Материалы для подготовки по теме объектно-ориентированное программирование по учебнику Н. Угриновича «Информатика и ИКТ 9 класс»

Материалы подготовлены учителем информатики: Белкиным В.В.

2.6. Кодирование алгоритмических структур основных типов на языке программирования Visual Basic

2.6.1. Линейный алгоритм

Существует большое количество алгоритмов (например, рассмотренные выше событийные процедуры), в которых команды должны быть выполнены последовательно одна за другой. Такие последовательности команд будем называть **сериями**, а алгоритмы, состоящие из таких серий, — **линейными**.

Алгоритм, в котором команды выполняются последовательно одна за другой, называется **линейным алгоритмом**.

Для того чтобы сделать алгоритм более наглядным, часто используют **блок-схемы**.

Различные элементы алгоритма изображаются с помощью различных геометрических фигур: для обозначения начала и конца алгоритма используются прямоугольники с закругленными углами, а для обозначения последовательности команд — прямоугольники (рис. 2.20).

На блок-схеме, изображенной на рис. 2.20, хорошо видна структура линейного алгоритма, по которой исполнителю (человеку) удобно отслеживать процесс его выполнения.

Рис. 2.20. Линейный алгоритм

Задания для самостоятельного выполнения

2.16. *Задание с развернутым ответом.* Нарисовать блок-схему событийной процедуры, описанной в проекте «Переменные».

2.6.2. Алгоритмическая структура «ветвление»

В отличие от линейных алгоритмов, в которых команды выполняются последовательно одна за другой, в алгоритмическую структуру «ветвление» входит **условие**. В зависимости от выполнения или невыполнения условия реализуется одна или другая последовательность команд (серий).

В алгоритмической структуре «ветвление» в зависимости от истинности или ложности **условия** выполняется одна или другая серия команд.

В условии два числа, две строки, две переменные, два арифметических или строковых выражения сравниваются между собой с использованием операций сравнения (`>`, `<`, `=`, `>=`, `<=`). Например: `5 > 3`, `"A" = "B"` и т. д. В зависимости от результата сравнения условие принимает значение **`True`** («истина») или **`False`** («ложь»).

Алгоритмическая структура «ветвление» может быть наглядно представлена с помощью блок-схемы. В языке программирования Visual Basic ветвление кодируется с использованием оператора условного перехода **`If ... Then ... Else ... End If`** (Если ... То ... Иначе ... Конец Если).

В операторе условного перехода после первого ключевого слова **`If`** должно быть размещено условие. Второе ключевое слово **`Then`** размещается на той же строке. Во второй строке размещается последовательность команд (Серия 1), которая должна выполняться, если условие принимает значение **`True`**. На третьей строке размещается ключевое слово **`Else`**. На четвертой строке размещается последовательность команд (Серия 2), которая должна выполняться, если условие принимает значение **`False`**. На пятой строке размещается конец инструкции ветвления **`End If`** (рис. 2.21).

Блок-схема	Язык программирования Visual Basic

	`If Условие Then`
	`  Серия 1`
	`[Else`
	`  Серия 2]`
	`End If`

Рис. 2.21. Алгоритмическая структура «ветвление»

В случае отсутствия серии команд, которую необходимо выполнить при ложности условия, используется сокращенная форма алгоритмической структуры «ветвление». В этом случае в операторе условного перехода отсутствует ключевое слово **Else** и, соответственно, последовательность команд «Серия 2». Тогда, если условие ложно, выполнение оператора условного перехода заканчивается и выполняется следующая строка программы.

Проект «Тест». Разработать проект, который использует алгоритмическую структуру «ветвление» для контроля знаний. Алгоритм контроля знаний должен последовательно реализовать следующие операции:

1) задать (вывести) вопрос и запросить ответ;
2) запомнить введенную с клавиатуры последовательность символов;
3) сравнить ответ, введенный с клавиатуры, с правильным ответом и в случае истинности результата сравнения вывести сообщение «Правильно», в случае ложности результата — сообщение «Неправильно».

В языке программирования Visual Basic задать вопрос и запросить ответ можно с помощью функции `InputBox()`, которая позволяет вводить данные с помощью диалогового окна ввода.

2.5.3. Функции ввода и вывода данных

Запомнить ответ пользователя можно путем присваивания переменной `Otvet` строки символов, введенной пользователем в текстовом поле окна ввода.

Проверку правильности введенного ответа и печать оценочных сообщений реализуем с помощью алгоритмической структуры «ветвление». В качестве условия используем сравнение значения переменной `Otvet` со строкой правильного ответа.

Полезно предусмотреть возможность подсчета количества сделанных ошибок с помощью счетчика ошибок (целочисленной переменной `N`). При каждом неправильном ответе необходимо к значению счетчика прибавлять единицу, для этого в последовательность команд, которая выполняется, если ответ неправильный, следует добавить: `N = N + 1`.

Создадим графический интерфейс проекта (рис. 2.23).

1. Поместить на форму:
   - надпись `Label1` для вывода оценочных сообщений;
   - надпись `Label2` для вывода количества ошибок;
   - кнопку `Button1` для запуска событийной процедуры.
2. Код событийной процедуры будет следующим:

```
Dim Otvet As String, N As Byte
Private Sub Button1_Click(…)
N = 0
Otvet = InputBox("Чему равен 1 байт?",
  "Первый вопрос")
If Otvet = "8 бит" Then
Label1.Text = "Правильно!"
Else
Label1.Text = "Неправильно!": N = N + 1
End If
Label2.Text = N
End Sub
```

3. Запустить проект и щелкнуть по кнопке *Начать проверку*. В появившемся диалоговом окне функции `InputBox()` ввести в текстовое поле ответ на вопрос и щелкнуть по кнопке *OK* (см. рис. 2.22).

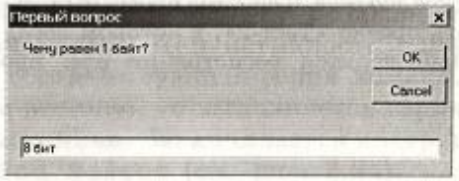

Рис. 2.22. Первый вопрос и ввод ответа в окне ввода функции `InputBox()`

4. На первую надпись будет выведено оценочное сообщение, а на вторую надпись — количество ошибок (см. рис. 2.23).

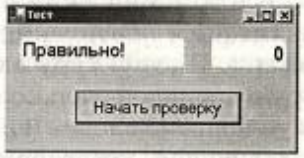

Рис. 2.23. Вывод результатов теста на надписи

Проект «Тест» хранится в папке ..\Informatika9\Тест — Windows-CD

В проекте «Тест» можно задать несколько вопросов. Для этого необходимо добавить в программный код событийной процедуры аналогичную последовательность команд, изменив первый аргумент в функции `InputBox()` и правильный ответ в условии ветвления оператора **If-Then-Else-End If**.

Контрольные вопросы

1. В каком случае в алгоритмической структуре «ветвление» выполняется последовательность команд «Серия 1»? Последовательность команд «Серия 2»?
2. В каком случае можно использовать сокращенную форму алгоритмической структуры «ветвление»?

Задания для самостоятельного выполнения

Windows-CD

2.17. Задание с развернутым ответом. Начертить блок-схему алгоритмической структуры «ветвление».

2.18. *Практическое задание.* В системе программирования Visual Basic 2005 создать проект «Тест», состоящий из нескольких вопросов и содержащий счетчик ошибок.

2.19. *Практическое задание.* В системе программирования Visual Basic 2005 создать проект «Тест-2», в котором перед началом тестирования производится регистрация (см. проект «Регистрация»).

2.6.3. Алгоритмическая структура «выбор»

Алгоритмическая структура «выбор» применяется для реализации ветвлений со многими вариантами серий команд. В структуру выбора входят несколько условий, которые последовательно проверяются. При истинности одного из условий (Условие 1, Условие 2 и т. д.) выполняется соответствующая последовательность команд (Серия 1, Серия 2 и т. д.). Если ни одно из условий не будет истинно, то будет выполнена последовательность команд «Серия».

В алгоритмической структуре **«выбор»** выполняется одна из нескольких последовательностей команд при истинности соответствующего **условия**.

Алгоритмическая структура «выбор» может быть наглядно представлена с помощью блок-схемы (рис. 2.24).

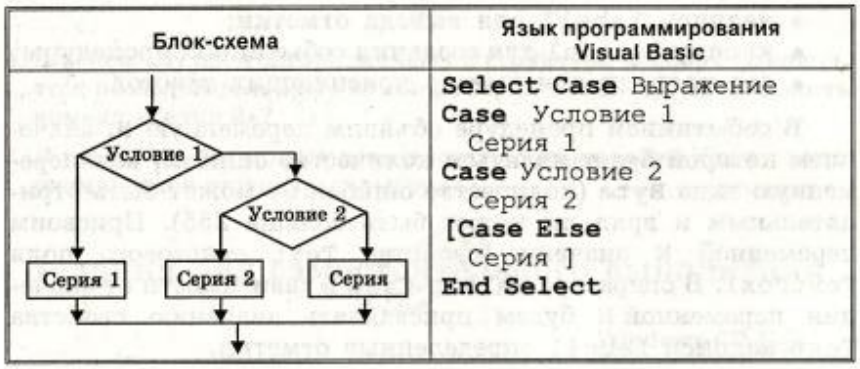

Рис. 2.24. Алгоритмическая структура «выбор»

В языке программирования Visual Basic инструкция выбора начинается с ключевых слов **Select Case**, после которых записывается переменная или выражение. После ключевых слов **Case** записываются условия, в которых заданная переменная или выражение сравнивается с определенными значениями. При истинности одного из условий выполняется соответствующая серия команд. Если ни одно из условий не истинно, то выполняется серия команд после ключевого слова **Else**. Заканчивается инструкция ключевыми словами **End Select** (см. рис. 2.24).

В случае отсутствия серии команд, которую необходимо выполнить при ложности всех условий, используется сокращенная форма алгоритмической структуры «выбор». В этом случае в операторе выбора отсутствуют ключевые слова **Case Else** и, соответственно, последовательность команд «Серия». Тогда, если все условия ложны, выполнение оператора выбора заканчивается и выполняется следующая строка программы.

Проект «Отметка». В качестве примера использования инструкции выбора разработать проект, который позволяет выставлять отметку за работу в зависимости от количества сделанных ошибок.

Создадим графический интерфейс проекта (рис. 2.25).

1. Поместить на форму:
   - текстовое поле TextBox1 для ввода количества ошибок;
   - надпись Label1 для вывода отметки;
   - кнопку Button1 для создания событийной процедуры;
   - две надписи для вывода поясняющих текстов.

В событийной процедуре объявим переменную N, значением которой будет являться количество ошибок, как переменную типа **Byte** (количество ошибок не может быть отрицательным и вряд ли может быть больше 255). Присвоим переменной N значение свойства Text текстового поля TextBox1. В операторе **Select Case** в зависимости от значения переменной N будем присваивать значению свойства Text надписи Label1 определенные отметки.

2.
```
Dim N As Byte
Private Sub Button1_Click(…)
N = TextBox1.Text
Select Case N
Case 0
Label1.Text = "Отлично"
Case 1
Label1.Text = "Хорошо"
Case 2
Label1.Text = "Удовлетворительно"
Case Else
Label1.Text = "Плохо"
End Select
End Sub
```

3. После запуска проекта на выполнение необходимо ввести в текстовое поле количество ошибок и щелкнуть по кнопке *Отметка*, на надпись будет выведена соответствующая отметка (см. рис. 2.25).

Рис. 2.25. Проект «Отметка»

Проект «Отметка» хранится в папке ..\informatika9\Отметка\ Windows-CD

Контрольные вопросы

1. В каком случае в алгоритмической структуре «выбор» выполняется последовательность команд «Серия 1»? Последовательность команд «Серия 2»?
2. В каком случае можно использовать сокращенную форму алгоритмической структуры «выбор»?

Задания для самостоятельного выполнения

Windows-CD

2.20. *Задание с развернутым ответом.* Начертить блок-схему алгоритмической структуры «выбор».

2.21. *Практическое задание.* В системе программирования Visual Basic 2005 создать проект «Отметка».

2.22. *Практическое задание.* В системе программирования Visual Basic 2005 создать проект «Тест с отметкой», в котором после тестирования (см. проект «Тест2») на надпись выводится отметка с учетом количества сделанных ошибок.

2.6.4. Алгоритмическая структура «цикл»

В алгоритмическую структуру «цикл» входит серия команд, выполняемая многократно. Такая последовательность команд называется **телом цикла**.

Циклические алгоритмические структуры бывают двух типов:

- **цикл со счетчиком**, в котором тело цикла выполняется определенное количество раз;
- **цикл по условию**, в котором тело цикла выполняется, пока истинно условие.

В алгоритмической структуре «цикл» серия команд (тело цикла) выполняется многократно.

Цикл со счетчиком. Алгоритмическая структура «цикл со счетчиком» используется, если известно заранее, какое число повторений тела цикла необходимо выполнить. Цикл

со счетчиком может быть зафиксирован графически, с помощью блок-схемы, а также записан на языке программирования Visual Basic с использованием оператора цикла **For ... Next** (рис. 2.26).

Блок-схема	Язык программирования Visual Basic

	`For Счетчик=НачЗнач`
	`  To КонЗнач [Step шаг]`
	`  Тело цикла`
	`Next [Счетчик]`

Рис. 2.26. Алгоритмическая структура «цикл со счетчиком»

Синтаксис оператора **For...Next** следующий: строка, начинающаяся с ключевого слова **For**, является заголовком цикла, а строка с ключевым словом **Next** — концом цикла, между ними располагаются операторы, являющиеся телом цикла.

В начале выполнения цикла значение переменной `Счетчик` устанавливается равным `НачЗнач`. При каждом «проходе» цикла переменная `Счетчик` увеличивается на величину шага. Если она достигает величины `КонЗнач`, то цикл завершается и выполняются следующие за ним операторы.

Проект «Коды символов». В качестве примера использования цикла со счетчиком создать проект, который должен выводить в поля списка числовые коды символов и соответствующие им символы.

3.1. Кодирование текстовой информации Информатика-8

Создадим графический интерфейс проекта (рис. 2.27).

1. Разместить на форме:
   - поле списка `ListBox1` для вывода числовых кодов символов;
   - поле списка `ListBox2` для вывода символов;
   - кнопку `Button1` для создания событийной процедуры.

Создадим событийную процедуру, в которой в качестве счетчика цикла используем целочисленную переменную `N`. В кодировке *Windows* первые 33 кода (десятичные коды с 0 по 32) соответствуют не знакам, а клавишам клавиатуры (клавишам управления курсором, клавишам «Пробел», «Ввод» и др.). Поэтому воспользуемся циклом со счетчиком с шагом −1, для того чтобы вывести на форму символы, начиная с символа с наибольшим числовым кодом 255.

Для преобразования числового кода в символ используем функцию `Chr()`, аргументом которой является число, а значением — символ. В теле цикла числовые коды символов и соответствующие им символы будут выводиться в поля списков с помощью метода `Items.Add()`.

2.
```
Dim N As Integer
Sub Button1_Click(…)
For N = 255 To 33 Step -1
ListBox1.Items.Add(N)
ListBox2.Items.Add(Chr(N))
Next N
End Sub
```

3. После запуска проекта на выполнение необходимо щелкнуть по кнопке *Пуск*. В полях списка будут напечатаны последовательности числовых кодов символов и соответствующих им символов. С помощью полос прокрутки можно ознакомиться со всеми кодами и их символами (см. рис. 2.27).

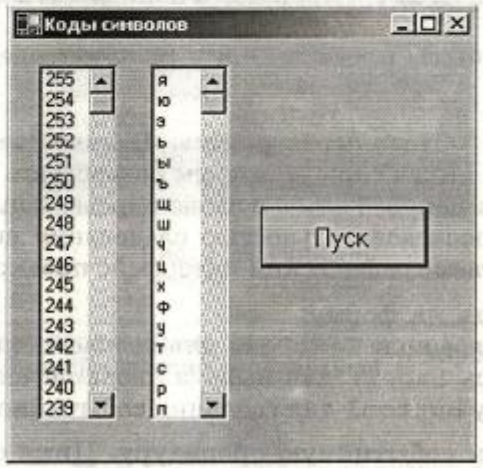

Рис. 2.27. Проект «Коды символов»

Проект «Коды символов» хранится в папке ..\informatika9\Коды символов **Windows-CD**

***Цикл с условием.** Алгоритмическая структура «цикл с условием» используется, если заранее неизвестно, какое количество раз необходимо повторить тело цикла. В этом случае количество повторений тела цикла зависит от истинности условия. Цикл с условием можно отобразить с помощью блок-схемы и записать на языке программирования Visual Basic с помощью инструкции **Do While...Loop**.

После ключевого слова **While** записывается условие продолжения цикла. Цикл выполняется до тех пор, пока выполняется условие, т. е. пока условие имеет значение «истина». Как только условие примет значение «ложь», выполнение цикла закончится. Если условие продолжения цикла стоит перед телом цикла, то такой цикл называется **циклом с предусловием** (рис. 2.28).

Блок-схема	Язык программирования Visual Basic
	**Do While** Условие Тело цикла **Loop**

Рис. 2.28. Алгоритмическая структура «цикл с предусловием»

***Проект «Слово-перевертыш».** В качестве примера использования цикла с предусловием разработать проект преобразования введенного слова в слово-перевертыш, т. е. в слово с обратной последовательностью следования символов.

Разработаем графический интерфейс проекта (рис. 2.29).

1. Поместить на форму:
   - текстовое поле `TextBox1` для ввода исходного слова;
   - надпись `Label1` для вывода слова-перевертыша;
   - кнопку `Button1` для создания событийной процедуры.

Создадим событийную процедуру. Цикл с предусловием будет выполняться, пока справедливо условие `N <= Len(TextBox1.Text)`, т. е. пока значение переменной `N` меньше или равно количеству символов в слове. (Количество символов во введенном слове является значением строковой функции `Len()`.)

В цикле символы последовательно вырезаются из введенного слова в прямой последовательности (слева направо) с использованием функции вырезания подстроки из строки `Mid(TextBox1.Text, N, 1)` и присваиваются строковой переменной `S`. Затем вырезанные символы (значения переменной `S`) в обратной последовательности (справа налево) присваиваются свойству `Label1.Text`, значением которого после завершения цикла будет слово-перевертыш.

2.5.2. Строковые функции

2.
```
Dim S As String, N As Byte
Private Sub Button1_Click()
Label1.Text = ""
N = 1
Do While N <= Len(TextBox1.Text)
S = Mid(TextBox1.Text, N, 1)
Label1.Text = S + Label1.Text
N = N + 1
Loop
End Sub
```

3. После запуска проекта на выполнение необходимо ввести в текстовое поле исходное слово и щелкнуть по кнопке *Пуск*. На надпись будет выведено слово-перевертыш (см. рис. 2.29).

Рис. 2.29. Проект «Слово-перевертыш»

Проект «Слово-перевертыш» хранится в папке ..\informatika9\Слово-перевертыш **Windows-CD**

Контрольные вопросы

1. В каких случаях используется алгоритмическая структура «цикл со счетчиком», а в каких случаях алгоритмическая структура «цикл с условием»?