ще раз перемістіть кнопку в різні місця форми.

Попрактикуйтесь якийсь час, розміщуючи елементи управління на формі. Використовуйте інші елементи управління, наприклад **labels (написи)**, **text boxes (текстові вікна), option buttons (кнопки вибору варіантів),** і **check boxes (флагові кнопки)**. Переміщуйте їх на формі, змінюйте їхні розміри. Спробуйте розмістити ваші елементи управління по групах, на одній лінії, щоб вони не виглядали безформним накопиченням.

Вправа 3. Видалення елементів управління з форми.

Клацніть на формі по елементу управління, який треба видалити. Елемент управління стане активним. Натисніть клавішу **Del** (delete – вида-

лити) на клавіатурі. Обраний елемент управління буде вилучений. Перед видаленням елемента управління переконайтеся, що ви видаляєте саме той елемент управління, який необхідно видалити.

Вправа 4. Створення додатку ПРИВІТ, який при натисканні командної кнопки друкує на полі форми фразу "**Привіт!**"

Встановіть властивість **Caption** для елементів управління **Form1** і **Command1** Привіт.

Подвійним клацання на кнопці **Command1** викличте **Программный код** (рис. 3.6) та введіть у шаблон коду обробки події **Click** оператор, виділений сірим кольором:

```
Private Sub Command_Click ()
Print "Привіт!»"
End Sub
```

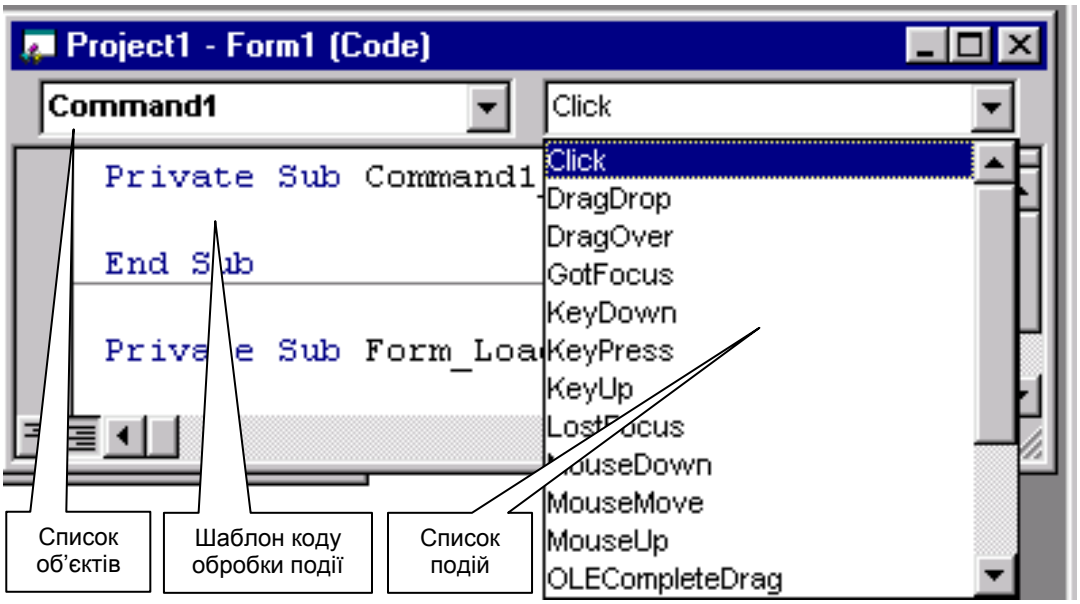


Рис. 3.6

У процесі створення програмного коду (процедури обробки) ні в якому разі не можна чіпати перший (**Private Sub …**) і останній (**End Sub)** оператори шаблону!

Запустіть додаток (рис. 3.7) на виконання, вибравши команду **Start** із меню **Run** або натиснувши кнопку **Start** на панелі інструментів, або клавішу <F5>. При натисканні (клацанні) мишею командної кнопки результат виконання додатку буде надрукований на полі форми (рис. 3.8).

Вправа 5. Створити додаток ГОДИННИК.

Встановіть для **Form1** властивість **Caption** Годинник. Встановіть на формі елементи управління **Label1** та **Timer1** (див. рис. 3.9).

Для **Label1** встановіть властивості **BackColor (фон)** – чорний, **ForeColor (текст)** – зелений, **Font (размер шрифта)** – 24.

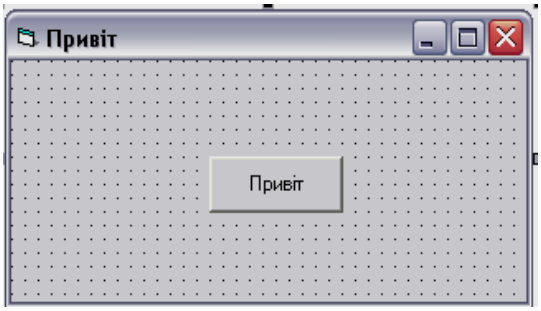


Рис. 3.7

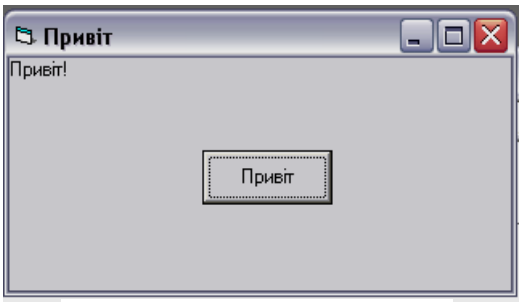


Рис. 3.8

Для **Timer1** встановить властивість **Interval** = 1000 і викличте **Программный код.** Введіть у шаблон для процедури оператор, виділений сірим кольором:

```
Private Sub Timer1_Timer()
Label1=Time
End Sub
```

Запустіть додаток на виконання. Елемент **Timer1** сховається, а на полі елемента **Label1** форми ви побачите поточний час, що обновляється кожну секунду (див. рис. 3.10).

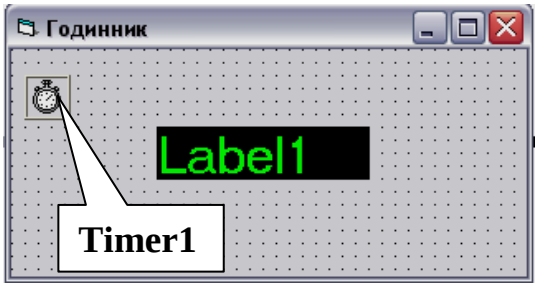


Рис. 3.9

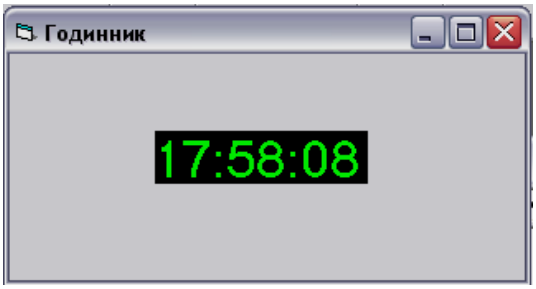


Рис. 3.10

3.5. Програмування лінійних алгоритмів

Лінійні алгоритми передбачають виконання тільки прямих розрахунків.

Вправа 6. Створити додаток РОЗРАХУНОК, який виконує операцію множення цілих чисел та виводить результат на поле форми.

На формі з кнопкою **Command1** викликати **Программный код** та ввести виділені сірим кольором оператори у шаблон, варіанти яких наведені у табл. 3.4.

Таблиця 3.4

Private Sub Command1_Click () c = 3*7 Print “c=”, c **End Sub**	**Private Sub Command1_Click ()** Print “c=”, c = 3*7 **End Sub**

Запустіть додаток на виконання. Після клацання мишею на командній кнопці ви побачите на формі надрукований результат c = 21.

Вправа 7. Створити додаток РОЗРАХУНОК, який виконує операцію множення чисел **a** та **b**, що вводяться, та виводить результат на поле форми.

Для введення даних звичайно використовується стандартна функція **Val(n)**, яка перетворює рядок на числовий вираз. Протилежна за призначенням стандартна функція **Str(a)** перетворює числовий вираз на рядок.

На формі встановіть елементи управління **Text1, Text2, Command1.** Для елемента **Command1** викличте **Программный код** та введіть оператори, виділені сірим кольором, у шаблон для процедури:

Private Sub Command1_Click ()
a=Val(Text1. Text)
b=Val(Text2. Text)
c = a*b
Print “c=”, c
End Sub

Запустіть додаток на виконання. Введіть будь-які числа в текстові бокси й натисніть командну кнопку. На формі буде надрукований результат.

Вправа 8. Для умов вправи 7 передбачити введення даних та виведення результатів у текстових вікнах. Поля для введення даних попередньо очищуються.

Встановіть для **Form1** властивість **Caption** Розрахунок. Встановіть на формі елементи управління елементи управління **Text1, Text2, Text3**, **Command1, Command2.**

Для елементів **Command1, 2** встановіть відповідно властивості **Caption** Очистити та Обчислити.

Викличте **Программный код** елемента **Command1** та введіть у шаблон для процедури виділені сірим кольором оператори:

Private Sub Command1_Click ()
Text1. Text = “”
Text2. Text = “”

Text3. Text = “”
End Sub

Для **Command2** викличте **Программный код** та введіть у шаблон для процедури виділені сірим кольором оператори:

Private Sub Command2_Click ()
a=Val(Text1. Text)
b=Val(Text2. Text)
Text3. Text = a*b
End Sub

Запустіть додаток на виконання. Натисніть спочатку командну кнопку **Очистити** й (елементи **Text1, 2, 3** приймуть пусте значення**)**, введіть будь-які числа в текстові бокси **Text1, 2** та натисніть кнопку **Обчислити**. У текстовому боксі **Text3** з′явиться результат.

Перевірте працездатність додатку при зміні арифметичної операції в операторі Text3. Text = a*b програмного коду **Command2**, зокрема, при діленні на 0.

Вправа 9. Створити додаток КАЛЬКУЛЯТОР, який виконує арифметичні операції, розташовані у контейнері **(Frame)** над числами **a** та **b**, що вводяться, та виводить результат у вікні з іменем **c=a<операція>b=**.

На формі попередньої вправи додатково встановіть елементи управління **Label1, Label2, Label3** із змістом (a=), (b=), (c=a<операція>b=) відповідно.

Для змінних використайте явне оголошення за допомогою оператора опису змінної типу **Dim** Ім’я_змінної [**As** Тип_даних]:

Private Sub Form_Load()
Dim a **As Single**
Dim b **As Single**
Dim c **As Single**
End Sub

Зовнішній вигляд форми додатка, розподіл та імена елементів управління, а також програмний код наведені відповідно на рис. 3.11, табл. 3.5 і табл. 3.6.

Запустіть додаток на виконання. Натисніть кнопку Очистити, введіть будь-які числові дані у текстові бокси **Text1, 2** та натисніть кнопку будь-якої операції. У текстовому боксі **Text3** з′явиться результат. Перевірте працездатність додатку для різних операцій та різних наборів даних, у тому числі ділення на 0.

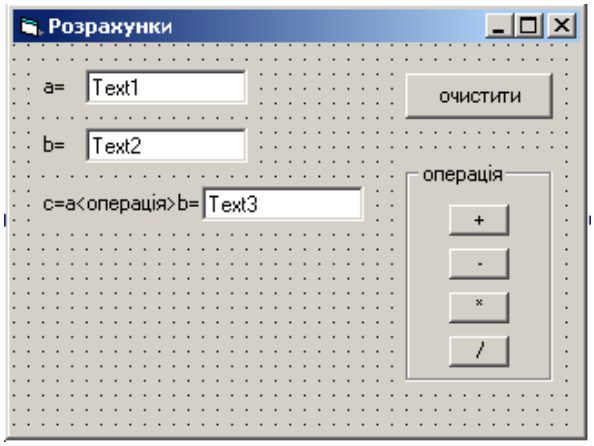


Рис. 3.11

Таблиця 3.6

```
Private Sub Command1_Click()
a = Val(Text1.Text)
b = Val(Text2.Text)
Text3.Text = a + b
End Sub
```

```
Private Sub Command2_Click()
a = Val(Text1.Text)
b = Val(Text2.Text)
Text3.Text = a - b
End Sub
```

```
Private Sub Command3_Click()
a = Val(Text1.Text)
b = Val(Text2.Text)
Text3.Text = a * b
End Sub
```

```
Private Sub Command4_Click()
a = Val(Text1.Text)
b = Val(Text2.Text)
Text3.Text = a / b
End Sub
```

```
Private Sub Command5_Click()
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
End Sub
```

Таблиця 3.5

Елемент управління	Caption
Command1	+
Command2	-
Command3	*
Command4	/
Command5	очистити
Form1	Розрахунки
Frame1	операція
Label1	a=
Label2	b=
Label3	c=a<операція>b=
Text1	Text1
Text2	Text2
Text3	Text3

3.6. Програмування розгалужених алгоритмів

Обчислювальний процес, що розгалужується, передбачає вибір одного з кількох можливих варіантів обчислень залежно від результату перевірки умови, яка має вигляд виразу логічного типу. Внаслідок перевірки цієї умови можливими є два варіанти:

True (Так) - умова виконується;

False (Ні) - умова не виконується.

Вирішення завдань такого типу вимагає застосування спеціалізованих умовних операторів.

У VB існують дві форми синтаксису умовного оператора (напівжирним шрифтом виділяються ключові слова, квадратними дужками – конструкції, які можуть бути відсутні).

Однорядкова форма:

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

End If

… .

Логіка роботи (семантика) такого умовного оператора.

1. Ключове слово **Else** відсутнє.

Якщо умовний вираз має істинне значення, то робиться перехід до виконання операторів після слова **Then**; виконання оператора припиняється і починають виконуватися записані за ним оператори програми.

2. Умовний оператор містить ключове слово **Else.**

Якщо умовний вираз має істинне значення, то робиться перехід до виконання операторів після слова **Then**. Якщо ж умовний вираз має хибне значення, то робиться перехід до виконання операторів після слова **Else.** Виконання умовного оператора припиняється і починають виконуватися записані за ним оператори програми.

Багаторядкова форма:

If умова **Then**

послідовність операторів 1

Else

послідовність операторів 2

End If

… .

Для перевірки відразу декількох умов існує умовний оператор, який крім головної умови розглядає ще й додаткові:

If головна умова **Then**

послідовність операторів А

ElseIf умова 1 **Then**

послідовність операторів обробки умови 1

ElseIf умова 2 **Then**
послідовність операторів обробки умови 2
...
ElseIf умова N **Then**
послідовність операторів обробки умови N
[**Else** послідовність операторів B]
End If
...

Семантика оператора: спочатку обчислюється значення головної умови в операторі **If**. Якщо воно дорівнює **True**, то буде виконаний оператор (оператори, якщо їх декілька) після оператора **Then**. Далі програма переходить до виконання оператора, записаного безпосередньо після оператора **End If.**

Якщо обидві умови мають істинні значення, то виконується послідовність операторів A. Якщо ж перша умова має хибне значення, то програма переходить до виконання оператора, записаного за оператором **End If.** Якщо ж перша умова є істинною, а друга – хибною, то виконується послідовність операторів B, а далі програма переходить до виконання оператора, записаного безпосередньо після оператора **End If.**

Приклад 2. Маємо:

X = Int (Rnd*10) + 1
If X = 7 **Then** Label1.Caption = “=7”
...

У цьому прикладі оператор 1 здійснює присвоювання змінній X значення випадкового числа в діапазоні від 1 до 10.

Оператор 2 реалізує умовний перехід. Якщо X має значення 7, то властивості **Caption** об’єкта **Label1** присвоюється значення =7.

Якщо ж умовний вираз має хибне значення, тобто $X \neq 7$, то виконання умовного оператора відразу припиняється і здійснюється перехід до наступного оператора.

Приклад 3. Визначення максимального з двох нерівних чисел X та У:

If X > Y **Then** max = X **Else** max = Y

Змінній max присвоюється значення максимального з порівняних чисел.

Для перевірки відразу декількох умов використовують також умовний оператор вибору варіанту такої конструкції:

Select Case ім′я змінної (або вираз)

Case умова 1
послідовність операторів обробки умови 1
Case умова 2
послідовність операторів обробки умови 2
...
Case умова N
послідовність операторів обробки умови N
End Select
…

Логіка роботи цього оператора така: при збігу присвоєного змінній значення з умовою 1 виконується послідовність операторів обробки умови 1 і управління передається оператору, що слідує за **End Select**; при незбігу – перевіряється умова 2 і т.д.

Вправа 10. Створити додаток, який визначає, що введене у текстовий бокс число більше або менше нуля. Для видавання результату порівняння використати елемент управління **Label1**.

Встановіть на формі елементи управління **Command1, Label1** та **Text1**.

Для елементів **Label1** та **Text1** видаліть властивості **Caption**.

Викличте **Программный код** елемента **Command1** та введіть у шаблон для процедури виділені сірим кольором оператори:

```
Private Sub Command1_Click()
Dim A As Single
A = Val(Text1.Text)
If A > 0 Then
Label1.Caption = " > 0"
Else
Label1.Caption = " < 0"
End If
End Sub
```

Запустіть додаток на виконання. Введіть будь-яке число у текстовий бокс **Text1** та натисніть командну кнопку. На полі елементу **Label1** з′явиться сповіщення про результат порівняння. Перевірте працездатність додатку для різних наборів даних.

Поверніться до режиму редагування форми, підберіть та встановіть для **Label1** властивість **Font (размер шрифта).**

Вправа 11. Додатково до умов вправи 10 перевіряти умову, що введене число А дорівнює нулю.

Програмний код додатку:

```
Private Sub Command1_Click()
```

```
Dim A As Single
A = Val(Text1.Text)
If A > 0 Then
Label1.Caption = " > 0"
ElseIf A < 0 Then
Label1.Caption = " < 0"
Else
Label1.Caption = " = 0"
End If
End Sub
```

Запустіть додаток на виконання, перевірте його працездатність для різних наборів даних.

Вправа 12. Із використанням оператора **ElseIf** створити додаток КАЛЬКУЛЯТОР, для якого операція задається у текстовому боксі.

Встановіть для **Form1** властивість **Caption** Калькулятор.

Встановіть на формі елементи управління **Label1, 2, 3,4, Text1, 2, 3, 4, Command1, 2** відповідно до зразка (рис. 3.12).

Для елементів **Label1, 2, 3, 4** встановіть властивість **Caption** відповідно "A=", "B=", "операція" і "результат".

Для елементів **Command1, 2** встановіть відповідно властивості **Caption** Очистити та Обчислити.

Викличте **Программный код** елемента **Command1** та введіть у шаблон для процедури виділені сірим кольором оператори:

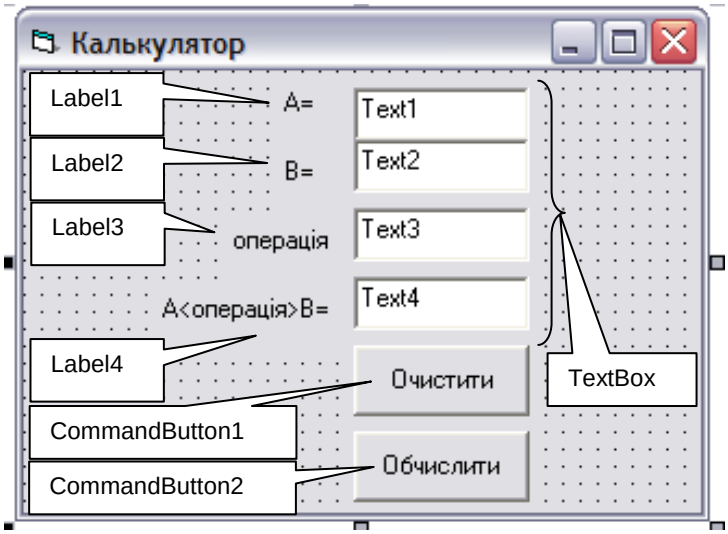


Рис. 3.12

```
Private Sub Command1_Click()
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text5.Text = ""
End Sub
```

Викличте **Программный код** елемента **Command2** та введіть у шаблон для процедури виділені сірим кольором оператори:

```
Private Sub Command2_Click()
Dim A As Single
Dim B As Single
Dim C As String
A = Val(Text1.Text)
B = Val(Text2.Text)
C = (Text3.Text)
If C = "+" Then
Text5.Text = A + B
ElseIf C = "-" Then
Text4.Text = A - B
ElseIf C = "*" Then
Text4.Text = A * B
ElseIf C = "/" Then
Text4.Text = A / B
End If
End Sub
```

Запустіть додаток на виконання. Натисніть кнопку **Очистити**, введіть будь-які числові дані у текстові бокси **Text1, 2,** а у **Text3** символ будь-якої арифметичної операції. Натисніть кнопку **Обчислити**. У текстовому боксі **Text4** з′явиться результат. Перевірте працездатність додатку для різних наборів даних та різних операцій, у тому числі ділення на 0.

Якщо властивість Name **Command1, 2** визначити відповідно як **ochist** і **obchisl**, то перші рядки відповідних шаблонів автоматично перейменуються на **Private Sub ochist_Click()** і **Private Sub obchisl_Click()**.

Вправа 13. Створити додаток КАЛЬКУЛЯТОР, для якого операція задається у текстовому боксі, з використанням оператора **Select Case**.

Виконується аналогічно вправі 12, за винятком частини програмного коду кнопки **Обчислити**, виділеної сірим кольором:

```
Private Sub Command2_Click()
Dim A As Single
Dim B As Single
Dim C As String
A = Val(Text1.Text)
B = Val(Text2.Text)
C = (Text3.Text)
Select Case C
Case "+"
Text4.Text = A + B
Case "-"
```

```
Text4.Text = A - B
Case "*"
Text4.Text = A * B
Case "/"
Text4.Text = A / B
End Select
End Sub
```

Запустіть додаток на виконання. Перевірте працездатність додатку для різних наборів даних та різних операцій.

3.7. Програмування циклічних алгоритмів

Циклічний обчислювальний процес – це такий процес оброблення інформації, для якого характерним є багаторазове виконання одного або кількох операторів алгоритму. Багато разів повторювані процеси називають циклами або повтореннями.

Розрізнюють два види циклів: з відомим і невідомим числом повторень.

Для програмування циклів із відомим числом повторень застосовується оператор циклу **For…Next**, який називається циклом із лічильником. Це багаторядковий оператор, синтаксис якого такий:

For Ім’я = Значення1 **To** Значення2 [**Step** Значення3]
Оператори, що повторюються (тіло циклу)
Next [Ім’я]

Тут Ім’я – це ім’я змінної, яку називають лічильником (індексом циклу); Значення 1 – початкове значення лічильника ; Значення 2 – його кінцеве значення; Значення 3 – величина, на яку змінюється значення лічильника при одному повторенні. Вона називається кроком циклу.

Тіло циклу - це та частина програми, яка має повторюватися один або декілька разів.

Конструкція [**Step** Значення3] може бути відсутня. При цьому за замовчуванням вважається, що лічильник змінює своє значення на одиницю (крок циклу дорівнює 1). Ім’я лічильника після ключового слова **Next** також може бути відсутнім.

Приклад 4. Підрахувати суму чисел від 1 до 10 та видати їх на поле форми.

По черзі наберіть програмний код командної кнопки **Command1** за варіантами і запустіть їх на виконання. Результат побачите після натискання кнопки **Command1.**

Варіант 1.

```
Private Sub Commandl_Click ( )
S = 0
S = 1 + 2 +3 + 4 + 5 + 6 + 7 + 8 + 9 +10.
Print “S = “, S
End Sub
```

Варіант 2.

```
Private Sub Commandl_Click ( )
S = 0
S=S+1: S=S+2: S=S+3: S=S+4: S=S+5: S=S+6: S=S+7: S=S+8:S= S+9: S=S+10.
End Sub
```

Варіант 3.

```
Private Sub Commandl_Click ( )
S = 0
For i = 1 To 10
S = S + i
Print “S = “, S
Next i
End Sub
```

Варіант 4.

```
Private Sub Commandl_Click ( )
S = 0
For i = 1 To 10
S = S + i
Next i
Print “S = “, S
End Sub
```

Перший та другий варіанти – прямі обчислення. Вони можуть бути доцільними при малій кількості чисел, що додаються. Третій та четвертий варіанти – циклічні обчислювальні процеси. У третьому варіанті результати додавання будуть друкуватися на кожному циклі, а у четвертому варіанті друкується тільки остаточний результат.

Приклад 5. Підрахувати суму чисел від 1 до 100.

У програмному коді варіанту 4 прикладу 4 зміниться тільки оператор початку циклу: **For** i = 1 To 100.

Розглянуті приклади демонструють виконання циклів **For…Next** при додатному кроці циклу (цикл завершується тоді, коли значення лічильника

стає більшим, ніж його кінцеве значення). Для випадку, коли вказується від'ємний крок, цикл завершується, якщо значення лічильника стає меншим, ніж його задане кінцеве значення.

Приклад 6.

У програмному коді командної кнопки **Command1** наберіть оператори, виділені сірим кольором:

```
Private Sub Commandl_Click ( )
For i = 10 To 2 Step -2
FontSize = 18 + i
Print "Visual BASIC"
Next i
End Sub
```

Запустіть додаток на виконання. Після натискання кнопки **Command1** рядок "Visual BASIC" буде надрукований п'ять разів, і при кожному повторенні розмір шрифту буде зменшуватися на два пункти:

Якщо при повтореннях змінюється не одна, а дві (або більше) величин, і при кожному значенні однієї величини інша величина „пробігає" всі свої значення, то для програмування таких циклічних процесів застосовуються вкладені цикли, конструкції яких мають такий вигляд:

For І'мя А = значення А1 **To** Значення А [**Step** Значення А3]
For І'мя В = значення В1 **To** Значення В [**Step** Значення В3]
Оператори, що повторюються
Next [І'мя А]
Next [І'мя В]

Тут І'мя А та І'мя В – лічильники двох циклів (зовнішнього циклу А і внутрішнього циклу В).

Принцип роботи вкладеного циклу такий: при кожному значенні лічильника зовнішнього циклу А лічильник внутрішнього циклу В „пробігає" всі свої значення, причому при кожному значенні лічильника внутрішнього циклу В виконуються оператори, що повторюються.

Приклад 7. Розробити програму для виведення на екранну форму таблиці чисел за зразком

1	2	3	4	5
1	4	9	16	25
1	8	27	64	125
1	16	81	256	625

У першому рядку таблиці розташовано натуральні числа від 1 до 5, у другому рядку – їх квадрати, в третьому – їх куби і т.д.

Позначимо через i – номер рядка, а через j – номер стовпця таблиці. З використанням вкладених циклів програма для виведення на екранну форму (рис. 3.13) таблиці має такий вигляд:

```
Private Sub Commandl_Click ( )
ni = 4
nj = 5
Cls
For i = 1 To ni
For j = 1 To nj
Print j^i,
Next j
Print j
Next i
End Sub
```

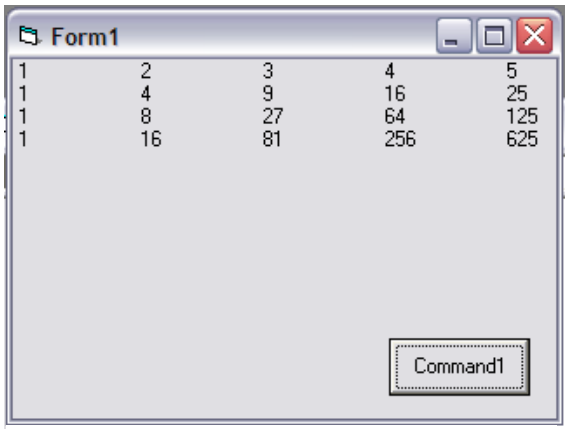


Рис. 3. 13

В програмі застосований метод очищення екранної форми Cls.

Запустіть додаток на виконання. Після натискання кнопки **Command1** на полі форми будуть надруковані результати у вигляді таблиці. Такий формат забезпечується символом “,”, набраним у кінці оператора **Print**.

Вправа 14. Створити додаток, який видає на екранну форму п’ять разів сповіщення Привіт!, кожний раз збільшуючи розмір шрифту на 2 пункти.

Встановіть властивість **Caption** для елементів управління **Form1** і **Command1** відповідно Привіт та Привіт!

Подвійним клацання на кнопці **Command1** викличте **Программный код** (рис. 3.10) та введіть у шаблон для процедури оператори, виділені сірим кольором: Після запуску додатку на виконання і натискання кнопки **Command1** поле форми буде мати вигляд, наведений на рис. 3.14.

```
Private Sub Command1_Click()
For i = 1 To 5
FontSize = 8 + 2 * i
Print “Привіт!”
Next i
End Sub
```

Рис. 3.14

Вправа 15. Розробити програму сортування заданої кількості цілих чисел за зростанням. Результати сортування видавати на екранну форму, кожний раз збільшуючи розмір шрифту на 2 пункти.

У програмному коді форми об′явіть масив змінних a(i) для введення:

Private Sub Form1
Dim a(1 To 100) **As Integer**
End Sub

Встановіть властивості елементів відповідно до табл. 3.7.

Наберіть оператори програмного коду відповідно до табл. 3.8.

Запустіть додаток на виконання. Після натискання кнопки **Command1** введіть n = 5, довільну послідовність цілих чисел (наприклад, 12, 2, 4, 7, 9). В результаті роботи додатку поле форми буде мати вигляд, наведений на рис. 3.15.

Таблиця 3.7

Елемент управління	Caption
Label1	n =
Text1	
Command1	Сортувати

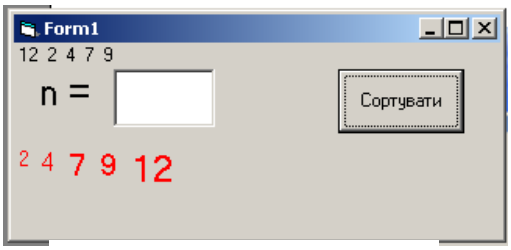


Рис. 3.15

Якщо елементи **Label1 і Text1** розмістити у правому нижньому куті форми, то у програмному коді можна видалити оператори пропуску п'яти рядків друку.

Таблиця 3.8

Програмний код	Коментар
Private Sub Command1_Click() n = Val(Text1.Text)	Введення кількості чисел n

If n = 0 **Then**
MsgBox "Введіть n!", 0, "n=0"
Сповіщення
Им'я форми
End If
	Перевірка значення n = 0
	При виконанні умови видавання сповіщення з використанням функції **MsgBox**

Закінчення табл. 3.8

Cls	Метод очищення екранної форми
For i = 1 **To** n a(i) = **InputBox** ("a(i)=", "a(i)") Сповіщення Им'я форми	Цикл введення змінних у масив з використанням функції **InputBox**
FontSize = 8 ForeColor = &H0& Print a(i); **Next** i	Встановлення розміру шрифту Встановлення чорного кольору шрифту Друк введеної змінної через один інтервал (заданий символом ";")

For k = n - 1 **To** 1 **Step** -1
For i = 1 **To** k
If a(i) > a(i + 1) **Then**
p = a(i + 1)
a(i + 1) = a(i)
a(i) = p
End If
Next i
Next k
	Сортування чисел способом перестановки

For i = 1 **To** 5 Print **Next** i	Пропуск п'яти рядків друку
ForeColor = &HFF&	Встановлення червоного кольору шрифту
For i = 1 **To** n FontSize = 8 + 2 * i Print a(i); **Next** i Text1.Text = "" **End Sub**	Цикл друку результатів сортування зі збільшенням розміру шрифту на кожному кроці на два пункти Очищення текстового боксу

Для програмування циклів з невідомим числом повторень використовуються оператори циклу з умовою.

Цикл з умовою (табл. 3.9) – це багаторядковий оператор, перший порядок якого починається з ключового слова **Do** (виконати), а останній – з ключового слова **Loop** (цикл); умови бувають двох типів: з ключовими словами **While** (умова продовження циклу - **True**) і **Until** (умова завершення циклу - **False**).

Таблиця 3.9

Передумова (спочатку перевіряється умова, а потім виконуються оператори циклу)	Післяумова (спочатку виконуються оператори циклу, а потім перевіряється умова)
Do While Умова Оператори, що повторюються **Loop**	**Do** Оператори, що повторюються **Loop While** Умова
Do Until Умова Оператори, що повторюються **Loop**	**Do** Оператори, що повторюються **Loop Until** Умова

Вправа 16. Знайти задане число з масиву випадкової послідовності цілих чисел та визначити номер його позиції у масиві.

Встановіть праворуч на формі елементи управління **Command1** та **Text1**.

Введіть оператори програмного коду відповідно до табл. 3.10.

Таблиця 3.10

Програмний код	Коментар
Private Sub Form1 **Dim** a(1 To 10) **As Integer** **End Sub**	Об′явлення масиву змінних
Private Sub Command1_Click() X = Val(Text1.Text)	Введення шуканого числа X
Cls	Метод очищення екранної форми
For i = 1 **To** n a(i) = Int (Rnd*10) + 1 Print a(i); **Next** i	Цикл генерації масиву випадкових змінних a(i) в діапазоні від 1 до 10 Друк генерованого числа
i = 1	
Do While a(i) <> X i = i + 1 **Loop** Print “i = ”; i **End Sub**	Цикл пошуку заданого числа у масиві Друк позиції шуканого числа

Запустіть додаток на виконання. Введіть в текстовий бокс значення шуканого числа. Після натискання кнопки **Command1** на полі форми додатку буде надрукований рядок генерованих випадкових чисел з діапазону від 1 до10 та номер позиції шуканого числа.

3.8. Створення головного меню проекту

Для створення головного меню проекту в рядку головного меню VB (рис. 3.16) натискають кнопку **Инструменты**. У спадному меню, що з'явилося, активують опцію **Редактор Меню**. У вікні, яке відкрилося, набирають потрібний заголовок (Довідка) і відповідне йому ім’я (help). Перший рядок відповідного шаблону буде мати вигляд **Private Sub help_Click** (див. вікно коду на рис. 3.16). В нашому прикладі процедура обробки цієї події викликає форму **Form1**.

Перехід до наступної команди головного меню проекту здійснюється натисканням кнопки **Следующий** вікна **Редактор Меню**. Спадне меню типу **Вид тексту** оформлюється шляхом натискання кнопки  , після чого набирають потрібний заголовок (для нашого прикладу – газета). Аналогічним чином формується спадне меню нижчого рівня.

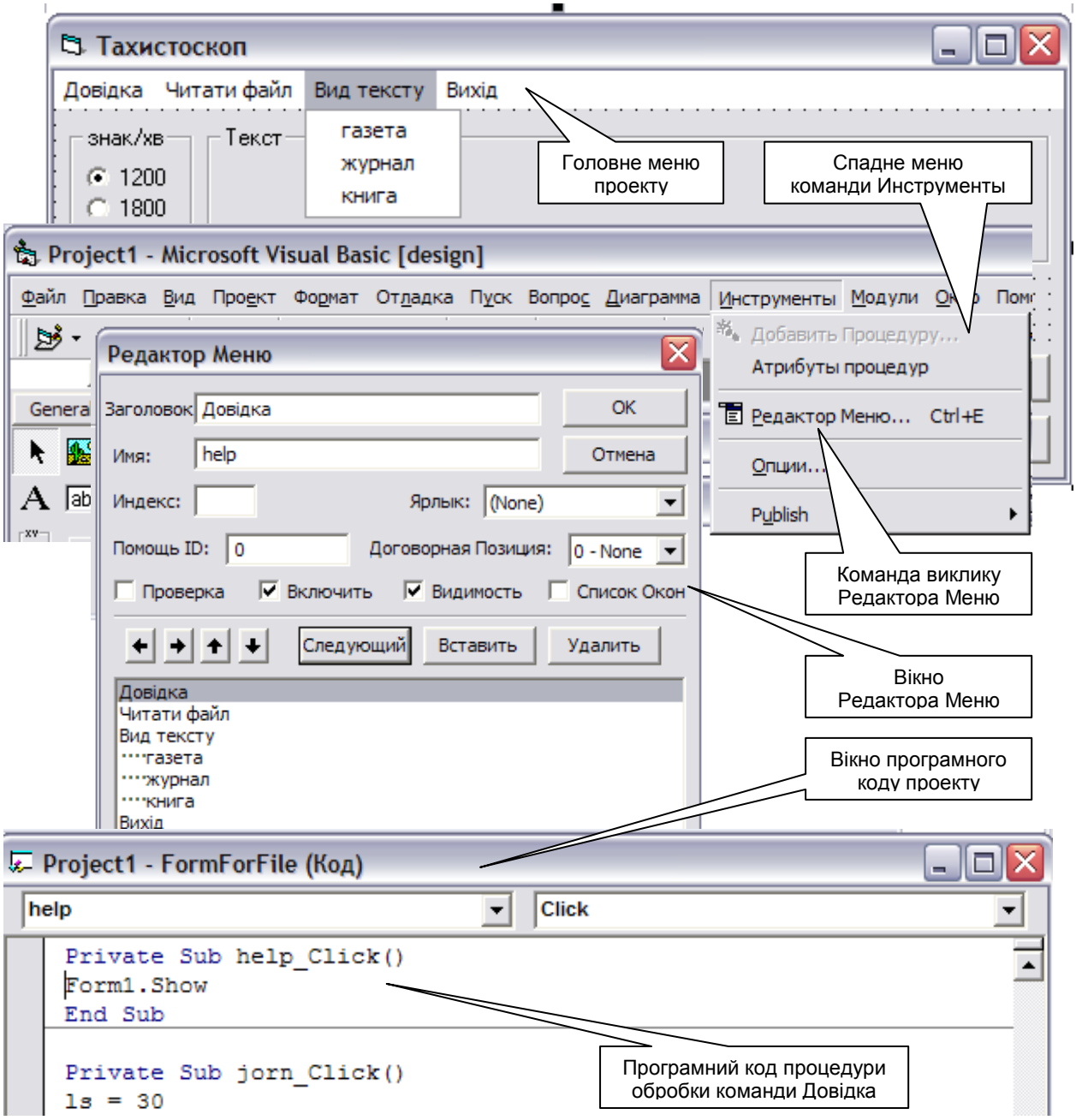


Рис. 3.16

3.9. Створення робочого файлу проекту

Після налагоджування додатку (проекту) зазвичай створюють так званий дистрибутив – файл проекту, що виконується, тобто робочий або exe-файл. Цей файл працює автономно, без підтримки середовища Visual BASIC, в якому створений. Для цього виконують таку низку дій.

Викликають команду **Файл** головного меню VB. У спадному меню, що відкриється, активують опцію **Делать Project exe…**. У вікні, що відкрилося, задають ім'я папки для збереження файлу та им'я файлу. Натискають кнопку **ОК**.

4. ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ

4.1. Питання до розділу 1

1. Що таке мови програмування?
2. Призначення машинно-орієнтованих мов програмування.
3. Призначення проблемно-орієнтованих (алгоритмічних) мови програмування.
4. Що таке система програмування?
5. Типи мовних процесорів.
6. Призначення асемблера.
7. У чому різниця між компілятором й інтерпретатором?
8. Призначення конвертора.
9. Для чого служить редактор текстів програм?
10. Призначення та склад бібліотеки стандартних програм.
11. Призначення програм для налагоджування.

4.2. Питання до розділу 2

1. Що таке алгоритм?
2. Що таке алгоритмізація?
3. Етапи алгоритмізації розрахункових задач.
4. Способи подання алгоритмів та їх сутність.
5. Сутність поняття “програмування”.
6. Порядок складання блок-схем алгоритмів.
7. Що таке програма?
8. Що таке оператор?
9. Що таке команда-директива?

4.3. Питання до розділу 3

1. Поняття **Проект (Project)** Visual BASIC.
2. **Форма (Form)** у системі VB.
3. **Елементи управління (Controls**) системи Visual BASIC.
4. Чим характеризується об’єкт VB?
5. Характеристика **Подія.**
6. Характеристика **Метод**.
7. **Властивість (property)** об’єкта VB.
8. Призначення процедур додатку VB.
9. Чим забезпечується процедура оброблення події?
10. Алфавіт Visual BASIC.
11. Синтаксис Visual BASIC.
12. Математичні операції VB.
13. Вбудовані функції Visual BASIC.
14. Ключові (зарезервовані слова) VB.

15. Стандартні типи даних.

16. Типи змінних (способи оголошення та визначення області видимості).

17. Структура робочого вікна системи Visual BASIC.

18. Порядок створення додатку в системі Visual BASIC.

5. ЗАВДАННЯ ДЛЯ САМОСТІЙНОГО ВІДПРАЦЬОВУВАННЯ

У процесі розроблення додатків доцільно дотримуватися такої послідовності дій:

- розробка алгоритму у вигляді блок-схеми;
- розробка екранної форми;
- встановлення елементів управління на екранній формі;
- визначення властивостей екранної форми та елементів управління;
- написання та введення програмного коду;
- перевірка працездатності програмного коду - контрольний розрахунок;
- створення дистрибутиву;
- збереження проекту.

Завдання 1. Для розробленого у вправі 9 додатку КАЛЬКУЛЯТОР добавити в елементі **Frame** кнопку **F,** призначеної для обчислення функції $y = \text{arctg}[\sqrt{ax^2 + b} + \ln(ax^2 + b)]$.

Процес обчислення доцільно розбити на елементарні операції:

k = a*c^2 + b;

l = Sqr (k);

m = Log (k);

y = Atn (l + m).

Виконати контрольний розрахунок на калькуляторі для a = 1, b = 1, c = x = 1. Результат обчислення дорівнює 1,1277... .

Завдання 2. Створити додаток, який визначає результат порівняння цілих довільних чисел **a** і **b**. Додатково виводити відповідне сповіщення, якщо **a** більше 0 і більше або менше 10.

Завдання 3. Створити додаток для обчислення за формулою $y = (a + b)/\sqrt{x - 1}$ із перевіркою умови та видаванням сповіщення "**x ≤ 1, Повторіть введення!**". Для введення змінної та виведення сповіщення використати оператори **Input Box** та **Msg Box**.

Завдання 4. Створити додаток для розрахунку зарплатні за виразом:

$$З = \begin{cases} Р * В_ф - \text{при відрядній простій формі оплати;} \\ Р * В_ф + П - \text{при відрядно преміальні й формі оплати;} \\ Р * (В_ф - В_д) + В_д * (Р + Р_д) - \text{при відрядно прогресивній формі оплати,} \end{cases}$$

де Р – відрядна розцінка за одиницю продукції;

$В_ф$ – фактичний заробіток;

П – премія;

$В_д$ – надплановий заробіток;

$Р_д$ – надбавка до розцінки.

Завдання 5. Розробити додаток для обчислення суми елементів вектора

$$a = \{a_1, a_2, \ldots, a_n\}.$$

Елементи вектора a_i – довільні цілі числа.

Завдання 6. У річному фінансовому плані на ремонт приміщення виділяються витрати на ремонтні роботи протягом кожного місяця. Потрібно організувати циклічне введення у ПК даних про витрати на ремонт приміщення за кожний місяць фінансового року (MP) і розрахувати сумарні витрати (CP) на рік:

$$\mathbf{CP} = \sum_{\mathbf{s=1}}^{\mathbf{n}} \mathbf{MP(s)},$$

де n - кількість місяців, протягом яких плануються ремонтні роботи.

Обчислення провести з урахуванням таких умов:

1. Період проведення ремонтних робіт (кількість місяців) задається користувачем у діапазоні від 1 до 12.

2. Значення витрат за кожний місяць вводиться у ПК послідовно за допомогою функції **InputBox**.

При програмуванні застосовувати цикл з умовою.

Завдання 7. Для умов вправи 16 застосувати цикли з умовою **Do... Loop While; Do Until ... Loop; Do... Loop Until.**

Завдання 8. Розробити додаток для обчислення функції

$$y = \sum_{i=1}^{n} f(x_i),$$

де x_i - значення змінної, заданої в інтервалі $x_1 \le x \ge x_n$ із кроком зміни Δx.

Алгоритм розв’язання побудувати за принципом накопичення суми. Умова виходу з циклу – задовільнення нерівності $x \le x_n$.

Завдання 9. Розробити програму обчислення функції

$$y = \operatorname{arctg}[\sqrt{ax^2 + b} + \ln(ax^2 + b)]$$

для значення аргументу $x = x_0$.

Завдання 10. Розробити програму обчислення коренів квадратного рівняння $ax^2 + bx + c = 0$, що знаходяться з виразу

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

Додаткові умови:

- вихідні дані - чисельні значення коефіцієнтів a, b, c;

- рішення не існує, якщо a = 0 або дискримінант $\Delta = b^2 - 4ac < 0$.

Завдання 11. Розробити додаток для обчислення функції при $x = x_0$

$$y = \frac{\sin(ax^2 + b)}{\sqrt{ax^2 + b}}.$$

Константи a, b задаються.

Доцільно попередньо розрахувати $z = ax^2 + b$ та перевірити його від′ємне значення.

Завдання 12. Розробити додаток-калькулятор з можливістю виконання арифметичних операцій та основних математичних функцій за зразком (рис. 5.1). Тут 1 – об'єкт Form, 2 – компоненти Frame, 3 – компонент TextBox, 4-10 – компоненти CommandButton, що реалізують подію Click (клацання на кнопці мишею). В табл. 5.1 наведено опис реакцій на події.

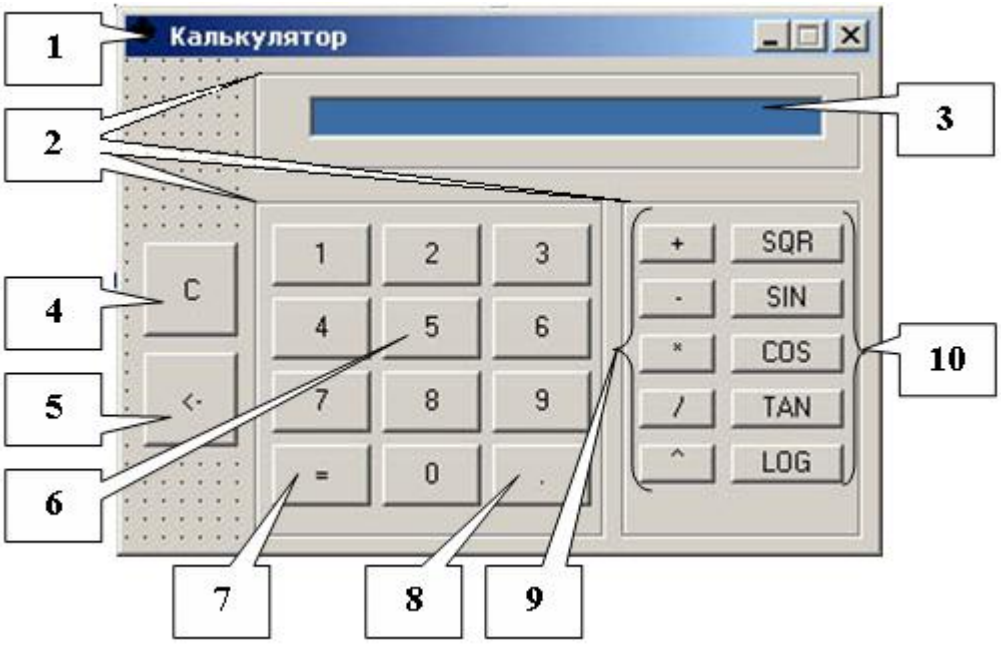


Рис. 5.1

У додатку А наведено програмний код додатку «Калькулятор». Операції, відмічені коментарем «Встановлення параметрів (властивостей) компонентів» можна виконати вручну.

Таблиця 5.1

Компонент	Реакція на подію	Примітка
1	Всі змінні і властивість Text компонента 3 = 0	
4	Всі змінні і властивість Text компонента 3 = 0	Кнопка СКИДАННЯ
5	У властивості Text компонента 3 видалити останній символ	Кнопка ЗАБІЙ
6	У властивості Text компонента 3 добавити останній символ, відповідний натиснутій кнопці	Кнопки набору чисел 0, 1, 2, …, 9
7	У властивість Text компонента 3 вивести результат	Кнопка РЕЗУЛЬТАТ
8	У властивості Text компонента 3 добавити останній символ, відповідний десятковій точці	Перевірити, чи не є це повтором
9	Виконати відповідну операцію	Перевірити / на 0
10	Обчислити відповідну функцію	Перевірити LOG, SQR

6. КОНТРОЛЬНІ ЗАВДАННЯ

Варіант	Функція	Вихідні дані			
1	$X = \frac{1}{\sqrt{a+b+c}}$		a	b	c
		1	1	1	2
		2	2	1	-3
		3	-2	-1	1
2	$X = \frac{1}{4a} + \frac{\sqrt{c}}{2b}$		a	b	c
		1	1	1	1
		2	0	1	3
		3	2	1	-1
3	$X = \frac{\sqrt{a}}{c+b}$		a	b	c
		1	1	1	1
		2	2	1	-1
		3	-2	1	1
4	$X = a + \sqrt{c} + \frac{1}{b}$		a	b	c
		1	1	1	1
		2	2	0	4
		3	2	1	-1
5	$X = \frac{1}{a+b} + \frac{1}{\sqrt{c}}$		a	b	c
		1	1	1	1
		2	1	-1	1
		3	2	1	-1
6	$X = \frac{72\sqrt{a}}{4b+c}$		a	b	c
		1	1	1	0
		2	2	1	-4
		3	-2	1	1
7	$X = \frac{\sqrt{c}}{(a+b)^2}$		a	b	c
		1	1	1	1
		2	2	-2	3
		3	2	1	-1
8	$X = \frac{a^2\sqrt{c}}{2b}$		a	b	c
		1	1	1	1
		2	2	0	1
		3	2	1	-1
9	$X = \frac{\sqrt{c}+a}{a+b}$		a	b	c
		1	1	1	1
		2	-1	1	1
		3	1	1	-1
10	$X = \frac{c}{10b} + \sqrt{a}$		a	b	c
		1	1	1	1
		2	1	0	2
		3	-1	1	1
11	$X = \frac{1}{a}\sqrt{bc}$		a	b	c
		1	1	1	1
		2	0	1	1
		3	1	-1	1

12	$X = \frac{1}{a + b + \sqrt{c}}$		a	b	c
		1	1	1	2
		2	0	0	0
		3	1	1	-1
13	$X = \frac{c}{2} + \frac{\sqrt{a^3}}{10b}$		a	b	c
		1	1	1	1
		2	1	0	1
		3	-1	1	1
14	$X = \frac{1}{a} + \frac{1}{b} + \frac{\sqrt{c}}{2}$		a	b	c
		1	1	1	4
		2	2	0	1
		3	1	1	-1
15	$X = \frac{\sqrt{a} + b^2}{c + b}$		a	b	c
		1	1	1	1
		2	1	1	-1
		3	-1	1	1
16	$X = \frac{(b-1)\sqrt{a}}{c-b} + 2b$		a	b	c
		1	1	1	2
		2	1	1	1
		3	-1	-1	1
17	$X = \frac{ab - bc}{2\sqrt{ac}} + 5$		a	b	c
		1	1	1	0
		2	0	1	0
		3	-1	-1	1
18	$X = \sqrt{\frac{a + b - c}{a + b}}$		a	b	c
		1	1	1	0
		2	1	-1	0
		3	-1	-1	0
19	$X = \frac{\sqrt{c-1}}{a-b}$		a	b	c
		1	2	1	1
		2	1	1	1
		3	2	1	0
20	$X = \sqrt{a} + \frac{b}{4 - c}$		a	b	c
		1	1	1	0
		2	1	1	4
		3	-1	1	1
21	$X = \frac{1}{\sqrt{a + b}} + \frac{1}{\sqrt{c}}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
22	$X = \frac{\sqrt{b^2 - 4ac}}{2a}$		a	b	c
		1	1	2	1
		2	0	1	1
		3	1	3	1
23	$X = \frac{a + b}{c} - \sqrt{ab}$		a	b	c
		1	1	1	1
		2	-1	1	1
		3	2	2	0

24	$X = \frac{(a+b)^2}{2\sqrt{c}}$		a	b	c
		1	1	1	1
		2	-1	1	0
		3	1	3	0
25	$X = \frac{a+b+c}{3} - \frac{3}{\sqrt{c}}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
26	$X = \frac{\sqrt{a^2 - b^2}}{c^2}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
27	$X = \frac{5c^2}{\sqrt{a-b}}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
28	$X = \frac{7}{b-a} + 2\sqrt{c}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
29	$X = \frac{9}{a+b+c} - \sqrt{c}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
30	$X = \frac{a+b}{4\sqrt{c}}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
31	$X = \frac{a+b+c}{c^2} - \sqrt{ab}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
32	$X = \frac{a^2 - ab + b^2}{8\sqrt{c}}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0
33	$X = \frac{a+b}{2c} - \sqrt{abc}$		a	b	c
		1	1	1	1
		2	-1	1	2
		3	1	3	0

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Браун С. Visual BASIC 6: Учебный курс/ С. Браун – СПб.: Питер, 2001. – 576 с.

2. Васильев П.П. Встроенные функции языка программирования Visual BASIC 6.0./ П.П. Васильев – М.: Диалог-МИФИ, 2000. – 160 с.

3. Зарецька І.Т. Інформатика/ І.Т. Зарецька, Б.Г. Колодязний, А.М. Гурій, О.Ю. Соколов. – К.: Навчальна книга, 2002. – 496 с.

4. Інформатика. Комп'ютерна техніка. Комп'ютерні технології/ За ред. О.І. Пушкаря – К.: Академія, 2001. - С. 143-163.

5. Кушниренко А.Г. Основы информатики и вычислительной техники/ А.Г. Кушниренко, Г.В. Лебедев, Р.А. Сворень. – М.: Просвещение, 1993. – 224 с.

6. Чернов Б.И. Программирование на алгоритмических языках Бейсик, Фортран, Паскаль/ Б.И. Чернов. – М.: Просвещение, 1991. – 192 с.

Додаток А

Програмний код додатку «Калькулятор»

```
' --------- Встановлення параметрів (властивостей) компонентів
Begin VB.Form Calc
   Caption         =   "Калькулятор"
   ClientHeight    =   3195
   ClientLeft      =   7515
   ClientTop       =   5085
   ClientWidth     =   4890
   FillColor       =   &H8000000F&
   FillStyle       =   7  'Diagonal Cross
   Icon            =   "Calc.frx":0000
   LinkTopic       =   "Form1"
   MaxButton       =   0   'False
   ScaleHeight     =   3195
   ScaleWidth      =   4890
   Begin VB.Frame Frame3
      Height          =   735
      Left            =   840
      TabIndex        =   4
      Top             =   0
      Width           =   3975
      Begin VB.TextBox Text1
         Alignment       =   1  'Right Justify
         BackColor       =   &H80000001&
         ForeColor       =   &H80000005&
         Height          =   285
         Left            =   360
         Locked          =   -1  'True
         TabIndex        =   5
         Top             =   240
         Width           =   3375
      End
   End
   Begin VB.CommandButton Cmd_zaboy
      Caption         =   "<-"
      Height          =   615
      Left            =   120
      MaskColor       =   &H000000FF&
      Style           =   1  'Graphical
      TabIndex        =   3
      Top             =   2160
      Width           =   615
   End
   Begin VB.CommandButton Cmd_sbros
      Caption         =   "C"
      Height          =   615
      Left            =   120
      MaskColor       =   &H000000FF&
      Style           =   1  'Graphical
      TabIndex        =   2
      Top             =   1200
      Width           =   615
```

```
End
Begin VB.Frame Frame2
   Height         =   2295
   Left           =   840
   TabIndex       =   1
   Top            =   840
   Width          =   2295
   Begin VB.CommandButton Cmd_dot
      Caption        =   "."
      Height         =   400
      Left           =   1560
      TabIndex       =   17
      Top            =   1680
      Width          =   650
   End
   Begin VB.CommandButton Cmd_0
      Caption        =   "0"
      Height         =   400
      Left           =   840
      TabIndex       =   16
      Top            =   1680
      Width          =   650
   End
   Begin VB.CommandButton Cmd_eq
      Caption        =   "="
      Height         =   400
      Left           =   120
      TabIndex       =   15
      Top            =   1680
      Width          =   650
   End
   Begin VB.CommandButton Cmd_9
      Caption        =   "9"
      Height         =   400
      Left           =   1560
      TabIndex       =   14
      Top            =   1200
      Width          =   650
   End
   Begin VB.CommandButton Cmd_8
      Caption        =   "8"
      Height         =   400
      Left           =   840
      TabIndex       =   13
      Top            =   1200
      Width          =   650
   End
   Begin VB.CommandButton Cmd_7
      Caption        =   "7"
      Height         =   400
      Left           =   120
      TabIndex       =   12
      Top            =   1200
      Width          =   650
   End
```

```
    Begin VB.CommandButton Cmd_6
      Caption         =   "6"
      Height          =   400
      Left            =   1560
      TabIndex        =   11
      Top             =   720
      Width           =   650
    End
    Begin VB.CommandButton Cmd_5
      Caption         =   "5"
      Height          =   400
      Left            =   840
      TabIndex        =   10
      Top             =   720
      Width           =   650
    End
    Begin VB.CommandButton Cmd_4
      Caption         =   "4"
      Height          =   400
      Left            =   120
      TabIndex        =   9
      Top             =   720
      Width           =   650
    End
    Begin VB.CommandButton Cmd_3
      Caption         =   "3"
      Height          =   400
      Left            =   1560
      TabIndex        =   8
      Top             =   240
      Width           =   650
    End
    Begin VB.CommandButton Cmd_2
      Caption         =   "2"
      Height          =   400
      Left            =   840
      TabIndex        =   7
      Top             =   240
      Width           =   650
    End
    Begin VB.CommandButton Cmd_1
      Caption         =   "1"
      Height          =   400
      Left            =   120
      TabIndex        =   6
      Top             =   240
      Width           =   650
    End
  End
  Begin VB.Frame Frame1
    Height          =   2295
    Left            =   3240
    TabIndex        =   0
    Top             =   840
    Width           =   1575
```

```
Begin VB.CommandButton Cmd_log
   Caption         =   "LOG"
   Height          =   255
   Left            =   720
   TabIndex        =   27
   Top             =   1680
   Width           =   735
End
Begin VB.CommandButton Cmd_tang
   Caption         =   "TAN"
   Height          =   255
   Left            =   720
   TabIndex        =   26
   Top             =   1320
   Width           =   735
End
Begin VB.CommandButton Cmd_sqr
   Caption         =   "SQR"
   Height          =   255
   Left            =   720
   TabIndex        =   25
   Top             =   240
   Width           =   735
End
Begin VB.CommandButton Cmd_cos
   Caption         =   "COS"
   Height          =   255
   Left            =   720
   TabIndex        =   24
   Top             =   960
   Width           =   735
End
Begin VB.CommandButton Cmd_sin
   Caption         =   "SIN"
   Height          =   255
   Left            =   720
   TabIndex        =   23
   Top             =   600
   Width           =   735
End
Begin VB.CommandButton Cmd_stepen
   Caption         =   "^"
   Height          =   255
   Left            =   120
   TabIndex        =   22
   Top             =   1680
   Width           =   495
End
Begin VB.CommandButton Cmd_delen
   Caption         =   "/"
   Height          =   255
   Left            =   120
   TabIndex        =   21
   Top             =   1320
   Width           =   495
```

```
    End
    Begin VB.CommandButton Cmd_umnog
      Caption        =   "*"
      Height         =   255
      Left           =   120
      TabIndex       =   20
      Top            =   960
      Width          =   495
    End
    Begin VB.CommandButton Cmd_minus
      Caption        =   "-"
      Height         =   255
      Left           =   120
      TabIndex       =   19
      Top            =   600
      Width          =   495
    End
    Begin VB.CommandButton Cmd_plus
      Caption        =   "+"
      Height         =   255
      Left           =   120
      TabIndex       =   18
      Top            =   240
      Width          =   495
    End
  End
End

' --------- Встановлення параметрів (властивостей) об’єктів і типів змінних

Attribute VB_Name = "Calc"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Dim Result As Double, oper As Boolean, kop As String, dot As Boolean

Static Function Log10(X)
  Log10 = Log(X) / Log(10#)
End Function

Private Sub ins_char(S)
If oper = True Then
   Text1.Text = ""
   oper = False
End If
Text1.Text = Text1.Text + S
End Sub
Private Sub operat()

' --------- Процедура виконання заданих арифметичних операцій

If kop = "" Then Result = Val(Text1.Text)
If kop = "+" Then Result = Result + Val(Text1.Text)
If kop = "-" Then Result = Result - Val(Text1.Text)
If kop = "*" Then Result = Result * Val(Text1.Text)
```

```
If kop = "^" Then Result = Result ^ Val(Text1.Text)
If kop = "/" Then
  If Val(Text1.Text) = 0 Then
    MsgBox ("Помилка. Ділення на 0")
  Else
    Result = Result / Val(Text1.Text)
  End If
End If
Text1.Text = Str(Result)
dot = True
oper = True
kop = ""
End Sub

' --------- Введення чисел

Private Sub Cmd_0_Click()
Call ins_char("0")
End Sub
Private Sub Cmd_1_Click()
Call ins_char("1")
End Sub
Private Sub Cmd_2_Click()
Call ins_char("2")
End Sub
Private Sub Cmd_3_Click()
Call ins_char("3")
End Sub
Private Sub Cmd_4_Click()
Call ins_char("4")
End Sub
Private Sub Cmd_5_Click()
Call ins_char("5")
End Sub
Private Sub Cmd_6_Click()
Call ins_char("6")
End Sub
Private Sub Cmd_7_Click()
Call ins_char("7")
End Sub
Private Sub Cmd_8_Click()
Call ins_char("8")
End Sub
Private Sub Cmd_9_Click()
Call ins_char("9")
End Sub

' --------- Введення операцій і функцій, їх виконання та обчислення

Private Sub Cmd_cos_Click()
Call operat
Result = Cos(Val(Text1.Text))
Text1.Text = Result
End Sub
Private Sub Cmd_delen_Click()
Call operat
```

```
kop = "/"
End Sub
Private Sub Cmd_dot_Click()
If dot Then
   Text1.Text = Text1.Text + "."
   dot = False
Else
   Beep
End If
End Sub
Private Sub Cmd_eq_Click()
Call operat
End Sub

Private Sub Cmd_log_Click()
Call operat
  If Val(Text1.Text) <= 0 Then                  ' логарифм
    MsgBox ("Помилка. Log(" + Trim(Text1.Text) + ")")
  Else
    Result = Log10(Val(Text1.Text))
  End If
Text1.Text = Str(Result)
End Sub

Private Sub Cmd_minus_Click()
Call operat
kop = "-"
End Sub

Private Sub Cmd_plus_Click()
Call operat
kop = "+"
End Sub

Private Sub Cmd_sbros_Click()
Call Form_Load
End Sub

Private Sub Cmd_sin_Click()
Call operat
Result = Sin(Val(Text1.Text))
Text1.Text = Str(Result)
End Sub

Private Sub Cmd_sqr_Click()
Call operat
If Val(Trim(Text1.Text)) < 0 Then             ' квадратний корінь
    MsgBox ("Помилка. Sqr(" + Trim(Text1.Text) + ")")
  Else
    Result = Sqr(Val(Text1.Text))
End If
Text1.Text = Str(Result)
End Sub

Private Sub Cmd_stepen_Click()
Call operat
kop = "^"
```

```
End Sub

Private Sub Cmd_tang_Click()
Call operat
Result = Tan(Val(Text1.Text))
Text1.Text = Str(Result)
End Sub

Private Sub Cmd_umnog_Click()
Call operat
kop = "*"
End Sub

Private Sub Cmd_zaboy_Click()
If Len(Text1.Text) > 0 Then
   If oper = False Then Text1.Text = Left(Text1.Text, Len(Text1.Text) - 1) Else Beep
End If
End Sub

Private Sub Form_Load()


Rem ********Завдання початкових значень для глобальних змінних
Result = 0   '--> тут накопичується результат
oper = False '--> прапорець, що показує, чи була запам'ятована арифметична операція
kop = ""     '--> код операції, яку треба виконати
dot = True   '--> прапорець, що показує, чи можна ставити символ <крапка>
Text1.Text = ""
End Sub
```

Навчальне видання

Сергій Олександрович Белінський
Валентин Євгенович Козлов
Володимир Олександрович Середа

ІНСТРУМЕНТАЛЬНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Навчальний посібник

Редактор *Ф.М. Сирнєв*
Комп'ютерна верстка: *С.О Белінський*
В.Є. Козлов

Підписано до друку 30.06.2011 р. Формат паперу 60x84/16. Різограф.
Папір офсетний. Ум. друк. арк. 3,25. Облік.-вид. арк. 4,48. Тираж 50 прим. Зам. №43

Редакційно-видавничий відділ Академії внутрішніх військ МВС України
Свідоцтво про державну реєстрацію ДК №2799 від. 22.03.07 р.
Друкарня Академії внутрішніх військ МВС України
61001, м. Харків, пл. Повстання, 3