**LOOPS**

Visual Basic loop structures allow us to run one or more lines of code repetitively. We can repeat the statements in a loop structure until a condition is True, until a condition is False, a specified number of times, or once for each element in a collection.

**While Loops**

The While...End While construction runs a set of statements as long as the condition specified in the While statement is True.



**Do Loops**

The Do...Loop construction allows us to test a condition at either the beginning or the end of a loop structure. We can also specify whether to repeat the loop while the condition remains True or until it becomes True.

**For Loops**

The For...Next construction performs the loop a set number of times. It uses a loop control variable, also called a counter, to keep track of the repetitions. We specify the starting and ending values for this counter, and we can optionally specify the amount by which it increases from one repetition to the next.

**For Each Loops**

The For Each...Next construction runs a set of statements once for each element in a collection. You specify the loop control variable, but you do not have to determine starting or ending values for it.



**While...End While Statement**

Runs a series of statements as long as a given condition is True.

**Syntax**

While condition

[ statements ]

[ Continue While ]

[ statements ]

[ Exit While ]

[ statements ]

End While

**Term Definition**

condition Required. Boolean expression. If condition is Nothing, Visual Basic treats it as False.

statements Optional. One or more statements following While, which run every time condition is True.

Continue While Optional. Transfers control to the next iteration of the While block.

Exit While Optional. Transfers control out of the While block.

End While Required. Terminates the definition of the While block.

**Example**

**Dim a As integer =1**

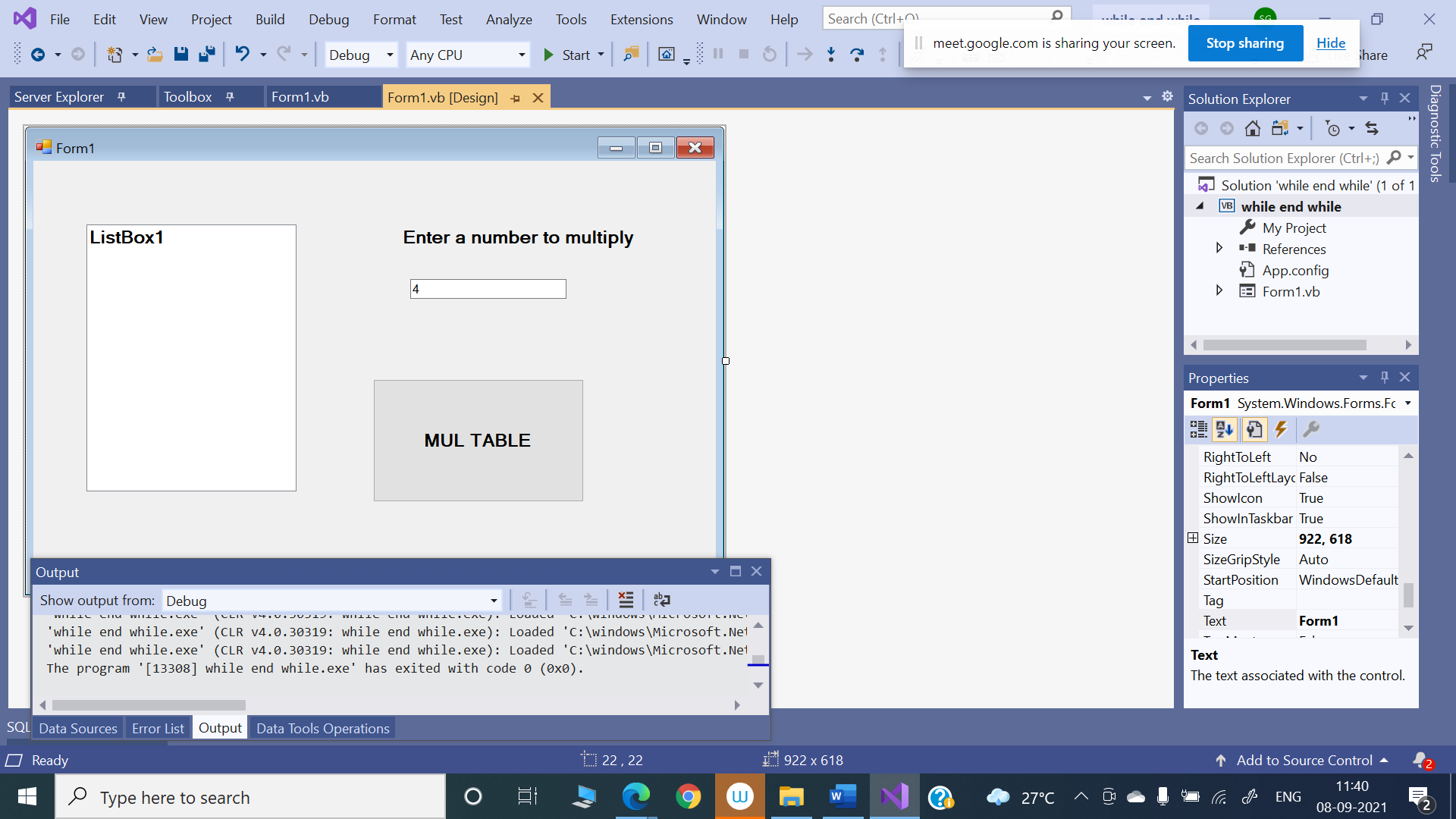
**While a<=10**

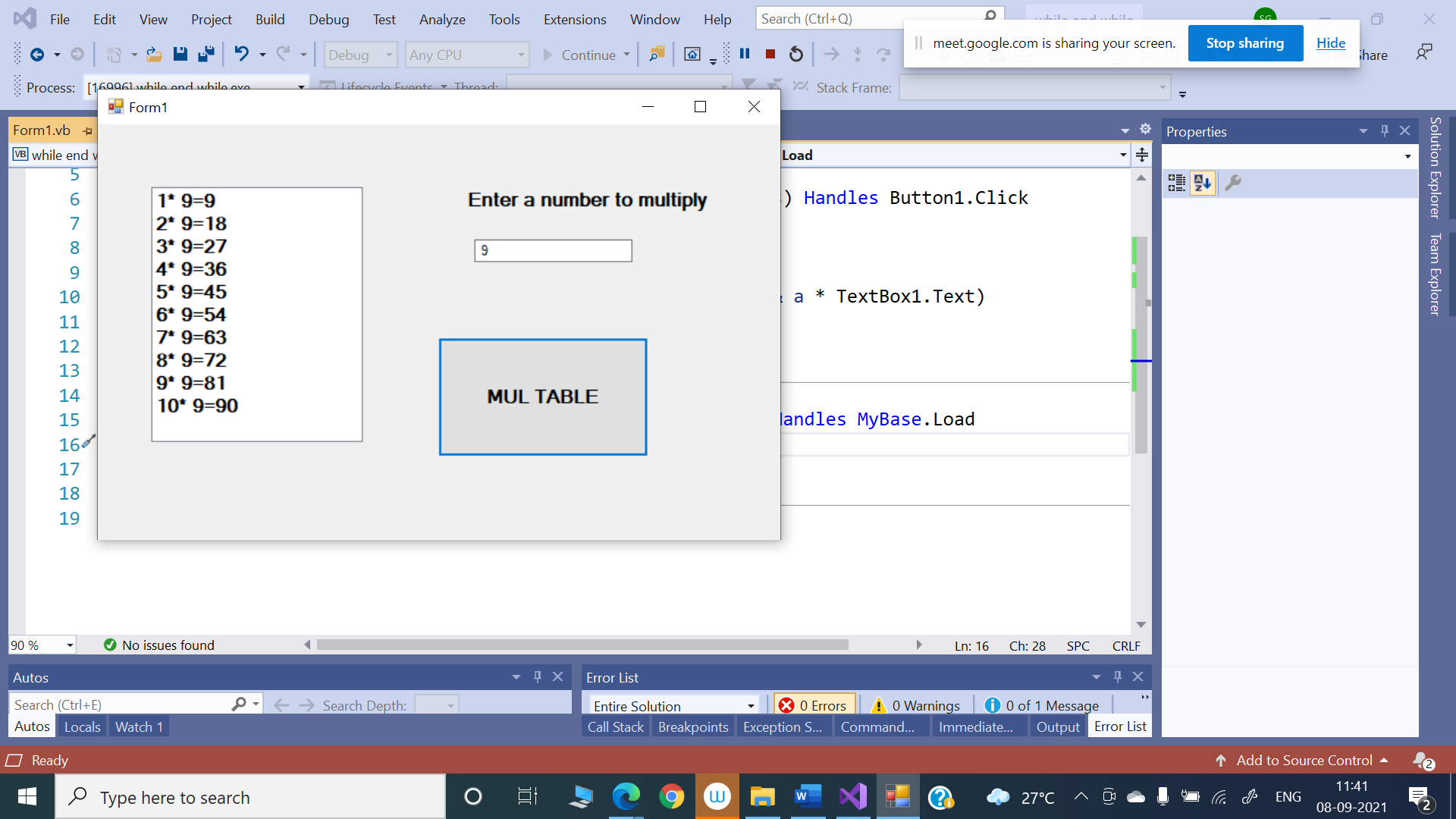
**ListBox1.Items.Add(a&"\*"& Textbox1.Text &"=" &a\*Textbox1.Text)**

**a=a+1**



**end while**





**Do….Loop**

Do { While | Until } condition



[ statements ]

[ Continue Do ]

[ statements ]

[ Exit Do ]

[ statements ]

Loop

-or-

Do

[ statements ]

[ Continue Do ]

[ statements ]

[ Exit Do ]

[ statements ]

Loop { While | Until } condition

Term Definition

Do Required. Starts the definition of the Do loop.

While Required unless Until is used. Repeat the loop until condition is False.

Until Required unless While is used. Repeat the loop until condition is True.

condition Optional. Boolean expression. If condition is Nothing, Visual Basic treats it as False.

statements Optional. One or more statements that are repeated while, or until, condition is True.

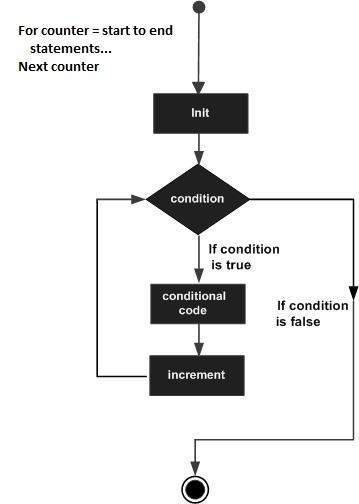
Continue Do Optional. Transfers control to the next iteration of the Do loop.

Exit Do Optional. Transfers control out of the Do loop.

Loop Required. Terminates the definition of the Do loop.

**For..Next**

Repeats a group of statements a specified number of times.





**Syntax**

For counter [ As datatype ] = start To end [ Step step ]

[ statements ]

[ Continue For ]

[ statements ]

[ Exit For ]

[ statements ]

Next [ counter ]

Part Description

counter Required in the For statement. Numeric variable. The control variable for the loop.

counter defines a new local variable that’s scoped to the entire For…Next loop.

datatype Optional. Data type of counter.

start Required. Numeric expression. The initial value of counter.

end Required. Numeric expression. The final value of counter.

step Optional. Numeric expression. The amount by which counter is incremented each time through the loop.

statements Optional. One or more statements between For and Next that run the specified number of times.

Continue For Optional. Transfers control to the next loop iteration.

Exit For Optional. Transfers control out of the For loop.

Next Required. Terminates the definition of the For loop.

Example 1

Dim a As Integer

' for loop execution

For a = 10 To 20



Console.WriteLine("value of a: {0}", a)



Next

Console.ReadLine()

Example 2

Dim a As Integer

' for loop execution

For a = 10 To 20 Step 2

Console.WriteLine("value of a: {0}", a)

Next

Console.ReadLine()

**For Each...Next**

It repeats a group of statements for each element in a collection. This loop is used for accessing and manipulating all elements in an array.

Syntax

For Each element [ As datatype ] In group

[ statements ]

[ Continue For ]

[ statements ]

[ Exit For ]

[ statements ]

Next [ element ]

Term Definition

element Required in the For Each statement. Variable which is used to iterate through the elements of the collection.

datatype Optional or element is already declared; The data type of element.

group Required. A variable with a type that's a collection type or Object. Refers to the collection over which the statements are to be repeated.

statements Optional. One or more statements between For Each and Next that run on each item in group.

Continue For Optional. Transfers control to the start of the For Each loop.

Exit For Optional. Transfers control out of the For Each loop.

Next Required. Terminates the definition of the For Each loop.

Example:

Dim anArray() As Integer = {1, 3, 5, 7, 9}

Dim arrayItem As Integer

'displaying the values

For Each arrayItem In anArray

Console.WriteLine(arrayItem)

Next

Console.ReadLine()