ASSIGNMENT-20 SOLUTIONS

1.
```
Sub Macro2()
    Application.Goto Reference:="R5C1:R10C3"
    ActiveWorkbook.Names.Add Name:="DataAnalytics", RefersToR1C1:= _
        "=Sheet1!R5C1:R10C3"
    Selection.FormulaR1C1 = "This is Excel VBA"
End Sub
```

2. The required VBA codes are:

a.
```
Sub Macro2()
    If Range("A2").Value Mod 2 = 0 Then
        Range("B2").Value = "Even"
    Else
        Range("B2").Value = "Odd"
    End If
    If Range("A3").Value Mod 2 = 0 Then
        Range("B3").Value = "Even"
    Else
        Range("B3").Value = "Odd"
    End If
    If Range("A4").Value Mod 2 = 0 Then
        Range("B4").Value = "Even"
    Else
        Range("B4").Value = "Odd"
    End If
    If Range("A5").Value Mod 2 = 0 Then
        Range("B5").Value = "Even"
    Else
        Range("B5").Value = "Odd"
    End If
    If Range("A6").Value Mod 2 = 0 Then
        Range("B6").Value = "Even"
    Else
        Range("B6").Value = "Odd"
    End If
    If Range("A7").Value Mod 2 = 0 Then
        Range("B7").Value = "Even"
    Else
        Range("B7").Value = "Odd"
    End If
    If Range("A8").Value Mod 2 = 0 Then
        Range("B8").Value = "Even"
    Else
        Range("B8").Value = "Odd"
    End If
    If Range("A9").Value Mod 2 = 0 Then
        Range("B9").Value = "Even"
```

```
    Else
        Range("B9").Value = "Odd"
    End If
    If Range("A10").Value Mod 2 = 0 Then
        Range("B10").Value = "Even"
    Else
        Range("B10").Value = "Odd"
    End If
    If Range("A11").Value Mod 2 = 0 Then
        Range("B11").Value = "Even"
    Else
        Range("B11").Value = "Odd"
    End If
End Sub
```

b.

```
Sub Macro2()
    Select Case Range("A2").Value Mod 2    ' Evaluate Number.
    Case 0
        Range("B2").Value = "Even"
    Case Else
        Range("B2").Value = "Odd"
    End Select

    Select Case Range("A3").Value Mod 2    ' Evaluate Number.
    Case 0
        Range("B3").Value = "Even"
    Case Else
        Range("B3").Value = "Odd"
    End Select

    Select Case Range("A4").Value Mod 2    ' Evaluate Number.
    Case 0
        Range("B4").Value = "Even"
    Case Else
        Range("B4").Value = "Odd"
    End Select

    Select Case Range("A5").Value Mod 2    ' Evaluate Number.
    Case 0
        Range("B5").Value = "Even"
    Case Else
        Range("B5").Value = "Odd"
    End Select

    Select Case Range("A6").Value Mod 2    ' Evaluate Number.
    Case 0
        Range("B6").Value = "Even"
    Case Else
```

```
    Range("B6").Value = "Odd"
End Select

Select Case Range("A7").Value Mod 2    ' Evaluate Number.
Case 0
    Range("B7").Value = "Even"
Case Else
    Range("B7").Value = "Odd"
End Select

Select Case Range("A8").Value Mod 2    ' Evaluate Number.
Case 0
    Range("B8").Value = "Even"
Case Else
    Range("B8").Value = "Odd"
End Select

Select Case Range("A9").Value Mod 2    ' Evaluate Number.
Case 0
    Range("B9").Value = "Even"
Case Else
    Range("B9").Value = "Odd"
End Select

Select Case Range("A10").Value Mod 2    ' Evaluate Number.
Case 0
    Range("B10").Value = "Even"
Case Else
    Range("B10").Value = "Odd"
End Select

Select Case Range("A11").Value Mod 2    ' Evaluate Number.
Case 0
    Range("B11").Value = "Even"
Case Else
    Range("B11").Value = "Odd"
End Select

End Sub
```

c. Sub Macro2()

```
For r = 2 To 11
    If Cells(r, 1).Value Mod 2 = 0 Then
        Cells(r, 2).Value = "Even"
    Else
        Cells(r, 2).Value = "Odd"
    End If
```

```
Next r

End Sub
```

3. There are three types of errors in VBA:
(a) Syntax Errors
(b) Runtime Errors
(c) Logical Errors.

4. Runtime errors occur when macro runs, and typically result from specific conditions present at that time.
We should always include some form of error handling in our macros to deal with runtime errors. Without any error handling, a runtime error causes a macro to stop immediately, and gives the user little information. To deal with runtime errors, you'll need to trap (catch) the errors, handle them, and then resume execution after the error is handled.

5. Here are some best practices we can use when it comes to error handling in Excel VBA:
i) Use 'On Error Go [Label]' at the beginning of the code. This will make sure any error that can happen from there is handled.
ii) 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.
iii) 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).
iv) 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. 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.UDF's are used to perform some customized functions which are not available by default in excel as a formula.
UDF to multiply two numbers:

```
Function UserMul(a,b)
UserMul=a*b
End Function
```