Q1. Explain the purpose of conditional statements in VBA. Write an example using If…ElseIf…Else that classifies sales performance as: "Excellent" if sales ≥ 1000, "Good" if sales ≥ 500, "Needs Improvement" otherwise.

Answer:

Conditional statements help VBA make decisions by executing specific blocks of code based on certain conditions.

```
Sub ClassifySales()
    Dim sales As Double
    sales = Range("B2").Value

    If sales >= 1000 Then
        MsgBox "Excellent"
    ElseIf sales >= 500 Then
        MsgBox "Good"
    Else
        MsgBox "Needs Improvement"
    End If
End Sub
```

Q2. Write a VBA code that calculates the total sales from the range B2:B10 and displays the result in a message box. Explain the purpose of each line of code used in the program.

Answer:

```
Sub TotalSalesB2toB10()
    Dim total As Double
    total =
Application.WorksheetFunction.Sum(Range("B2:B10"))
```

```
    MsgBox "Total Sales: " & total
End Sub
```

- `Dim total As Double` — declares a variable to store total.
- `Application.WorksheetFunction.Sum(Range("B2:B10"))` — calculates the sum of all cells in the range.
- `MsgBox` — displays the result in a message box.

Q3. Explain the use of loops in VBA programming. Write a For…Next loop that fills cells A1 to A5 with the values 10, 20, 30, 40, and 50 respectively.

Answer:
Loops repeat a set of actions multiple times without writing duplicate code.

```
Sub FillA1toA5()
    Dim i As Integer
    For i = 1 To 5
        Cells(i, 1).Value = i * 10
    Next i
End Sub
```

Q4. Discuss the importance of error handling in VBA. Write a short example using On Error GoTo to handle division by zero, and explain how the program flow changes when an error occurs.

Answer:
Error handling prevents program crashes and provides user-friendly messages.

```
Sub SafeDivide()
    On Error GoTo ErrHandler
```

```
    Dim x As Double, y As Double, result As Double

    x = 10
    y = 0
    result = x / y

    MsgBox "Result: " & result
    Exit Sub

ErrHandler:
    MsgBox "Error: Division by zero is not allowed."
End Sub
```

When an error occurs, VBA jumps to `ErrHandler`, shows a message, and safely exits the procedure.

Q5. Describe the role of UserForms in VBA for Excel. Mention two common controls used in UserForms and explain how they help in user interaction.

Answer:
UserForms allow users to interact with VBA programs through a graphical interface for data entry and commands.

Two common controls:

1. **TextBox:** Accepts user input such as names or numbers.

2. **CommandButton:** Executes code when clicked (e.g., saves data or performs calculations).

Q6. Differentiate between dynamic arrays and fixed arrays in VBA. Provide syntax examples to declare and resize each type.

Answer:

Aspect	Fixed Array	Dynamic Array
Declaration	`Dim num(1 To 5) As Integer`	`Dim num() As Integer`
Resizing	Not resizable	`ReDim num(1 To 5)`
Preserve Data	Not needed	`ReDim Preserve num(1 To 10)`

Example:

```
Dim num() As Integer
ReDim num(1 To 5)
num(1) = 10
ReDim Preserve num(1 To 10)
```

Q7. What are User Defined Functions (UDFs) in VBA? Write a VBA function named DiscountPrice that calculates and returns a 10% discount on a given price.

Answer:

UDFs are custom functions created by users to perform calculations in Excel.

```
Function DiscountPrice(price As Double) As Double
    DiscountPrice = price * 0.9
End Function
```

Used in Excel as:
`=DiscountPrice(200)` → Returns **180**

Q8. Explain the difference between properties and methods in the Excel Object Model with suitable examples of each.

Answer:

Type	Description	Example
Property	Attribute or characteristic of an object	`Range("A1").Value = 100`
Method	Action performed by an object	`Range("A1").ClearContents`

Q9. Describe the use of the With…End With statement in VBA. Explain how it improves readability and efficiency, using a formatting example for range A1:D1.

Answer:
`With…End With` allows performing multiple actions on the same object efficiently.

```
Sub FormatHeader()
    With Range("A1:D1")
        .Font.Bold = True
        .Interior.Color = RGB(200, 200, 255)
        .HorizontalAlignment = xlCenter
    End With
End Sub
```

It improves readability by avoiding repetition of the same object reference.

Q10. Write a VBA code that creates a new worksheet named “Summary”, inserts headers “Region”, “Sales”, and “Target” in A1:C1, and formats the header row to be bold and center-aligned. Explain how the code uses Excel’s object model.

Answer:

```
Sub CreateSummary()
    Dim ws As Worksheet
    Set ws = ThisWorkbook.Worksheets.Add
    ws.Name = "Summary"

    With ws.Range("A1:C1")
        .Value = Array("Region", "Sales", "Target")
        .Font.Bold = True
        .HorizontalAlignment = xlCenter
    End With
End Sub
```

This code uses:

- **Workbook object** → to add a new worksheet.
- **Worksheet object** → to rename and access cells.
- **Range object** → to insert and format headers.