

ПЕРЕМЕННЫЕ

Иногда необходимо запомнить или записать кое-что на память. Мы для этого пользуемся, например, записной книжкой (простой или электронной) или записями на бумаге.

При выполнении вычислений или обработке информации тоже требуется запоминать значения некоторых данных. Для каждого из этих данных выделяется участок памяти компьютера, состоящий из одного или нескольких байтов. Для того чтобы эти участки памяти можно было различать, а также иметь возможность сослаться на содержимое некоторого конкретного участка памяти, ему сопоставляют имя. Вот эта пара, включающая участок памяти и сопоставленное ему имя, и есть переменная.

Итак, переменная – это в некотором смысле ячейка для хранения информации, например, числа, строки символов. При этом имеется возможность неоднократно считывать значение переменной, а также возможность записывать в эту ячейку другое значение. Переменная может изменять свое значение в процессе выполнения программы. Ее значение может также оставаться неизменным от начала до конца выполнения программы, но принимать разные значения в разных прогонах программы.

В VB.NET переменные являются объектами. Они как объекты обладают свойствами, к ним как к объектам применимы методы.

Имена переменных в VB должны удовлетворять следующим требованиям:

- начинаться с буквы;
- включать только буквы, цифры, символ подчеркивания (_), который на клавиатуре находится под тире (-).

Обратите внимание, имя не может содержать пробел (), точку (.), запятую (,), восклицательный знак (!) или символы (@), (&), (\$), (#). Не следует использовать имена, совпадающие с зарезервированными словами языка.

В языке VB не различаются строчные и прописные буквы.

Множество возможных значений переменной, допустимые операции, которые к ней применимы, количество байтов, отведенных для нее, определяется типом переменной. В каждом языке программирования определена своя система типов переменных.

К константам относится все то, что сказано выше о переменных. Но есть одно существенное отличие: константа не может изменить значение при выполнении программы. Переменные и константы – это данные. Познакомимся с системой типов данных VB.NET.

1. Основные базовые типы данных VB.NET

Табл. 1. Основные базовые типы данных VB.NET.

Название типа	Длина области памяти в байтах	Диапазон значений
SByte	1	От – 128 до 127
Byte	1	0 – 255
Boolean (логический)	2	True или False
Char	2	Любой символ Unicode в диапазоне 0 – 65535
Short (короткий целый)	2	От -32 768 до 32 767
Integer (целый)	4	От -2 147 483 648 до 2 147 483 647
Single (числа с дробной частью одинарной точности)	4	Отрицательные числа от $-3,402823 \cdot 10^{38}$ до $-1,401298 \cdot 10^{-45}$ и положительные от $1,401298 \cdot 10^{-45}$ до $3,402823 \cdot 10^{38}$. Точность 6 – 7 десятичных разрядов.
Long (длинный целый)	8	От -9 223 372 036 854 775 808 до

Название типа	Длина области памяти в байтах	Диапазон значений
		9 223 372 036 854 775 807
Double (числа с дробной частью двойной точности)	8	Отрицательные числа от $-1,79769313486232 \cdot 10^{308}$ до $-4,9406564841247 \cdot 10^{-324}$ и положительные от $4,9406564841247 \cdot 10^{-324}$ до $1,79769313486232 \cdot 10^{308}$. Точность 14 – 15 десятичных разрядов.
Date (дата/время)	8	От 0:00:00 1 января 0001 г. до 23:59:59 31 декабря 9999 г.
Decimal (целые числа и числа с дробной частью)	16	Масштабируемый показателем степени 10^P , где P изменяется в пределах от 0 до 28. Число имеет P разрядов дробной части. Максимальное значение равно 79 228 162 514 264 337 593 543 950 335, минимальное значение равно -79 228 162 514 264 337 593 543 950 335.
String (строка переменной длины)	Зависит от платформы. Обычно 2 байта на символ.	Приблизительно до 2 миллиардов символов Unicode
Object (объект)	4	Любой объект

Если, например, в программе имеется переменная типа Integer с именем КоличествоСтудентов, то где-нибудь в программе можно написать:

```
КоличествоСтудентов = 1000
```

и для всех инструкций, которые появятся после этого, переменная КоличествоСтудентов будет всегда равна числу 1000 – пока, возможно, не появится инструкция:

```
КоличествоСтудентов = 1050
```

КоличествоСтудентов в этой инструкции является именем переменной, а 1050 в этом примере – её значением.

Инструкцию `КоличествоСтудентов = 1050` называют присвоением: переменная КоличествоСтудентов получает значение, равное 1050.

Эту инструкцию нельзя понимать в том смысле, что левая часть равна правой части. Присвоение – это действие, заключающееся в том, что значение правой части записывается в ячейку памяти, отведенную для хранения значения переменной, имя которой указано слева от знака равенства в инструкции присвоения. Будет ошибкой записать эту инструкцию так:

```
1050 = КоличествоСтудентов
```

Слева от знака равенства должна находиться переменная.

Переменные типов Short, Integer, Long, Single, Double, Decimal принимают числовые значения.

Если в программе имеется переменная а типа Single, то такой переменной можно присвоить числовое значение с дробной частью, например:

```
а = - 62.697
```

В этой инструкции присваивания справа от знака равенства находится константа с плавающей точкой. В VB для отделения целой части от дробной части применяется символ (.).

Возможен другой способ записи констант с плавающей точкой – константа с порядком. Например, $1.5E-16$ означает $1.5 \cdot 10^{-16}$ (или иначе 0.00000000000000015).

К числовым переменным можно применять арифметические операции сложения (+), вычитания (–), умножения (*), деления (/), возведения в степень (^).

Значением переменной типа String может быть символ или строка символов.

Значением переменной типа Date может быть, дата, время или дата и время.

Переменная типа Boolean может принимать всего два значения. Такая переменная может иметь значение True (истина) или значение False (ложь).

Тип Object является универсальным. Переменные типа Object могут принимать числовые значения, значения символов и строк символов, значение даты и времени. Остальные типы данных пока комментировать не будем.

Если UserName является переменной типа String, тогда можно написать: `UserName = "Иван"` (здесь очень важны кавычки, так как иначе компьютер может принять Иван за имя переменной). Этот пример показывает, что константа типа String должна быть заключена в двойные кавычки.

К строковым переменным и константам можно применять операцию сцепления, которая обозначается символом (&) или символом (+). Например, можно написать:

```
UserName = UserName & " Иванов"
```

После выполнения этой инструкции переменная UserName будет иметь значение "Иван Иванов".

Если ДеньРождения и EndOfTime являются именами переменных типа Date, тогда можно записать:

```
ДеньРождения = #29.10.1970#
```

```
EndOfTime = #8:30#
```

(как символы выделяют кавычками, так дату или время выделяют символом (#)). Дату и время можно поместить в одну переменную. Для переменной DateAndTime типа Date может быть записано:

```
DateAndTime = #13.2.2000 11:30#
```

2. Структура проекта

Проект обычно включает одну или несколько форм, а также может включать модули. С формой Вы уже имели дело при выполнении предыдущих заданий. Форма при выполнении проекта отображается на экране монитора в виде окна. Она может содержать управляющие элементы (надписи, текстовые поля и т.д.), а также содержит программный код. Модуль подобно форме тоже содержит код. Но в отличие от формы модуль не отображается на экране монитора при выполнении проекта и не может содержать управляющие элементы.

3. Объявление переменных и констант

Если переменная (константа) применяется в проекте, она должна быть объявлена до первого обращения к ней. Существуют следующие четыре уровня объявления переменных и констант:

Уровень блока. Блоком называется последовательность инструкций, заканчивающаяся одной из строк Next, End If, Loop. Имя объявленное внутри блока, вне этого блока не действует.

Уровень процедуры (локальный уровень). Имя, объявленное в процедуре, действует только внутри этой процедуры и не действует вне ее.

Уровень формы (модуля). Имя, объявленное в форме или модуле (но не внутри процедуры), действует во всех процедурах этой формы (модуля), но не действует в других формах и модулях.

Уровень проекта (глобальный уровень). Имя, объявленное в модуле (но не внутри процедуры) с предваряющим словом **Public** (общий), действует во всех формах и модулях проекта.

Синтаксис инструкции объявления переменной:

Static/Public/Private/Dim ИмяПеременной **As** Тип [= Значение для инициализации]

В подобных определениях синтаксиса прямоугольные скобки [] означают, что конструкция, находящаяся внутри этих скобок, не обязательна. Символ (/) означает, что должно быть выбрано одно из слов, между которыми он поставлен. Сами же символы ([, (, /) в текст объявления не включаются.

Результатом объявления переменной является выделение памяти под эту переменную.

Возможно отменить требование обязательного объявления переменных. Для этого в верхней части кода проекта (до предложения **Public Class Form1**) следует поместить инструкцию **Option Explicit Off**. В этом случае переменной, которая не объявлена, будет принудительно назначен тип **Object**. Однако применение такой практики решительно не рекомендуется. Причина – открывается возможность создания трудно обнаруживаемых ошибок.

Если при объявлении переменной ее значение для инициализации не задано, то переменная любого числового типа получит значение 0, типа **Boolean** – значение **False**, типа **String** – значение "" (пустая строка).

Пример объявления целочисленной переменной, которая после ее объявления получит значение, равное 1:

```
Dim i As Integer = 1
```

При объявлении константы необходимо задать ее имя, тип, область действия и значение.

Синтаксис объявления константы:

[**Public/Private**] **Const** ИмяКонстанты **As** Тип = Значение

Приведенное определение синтаксиса означает, что объявление константы начинается с обязательного слова **Const** (константа). Перед **Const** может стоять одно из слов: **Private** (локальный) или **Public** (общий), задающих область действия константы. Затем следует имя константы. После имени должно стоять слово **As** и наименование типа. Затем следует знак равенства и значение константы.

В определении подчеркнуто наименование той области действия, которая может быть задана по умолчанию.

В следующем примере в модуле объявляется (но не внутри процедуры) глобальная константа **Age** целого типа, и ей присваивается значение 54.

```
Public Const Age As Integer = 54
```

Допускается также описание нескольких констант в одной строке. В следующем примере описываются локальные константы **Ag** и **Wg** типа **Single**:

```
Const Ag As Single = 3.14, Wg As Single = 2.78
```

Зарезервированное слово **Dim** (размерность) при объявлении переменных применяется чаще всего.

Статические переменные, могут быть объявлены только на уровне процедуры со словом **Static** вместо слова **Dim**. Они сохраняют свои значения даже после выхода из процедуры при повторном входе в эту процедуру.

Вот пример объявления переменной типа строки символов:

```
Dim strName As String
```

В одной строке можно объявить несколько переменных:

```
Dim a, b As Integer, c As Long
```

```
Dim e, f, g As Integer
```

```
Dim a1 As Single = 5.3, a2 As Integer = 34
```

В первой строке объявлены две переменные типа `Integer` и одна переменная типа `Long`. Во второй строке – три переменных типа `Integer`. В третьей строке объявлены и инициализированы две переменные.

4. Преобразование и совместимость типов

В этом разделе речь будет идти о тех ситуациях, когда при выполнении операции или инструкции участвуют данные разных типов. Например, что произойдет, если переменной целого типа присваивается числовое значение типа `Single`?

По умолчанию в отношении типов VB проявляет максимальную терпимость – когда это возможно, преобразование одного типа в другой будет выполнено автоматически и программисту нет нужды об этом заботиться.

Однако можно отменить автоматическое преобразование типов, что делать не рекомендуется. Для этого в верхней части кода проекта (до предложения `Public Class Form1`) следует поместить инструкцию `Option Strict On`. В этом случае в программном коде следует в каждом случае предусматривать явное преобразование типа.

Если автоматическое преобразование типа не отменено, то соблюдаются перечисленные ниже правила:

- ◆ При преобразовании числа с плавающей точкой в целое происходит округление до ближайшего целого.
- ◆ При преобразовании целого числа в число с плавающей точкой дробная часть принимается равной нулю.
- ◆ В случае преобразования целого типа в другой целый тип возможна ситуация, когда целый тип с большим диапазоном значений преобразуется в целый тип с меньшим диапазоном значений. Если значение, присваиваемое «короткому» типу, выйдет за пределы диапазона его допустимых значений, произойдет генерация исключения.
- ◆ Строковые и числовые типы совместимы. Можно присвоить числовое значение строковой переменной и наоборот.

Например, не приведет к исключению присвоение свойству `Text` некоторого текстового поля (оно имеет тип `String`) значения переменной типа `Integer`, равное 125. Произойдет преобразование целого числового значения 125 в строку символов "125".

И наоборот. При присвоении переменной целого типа значения свойства `Text` некоторого текстового поля (пусть, например, в этом текстовом поле набрано три символа 125), произойдет преобразование строки символов "125" в целое число, и целочисленная переменная получит значение 125. Однако строка символов должна быть такой, чтобы она могла трактоваться как число.

Возможно также выполнение арифметической операции, например умножения (*), когда один или оба операнда являются строкой символов, но при условии, что их значения, можно интерпретировать как числа. При этом особую осторожность следует соблюдать с операцией (+), которая в зависимости от контекста может означать либо операцию сложения, либо операцию сцепления.

В операции `операнд1 + операнд2` символ (+) VB будет воспринимать как:

- операцию сложения, если оба операнда имеют числовой тип;
- операцию сложения, если один операнд имеет числовой тип, а второй является строкой символов, значение которой может быть интерпретировано как число;
- операцию сцепления, если один операнд имеет числовой тип, а второй является строкой символов, значение которой не может быть интерпретировано как число;
- операцию сцепления, если оба операнда имеют строковый тип независимо от их значений.

Теперь можно приступить к разработке проекта.

5. Разработка проекта



1. Создайте новый проект с именем Переменные, следуя приложению 1.
2. Если окно конструктора форм не открыто, то откройте его щелчком на кнопке  View Designer (просмотреть конструктор), расположенной на панели инструментов окна обозревателя решений. Если же эта кнопка на панели инструментов отсутствует, то предварительно щелкните в окне обозревателя решений на компоненте проекта Form1.vb.
3. Задайте свойству Text формы значение Переменные.
4. Поместите на открывшейся форме шесть текстовых полей, три надписи и три кнопки Button. Ваша форма должна выглядеть так, как показано на рис. 1.



Рис. 1. Вид формы

5. На панели инструментов окна обозревателя решений щелкните на кнопке  View Code (просмотреть код) и перед строкой кода формы `End Class` для объявления переменных уровня формы введите код, показанный на листинге 1.

Листинг 1. Объявление переменных уровня формы

```
Dim i As Integer
'Объявлена переменная i целого типа
Dim r As Single
'Объявлена переменная r с дробной частью
'обычной точности
'Ниже объявлены
'переменные st1 и st2 строкового типа
Dim st1, st2 As String
```

На примере этого фрагмента программного кода можно увидеть, как документируется программа с помощью комментариев. Все, что находится в строке правее символа (') рассматривается как комментарий, присутствующий в программном коде, но не оказывающий влияния на результаты его выполнения.

6. Над окном кода слева в поле Class Name раскройте список имен классов находящихся на форме объектов и выберите объект Button1. Над окном кода справа в поле Method Name раскройте список имен методов и выберите метод Click. В окне кода появится заготовка подпрограммы Button1_Click, которая будет выполняться при щелчке на кнопке Button1.

7. Введите в этой заготовке программный код, чтобы он соответствовал листингу 2.

Листинг 2. Код подпрограммы Button1_Click.

```
Private Sub Button1_Click(ByVal sender As Object, _
ByVal e As System.EventArgs) Handles Button1.Click
    i = TextBox1.Text
    r = TextBox2.Text
    Label1.Text = r + i
End Sub
```

8. Щелкните на кнопке Start Debugging для запуска проекта. В появившемся окне формы наберите в поле TextBox1 значение 2, а также наберите в поле TextBox2 значение – 1,3. После этого щелкните на кнопке Button1. Последнее приведет к выполнению

инструкций процедуры `Button1_Click` и надпись отобразит результат $2 + (-1.3)$. На рис. 2 показан вид окна формы после указанных действий.

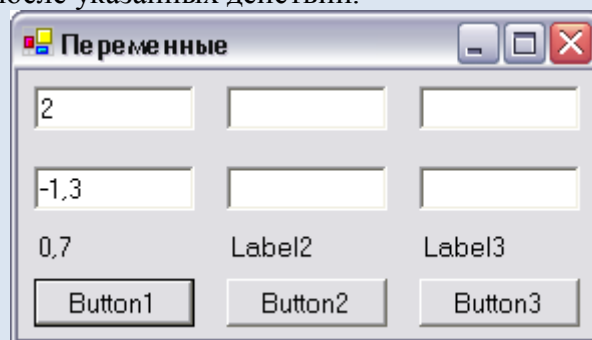


Рис. 2. Результат выполнения подпрограммы `Button1_Click`

Обратите внимание, при вводе строки в поле `TextBox2`, которая должна интерпретироваться программой как число с плавающей точкой, для разделения целой и дробной частей применена не точка, а запятая. Это связано с тем, что стандартной настройкой операционной системы в качестве разделителя целой и дробной части числа предусмотрена запятая. Вид разделителя можно изменить. Для этого необходимо выполнить команды Настройка и Панель управления меню кнопки Пуск панели задач. Затем следует выбрать Язык и стандарты и на вкладке Числа в окне Разделитель дробной и целой частей числа установить нужный разделитель и щелкнуть на кнопке ОК. Для защиты от недостаточно квалифицированных пользователей команда Настройка в учебных компьютерных классах ИВЦ МЭИ исключена из меню кнопки Пуск, поэтому изменить разделитель целой и дробной частей числа пользователь, не имеющий прав администратора, не может.

Итак, будем исходить из того, что при записи констант в программном коде следует применять точку для разделения целой и дробной частей, а во входном потоке при вводе значений переменных надо применять запятую.

Остановимся подробнее на существенных, но на первый взгляд не заметных, особенностях выполнения инструкций процедуры `Button1_Click`. При выполнении этих трех инструкций четыре раза осуществляется преобразование типа.

Так, при выполнении инструкции присвоения `i = TextBox1.Text` тип `String` значения свойства `Text` текстового окна `TextBox1` преобразуется в тип `Integer` переменной `i`.

При выполнении инструкции присвоения `r = TextBox2.Text` тип `String` значения свойства `Text` текстового окна `TextBox2` преобразуется в тип `Single` переменной `r`.

В инструкции `Label1.Text = r + i` при выполнении арифметической операции сложения тип `Integer` операнда `i` преобразуется к типу `Single` операнда `r`, так как арифметическая операция может быть выполнена над однотипными операндами, а тип с плавающей точкой старше целого типа.

Наконец, результат сложения типа `Single` преобразуется к типу `String` значения свойства `Text` надписи `Label1`. Обратите внимание, все эти преобразования типов VB выполнил автоматически без указания об этом в коде с Вашей стороны.

9. Остановите выполнение проекта, щелкнув на кнопке `Stop Debugging` стандартной панели инструментов, и сохраните проект.

10. Щелкните два раза на кнопке `Button2`. В окне кода появится заготовка подпрограммы `Button2_Click`.

11. Введите в этой заготовке программный код, соответствующий листингу 3.

Листинг 3. Код подпрограммы `Button2_Click`.

```
Private Sub Button2_Click(ByVal sender As System.Object, _
```

```

ByVal e As System.EventArgs) Handles Button2.Click
    st1 = TextBox3.Text
    st2 = TextBox4.Text
    Label2.Text = st1 + st2
End Sub

```

12. Щелкните на кнопке Start Debugging для запуска проекта и в появившемся окне формы наберите в текстовом поле TextBox3 строку Вас, а также наберите в текстовом поле TextBox4 строку илий. После этого щелкните на кнопке Button2. Это приведет к выполнению инструкций процедуры Button2_Click, и надпись отобразит результат Василий операции сцепления, как это показано на рис. 3.

13. Полученный результат Вас не должен удивлять. Ведь в инструкции Label2.Text = st1 + st2 символ операции (+) воспринимается как символ операции сцепления, а не операции сложения, поскольку оба операнда st1 и st2 имеют строковый тип. А что произойдет, если в текстовых окнах задать числа?

14. Удалите в текстовом поле TextBox3 строку Вас и наберите 25, а также удалите в текстовом поле TextBox4 строку илий и наберите 15. После этого щелкните на кнопке Button2. На рис. 4 показан результат. Опять произошло сцепление операндов. Этого следовало ожидать, поскольку оба операнда (свойства Text) имеют строковый тип. Посмотрите, что произойдет, если один операнд будет иметь строковый, а другой – числовой, например, целый тип.

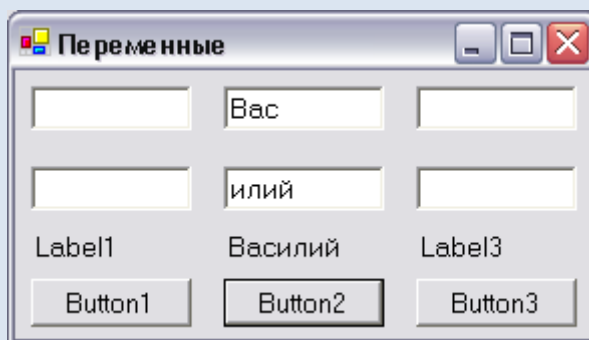


Рис. 3. Результат выполнения подпрограммы Button2_Click

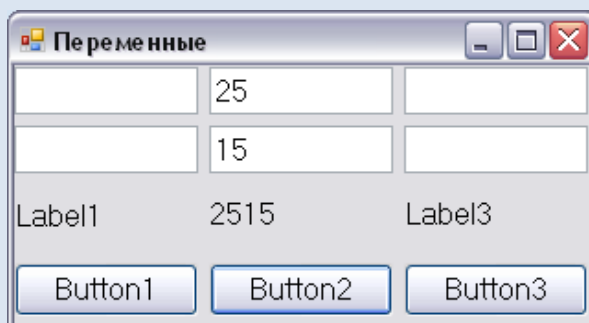


Рис. 4. Результат повторного выполнения подпрограммы Button2_Click

15. Остановите выполнение проекта, щелкнув на кнопке Stop Debugging стандартной панели инструментов, и сохраните проект.

16. Щелкните два раза на кнопке Button3. В окне кода появилась заготовка процедуры Button3_Click.

17. Введите в этой заготовке программный код, соответствующий листингу 4.

Листинг 4. Код подпрограммы Button3_Click.

```

Private Sub Button3_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles Button3.Click
    st1 = TextBox5.Text

```



```

st2 = TextBox6.Text
Label3.Text = 25 + st1 + st2
End Sub

```

18. Щелкните на кнопке Start Debugging для запуска проекта и в появившемся окне формы наберите 2 в текстовом поле TextBox5, а также наберите 3 в текстовом поле TextBox6.

19. После этого щелкните на кнопке Button3. Это приведет к выполнению инструкций подпрограммы Button3_Click, и надпись отобразит результат операции, что и показано на рис. 5.

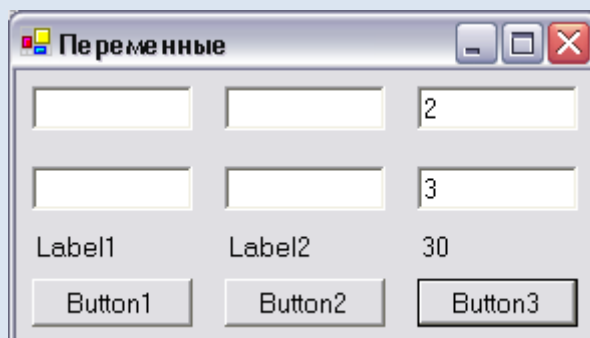


Рис. 5. Результат выполнения подпрограммы Button3_Click

При выполнении вычисления значения выражения $25 + st1 + st2$ первой выполняется операция $25 + st1$. В этой операции один операнд (константа 25) имеет числовой тип, а другой (строка st1) имеет значение, которое может быть интерпретировано как число. Поэтому здесь символ (+) воспринят как операция сложения. Результат операции сложения (27) тоже имеет числовой тип, поэтому символ (+) в следующей операции $27 + 3$ также воспринят как операция сложения и, в конце концов, получен понятный результат 30.

20. Остановите выполнение проекта, щелкнув на кнопке Stop Debugging панели инструментов, и сохраните проект.

21. Измените порядок следования операндов в инструкции присвоения `Label3.Text = 25 + st1 + st2`.

Замените ее инструкцией `Label3.Text = st1 + st2 + 25`.

22. Щелкните на кнопке Start Debugging для запуска проекта и в появившемся окне формы снова наберите 2 в текстовом поле TextBox5, а также наберите 3 в текстовом поле TextBox6.

23. После этого щелкните на кнопке Button3. Надпись отобразит, на первый взгляд, неожиданный результат вычислений (рис. 6).

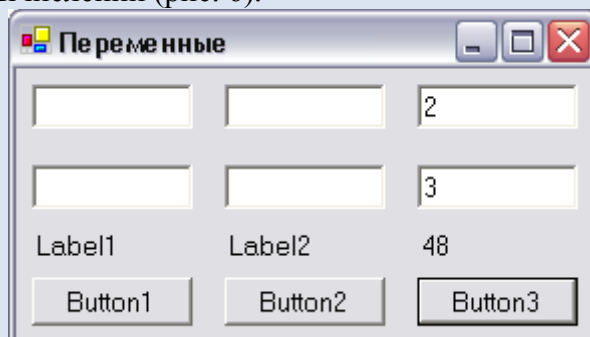


Рис. 6. Результат повторного выполнения подпрограммы Button3_Click

Изменение порядка слагаемых привело к изменению значения результатов вычислений! Почему? Дело в том, что первой теперь выполняется операция $st1 + st2$, в которой символ (+) воспринимается как операция сцепления, поскольку оба

- множество значений, которые может принимать данное;
- операции, которые можно выполнять с данными этого типа?

17. Какой тип объявлен для каждой из трех переменных инструкцией:

`Dim a, b, c As Integer`

18. Являются ли эти два варианта объявления переменных эквивалентными по результату:

Первый вариант	Второй вариант
<code>Dim s, d As Single</code>	<code>Dim s As Single</code> <code>Dim d As Single</code>

19. Где должна быть объявлена переменная, чтобы она действовала только внутри процедуры:

- в коде модуля (но не внутри процедуры);
- в коде формы (но не внутри процедуры);
- в этой процедуре.

20. Где должна быть объявлена переменная, чтобы она действовала во всех процедурах, объявленных в форме, но не действовала в других формах и модулях:

- в коде модуля (но не внутри процедуры);
- в коде этой формы (но не внутри процедуры);
- в любой процедуре формы с зарезервированным словом `Public`.

21. Где должна быть объявлена переменная, чтобы она действовала во всех процедурах всех форм и модулей проекта:

- в коде формы с зарезервированным словом `Public` (но не внутри процедуры);
- в коде модуля с зарезервированным словом `Public` (но не внутри процедуры);
- в коде любой процедуры с зарезервированным словом `Public`.

22. К чему приведет объявление переменной `Dim f As Single`:

- для переменной `f` будет отведена ячейка памяти длиной 4 байта;
- Для переменной `f` будет отведена ячейка памяти длиной 4 байта и в нее будет записано значение 0;
- Для переменной `f` будет отведена ячейка памяти длиной 8 байтов и в нее будет записано значение 0.

23. К чему приведет объявление переменной: `Dim k As Integer`:

- для переменной `k` будет отведена ячейка памяти длиной 2 байта и в нее будет записано значение 0;
- для переменной `k` будет отведена ячейка памяти длиной 4 байта;
- для переменной `k` будет отведена ячейка памяти длиной 4 байта и в нее будет записано значение 0.

24. Каков результат выражения `"2" + "7" + 12`?

25. Какое значение получит переменная `k`:

```
Dim st1 As String, st2 As String, k As Integer
st1 = 12
st2 = 15
k = st2 + st1
```