СІМ СПОСОБІВ ПРОГРАМУВАННЯ В СЕРЕДОВИЩІ VBA

Глинський Я.М., Ряжська В.А.

У цій статті на базі двох *парадигм* програмування: 1) процедурного і 2) об'єктно-орієнтованого (об'єктного) — розглянемо сім *способів* (підходів, методів) створення програм у середовищі Visual Basic for Applications (VBA) з дотриманням структурного і модульного *стилів* програмування.

Парадигма програмування (концепція, методологія) — це сукупність принципів розробки програм, що формують спосіб мислення програміста. Розрізняють чотири головні парадигми: процедурне, об'єктне, логічне та функційне програмування. Є й детальніші класифікації. Парадигма програмування не визначається однозначно мовою програмування, оскільки сучасні мови допускають використання різних парадигм. Таке переплетіння парадигм потребує подальшої класифікації способів створення програм. Спосіб (синонім *метод*) програмування конкретизує процес розробки програм у конкретному середовищі програмування і визначає засоби, які для цього використовують. Процедурна чи об'єктна парадигми програмування визначають, як програміст розробляє програму: як сукупність даних і дій над ними, оформлених засобами процедур, чи як сукупність взаємодіючих об'єктів, кожен з яких є екземпляром певного класу, а класи є членами певної ієрархії наслідування. Спосіб визначає конкретну реалізацію парадигми (парадигм). Існує також поняття «стиль програмування». Стиль програмування визначає, як оформлені й побудовані програми. Загальноприйнятими стилями створення великих програм є структурний стиль, який передбачає декомпозицію алгоритму, використання базових алгоритмічних структур для реалізації його частин і структуроване написання текстів кодів, а також модульний стиль, який передбачає об'єднання процедур у незалежні блоки, які називають модулями, а модулів у бібліотеки. Зауважимо, що для невеликих навчальних кодів поняття стилю є несуттєвим, оскільки такі коди, зазвичай, є однопроцедурними й актуальною залишається тільки одна властивість структурного програмування — структуроване написання тексту коду.

Розвиваючи тему, підняту в [1], ми акцентуємо на важливості застосування мови Visual Basic під час вивчення розділу алгоритмізації та програмування в курсі базової інформатики. Ми зосереджуємо увагу на середовищі VBA невипадково. Це результат багаторічних спостережень за тенденціями в освіті й головно за інтересами і можливостями сучасних учнів і студентів. Для студентів, які планують займатися програмуванням на професійному рівні, C# і Java — без сумніву кращий вибір. Тому як суб'єктів навчання за нашою методикою ми розглядаємо студентів некомп'ютерних напрямів підготовки (зокрема студентів і випускників педагогічних вузів відповідного фаху), а також учнів загальноосвітніх шкіл.

Власне у VBA з найменшими затратами можна реалізувати основні три способи програмування: 1) процедурне, 2) візуальне, 3) об'єктне. Правда, це можна не менш успішно зробити і в інших середовищах. Але тільки у VBA можна реалізувати інші способи: 4) різновид процедурного програмування у консольному вікні негайного виконання **Immediate**; 5) різновид візуального програмування безпосередньо в документі офісної програми-додатка, наприклад, MS Word чи MS Excel тощо, без застосування бібліотек класів і колекцій об'єктів; 6) різновид об'єктного програмування безпосередньо в документі офісної програми із застосуванням об'єктних моделей офісних програм, бібліотек класів і колекцій об'єктів; 7) програмування зі створенням макросів.

Продемонструємо методику ознайомлення з цими способами програмування на прикладі розв'язування навчальної задачі.

Спосіб 1. Відкриємо програму MS Word і запишемо умову задачі.

Задача про клумбу. Задано радіус клумби. Обчислити її площу і довжину огорожі.

Щоб побудувати математичну модель задачі, введемо позначення:

r — радіус кола (дано),

L — довжина кола (треба обчислити),

S — площа круга (треба обчислити).

Модель і метод розв'язування задачі — дві відомі формули:

$L=2\pi r$,

$S=\pi r^2$.

Алгоритм розв'язування задачі такий:

<Ввести r>

$L=2\pi r$

$S=\pi r^2$

<Вивести L, S>

Код мовою Visual Basic має такий вигляд:

```
Dim r As Single, L As Single, S As Single
Const pi = 3.1415
<Ввести r>
L=2*pi*r
S=pi*r^2
<Вивести L, S>
```

Необов'язковою командою **Dim** оголошуємо тип змінних *r*, *L*, *S* як дійсний числовий. Командою **Const** оголошуємо константу *pi* і запам'ятовуємо, що грецьких букв у кодах використовувати не можна. У кодах у числах замість десятково коми треба писати крапку. Множення позначають зірочкою, а піднесення до степеня — символом «^».

Команди у кутових дужках несправжні. Замість них треба записати конкретні команди. Тут можливі різні конкретні команди, оскільки є різні способи введення даних і виведення результатів. Розглянемо деякі способи.

Увести значення радіуса можна за допомогою команди присвоєння одним із трьох способів:

1) безпосередньо в коді, наприклад, так: r=8.5;

2) в діалоговому режимі під час виконання коду за допомогою функції **InputBox**, наприклад, так: r=InputBox("А тепер введіть значення радіуса");

3) у візуальному програмуванні за допомогою елемента керування класу «текстове поле», наприклад, так: r=TextBox1.Text.

Вивести на екран результати обчислень *L* та *S* можна трьома способами:

1) у консольне вікно негайного виконання Immediate командою **Debug.Print** так: **Debug.Print** "Довжина=" L, "Площа ="S;

2) у вікно повідомлень такою командою:

MsgBox "Довжина=" & L & " Площа =" & S ;

3) у візуальному програмуванні на форму двома способами: у текстові поля, наприклад, так:

Textbox2.Text=L,

TextBox3.Text=S,

або в елементи керування класу «написи», наприклад, так:

Label1.Caption=L,

Label2.Caption=S.

Продемонструємо усі описані способи.

Відкриємо середовище VBA комбінацією клавіш **Alt+F11**. Виконаємо команди з головного меню **Insert \Module**. Цим самим ми відкриємо вікно модуля **Module1** для написання коду. Уведемо текст **sub klumba** і натиснемо **Enter**. Отримаємо таку заготовку процедури:

```
Sub klumba()
 <Сюди вводять код>
End Sub
```

У цю заготовку введемо текст коду:

```
Dim r As Single, L As Single, S As Single
Const pi = 3.1415
r = 8.5
L=2*pi*r
S=pi*r^2
Debug.Print "Довжина=" L, "Площа ="S
MsgBox "Довжина=" & L & " Площа =" & S
```

Тут значення (нехай 8,5) змінній *r* надаємо командою присвоєння. Результати виводимо у консольне вікно **Immediate** і одночасно у вікно повідомлень.

Запустимо створений код на виконання. Курсор має знаходитися в тексті коду **klumba**. Натиснемо на зеленому трикутнику на панелі інструментів, що позначає команду **Run**, і отримуємо результати як у вікні **Immediate**, так і у вікні повідомлень.

Проведемо експеримент. Закриємо вікно повідомлень, натиснувши на кнопку **ОК**, і змінимо значення радіуса в коді на 10,5. Виконаємо код повторно. Отримаємо нові результати. Отже, щоб отримати результати для іншого значення радіуса, треба внести зміни в код і повторно його виконати.

Виконаємо завдання на модифікацію коду: нехай потрібно ввести значення радіуса в режимі діалогу.

Повернемося в середовище VBA. У коді замість команди r=8.5 запишемо команду r=InputBox("Введіть значення радіуса"). Запустимо код на виконання. Отримаємо діалогове вікно, куди введемо потрібне число, наприклад 8,5 чи інше. Зверніть увагу! У діалогових вікнах у десяткових числах може використовуватися кома, а не крапка. Натиснемо **ОК** і отримаємо результати. Отже, використання режиму діалогу дає змогу вводити вхідні дані на етапі виконання коду без внесення змін у код.

Підготуємо звіт про проведену роботу. Для цього натисканням клавіш **Alt+PrintScreen** скопіюємо зображення активного вікна, що містить текст коду і результати роботи коду в буфер обміну, і вставимо це зображення комбінацією клавіш **Ctrl+V** у документ1, що містить умову задачі. Звіт матиме вигляд, як на рис. 1. Підготовка звіту демонструє органічне поєднання процесу програмування з роботою з текстовим документом, чого не можна досягнути під час використання інших середовищ програмування.

Висновок. Ми продемонстрували використання VBA у програмі MS Word для реалізації способу процедурного програмування, що відповідає однойменній парадигмі.

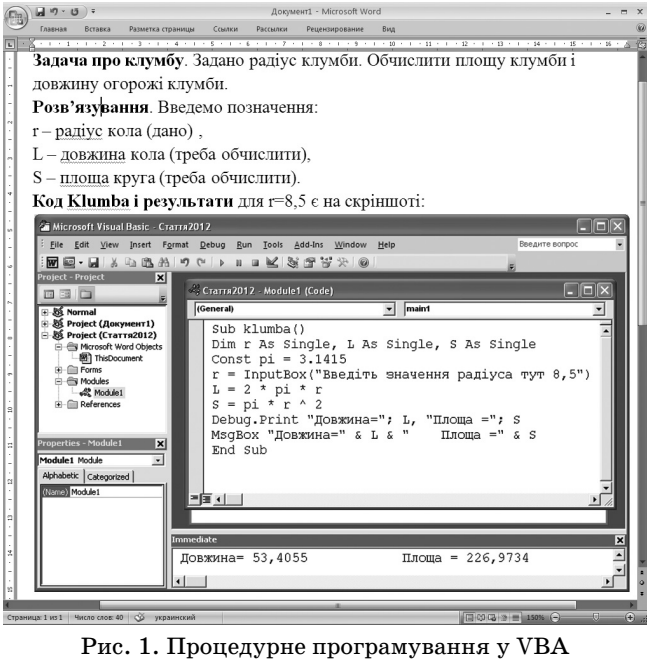

Рис. 1. Процедурне програмування у VBA

Спосіб 2. Тепер розв'яжемо цю ж задачу способом візуального програмування, де під візуальним програмуванням ми розуміємо розробку візуального інтерфейсу користувача з використанням стандартних графічних об'єктів і з написанням кодів опрацювання стандартних подій. Для розв'язування задачі використаємо об'єкт форму і такі візуальні елементи керування: три написи, три текстові поля, дві кнопки.

Запишемо в MS Word умову задачі, як вище. Увійдемо у середовище програмування VBA і вставимо в проект форму **UserForm1** командами **Insert\ UserForm**.

Підпишемо форму, задавши значення її властивості **Caption** як «Задача про клумбу». Для тла форми використаємо будь-яку картинку. За допомогою властивості **Font** задамо великий шрифт текстів для всіх елементів керування, що будуть на формі.

Розташуємо на формі (вертикально) три елементи керування класу «написи». Для цього тричі клацнемо мишею на значку елемента **Label** в **Toolbox** і у формі. Вирівняємо елементи і підпишемо їх так: **Радіус**, **Довжина**, **Площа**. Поряд з ними вставимо за допомогою **Toolbox** три текстові поля **Textbox1**, **Textbox2**, **Textbox3** для відповідних числових величин задачі про клумбу. Вставимо дві кнопки і підпишемо їх за допомогою вікна властивостей як **Обчислити** і **Кінець роботи**.

Сценарій проекту такий: після запуску проекту користувач має ввести в перше текстове поле значення радіуса, клацнути на кнопці **Обчислити** і у двох інших полях отримати результати. Після експериментів з різними значеннями радіуса і виконання всіх обчислень, треба закінчити роботу з проектом, клацнувши на кнопці **Кінець роботи**.

Створимо код кнопки **Обчислити**. Для цього треба в конструкторі двічі клацнути на кнопці — відкриється вікно коду проекту із заготовкою коду кнопки **CommandButton1_Click**. У цю заготовку впишемо чи вставимо за допомогою буфера обміну такий код:

```
Dim r As Single, L As Single, S As Single
Const pi = 3.1415
r = Textbox1.Text
L = 2 * pi * r
S = pi * r ^ 2
Textbox2.Text =L
Textbox3.Text = S
Debug.Print "Довжина=" L, "Площа ="S
```

Значення радіуса введемо з текстового поля, а результати виведемо у два текстові поля і продублюємо у вікні **Immediate**.

Створимо код кнопки **Кінець роботи**, увівши в заготовку коду кнопки одну команду **End**.

Запустимо проект на виконання. Отримаємо вікно працюючого проекту. Уведемо в перше текстове поле значення радіуса 8,5 через десяткову кому, клацнемо на кнопці **Обчислити** й отримаємо результати. Тепер уведемо нове значення радіуса 12,5. Натиснемо кнопку **Обчислити** й отримаємо інші результати.

Створимо звіт. Скопіюємо вікно проекту в буфер обміну **(Alt + PrintScreen)** і вставимо його в документ. Скопіюємо код кнопки **Обчислити** або ціле вікно з кодом і вставимо його в документ (рис. 2).

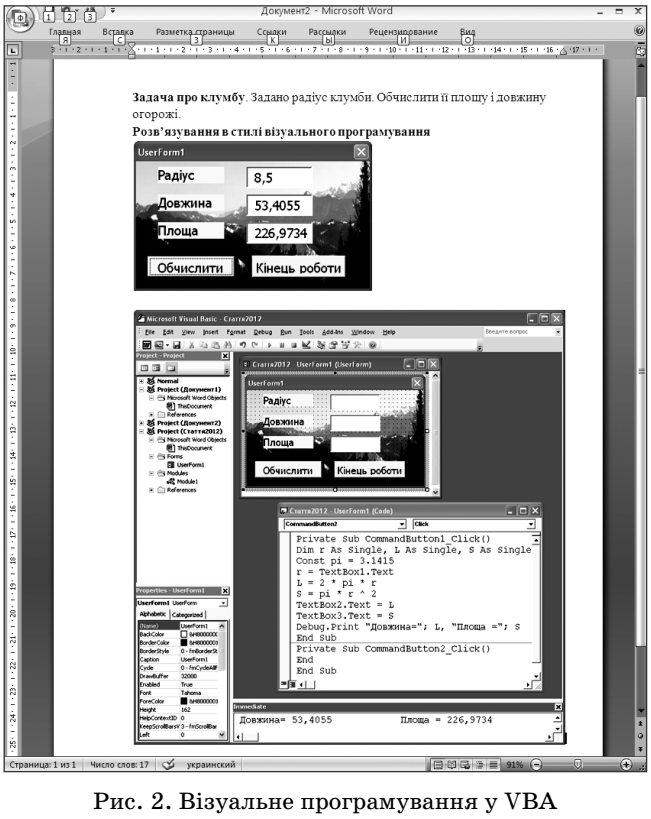

Рис. 2. Візуальне програмування у VBA

Висновок. Ми маємо змогу обчислювати результати для різних вхідних значень без перезапускання проекту. Потрібні результати отримують не в момент запуску проекту, а після настання події клацання на кнопці. Такий спосіб програмування називають візуальним або подійно-орієнтованим програмуванням. У його основі лежить як процедурна, так і об'єктна парадигми програмування.

Спосіб 3. Розв'яжемо задачу про клумбу способом об'єктного програмування. Запишемо в MS Word умову задачі і перейдемо в середовище VBA. Вставимо за допомогою головного меню в проект модуль класу **Class1**, який відразу перейменуємо на **klumba**. Складемо код класу, використовуючи поняття інкапсуляції, тобто створимо клас, що має стале поле pi, поле-властивість **r** (радіус) і два поля-методи: **pl** (для обчислення площі круга) і **dov** (для обчислення довжини кола). Вставимо модуль **Module1** і запишемо головний код проекту, де командою **Dim k As klumba** спочатку оголосимо про наміри створити об'єкт **k** класу **klumba**, а командою **Set k = New klumba** реалізуємо намір створити об'єкт **k**. Доступ до полів об'єкта дають складені імена **k.r, k.pl(), k.dov()**. Виконаємо проект і отримаємо результати (рис. 3).

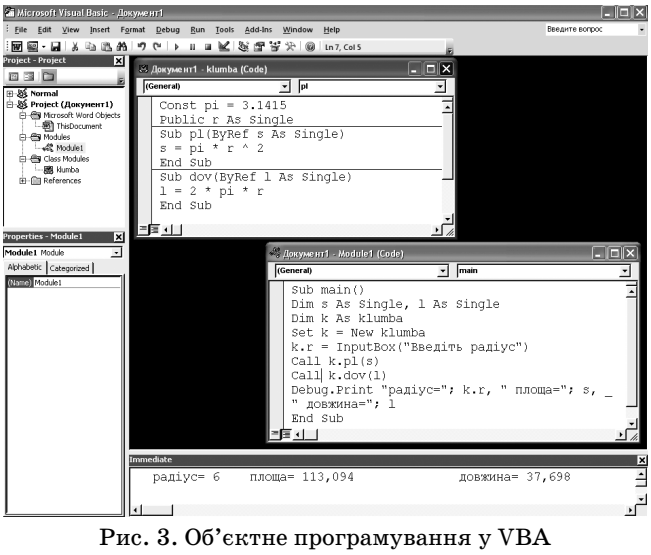

Рис. 3. Об'єктне програмування у VBA

Висновок. Ми продемонстрували спосіб розв'язування навчальної задачі із застосуванням елементів об'єктного програмування, формально створивши клас, що володіє двома методами, які дають розв'язок задачі. Головний код не містить команд перетворення вхідних даних. Відразу зауважимо, що можливості повноцінного об'єктного програмування у VBA дещо обмежені, тому VBA не можна рекомендувати для професійного вивчення парадигми об'єктно-орієнтованого програмування. Зазначимо, що цих недоліків не мають сучасні версії мови, наприклад, Visual Basic 2010 та ін.

Розглянемо похідні від основних, але достатньо незалежні й корисні способи програмування.

Спосіб 4. Цей спосіб програмування можна назвати «негайним» за назвою вікна **Immediate**, у якому він реалізується.

Запишемо в MS Word умову задачі, перейдемо в середовище VBA і відкриємо консольне вікно **Immediate** командами **View\Immediate Window**. Уведемо в ньому код розв'язування задачі і після введення кожної команди **Print** відразу ж отримуватимемо результати без запуску коду на виконання (рис. 4).

Зауважимо, що таке програмування має обмежені можливості. Можна складати найпростіші лінійні, розгалужені та циклічні коди. Тут не можна застосовувати команди **Dim**, масиви, складені команди (наприклад «розгалуження» в «циклі») тощо. Програмування підтримується однопрохідним інтерпретатором, де виконання коду відбувається рядок за рядком відразу після введення рядка, тобто після натискання на клавішу **Enter**.

Але в якому ще іншому середовищі можна так просто розв'язати задачу табулювання декількох функції з виведенням таблиці результатів на екран, як це показано далі? Розглянемо задачу: вивести таблицю значень ра-

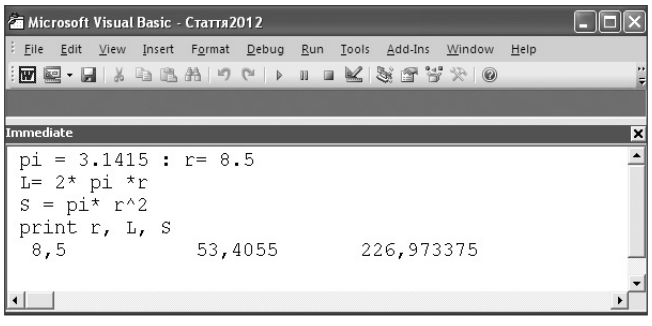

Рис. 4. «Негайне» програмування у VBA

діуса, довжини кола і площі круга для значень радіуса від 1 до 8,5 з кроком 0,5. Код, який треба ввести у вікні **Immediate** для розв'язування задачі, такий:

```
pi = 3.1415
For r = 1 To 8.5 Step 0.5 : _
Print r, 2*pi*r, pi * r ^ 2 : _
Next r
```

Висновок. «Негайне» програмування — це спрощений, але продуктивний різновид процедурного програмування, що може бути корисним для навчання початківців, оскільки дає змогу швидко отримати результати.

Спосіб 5. Розглянемо спосіб програмування безпосередньо в документі MS Word із застосуванням візуальних елементів.

Для цього на стрічку MS Word 2007 спочатку додамо закладку **Разработчик**, відкриємо режим конструктора, панель елементів керування Visual Basic (тут елементи керування називаються елементами ActiveX). Вставимо елементи класів «написи», «текстові поля» і «кнопки» безпосередньо в документ під умовою задачі. Двічі клацнемо на кнопці **Обчислити** і в заготовку процедури реакції на подію «клік» уведемо той же код, що розглядався раніше. Закриємо режим конструктора, уведемо в перше текстове поле значення радіуса 8,5 і після клацання на кнопці **Обчислити** отримаємо результати у двох інших полях (рис. 5).

Висновок. Цей різновид візуального програмування дає змогу максимально сконцентрувати увагу на роботі з офісним документом і реалізувати програмування у фоновому режимі.

Спосіб 6. Розглянемо об'єктне програмування, що базується на використанні об'єктних моделей офісних документів, бібліотек класів і колекцій об'єктів, що і є суттю VBA-програмування. До цього часу ми не застосовували всіх можливостей VBA, оскільки нашою головною метою було продемонструвати зручність

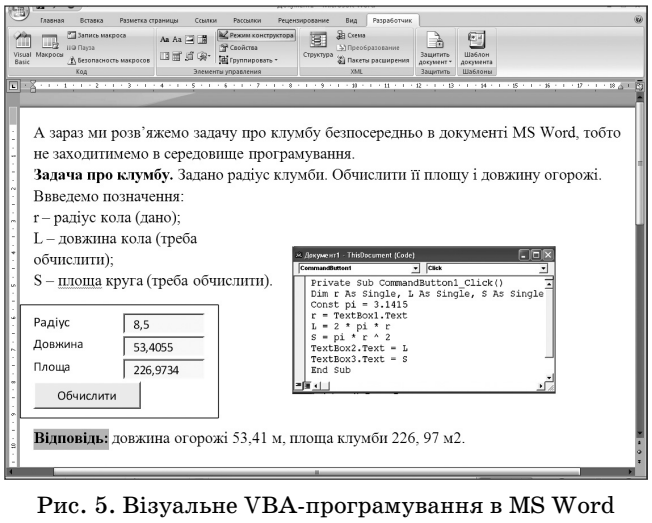

Рис. 5. Візуальне VBA-програмування в MS Word

середовища VBA для реалізації традиційних алгоритмів і програм, які зазвичай, реалізують в середовищах TP7 чи Delphi так само чи дещо складніше.

Відповідна задача без коду розглядалася в [1]. Нагадаємо, що у Word-документ треба вставити три 3x3 таблиці і скласти код кнопки, який заповнюватиме дві таблиці A і B випадковими цілими числами від 0 до 10, а третю таблицю — за правилом C=A+B.

На рис. 6 подано таблиці і об'єктний VBA-код кнопки **Пуск**. Пояснення до коду та інші зразки «чистого» VBA-програмування наведено в [2].

Висновок. Цей спосіб програмування суттєво використовує об'єктні моделі офісних програм. Таке VBA-програмування входить до програм навчання студентів економічних напрямів підготовки, а в загальноосвітніх школах може вивчатися факультативно.

Спосіб 7. Розглянемо спосіб програмування у VBA з використанням макросів. Макрос — це деякий VBA-код, виконання якого ініціює користувач за допомогою команд **Макроси\Виконати**, або деякої комбінації клавіш, або кнопки на панелі керування або деякого візуального елемента керування, заздалегідь вставленого у документ. Є два способи створення макросів: автоматичний і ручний. Для автоматичного створення макроса не потрібно жодних знань з програмування, оскільки макрос створює середовище VBA самостійно, відстежуючи всі дії користувача після виконання команд **Макроси\Записати макрос**. Код макроса можна переглянути в модулі **NewMacros**, який належить шаблону **Normal** документа. Зазвичай він є зразком об'єктного програмування з використанням об'єктної моделі офісної програми і може бути корисним для демонстраційного навчання. Макрос можна створити вручну, оскільки будь-якій процедурі можна надати статус макроса, скопіювавши її в модуль **NewMacros**, чи створивши таку процедуру командами **Макроси\Створити**.

Висновки. Макроси використовують для автоматизації робіт в офісних документах. Використання

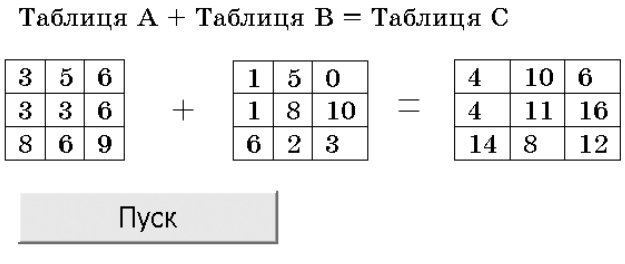

Рис. 6.1.

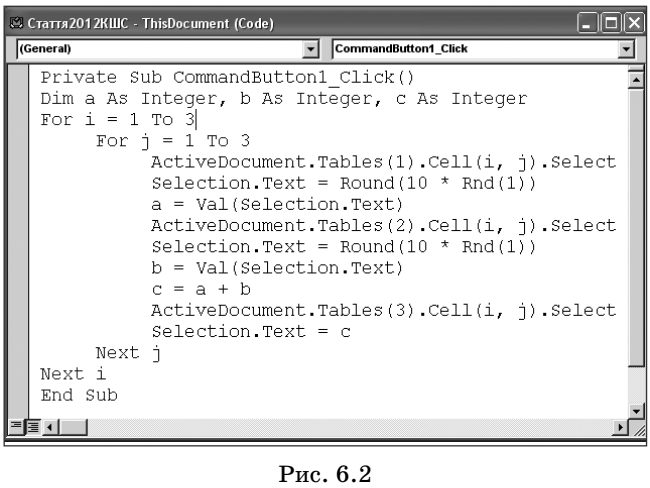

Рис. 6.2

їх для незалежного від документа програмування сенсу немає, оскільки для цього у VBA є зручніші засоби, які були розглянуті вище.

Зауваження. Наміри початківців програмувати у VBA будь-яким способом часто зазнають невдачі після отримання повідомлення про неможливість роботи з макросами «macros disable» під час першої спроби запустити код на виконання. Ця проблема розв'язується так: у випадку роботи з давнішими версіями офісних програмах потрібно понизити рівень безпеки комп'ютера до середнього чи низького, а в нових версіях треба виконати такий алгоритм: слід послідовно натиснути головну офісну кнопку, кнопки **Параметри Word**, **Центр керування безпекою**, **Параметри центру керування безпекою**, **Параметри макросів**, увімкнути радіокнопку **Увімкнути всі макроси**, а також перемикач **Довіряти доступ до об'єктної моделі проектів VBA**. Документ треба закрити зі збереженням і відкрити його повторно.

Висновки. Погоджуючись з аргументами про важливість навчання об'єктно-орієнтованому програмуванню [3], ми вважаємо, що шлях до нього (як це видно з рис. 3) пролягає через процедурне програмування, яке є основною всіх інших, розглянутих тут, способів. Візуальне програмування — це засіб створення графічного інтерфейсу до програм користувача, що на стадії навчання учнів і студентів є вагомим мотивуючим фактором, оскільки розробка сценаріїв проектів і конструювання форми дає змогу реалізувати їхні креативні задуми і є цікавим видом діяльності, що компенсує незацікавлення чи небажання займатися процедурним програмуванням у консольних вікнах, а також може слугувати вступом до об'єктного програмування. Візуальне програмування не вважають парадигмою програмування. Ми показали, що це є спосіб програмування, що базується на двох парадигмах. За допомогою описаних способів у VBA можна реалізувати всі типові алгоритми, які розглядаються в курсах базової інформатики в загальноосвітній і вищих школах, причому це можна зробити швидко, ефективно і наочно, обравши спосіб програмування відповідно до мети навчання й умови задачі. Ми рекомендуємо застосовувати перші два способи: процедурне і візуальне програмування. Інші способи мають специфічні застосування. У посібнику [2] достатньо повно описано принципи розв'язування задач від моделювання до програмування різними способами у VBA і паралельно у середовищі Visual Basic 2010. Акцентуємо на необхідності перейти до нової моделі навчання, суть якої полягає у відмові від використання недоступних середньому учневі задач, наприклад, про НСД, Ханойські вежі, впорядкування масивів та інших на користь більш доступних і зрозумілих задач. Слід вважати моделювання й алгоритмізацію найважливішими етапами навчання, а етап програмування розглядати як засіб реалізації алгоритму і взаємодії користувача з комп'ютером з метою проведення експериментів для візуалізації результатів і прийняття певних рішень. Ми хочемо надати новий імпульс для створення ефективних методик масового, але диференційованого, навчання алгоритмізації і програмування, щоб довести гіпотезу про те, що основи алгоритмізації і програмування варто вивчати всім учням загальноосвітніх шкіл і студентам загальнотехнічних напрямів підготовки і заперечити гіпотезу щодо доцільності зосереджувати таке навчання тільки в рамках профільної підготовки в середній і фахової підготовки у вищій школах.

Для підтримки нашої гіпотези і методики навчання створено не лише навчальний посібник [2], але й серію навчальних відеофільмів.

Останнім часом усе частіше спостерігається невміння, а часто просто небажання, учнів і студентів, працювати з підручниками і навчальними посібниками з інформатики чи з інших дисциплін. Причин цього явища є три: 1) втрата учнями шкіл навичок працювати з підручниками; 2) невдалий, громіздкий чи нецікавий виклад матеріалу у підручниках; 3) бурхливий розвиток сучасних інформаційно-комунікаційних технологій і різних альтернативних засобів і способів навчання. Одним із таких засобів є вільнодоступний в інтернеті медіаконтент практично на будь-яку тему навчання, який лежить в основі нового способу навчання — мобільного. Мобільне навчання є різновидом дистанційного [4, 5] і реалізується за допомогою сучасних технічних засобів (нетбуків, планшетів, смартфонів тощо) і відповідного навчального програмного забезпечення, яке має задовольняти дещо іншим вимогам, ніж традиційні електронні засоби, реалізовані у вигляді інтерактивних електронних підручників і посібників, систем тестування тощо. Одним з елементів наповнення контенту для мобільного навчання є короткі навчальні відеофільми, які динамічно розкривають центральні теми курсів чи окремі питання і які можна переглядати багаторазово у зручний час з довільного місця перебування (мобільність у часі і просторі). Деякі такі відеофільми на тему вивчення алгоритмізації і програмування в середній і вищій школах створено, розміщено для вільного доступу на каналі hlynsky1 [6] відеосервісу **YouTube** і апробовано в навчальному процесі. Результати апробації дають змогу стверджувати, що отримано значний педагогічний ефект.

Комп'ютерні відеофільми є ефективною формою демонстраційного начання і важливим елементом наповнення сучасного електронного навчально-методичного комплексу. Вони слугують доповнювальним засобом до твердої копії навчального посібника (підручника), що веде до зростання ролі класичного підручника і до перегляду способів створення таких підручників. Відеофільми дають змогу вчителеві зосередитися на питаннях фундаменталізації навчання, оскільки вивільняють його від висвітлення прагматичних питань, де без застосування лексики «клік мишею», «клік на кнопці» обійтися неможливо.

Література

1. *Глинський Я.М.* Чому ми вибираємо Visual Basic / Глинський Я.М., Ряжська В.А. // Комп'ютер у школі та сім'ї. — 2011. — №8. — С. 9–12.
2. *Глинський Я.М.* Інформатика. Основи алгоритмізації і програмування мовою Visual Basic: навч. посібн. — Львів: СПД Глинський, 2011. — 272 с.
3. *Шевчук П.Г.* Навчання об'єктно-орієнтованому програмуванню в загальноосвітних навчальних закладах засобами мови C# // Теорія та методика навчання математики, фізики, інформатики: збірник наукових праць. — Вип. IX. — Кривий Ріг : Вид. відділ НметАУ, 2011. — С. 595–601.
4. *Куклев В. А.* Становление системы мобильного обучения в открытом дистанционном образовании [Електронний ресурс]. — Режим доступу: http://www.conf.muh.ru/091012/thesis_Kuklev.htm.
5. *Поліщук О.П.* Сучасні тенденції розвитку теорії і методики навчання інформатики / Поліщук О.П., Теплицький І.О., Семеріков І.О. // Теорія та методика навчання математики, фізики, інформатики: збірник наукових праць. — Вип. IX. — Кривий Ріг: Вид. відділ НметАУ, 2011. — С. 509–514.
6. [Електронний ресурс]. — Режим доступу: http://www.youtube.com/hlynsky1.