Министерство Высшего и среднего специального образования
Республики Узбекистан
Самаркандский государственный университет
кафедра информационных технологий

КУРСОВАЯ РАБОТА

ОРГАНИЗАЦИЯ ОБРАБОТКИ ФАЙЛОВ В СИСТЕМЕ VISUAL BASIC

Студент 2-го курса: Адилов Адхам

Научный руководитель: Ахатов А.Р.

САМАРКАНД – 2012

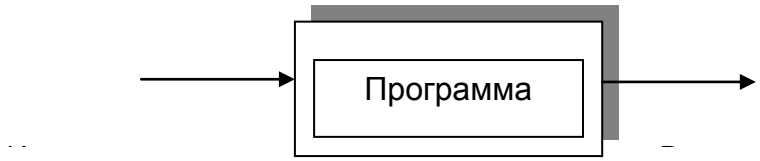
С О Д Е Р Ж А Н И Е

ВВЕДЕНИЕ

В настоящей работе обсуждаются вопросы формирования и организации обработки файлов на основе проектирования констант собственных и констант Visual Basic, а также переменных. Обсуждаются способы объявления переменных: явный, неявный, с помощью суффикса, рассматривается вопрос, как настроить среду VB, чтобы требовалось явное объявление переменных. В работе создаются на проверку два проекта. В первом проекте на форме распечатывается стихотворение, второй – калькулятор с четырьмя арифметическими действиями.

1. Данные в *Visual Basic.* Типы данных

Любая программа, которую вы составляете, нацелена на обработку каких-либо данных. Данные поступают в программу в своем первоначальном виде (исходные данные) и в обработанном виде «выходят» из программы (выходные данные).



Данные могут быть постоянными величинами и переменными величинами. Постоянные величины или константы – это такие величины, которые в процессе выполнения программы не изменяют своих значений. Переменные же величины в процессе выполнения программы свои значения изменяют.

В языке *Visual Basic,* как и в любом другом языке, постоянные и переменные величины должны быть отнесены к какому-либо типу данных. В каждом языке существует набор стандартных типов данных. В *Visual Basic* кроме этого можно объявить свой собственный тип данных как комбинацию стандартных и отнести данные к этому типу.

Стандартные типы данных, поддерживаемые языком *Visual Basic,* приведены в таблице.

В таблице представлены различные типы данных, использующиеся в *Visual Basic*. Понимание того, как связаны тип данных и занимаемая ими память важно для оптимизации кода. Везде, где это возможно, нужно использовать тип данных, требующий меньший объем памяти, чтобы

сэкономить системные ресурсы. Но сначала нужно научиться создавать и использовать постоянные и переменные величины.

Типы данных *Visual Basic.*

Тип данных	**Объем занимаемой памяти**	**Краткая запись**
Целые типы		
Integer (целое)	**2 байта**	**%**
Long (целое двойной длины)	**4 байта**	**&**
Byte (Байт)	**1 байт**	
Boolean (булево)	**2 байта**	
Плавающие типы		
Single Десятичные числа	**4 байта**	**!**
Double Десятичные числа	**8 байт**	**#**
Строковые типы		
String Текстовая информация	**1 байт на каждый символ**	**$**
Объектные типы		
Object Рисунки или ссылки на любой другой объект	**4 байта**	
Variant-типы		
Variant (числовые типы)	**16 байт**	
Variant (строковые типы)	**22 байта +длина строки**	
Прочие типы		
Currency Число в денежном формате	**8 байт**	**@**
Date Дата	**8 байт**	

2. Константы *Visual Basic*

Чаще всего константы используются для значений, которые трудно запомнить. *Visual Basic* предлагает целый ряд констант. Все эти константы имеют префикс *vb.* Например, константа *vbActiveTitleBar* – цвет панели

заголовка *Windows* имеет значение 2147483646. Конечно проще запомнить имя константы, чем ее значение. Или константа *vbGreen* (зеленый цвет) имеет значение 65280.

Если в программе используются константы, чтение такой программы облегчается. Можно, например, использовать константы *Visual Basic* для дней недели: *vbSunday, vbMonday*, и т. д., а не их значения 1, 2, и. т. д. Когда в программе встречаются анонимные числа, человеку, читающему текст программы, приходится с раздражением гадать, откуда они взялись. Если же вместо них он увидит имя константы, которое несет в себе смысловую нагрузку, программа становится понятнее, и чувство раздражения сменяется на чувство удовлетворения от понимания идеи автора программы.

В следующем разделе урока попробуем сделать маленький тренировочный проект с использованием некоторых констант, предлагаемых *Visual Basic.*

3. Проект *prjConstVB*

Цель этого раздела – научиться использовать константы *Visual Basic* в своих программах. На форму, которую назовем *frmDays*, поместим одну командную кнопку *Command1*, после щелчка по которой мышкой пользователь на форме увидит распечатанные названия дней недели и соответствующие им значения констант дней недели *Visual Basic* разными цветами. При этом будут использоваться константы цветов *Visual Basic.*

В следующих таблицах будут приведены имена и значения констант *Visual Basic*, которые вы должны использовать в проекте.

Таблица цветовых констант *Visual Basic*

Имя константы	Значение
VbBlack	*0*
VbRed	*255*
VbGreen	*65280*
VbYellow	*65535*
VbBlue	*16711680*
VbMagenta	*16711935*
VbCyan	*16776960*
VbWhite	*16777215*
VbButtonFace	*-2147483633*

Таблица констант дней недели *Visual Basic*

Имя константы	Значение
VbSunday	***1***
VbMonday	***2***
VbTuesday	***3***
VbWednesday	***4***
VbThursday	***5***
VbFriday	***6***
VbSaturday	***7***

Еще используем в проекте константу *Visual Basic vbTab,* с помощью которой будем делать промежутки между названием дня недели и значением соответствующей константы.

Код разместим в процедуре командной кнопки:

```
Private Sub Command1_Click( )
  BackColor = vbButtonFace
                                  ‘свойству BackColor формы
                                  ‘присвоим значение
                                  константы ‘vbButtonFace
   ForeColor = vbWhite
                                  ‘свойству ForeColor формы
                                  ‘присвоим значение
                                  константы ‘vbWhite

  Print “Воскресенье”; vbTab; vbTab ; vbSunday
                                  ‘печатаем слово
                                  <Воскресенье> ‘и через два
                                  промежутка ‘значение
                                  константы vbSunday. ‘Этот
                                  ‘меняем значение свойства
ForeColor = vbYellow              ‘ForeColor на vbYellow

  Print “Понедельник” ; vbTab; vbTab; vbMonday
```

Для того, чтобы проект работал для всех дней недели нужно дописать код программы так, чтобы каждый следующий день печатался на форме новым цветом. Необходимо использовать константы *Visual Basic*, приведенные в таблицах. После этого проект запускается на компьютере.

4. Создание своих собственных констант

Часто бывает разумным вместо неоднократно повторяющихся строк или чисел использовать имя константы. В этом случае вы должны создать свою собственную константу. Для того, чтобы использовать константу, ее в программе сначала нужно определить. Определяются константы с помощью ключевого слова *Const*, которое присваивает константе имя и значение:

Const CONSTANT_NAME [As ConstantType] = значение

Таким образом нужно задать имя константы и (необязательно) тип данных, которые константа будет хранить. Ключевое слово *Const* в начале оператора сообщает *Visual Basic* о том, что этот оператор объявляет константу. При объявлении типа константы используются типы данных, приведенные в таблице в уроке 1. Завершает создание константы знак «равенства» (=) и присвоение константе значения. Если создается строка, то она заключается в кавычки.

В именах констант обычно используют только прописные буквы, соединяя слова, из которых образовано имя константы, знаком подчеркивания (_). В последнее время стало привычным для имен констант использование комбинации прописных и строчных букв и префикса, составленного из строчных букв.

Приведем пример объявления константы с именем *numSentense*

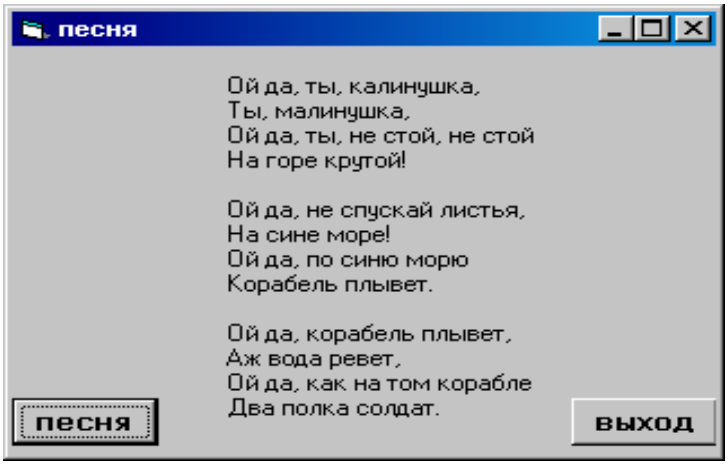
Const numSentense As String = “В лесу растут грибы”

Приведем еще один пример объявления константы:

Const CUR_SUMMA As Integer = 255

5. Проект *prjSong*

Теперь попробуем сделать небольшой проект, в котором на форме после щелчка пользователем по соответствующей командной кнопке, будут распечатываться слова песни. Текст песни можете взять любой, но в нем обязательно должна быть одна или несколько повторяющихся фраз. Можете воспользоваться следующим текстом:



Форму назовем *frmSong* (свойство *name)*, свойству *Caption* формы присвоим значение “*Песня*”.

На форму поместим две командные кнопки: одну *cmdSong*, при нажатии на которую на форме появлялись бы слова песни, и *cmdExit* для выхода из проекта. Свойству *Caption* кнопки *cmdSong* присвоим значение «*Песня»,* свойству *Caption* кнопки *cmdExit* – “*Выход*”.

Приступим к составлению кода проекта. Начнем с кода для кнопки *cmdExit.* Здесь мы будем очищать форму с помощью метода *Cls* и выходить из программы. Дополните код :

```
Private Sub cmdExit_Click( )

End Sub
```

Текст песни распечатываться будет с помощью методов *Print* и функции *Tab* в коде *cmdSong_Click( ).* Объявим в этом коде сначала константу с повторяющимся текстом с помощью строки:

```
Const strZapev As String = “Ой да, ”
```

С помощью констант *Visual Basic* зададим в этом коде свойства формы *ForeColor* и *BackColor* и, используя метод *Print* и функцию *Tab,* распечатаем строки песни, причем повторяющиеся слова будем распечатывать с помощью

объявленной ранее константы. Строка кода, распечатывающая первую строку песни, может выглядеть, например, следующим образом:

Print Tab(20); strZapev ; “ты, калинушка,”

Составив код и нужно запустить проект на компьютере.

Private Sub cmdSong_Click ( )

End Sub

6. Переменные величины. Явное объявление переменных

Переменные величины или просто переменные, также как и константы, используются для хранения некоторых необходимых программе данных. Переменная также, как и константа имеет имя. Имена переменным можно давать как простые, так и сложные. И, хотя имена переменным выбираются произвольно, существуют следующие ограничения:

-имя переменной должно начинаться с буквы, а не с цифры или другого символа;

-имя не должно содержать точки;

-длина имени не должна превышать 255 символов;

-имя должно быть уникальным в пределах данной процедуры или модуля (это ограничение зависит от области видимости данной переменной, что мы обсудим позже).

Например, нельзя использовать вот такие имена для переменных:

1Week - нельзя начинать имя переменной с цифры;
Jan.To.Dat - нельзя использовать точки;
Number One - между символами нельзя оставлять пробелы.
Следующие имена можно использовать для переменных:

NumberOne INumOne Number1

Лучше выбирать имена переменных так, чтобы они несли в себе информацию о назначении переменной, но не нужно делать их слишком длинными, т. к. в программе вам придется на них ссылаться.

Переменные получают свои значения в период выполнения программы и сохраняют их, пока им не будет присвоено новое значение. Нельзя не отметить, что в программах на *Visual Basic* свойства элементов управления тоже рассматриваются как некоторые переменные.

В переменных можно хранить практически любые данные. Переменная может содержать число, строку текста или экземпляр объекта (в том числе формы), элементы управления, элемент базы данных. В переменной можно хранить информацию любого типа, но разные типы переменных предназначены для эффективной работы с различными типами информации. Чтобы программа смогла воспользоваться переменной, ее необходимо объявить. Способов объявления переменной *Visual Basic* предлагает несколько. Сначала подробно поговорим *о явном* объявлении переменных.

При явном объявлении переменной обязательно указывается ее имя и тип. Синтаксис объявления переменной следующий:

Public / Private / Dim ИмяПеременной [As ИмяТипа]

Начинается объявление с одного из зарезервированных ключевых слов: *Public, Private* или *Dim,* которое сообщает *Visual Basic,* что вы хотите объявить переменную. Выбор конкретного зарезервированного слова зависит от того, какой вы себе представляете область видимости вашей переменной. Об этом мы поговорим немного позже.

As – ключевое слово, которое говорит *Visual Basic*, что вы определяете тип данных для этой переменной. Если вы не определяете тип данных (*Visual Basic* это допускает), ей присваивается тип *Variant*. Необходимо учитывать, что переменные типа *Variant* занимают больше места в памяти и обращение к ним происходит несколько медленнее.

В одной строке можно объявить несколько переменных, при этом, правда, следует обращать внимание на указание имени типа:

Dim NumText As Integer, b As Integer, c As Long
Private numberOne As Integer, f,d

В первой строке объявлены две переменные типа *Integer* и одна переменная типа *Long.* Во второй строке – три переменные, причем одна из них получит тип *Integer,* а две другие по умолчанию *Variant.*

Когда явно объявляете переменную определенного типа, вы даете инструкции *Visual Basic,* как «сформировать» переменную с учетом типа данных, для хранения которых она предназначена. Понимание того, как связаны тип данных переменной и занимаемая ею память, оказывается важным для оптимизации кода. Везде, где это возможно, следует использовать

переменные, требующие меньший объем памяти, чтобы сэкономить системные ресурсы.

В отличие от других языков *Visual Basic* не требует обязательного объявления переменной перед ее использованием. Если переменная не объявлена, *Visual Basic* использует тип данных, заданный по умолчанию, - это тип *Variant (*произвольный). Тип *Variant* может содержать любую информацию. Частое использование этого типа для хранения информации имеет два существенных недостатка – влечет за собой лишнюю трату ресурсов памяти и может привести к непредсказуемому поведению заданных по умолчанию значений. Поэтому все же лучше объявлять переменные перед их использованием. Это сделает код более надежным и в конечном итоге сохранит время.

7. Функции *Visual Basic Val( ) и Str( )*

Visual Basic располагает большим набором встроенных функций, предназначенных для решения разнообразных задач. Постепенно мы будем знакомиться с многими из них. Для проекта, который нам предстоит сделать на следующем уроке, нам потребуются две функции: *Val( )* и *Str( ).* Рассмотрим подробнее, как они “работают”.

Функция ***Val(Строка)*** преобразует строку цифровых символов в число, причем преобразование заканчивается на первом нецифровом символе в строке.

Функция ***Str(Число)*** преобразует числовое значение в строку.

Чтобы понять, как “работают” эти функции, сделаем маленький пример. Откроем новый проект. На форме поместим две командные кнопки: первую *cmdVal* со значением свойства *Caption - Val,* вторую *cmdStr* со значением свойства *Caption - Str.* В общей части кода, сразу под строкой *Option Explicit* объявим три переменные:

```
Dim StrIn As String, StrOut As String
Dim Num As Single
```

Код командной кнопки ***cmdVal*** сделаем следующий:

```
Private Sub cmdVal_Click()
   StrIn = “01234dfghgj”
   Num = Val(StrIn)
   Print StrIn
   Print Num
```

```
End Sub
```

В этом коде переменной ***StrIn,*** объявленной как строка символов, присваивается значение *01234hjjkk,* затем с помощью функции *Val* оно преобразовывается в число и присваивается переменной *Num.* Если теперь запустить проект и щелкнуть мышкой на кнопке *cmdVal,* на форме распечатаются значения: *01234hjjkk* и 1234. То есть функция *Val* “отсекла” нечисловые символы а числовые символы преобразовала в число.

Код командной кнопки ***cmdStr*** сделаем следующий:

```
Private Sub cmdStr_Click()
   StrOut = Str(Num)
   Print StrOut
End Sub
```

В нем мы преобразовываем число, содержащееся в ***Num***, в строку. После двух преобразований строка в переменной ***StrOut*** примет значение 1234.

8. Объявление типа переменной с помощью суффикса

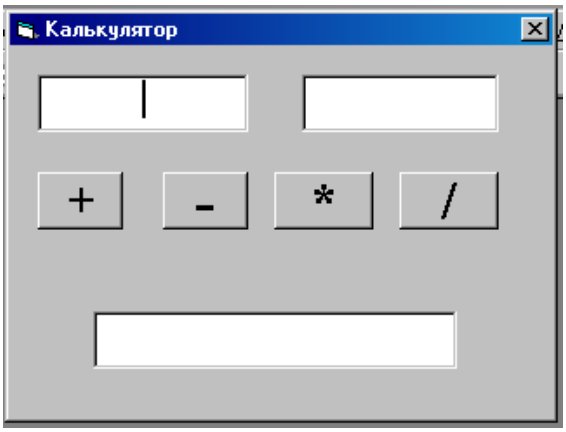
В трех предыдущих пунктах занятия при объявлении переменных мы для указания типа использовали суффикс *As.* Существует еще один способ объявления переменной и определения ее типа данных. Он использует *суффиксы типов данных.* В этом случае тип данных переменной определяется с помощью добавления в конец ее имени специального символа описания типа, который при первом присваивании значения этой переменной автоматически дает указание отнести ее к определенному типу, и поэтому использование ключевого слова *As* не требуется. Каждому типу данных сопоставлен свой символ описания типа, и все они приведены в таблице *Типы данных* в пункте занятия. Пример использования суффиксов для объявления типов данных:

```
Private NumVal%
Static CaccRes!, InputVal#
Dim InputMsg$
```

Для того, чтобы приобрести некоторый опыт в работе с переменными и явным их объявлением, сделаем проект *prjCalc* – калькулятор.

На форму, которую назовем *frmCalc,* поместим три текстовых окна. В первое с именем *txtFirst* пользователь будет вводить значение первого

исходного числа, во второе с именем *txtSecond* пользователь будет вводить второе исходное число. В третьем с именем *txtResult* будет выводиться результат после выполнения соответствующего действия. Кроме этого на форму поместим четыре командные кнопки, соответствующие арифметическим действиям. При щелчке на кнопке с именем *cmdPlus* будет выполняться действие сложение первого и второго исходного чисел. Свойству *Caption* этой кнопки присвоим значение «+». При щелчке на кнопке с именем *cmdMinus* будет выполняться вычитание второго из первого исходных чисел. Свойству *Caption* этой кнопки присвоим значение «-». При щелчке на кнопке с именем *cmdMult* будет выполняться действие умножение первого и второго исходных чисел. Свойству *Caption* этой кнопки присвоим значение «*». При щелчке на кнопке с именем *cmdDiv* будет выполняться действие деление первого исходного числа на второе. Свойству *Caption* этой кнопки присвоим значение «/». После того, как мы сформируем графический интерфейс пользователя, форма будет выглядеть следующим образом:



Наш программный код будет состоять из четырех процедур, каждая из которых соответствует щелчку мышкой на одной из командных кнопок. Кнопок на форме четыре – процедуры четыре.

В общей части кода, сразу после строки *Option Explicit* объявим необходимые для работы программы переменные: *NumberFirst* – переменная, в которой будет храниться значение, введенное пользователем в окошко *txtFirst; NumberSecond* – переменная, в которой будет храниться значение, введенное пользователем в окошко *txtSecond; NumberResult!* – переменная, в которой

будет храниться результат вычислений данного действия. Тип определим для них – десятичные числа. Соответствующие строки кода будут выглядеть следующим образом:

Dim NumberFirst As Single, NumberSecond As Single
Dim NumberResult!

Теперь составим код для командной кнопки *cmdPlus:*

```
Private Sub cmdPlus_Click( )
    NumberFirst = Val(txtFirst.Text)
    NumberSecond = Val(txtSecond.Text)
    NumberResult! = NumberFirst + NumberSecond
    txtResult.Text = Str(NumberResult!)
End Sub
```

В первой строке кода переменной *NumberFirst* присваивается значение свойства *Text* окошка *txtFirst,* с помощью функции *Val( )* преобразованное в число. Во второй строке кода переменной *NumberSecond* присваивается значение свойства *Text* окошка *txtSecond,* с помощью функции *Val( )* преобразованное в число. В третьей строке переменной *NumberResult!* присваивается значение суммы *NumberFirst* и *NumberSecond.* В четвертой строке свойству *Text* окошка *txtResult* присваивается значение переменной *NumberResult!,* преобразованное в строку цифровых символов.

Самостоятельно в тетради составьте коды трех оставшихся процедур.

Private Sub cmdMinus_Click( )
End Sub

Private Sub cmdMult_Click( )
End Sub

Private Sub cmdDiv_Click( )
End Sub

Теперь проект набирается и запускается на компьютере и проверяется правильность составленного кода.

ЛИТЕРАТУРА

1. Кетков Ю., Кетков А. Практика программирования.Visual Basic, C++ Builder, Delphi.-Санкт-Петербург «БХВ- Петербург», 2002
2. Реселман Боб, Ричард Писли и др. Использование Visual Basic 6. Вильямс,1999
3. Тим Андерсон. Кадам ба кадам, Тошкент 2002
4. Абрамов С.А. и др. Задачи по программированию - М.:Наука, 1988 - 224 с.
5. Шафрин Ю. Основы компьютерной технологии. Бишкек, 1998, 560 с;
6. Вирт Н. Алгоритмы + структуры данных=программы. М.: Мир, 1985.-406 с
7. Брябрин Р.М. Программное обеспечение персональных ЭВМ. М.,: Наука, 1990, 272 бет;