1. Write a VBA code to enter your name in A1 Cell using Input Box and once you enter the name display a message box that says the name has been entered.

Answer: Sub EnterName()

Dim userName As String

' Prompt the user to enter their name using an input box

userName = InputBox("Please enter your name:", "Enter Name")

' Check if the user entered a name

If userName <> "" Then

' Enter the name into cell A1

Range("A1").Value = userName

' Display a message box confirming that the name has been entered

MsgBox "Your name, " & userName & ", has been entered in cell A1.", vbInformation

Else

' Display a message box if the user did not enter a name

MsgBox "No name entered.", vbExclamation

End If

End Sub

1. What are Userforms? Why are they used? How to fill a list box using for loop.

Answer: Userforms are custom dialog boxes or forms that you can create in VBA. They allow you to create custom interfaces for users to interact with your Excel application. Userforms can contain various controls such as text boxes, list boxes, combo boxes, buttons, checkboxes, and more, enabling users to input data, make selections, or trigger actions within your Excel application.

Userforms are used for various purposes, including:

Data Entry: You can create userforms to collect data from users in a structured and user-friendly manner.

Data Display: Userforms can display information or reports to users in a more organized and visually appealing format compared to worksheets.

User Interaction: Userforms allow users to interact with your Excel application, providing a more intuitive and guided user experience.

Custom Tools: Userforms can host custom tools or utilities, enabling users to perform specific tasks or operations within your Excel application.

To fill a list box using a for loop in VBA, you first need to add a list box control to your userform and then write VBA code to populate the list box with items. Here's how you can do it:

Add a List Box to the Userform:

Open the VBA editor by pressing Alt + F11.

Insert a new userform by going to Insert > Userform.

Drag and drop a list box control from the toolbox onto the userform.

Write VBA Code to Fill the List Box:

Double-click on the userform to open its code window.

Write VBA code to populate the list box using a for loop. Here's an example code that fills the list box with numbers from 1 to 10:

Private Sub UserForm\_Initialize()

Dim i As Integer

' Fill the list box with numbers from 1 to 10 using a for loop

For i = 1 To 10

ListBox1.AddItem i

Next i

End Sub

Replace ListBox1 with the name of your list box control if it's different.

Run the Userform:

Close the code window and return to Excel.

Run the userform by going to View > Userform.

The list box should be filled with numbers from 1 to 10 when the userform is displayed.

1. What is an array? Write a VBA code to enter students and their marks from the below table.

Answer: An array is a data structure that can hold multiple values of the same data type under a single variable name. In VBA, arrays are used to store collections of values that are related or belong together. Arrays can be one-dimensional, two-dimensional, or multi-dimensional, depending on the number of indices used to access the elements.

Sub EnterStudentsAndMarks()

Dim studentData() As Variant

Dim ws As Worksheet

Dim numRows As Integer

Dim i As Integer

' Assuming the data is stored in a worksheet named "StudentData" starting from cell A1

Set ws = ThisWorkbook.Sheets("StudentData")

' Determine the number of rows in the table

numRows = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row

' Resize the array to accommodate the data

ReDim studentData(1 To numRows, 1 To 2)

' Enter students and their marks into the array

For i = 1 To numRows

studentData(i, 1) = ws.Cells(i, 1).Value ' Assuming student names are in column A

studentData(i, 2) = ws.Cells(i, 2).Value ' Assuming marks are in column B

Next i

' Example: Display the array content in the Immediate Window (Ctrl + G to open)

For i = 1 To numRows

Debug.Print "Student: " & studentData(i, 1) & ", Marks: " & studentData(i, 2)

Next i

End Sub

This code does the following:

Sets a reference to the worksheet containing the student data.

Determines the number of rows in the table.

Resizes the array to accommodate the data.

Enters the student names and their marks into the array.

Prints the student names and their marks from the array to the Immediate Window for demonstration purposes.

1. Use the following data to create a pie chart using VBA code. Use Font - ‘Times new Roman’, Size -14, Bold, Title - Piechart’ and you are per to use colours as per your taste.

Answer:

Sub CreatePieChart()

Dim ws As Worksheet

Dim chartObj As ChartObject

Dim chartDataRange As Range

Dim chartTitle As String

Dim i As Integer

' Set the worksheet where the chart will be created

Set ws = ThisWorkbook.Sheets("Sheet1") ' Change "Sheet1" to your sheet name

' Define the data range for the chart

Set chartDataRange = ws.Range("A1:B5") ' Assuming your data range is from A1:B5

' Add a new chart object to the worksheet

Set chartObj = ws.ChartObjects.Add(Left:=100, Width:=375, Top:=75, Height:=225)

' Set the chart type to Pie

chartObj.Chart.ChartType = xlPie

' Set the chart data source

chartObj.Chart.SetSourceData Source:=chartDataRange

' Set chart title and format

chartTitle = "Piechart"

chartObj.Chart.HasTitle = True

chartObj.Chart.ChartTitle.Text = chartTitle

chartObj.Chart.ChartTitle.Font.Name = "Times New Roman"

chartObj.Chart.ChartTitle.Font.Size = 14

chartObj.Chart.ChartTitle.Font.Bold = True

' Set font properties for data labels

For i = 1 To chartObj.Chart.SeriesCollection(1).Points.Count

chartObj.Chart.SeriesCollection(1).Points(i).DataLabel.Font.Name = "Times New Roman"

chartObj.Chart.SeriesCollection(1).Points(i).DataLabel.Font.Size = 14

chartObj.Chart.SeriesCollection(1).Points(i).DataLabel.Font.Bold = True

Next i

' Format colors as per your taste

chartObj.Chart.SeriesCollection(1).Points(1).Interior.Color = RGB(255, 0, 0) ' Red

chartObj.Chart.SeriesCollection(1).Points(2).Interior.Color = RGB(0, 255, 0) ' Green

chartObj.Chart.SeriesCollection(1).Points(3).Interior.Color = RGB(0, 0, 255) ' Blue

chartObj.Chart.SeriesCollection(1).Points(4).Interior.Color = RGB(255, 255, 0) ' Yellow

chartObj.Chart.SeriesCollection(1).Points(5).Interior.Color = RGB(255, 0, 255) ' Magenta

' Optionally, you can adjust the position and size of the chart object

' chartObj.Left = 100

' chartObj.Top = 75

' chartObj.Width = 375

' chartObj.Height = 225

End Sub

1. Check the dataset in the link given below and create a pivot table using VBA showing the sales for the year from stationary category.

<https://docs.google.com/spreadsheets/d/1IRSEnmgz8Ro276GslknRNk0zlrB5CZH1YrnT71kqFM/edit?usp=sharing>

Answer:

Sub CreatePivotTable()

Dim ws As Worksheet

Dim pt As PivotTable

Dim ptCache As PivotCache

Dim ptRange As Range

' Assuming your data is stored in a worksheet named "SalesData"

Set ws = ThisWorkbook.Sheets("SalesData")

' Define the range containing your data (adjust the range as per your data)

Set ptRange = ws.Range("A1:D100") ' Assuming your data range is from A1 to D100

' Create a new pivot cache

Set ptCache = ThisWorkbook.PivotCaches.Create( \_

SourceType:=xlDatabase, \_

SourceData:=ptRange)

' Create a new pivot table on a new worksheet

Set pt = ptCache.CreatePivotTable( \_

TableDestination:=ws.Cells(1, 6), \_

TableName:="PivotTable1")

' Add fields to the pivot table

With pt

.PivotFields("Year").Orientation = xlRowField

.PivotFields("Category").Orientation = xlRowField

.PivotFields("Sales").Orientation = xlDataField

End With

End Sub

1. Write step by step procedure to protect your workbook using a password.

Answer: To protect your workbook using a password in Excel, follow these step-by-step procedures:

Open the Workbook: Open the Excel workbook that you want to protect with a password.

Navigate to the Protect Workbook Option:

Click on the "File" tab in the top-left corner of the Excel window.

In the File menu, click on "Info" from the left sidebar.

Click on the "Protect Workbook" dropdown arrow located on the right side of the screen.

Choose Encrypt with Password:

From the "Protect Workbook" dropdown menu, select the "Encrypt with Password" option.

Enter the Password:

A dialog box titled "Encrypt Document" will appear.

In the dialog box, enter the password you want to use to protect the workbook. Make sure to choose a strong password that is difficult to guess.

After entering the password, click on the "OK" button.

Confirm the Password:

In a new dialog box titled "Confirm Password", re-enter the same password to confirm it.

Make sure to enter the password exactly as you did in the previous step.

After confirming the password, click on the "OK" button.

Save the Workbook:

Once the password is set, save the workbook to apply the protection.

Click on the "File" tab again and select "Save As" to save the workbook with the password protection.

Close and Re-Open the Workbook:

Close the workbook and then re-open it to test the password protection.

When you try to open the workbook, Excel will prompt you to enter the password. Enter the correct password to access the workbook.