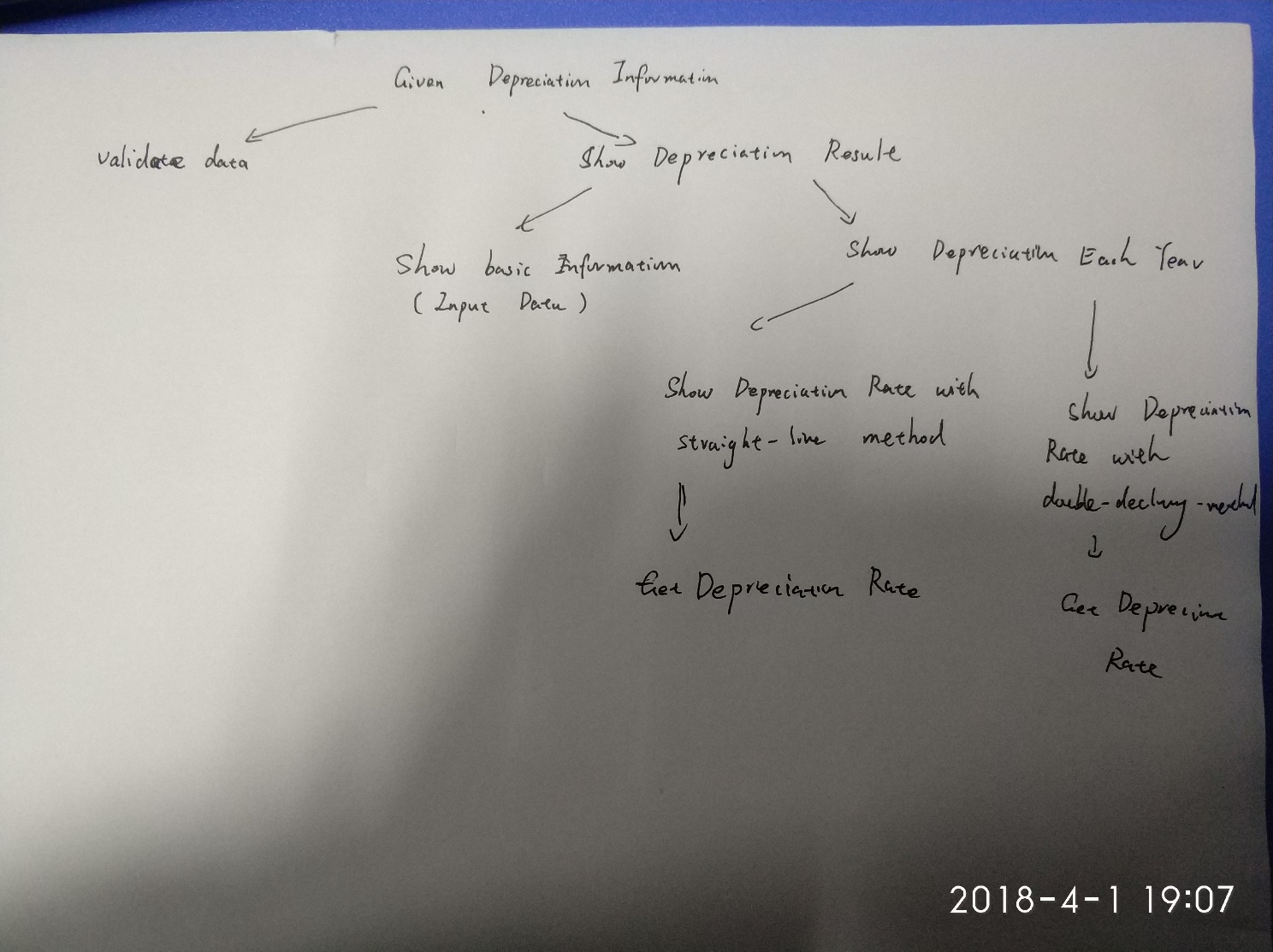
1. **A Structured Diagram**



1. **Pseudocode**

Click button Calcalate (Sub procedure btnCalculate\_Click)

VALIDATE data (Function CheckData)

SHOW depreciation result (Sub procedure ShowDepreciationResult)

SHOW basic info (Sub procedure ShowBasicInfo)

If Method = straight-line

Compute depreciation rate (Function GetDepreciation Rate)

SHOW Depreciation with straight-line Method (Sub procedure ShowSLDepreciation)

For 1 to usageLife

Calculate depreciation value, current value based on depreciation method

OUTPUT each year depreciation info

Else

Compute depreciation rate (Function GetDepreciation Rate)

SHOW Depreciation with double-declining Method (Sub procedure ShowDDBDepreciation

For 1 to usageLife

Calculate depreciation value, current value based on depreciation method

OUTPUT each year depreciation info

Click Button Clear (Sub procedure btnClear\_Click)

Clear all the text box and reset the radio box

Click Button Exit (Sub procedure btnExit\_Click)

Exit Application

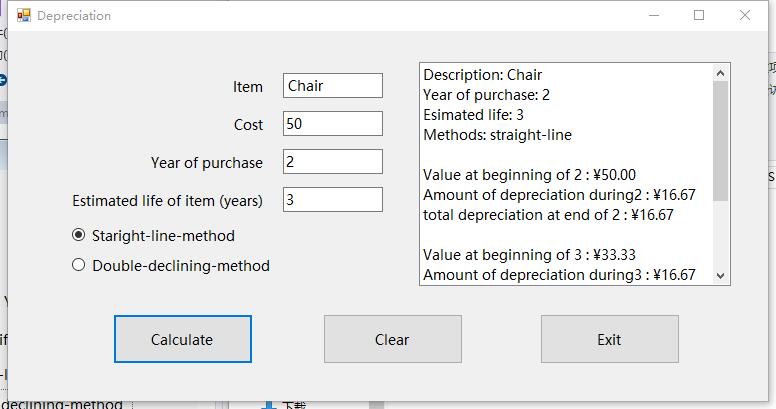
1. **Object table**

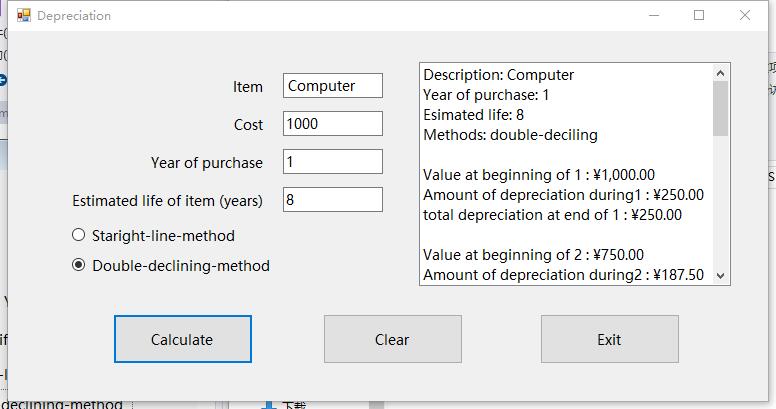
| **Object** | **Property** | **Setting** |
| --- | --- | --- |
| lblCost | Text | Item price |
| lblItem | Text | Item name |
| lblYearOfPurchase | Text | Purchase year |
| lblEstimatedLifeOfItem | Text | Item‘s estimated life |
| txtItem | Text | Item name |
| txtCost | Text | Item price |
| txtYearOfPurchase | Text | Purchase year |
| txtEstimatedLifeOfItem | Text | Item‘s estimated life |
| radStraight | Text | Straight-line method |
| radStraight | Checked | True |
| radDouble | Text | Double-declining  method |
| radDouble | Checked | False |
| lstResult | Text | Show info |
| btnCalculate | Text | Calculate value |
| btnClear | Text | Empty value |
| btnExit | Text | Exit procedure |
| grpMethods | Text | Depreciation Methods |

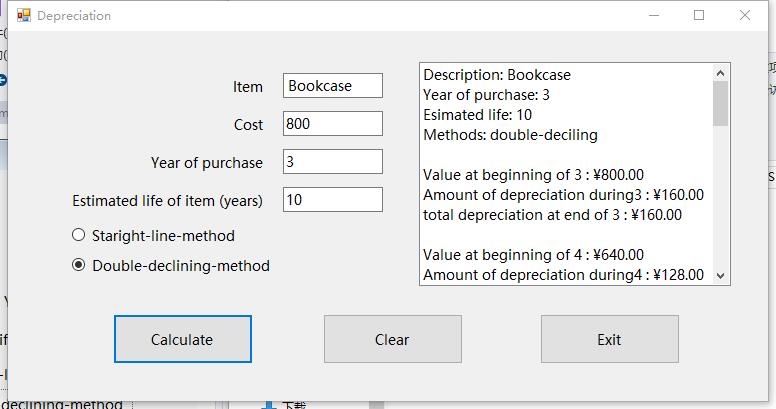
**3. Table of Products and details**

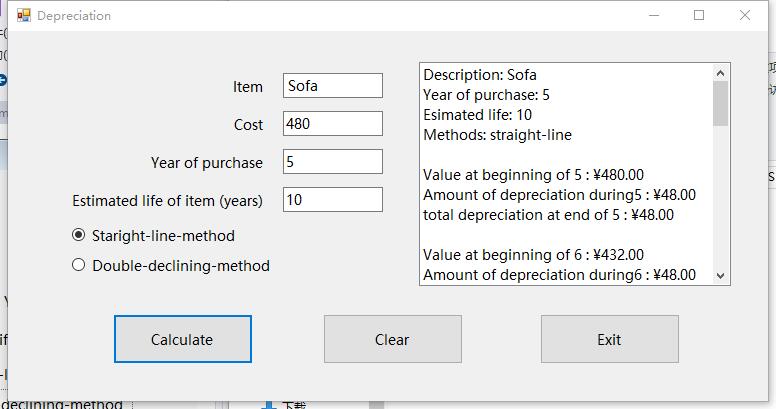
| **Product Name** | **Original Cost** | **Years of purchase** | **Estimated Life of Product** | **Depreciation Type** |
| --- | --- | --- | --- | --- |
| Chair | $50 | 2 | 3 | Straight line |
| Computer | $1000 | 1 | 8 | Double-declining |
| Bookcase | $800 | 3 | 10 | Double-declining |
| Sofa | $480 | 5 | 10 | Straight line |

1. **Deliver 4 Printouts**









**5. Coding Design**

Public Class Depreciation

' Task: Check data format

' Check wheter the data format is correct.

' Return a string to show the specific problem of current format

Function CheckData() As String

If txtItem.Text = "" Then

Return "Please enter right item value!"

ElseIf txtCost.Text = "" Or IsNumeric(txtCost.Text) = False Then

Return "Please enter right cost value!"

ElseIf txtYearOfPurchase.Text = "" Or IsNumeric(txtYearOfPurchase.Text) = False Then

Return "Please enter right Tear of Purchase value!"

ElseIf txtEstimatedLifeOfItem.Text = "" Or IsNumeric(txtEstimatedLifeOfItem.Text) = False Then

Return "Please enter right Estimated Life of Item!"

ElseIf CDbl(txtEstimatedLifeOfItem.Text) = 0 Then

Return "Please enter a Estimated Life Of Item value bigger than 0 !!"

End If

' if all the format is right, return a string OK

Return "OK"

End Function

' Task: Show basic info

' Show the input data in the listbox

' Including item name, year of purchase, esimated life and depreciation method

Private Sub ShowBasicInfo(ByVal startYear As String, ByVal usageLife As String, ByVal itemName As String, ByVal depreciationMethod As String)

lstResult.Items.Clear()

lstResult.Items.Add("Description: " & itemName)

lstResult.Items.Add("Year of purchase: " & startYear)

lstResult.Items.Add("Esimated life: " & usageLife)

lstResult.Items.Add("Methods: " & depreciationMethod)

lstResult.Items.Add("")

End Sub

' Task: Compute depreciation rate

' Check the depreciation method (straight-line or double-declining-balance)

' Return depreciation based on method

Function GetDepreciationRate(ByVal usageLife As Double, ByVal depreciationMethod As String) As Double

Dim depreciationRate As Double

If depreciationMethod = "straight-line" Then

depreciationRate = 1.0 / usageLife

Else

depreciationRate = 2.0 / usageLife

End If

Return depreciationRate

End Function

' Task: Show straight-line method depreciation each year

' Show the depreciation infomation each year in the listbox (using straight-line method)

' Including value at beginning of the year, Amount of depreciation during the year and total depreciation at end of the year

Private Sub ShowSLDepreciation(ByVal startYear As Integer, ByVal usageLife As Double, ByVal itemValue As Double, ByVal itemName As String, ByVal depreciationMethod As String)

Dim depreciationRate, depreciationValue, totalDepreciationValue As Double

Dim currentValue As Double = itemValue

' get depreciation rate

depreciationRate = GetDepreciationRate(usageLife, depreciationMethod)

' staright-line method has a const depreciation value each year

depreciationValue = depreciationRate \* itemValue

' init total depreciation as 0

totalDepreciationValue = 0

For index = 1 To usageLife

If currentValue <= 0 Then

Exit For

End If

lstResult.Items.Add("Value at beginning of " & CStr(startYear) & " : " & currentValue.ToString("C"))

' compute total depreciation and current value

currentValue -= depreciationValue

totalDepreciationValue += depreciationValue

lstResult.Items.Add("Amount of depreciation during" & CStr(startYear) & " : " & depreciationValue.ToString("C"))

lstResult.Items.Add("total depreciation at end of " & CStr(startYear) & " : " & totalDepreciationValue.ToString("C"))

lstResult.Items.Add("")

startYear += 1

Next

End Sub

' Task: Show double-declining-balance method depreciation each year

' Show the depreciation infomation each year in the listbox (using double-declining-balance method)

' Including value at beginning of the year, Amount of depreciation during the year and total depreciation at end of the year

Private Sub ShowDDBDepreciation(ByVal startYear As Integer, ByVal usageLife As Double, ByVal itemValue As Double, ByVal itemName As String, ByVal depreciationMethod As String)

Dim depreciationRate, depreciationValue, totalDepreciationValue As Double

Dim currentValue As Double = itemValue

' get depreciation rate

depreciationRate = GetDepreciationRate(usageLife, depreciationMethod)

' init total depreciation as 0

totalDepreciationValue = 0

For index = 1 To usageLife

If currentValue <= 0 Then

Exit For

End If

lstResult.Items.Add("Value at beginning of " & CStr(startYear) & " : " & currentValue.ToString("C"))

' staright-line method has a changing depreciation value each year

depreciationValue = currentValue \* depreciationRate

' compute total depreciation and current value

totalDepreciationValue += depreciationValue

currentValue -= depreciationValue

lstResult.Items.Add("Amount of depreciation during" & CStr(startYear) & " : " & depreciationValue.ToString("C"))

lstResult.Items.Add("total depreciation at end of " & CStr(startYear) & " : " & totalDepreciationValue.ToString("C"))

lstResult.Items.Add("")

startYear += 1

Next

End Sub

' Task: Show depreciation result

' Show depreciation in the listbox based on the input data

' Result includes basic info and each year depreciation

Private Sub ShowDepreciationResult(ByVal startYear As Integer, ByVal usageLife As Double, ByVal itemValue As Double, ByVal itemName As String, ByVal depreciationMethod As String)

' Show basic info

ShowBasicInfo(startYear, CStr(usageLife), CStr(itemName), depreciationMethod)

' Show each year depreciation base on depreciation method

If depreciationMethod = "straight-line" Then

ShowSLDepreciation(startYear, usageLife, itemValue, itemName, depreciationMethod)

Else

ShowDDBDepreciation(startYear, usageLife, itemValue, itemName, depreciationMethod)

End If

End Sub

' Event: Click Button Calculate

' Get the depreciation result

Private Sub btnCalculate\_Click(sender As Object, e As EventArgs) Handles btnCalculate.Click

Dim method As String

If CheckData() = "OK" Then

' check and get the depreciation method

If radStraight.Checked = True Then

method = "straight-line"

Else

method = "double-declining-balance"

End If

' compute and show the depreciation result

ShowDepreciationResult(CInt(txtYearOfPurchase.Text), CDbl(txtEstimatedLifeOfItem.Text), CDbl(txtCost.Text), txtItem.Text, method)

Else

' return a warning and show where the input is wrong

lstResult.Items.Clear()

MessageBox.Show(CheckData(), "Warning")

End If

End Sub

' Event: Click Button Clear

' Clear all the textbox and reset the radiobox

Private Sub btnClear\_Click(sender As Object, e As EventArgs) Handles btnClear.Click

txtCost.Clear()

txtItem.Clear()

txtYearOfPurchase.Clear()

txtEstimatedLifeOfItem.Clear()

lstResult.Items.Clear()

radDouble.Checked = False

radStraight.Checked = True

End Sub

' Event: Click Button Exit

' Exit the application

Private Sub btnExit\_Click(sender As Object, e As EventArgs) Handles btnExit.Click

End

End Sub

End Class