Chapter 8 GPD CIS-233 01

chef shack inventory depreciation application

This application opens the inventory text file so that the user can select an item and the type of depreciation method. The depreciation is calculated for the five-year life of the inventory item.

**4/22/2022**

Jeffrey Annan Koranteng

.Net I Program

Contents

[Snapshots 2](#_Toc101550202)

[Compute Depreciation by Method Initial Form 2](#_Toc101550203)

[Straight-Line Method Calculation for an Item 2](#_Toc101550204)

[Double-Declining Method Calculation for an Item 3](#_Toc101550205)

[Sorted Inventory Listing (DisplayInventory) Form 3](#_Toc101550206)

[Compute Depreciation Form mnuClear clicked Screen 4](#_Toc101550207)

[Missing Item Selection Message Box 4](#_Toc101550208)

[Missing Depreciation Method MsgBox 4](#_Toc101550209)

[Source Code 4](#_Toc101550210)

[frmDepreciation 4](#_Toc101550211)

[frmDisplayInventory 8](#_Toc101550212)

## Snapshots

### Compute Depreciation by Method Initial Form

Graphical user interface, text

Description automatically generated

### Straight-Line Method Calculation for an Item

Table

Description automatically generated

### Double-Declining Method Calculation for an Item

Table

Description automatically generated

### Sorted Inventory Listing (DisplayInventory) Form

Graphical user interface, text, application

Description automatically generated

### Compute Depreciation Form mnuClear clicked Screen

Graphical user interface, text

Description automatically generated

|  |  |
| --- | --- |
| Missing Item Selection Message Box Graphical user interface, text, application, chat or text message  Description automatically generated | Missing Depreciation Method MsgBox Graphical user interface, text, application, chat or text message  Description automatically generated |

## Source Code

### frmDepreciation

' Program Name: Chef Shack Food Truck Depreciation Windows Application

' Author: Jeffrey Annan Koranteng

' Date: April 21, 2022

' Purpose: The Chef Shack Inventory Windows Application determines

' the depreciation based on a 5 year life of items in inventory

' using the straight-line and double-declining balance methods.

'

Option Strict On

Public Class frmDepreciation

' Class Level Private variables

Private \_intLifeOfItems As Integer = 5

Public Shared \_intSizeOfArray As Integer = 7

Public Shared \_strInventoryItem(\_intSizeOfArray) As String

Private \_strItemId(\_intSizeOfArray) As String

Private \_decInitialPrice(\_intSizeOfArray) As Decimal

Private \_intQuantity(\_intSizeOfArray) As Integer

Private Sub frmDepreciation\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

' This frmDepreciation Load event reads the inventory text file and

' fills the Listbox object with the items

' Initialize an instance of the StreamReader object and declare variables

Dim objReader As IO.StreamReader

Dim strLocationAndNameOfFile As String = "c:\Bob\Inventory.txt"

Dim intCount As Integer = 0

Dim intFill As Integer

Dim strFileError As String = "The file is not available. Restart when the file is available."

If IO.File.Exists(strLocationAndNameOfFile) Then

objReader = IO.File.OpenText(strLocationAndNameOfFile)

' Read the file line by line until the file is completed

Do While objReader.Peek <> -1

\_strInventoryItem(intCount) = objReader.ReadLine()

\_strItemId(intCount) = objReader.ReadLine()

\_decInitialPrice(intCount) = Convert.ToDecimal(objReader.ReadLine())

\_intQuantity(intCount) = Convert.ToInt32(objReader.ReadLine())

intCount += 1

Loop

objReader.Close()

' The Listbox object is filled with the Inventory IDs

For intFill = 0 To (\_strItemId.Length - 1)

lstInventoryId.Items.Add(\_strItemId(intFill))

Next

Else

MsgBox(strFileError, , "Error")

Close()

End If

End Sub

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

' The BtnCalculateDepreciation click event calls the depreciation Sub procedures

' Declare variables

Dim intSelectedItemId As Integer

Dim strMissingSelection As String = "Missing Selection"

Dim strSelectDepreciationError As String = "Select Depreciation Method"

Dim strSelectInventoryItemIDError As String = "Select an Inventory Item ID"

' If the Listbox and a Depreciation Radio Button are selected,

' call the depreciation procedure

If lstInventoryId.SelectedIndex >= 0 Then

intSelectedItemId = lstInventoryId.SelectedIndex

If radStraightLine.Checked Then

StraightLineDepreciation(intSelectedItemId)

ElseIf radDoubleDeclining.Checked Then

DoubleDecliningDepreciation(intSelectedItemId)

Else

MsgBox(strSelectDepreciationError, , strMissingSelection)

End If

Else

MsgBox(strSelectInventoryItemIDError, , strMissingSelection)

End If

End Sub

Private Sub StraightLineDepreciation(ByVal intItemId As Integer)

' This sub procedure computes and displays the straight line depreciation for the selected item

' Declare variables

Dim intStraightPresentYear As Integer

Dim decStraightPresentYearValue As Decimal = 0

Dim decStraightDepreciation As Decimal

Dim decStraightTotal As Decimal

Dim strDepreciationItem As String = "The depreciation of the item: "

Dim strQuantityMessage As String = "Quantity: "

' The procedure MakeObjectsVisible is called to display the Form objects

MakeObjectsVisible()

' Display the item and quantity of the selected items

lblItem.Text = strDepreciationItem & \_strInventoryItem(intItemId)

lblQuantity.Text = strQuantityMessage & \_intQuantity(intItemId).ToString()

' The formular for straaight-line depreciation

decStraightDepreciation = \_decInitialPrice(intItemId) / \_intLifeOfItems

decStraightPresentYearValue = \_decInitialPrice(intItemId)

' The loop repeats for the life of the items

For intStraightPresentYear = 1 To \_intLifeOfItems

' Accumulates the total of depreciation

decStraightTotal += decStraightDepreciation

' Displays the depreciation amounts

lstYear.Items.Add(intStraightPresentYear.ToString())

lstPresentValue.Items.Add(decStraightPresentYearValue.ToString("C"))

lstYearDepreciation.Items.Add(decStraightDepreciation.ToString("C"))

lstTotalDepreciation.Items.Add(decStraightTotal.ToString("C"))

decStraightPresentYearValue -= decStraightDepreciation

Next

End Sub

Private Sub DoubleDecliningDepreciation(ByVal intItemId As Integer)

' This Sub Procedure computes and displays the double declining

' balance depreciation for the item selected

Dim intDoublePresentYear As Integer

Dim decDoublePresentYearValue As Decimal = 0

Dim decDoubleDepreciation As Decimal

Dim decDoubleTotal As Decimal

' The procedure MakeObjectsVisible is called to display the Form objects

MakeObjectsVisible()

' Display the item and quantity of the selected item

lblItem.Text = "The depreciation of the item: " & \_strInventoryItem(intItemId)

lblQuantity.Text = "Quantity: " & \_intQuantity(intItemId).ToString()

decDoublePresentYearValue = \_decInitialPrice(intItemId)

' The loop repeats for the life of the items

For intDoublePresentYear = 1 To \_intLifeOfItems

' The formular for double-declining depreciation inside the loop to repeat the process

decDoubleDepreciation = (decDoublePresentYearValue \* 2D) / \_intLifeOfItems

' Accumulates the total of depreciation

decDoubleTotal += decDoubleDepreciation

' Displays the depreciation amount

lstYear.Items.Add(intDoublePresentYear.ToString())

lstPresentValue.Items.Add(decDoublePresentYearValue.ToString("C"))

lstYearDepreciation.Items.Add(decDoubleDepreciation.ToString("C"))

lstTotalDepreciation.Items.Add(decDoubleTotal.ToString("C"))

decDoublePresentYearValue -= decDoubleDepreciation

Next

End Sub

Private Sub MakeObjectsVisible()

' This procedure displays the objects showing results

lblItem.Visible = True

lblQuantity.Visible = True

lblYear.Visible = True

lstYear.Visible = True

lblPresentValue.Visible = True

lstPresentValue.Visible = True

lblYearDepreciation.Visible = True

lstYearDepreciation.Visible = True

lblTotalDepreciation.Visible = True

lstTotalDepreciation.Visible = True

' The previous data is removed

lstYear.Items.Clear()

lstPresentValue.Items.Clear()

lstYearDepreciation.Items.Clear()

lstTotalDepreciation.Items.Clear()

End Sub

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

' The mnuDisplay click event creates an instance of the frmDisplayInventory

Dim frmSecond As New frmDisplayInventory

' Hide this form and show the Display Inventory form

Hide()

frmSecond.ShowDialog()

End Sub

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

' The mnuClear click event clears and resets the form

lstInventoryId.SelectedIndex = -1

radStraightLine.Checked = False

radDoubleDeclining.Checked = False

lblItem.Visible = False

lblQuantity.Visible = False

lblYear.Visible = False

lstYear.Visible = False

lstYear.Items.Clear()

lblPresentValue.Visible = False

lstPresentValue.Visible = False

lstPresentValue.Items.Clear()

lblYearDepreciation.Visible = False

lstYearDepreciation.Visible = False

lstYearDepreciation.Items.Clear()

lblTotalDepreciation.Visible = False

lstTotalDepreciation.Visible = False

lstTotalDepreciation.Items.Clear()

End Sub

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

' The mnuExit click event closes the application

Application.Exit()

End Sub

End Class

### frmDisplayInventory

' The frmDisplayInventory class is open by frmDepreciation

' and displays the inventory file in sorted order

Option Strict On

Public Class frmDisplayInventory

Private Sub frmDisplayInventory\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

' The frmDisplayInventory load event is a second form that

' displays the sorted inventory items

Dim strItem As String

' Sorts the \_strInventoryItem array

Array.Sort(frmDepreciation.\_strInventoryItem)

' Displays the \_strInventoryItem array

For Each strItem In frmDepreciation.\_strInventoryItem

lstDisplay.Items.Add(strItem)

Next

End Sub

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

' This Sub procedure opens the first form

Dim frmFirst As New frmDepreciation

Hide()

frmFirst.ShowDialog()

End Sub

End Class