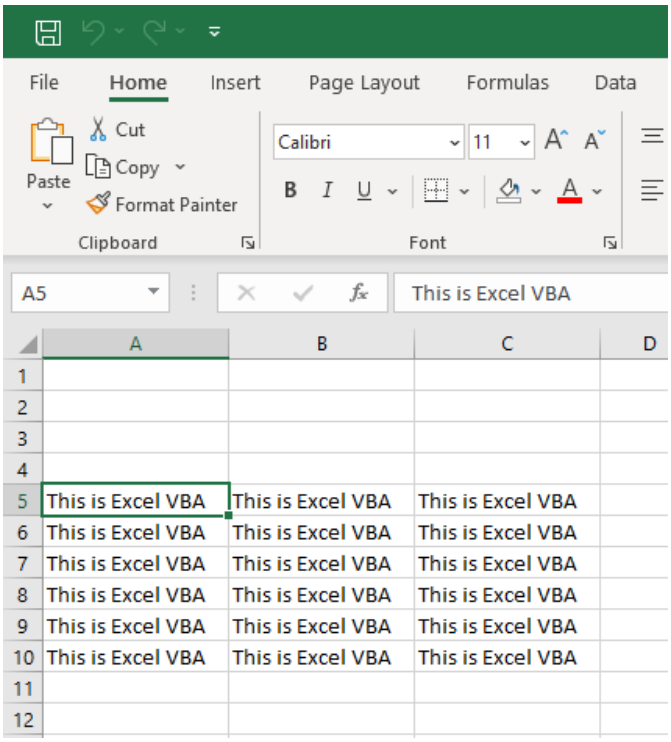
Excel Assignment - 20

1. Write a VBA code to select the cells from A5 to C10. Give it a name “Data Analytics” and fill the cells with the following cells “This is Excel VBA”.

   VBA Code:

```
Sub filling_cell()
Dim Data_Analytics As Range
Set Data_Analytics = Range("A5:C10")
Data_Analytics.Value = "This is Excel VBA"
End Sub
```



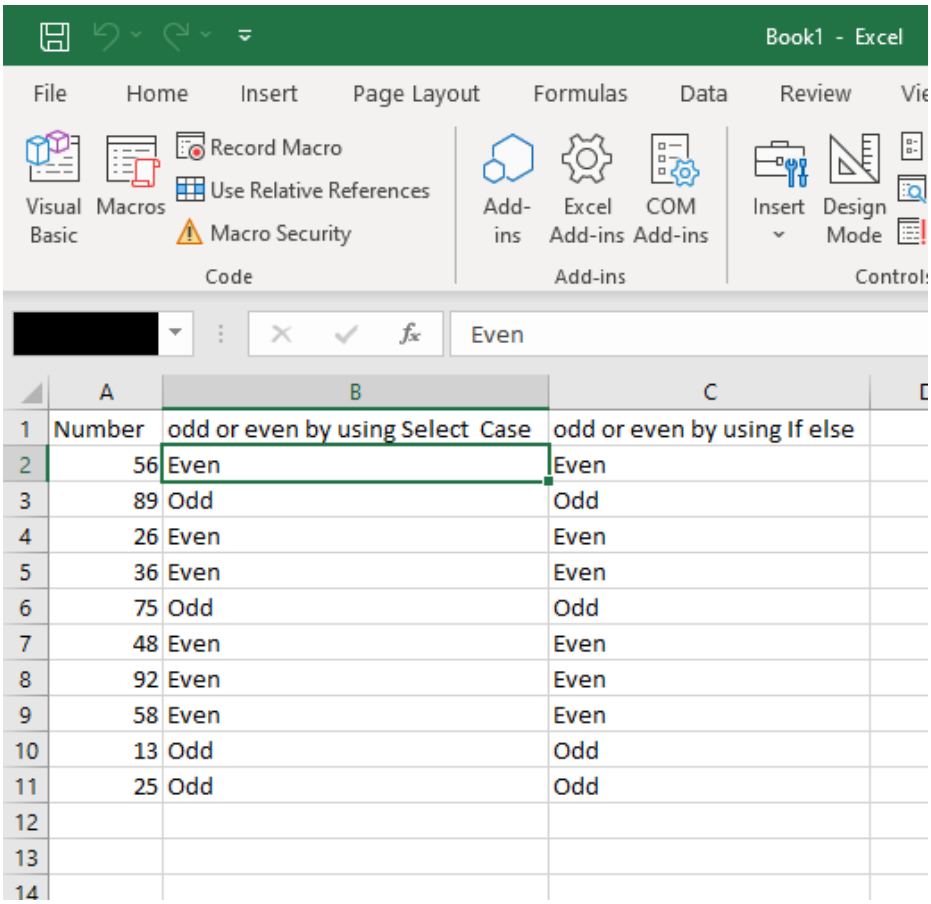
Number	Odd or even
56	
89	
26	
36	
75	
48	
92	
58	
13	
25	

2. Use the above data and write a VBA code using the following statements to display in the next column if the number is odd or even

    a. IF ELSE statement
    b. Select Case statement
    c. For Next Statement

```
Sub even_odd()
For i = 2 To 11
num = Cells(i, 1).Value
Select Case num Mod 2
   Case 0
   Cells(i, 2).Value = "Even"
   Case 1
   Cells(i, 2).Value = "Odd"
End Select
If num Mod 2 = 0 Then
Cells(i, 3).Value = "Even"
Else
```

```
Cells(i, 3).Value = "Odd"
End If
Next
End Sub
```



	A	B	C
1	Number	odd or even by using Select Case	odd or even by using If else
2	56	Even	Even
3	89	Odd	Odd
4	26	Even	Even
5	36	Even	Even
6	75	Odd	Odd
7	48	Even	Even
8	92	Even	Even
9	58	Even	Even
10	13	Odd	Odd
11	25	Odd	Odd
12			
13			
14			

3. What are the types of errors that you usually see in VBA?

In Visual Basic, errors fall into one of three categories: syntax errors, run-time errors, and logic errors.

Syntax Errors: *Syntax errors* are those that appear while you write code. If you're using Visual Studio, Visual Basic checks your code as you type it in the **Code Editor** window and alerts you if you make a mistake, such as misspelling a word or using a language element improperly. If you compile from the command line, Visual Basic displays a compiler error with information about the syntax error. Syntax errors are the most common type of errors. You can fix them easily in the coding environment as soon as they occur.

Run-Time Errors: *Run-time errors* are those that appear only after you compile and run your code. These involve code that may appear to be correct in that it has no syntax errors, but that will not execute. For example, you might correctly write a line of code to open a file. But if the file does not exist, the application cannot open the file, and it throws an exception. You can fix most run-time errors by rewriting the faulty code or by using exception handling, and then recompiling and rerunning it.

Logic Errors: Logic errors are those that appear once the application is in use. They are most often faulty assumptions made by the developer or unwanted or unexpected results in response to user actions. For example, a mistyped key might provide incorrect information to a method, or you may assume that a valid value is always supplied to a method when that is not the case. Although logic errors can be handled by using exception handling (for example, by testing whether an argument is Nothing and throwing an ArgumentNullException), most commonly they should be addressed by correcting the error in logic and recompiling the application.

4. How do you handle Runtime errors in VBA?

To handle an error inline, use the Resume Next statement with On Error. Any errors that occur during runtime cause Info Connect to continue executing the macro at the next statement. If an error occurs, it is handled by opening a dialog box, and passing control to another procedure or to a routine within the same procedure.

5. Write some good practices to be followed by VBA users for handling errors

- Use 'On Error Go [Label]' at the beginning of the code.
- Use 'On Error Resume Next' ONLY when you're sure about the errors that can occur.
- When using error handlers, make sure you're using Exit Sub before the handlers.
- Use multiple error handlers to trap different kinds of errors.

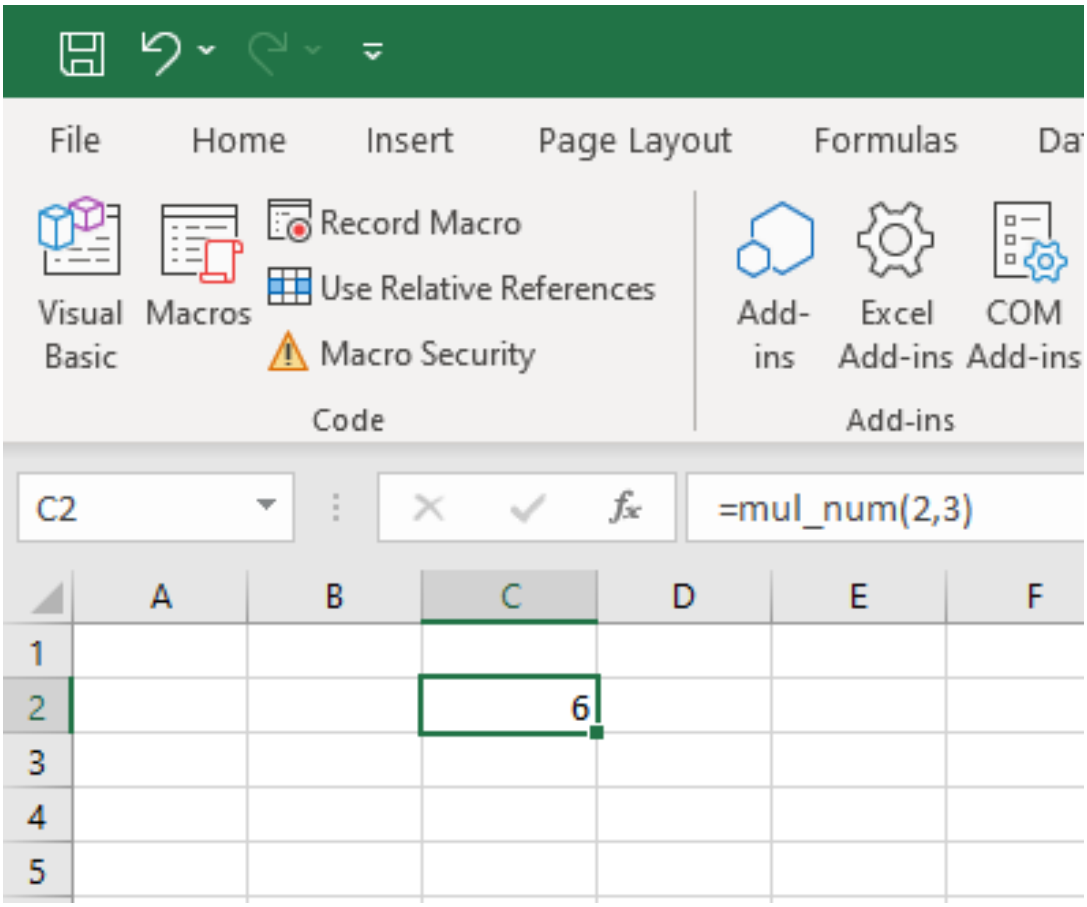
6. What is UDF? Why are UDF's used? Create a UDF to multiply 2 numbers in VBA

A User Defined Function is a procedure (a group of commands) written in VBA that (usually) accepts inputs and returns a result. A UDF cannot modify the formatting of a cell or workbook or move values around on a worksheet.

User-defined functions are functions that you use to organize your code in the body of a policy. Once you define a function, you can call it in the same way as the built-in action and parser functions. Variables that are passed to a function are passed by reference, rather than by value.

VBA Code:

```
Function mul_num(x As Integer, y As Integer) As Integer
mul_num = x * y
End Function
```




iNeuron