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”

Number Odd or even

56
89
26
36
75
48
92
58
13
25

Ans.
```
Sub createDataAnalytics()

    ' Select the range A5 to C10
    Range("A5:C10").Select

    ' Name the range "Data Analytics"
    ActiveWorkbook.Names.Add Name:="Data Analytics", RefersTo:=Selection

    ' Fill the range with the specified data
    Selection.Value = "This is Excel VBA"
    Selection.Offset(0, 1).Value = "Number"
    Selection.Offset(0, 2).Value = "Odd or Even"

    Selection.Offset(1, 0).Value = 56
    Selection.Offset(2, 0).Value = 89
    Selection.Offset(3, 0).Value = 26
    Selection.Offset(4, 0).Value = 36
    Selection.Offset(5, 0).Value = 75
    Selection.Offset(6, 0).Value = 48
    Selection.Offset(7, 0).Value = 92
    Selection.Offset(8, 0).Value = 58
    Selection.Offset(9, 0).Value = 13
    Selection.Offset(10, 0).Value = 25

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

```
Sub classifyNumbers()
 Dim dataRange As Range
   Set dataRange = Range("Data Analytics")

   ' Loop through each cell in the data range and classify the numbers as even or odd
   For Each cell In dataRange.Offset(1).Resize(dataRange.Rows.Count - 1, 1)
      If cell.Value Mod 2 = 0 Then
         cell.Offset(0, 2).Value = "Even"
      Else
         cell.Offset(0, 2).Value = "Odd"
      End If
   Next cell

End Sub
```

b. Select Case statement

```
Sub classifyNumbers()

   Dim dataRange As Range
   Set dataRange = Range("Data Analytics")

   ' Loop through each cell in the data range and classify the numbers as even or odd
   For Each cell In dataRange.Offset(1).Resize(dataRange.Rows.Count - 1, 1)
      Select Case cell.Value Mod 2
         Case 0
            cell.Offset(0, 2).Value = "Even"
         Case 1
            cell.Offset(0, 2).Value = "Odd"
      End Select
   Next cell

End Sub
```

c. For Next Statement

```
Sub classifyNumbers()

   Dim dataRange As Range
   Set dataRange = Range("Data Analytics")

   ' Loop through each cell in the data range and classify the numbers as even or odd
   For i = 2 To dataRange.Rows.Count
      If dataRange.Cells(i, 1).Value Mod 2 = 0 Then
         dataRange.Cells(i, 3).Value = "Even"
      Else
         dataRange.Cells(i, 3).Value = "Odd"
      End If
   Next i

End Sub
```

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

Ans. In VBA, there are three types of errors that you can encounter:

Syntax errors

Runtime errors

Logic errors

4. How do you handle Runtime errors in VBA?

Ans. Runtime errors in VBA can be handled using error handling techniques, which allow you to gracefully handle errors that occur during code execution. There are two main methods for handling runtime errors in VBA:

On Error statement
Err object

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

Ans. following can be good practises:

- Always use error handling code in your procedures to catch and handle errors that may occur.
- Use the "On Error" statement to enable error handling in your procedure.
- Use descriptive and meaningful error messages that provide information about the error that occurred and how to resolve it.
- Use "Debug.Print" statements to output information about errors to the Immediate Window for debugging purposes.
- Use "Err.Raise" to raise custom errors that can be caught and handled in your error handling code.
- Use "Resume" statements to continue execution of your code after an error has occurred, but make sure you handle the error appropriately first.
- Always test your error handling code to ensure it works as expected and catches all possible errors.
- Keep your error handling code organized and separated from your main code for better readability and maintainability.
- Use comments to document your error handling code and explain what each section does.
- Use tools like VBA code analyzers to detect potential errors in your code and suggest best practices for error handling.