**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**

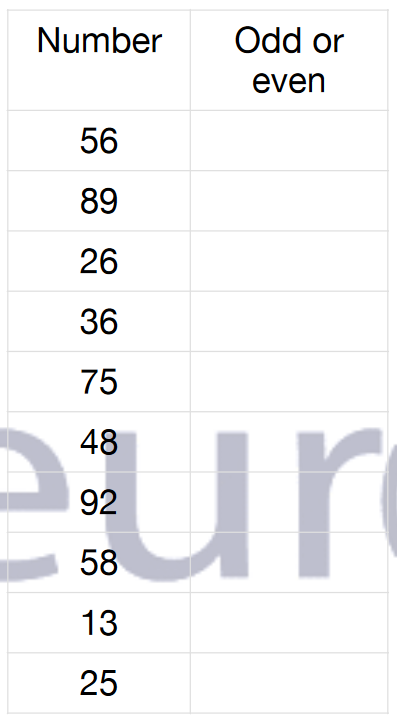
**VBA”.**

Sub SelectRange()

Range(“A5:C10”).Select

Range(“A5:C10”).Value=”This is Excel VBA”

End Sub

****

**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 CheckEvenOrOddNumber()

Dim p As Range

For Each p In Worksheets("Sheet1").Range("A2:A11")

If IsNumeric(p.Value) Then

If p.Value Mod 2 = 0 Then

p.Offset(0, 1).Value = "Even"

Else

p.Offset(0, 1).Value = "Odd"

End If

End If

Next p

End Sub

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

There are four types of errors in Excel VBA:

1. Syntax errors
2. Compilation errors
3. Runtime errors
4. Logical Errors

A ***Syntax error***, as the name suggests, occurs when VBA finds something wrong with the syntax in the code.

***Compile errors*** occur when something is missing that is needed for the code to run.

***Runtime errors*** are those that occur when the code is running.

***Logical errors*** would not make your code stop but can lead to wrong results.

**4. How do you handle Runtime errors in VBA?**

 To deal with runtime errors, we trap (catch) the errors, handle them, and then resume execution after the error is handled:

[Trapping Runtime Errors](https://www.microfocus.com/documentation/infoconnect-desktop/17-1/vba-guide/HandlingRuntimeErrors.html#trapping)

[Handling Errors Inline](https://www.microfocus.com/documentation/infoconnect-desktop/17-1/vba-guide/HandlingRuntimeErrors.html#handling)

[Using Error Handling Routines](https://www.microfocus.com/documentation/infoconnect-desktop/17-1/vba-guide/HandlingRuntimeErrors.html#routines)

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

**errors**

Here are some best practices you can use when it comes to error handling in Excel VBA.

1. Use ‘**On Error Go [Label]**’ at the beginning of the code. This will make sure any error that can happen from there is handled.
2. Use **‘On Error Resume Next’** ONLY when you’re sure about the errors that can occur. Use it with expected error only. In case you use it with unexpected errors, it will simply ignore it and move forward. You can use ‘On Error Resume Next’ with ‘**Err.Raise’** if you want to ignore a certain type of error and catch the rest.
3. When using error handlers, make sure you’re using **Exit Sub** before the handlers. This will ensure that the error handler code is executed only when there is an error (else it will always be executed).
4. Use multiple error handlers to trap different kinds of errors. Having multiple error handler ensures that an error is properly addressed. For example, you would want to handle a ‘type mismatch’ error differently than a ‘Division by 0’ run-time error.

**6. What is UDF? Why are UDF’s used? Create a UDF to multiply 2**

**numbers in VBA.**

Sometime Excel function are not enough to handle User requirement. In such cases, with VBA users can create custom functions to meet their requirement. These functions are called User Defined Functions (UDF).

Sub Multiply\_Numbers()

Dim Number\_1 As Integer

Dim Number\_2 As Integer

Dim Number\_3 As Integer

Number\_1 = 10

Number\_2 = 5

Answer = Number\_1 \* Number\_2

End Sub