NAME   AHMAD ULLAH KHAN

ID   6958

SUBJECT   VISUAL PAROGRAMMING

Q1. How to write Hello program in C# and explain in detail?

ANS:

```
// Hello World! program
namespace HelloWorld
{
   class Hello {
      static void Main(string[] args)
      {
         System.Console.WriteLine("Hello World!");
      }
   }
}
```

OUTPUT

```
Hello World!
```

`1: // Hello World! Program`

`//` indicates the beginning of a comment in C#. Comments are not executed by the C# compiler

`2: namespace HelloWorld{...}`

The namespace keyword is used to define our own namespace. Here we are creating a namespace called `HelloWorld`.

`3: class Hello{...}`

The above statement creates a class named – Hello in C#. Since, C# is an object-oriented programming language, creating a class is mandatory for the program’s execution.

```
4: static void Main(string[] args){...}
```

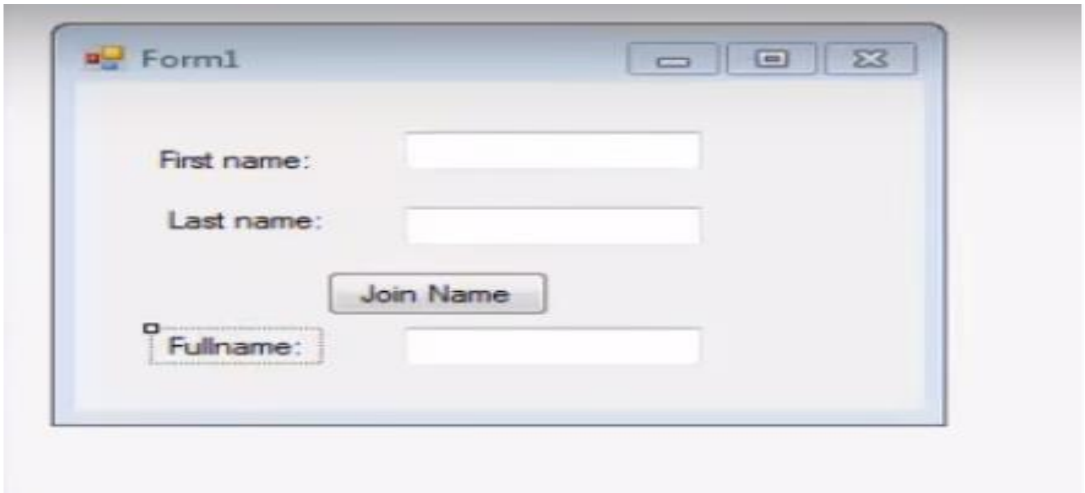
Main() is a method of class Hello. The execution of every C# program starts from the Main() method. So it is mandatory for a C# program to have a Main() method.

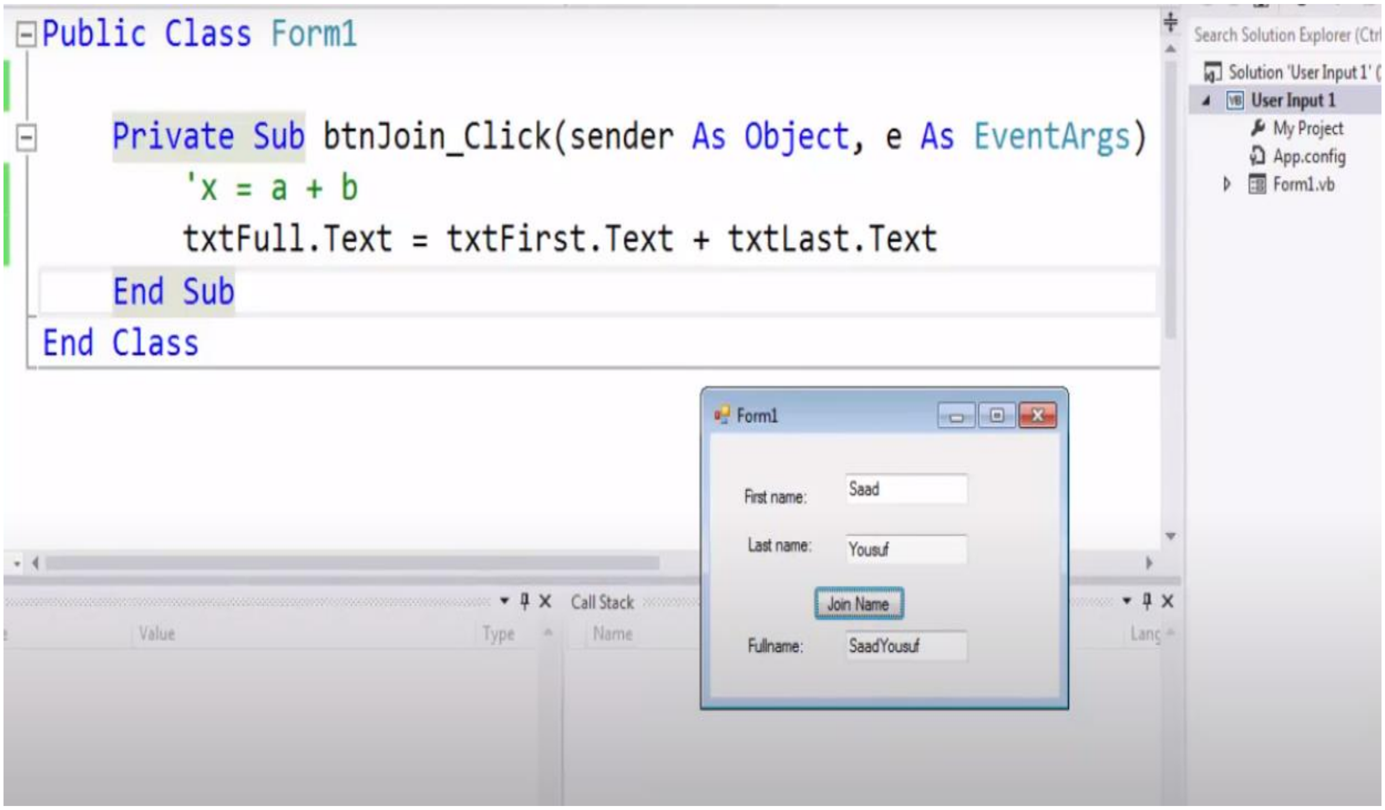
```
5: System.Console.WriteLine("Hello World!");
```

For now, just remember that this is the piece of code that prints **Hello World!** to the output screen.You’ll learn more about how it works in the later chapters

Q2. a. Write a simple program C # by taking 2 different strings in text boxes   concatenate them and display with the help of message box and explain in detail.

Ans:





```
Public Class Form1

    Private Sub btnJoin_Click(sender As Object, e As EventArgs)
        'x = a + b
        txtFull.Text = txtFirst.Text + txtLast.Text
    End Sub
End Class
```

Form1

First name: Saad

Last name: Yousuf

Join Name

Fullname: SaadYousuf

Q2 b. Write about different types of type conversions in available in C #.

ANS

C# provides the following built-in type conversion methods −

1 ToBoolean

Converts a type to a Boolean value, where possible.

2 ToByte

Converts a type to a byte.

3 ToChar

Converts a type to a single Unicode character, where possible.

4 ToDateTime

Converts a type (integer or string type) to date-time structures.

5 ToDecimal

Converts a floating point or integer type to a decimal type.

6 ToDouble

Converts a type to a double type.

7 ToInt16

Converts a type to a 16-bit integer.

8 ToInt32

Converts a type to a 32-bit integer.

9 ToInt64

Converts a type to a 64-bit integer.

10 ToSbyte

Converts a type to a signed byte type.

11 ToSingle

Converts a type to a small floating point number.

12 ToString

Converts a type to a string.

13 ToType

Converts a type to a specified type.

14 ToUInt16

Converts a type to an unsigned int type.

15 ToUInt32

Converts a type to an unsigned long type.

16 ToUInt64

Converts a type to an unsigned big integer.

Q3:

a) What are constant in C# discuss in detail?

Ans:

A constant is a meaningful name that takes the place of a number or string that does not change. Constants store values that, as the name implies, remain the same throughout the execution of an application. You can greatly improve the readability of your code and make it easier to maintain by using constants.

The constants are treated just like regular variables except that their values cannot be modified after their definition.

Integer Literals

An integer literal can be a decimal, or hexadecimal constant. A prefix specifies the base or radix: 0x or 0X for hexadecimal, and there is no prefix id for decimal.

An integer literal can also have a suffix that is a combination of U and L, for unsigned and long, respectively. The suffix can be uppercase or lowercase and can be in any order.

Here are some examples of integer literals −

```
212       /* Legal */
215u       /* Legal */
0xFeeL     /* Legal */
```

Following are other examples of various types of Integer literals −

```
85        /* decimal */
0x4b      /* hexadecimal */
30        /* int */
30u       /* unsigned int */
30l       /* long */
30ul      /* unsigned long */
```

Floating-point Literals

A floating-point literal has an integer part, a decimal point, a fractional part, and an exponent part. You can represent floating point literals either in decimal form or exponential form.

Here are some examples of floating-point literals −

```
3.14159       /* Legal */
314159E-5F    /* Legal */
510E          /* Illegal: incomplete exponent */
210f          /* Illegal: no decimal or exponent */
.e55          /* Illegal: missing integer or fraction */
```

b. Write a program on string literal and explain in detail

```
Option Strict On



Module Example

   Sub Main()

      Dim str1 As String = "There are 10"
      Dim str2 As String = " apples"
```

```
        Console.WriteLine(str1 + str2)

        Console.WriteLine("The length of the first string is " _
            + str1.Length.ToString() + " characters")

    End Sub

End Module
```

In the preceding example, we create two string variables. Then we add them and compute the length of the first string.

```
Dim str1 As String = "There are 10"
```

A string variable is declared and initiated.

```
Console.WriteLine(str1 + str2)
```

Two strings are concatenated. We use the + operator to add two strings.

```
Console.WriteLine("The length of the first string is " _
    + str1.Length.ToString() + " characters")
```

The Length property is used to determine the length of the string.

```
$ ./basics.exe
There are 10 apples
The length of the first string is 12 characters
```