CS30_Project # 2-Chapter 3 Variables, Input, and Output- Total = 50.0 pts

Due date: (By 11:30 p.m. (Pacific Time), Monday, 02/04/2019 in Canvas)

This PDF file consists of eleven (11) pages

Please read all of the following instructions carefully!

Please note: For full credit, the following three (3) assignments must be completed independently by each online student by the above due date and submitted to Canvas.

For this project students should implement programming constructs pertaining to Visual Programming Topics covered Chapter Three (3) of the Textbook.

Supplementary Materials:

1. PowerPoint Presentation.
2. PDF file, containing Chapter 3-notes for online CS30
3. PDF file, containing The Programming Process.
4. A compressed folder containing Chapter_3_Solution to all Programming Examples and Chapter_3_Solution to all odd-numbered Programming Exercises.

These documents are uploaded to the "***Content***" module of Canvas.

Late Assignment are not accepted. There is no grace period for any online assignment. All assignments are required to be submitted/uploaded to Canvas LMS, on or before their deadlines.

Under no circumstance should any online assignment be submitted to the course instructor's campus email account. No Exceptions!

As stated in the course syllabus, effective consistency with fundamental and basic topics covered in this online course must be observed. It is strongly recommended that you adhere to using **ONLY** programming elements (materials) that are presented (covered) in chapter 3 of our textbook. **Please refrain from including materials that may be found on the Internet or other sources**. Furthermore, students are required to maintain academic integrity and taking the necessary step to avoid **plagiarism.**

Failure to do so is subject to substantial penalty.

Steps for downloading This Programming Project

1. Log in Canvas LMS, and on the **Student View** page go to "***CONTENT***" module.

2. Click on the link to download this assignment **PDF file.**

3. Save the assignment PDF file under suitable folder on your computer. Record the **submission due date**.

Steps for creating Visual Basic projects:

When you write a Visual Basic application, you should follow number of important steps, for planning the project and then repeat these steps for creating the project. These steps involve setting up the user interface, defining the properties, and then creating the code.

Planning

- ***Design the user interface***: When you plan the user interface, you draw a sketch of the screens you will see when running your project. On your sketch, show the forms and all the controls that you plan to use. Indicate the names that you plan to give the form and each of the objects on the form.
- ***Plan the properties***: For each object (control) that you will place on the form, write down the properties that you plan to set or change during the design of the form.
- ***Plan the Visual Basic code***. In this step, you plan the classes and procedures that will execute when your project runs. You will determine which events require action to be taken and then make a step-by-step plan for those actions. Later, when you actually write the Visual Basic code, you must follow the language syntax rules. But during the planning stage, you will write out the actions using ***pseudocode***, which is an English-Like expression or comment that describes the action. For example, you must plan for the event that occurs when the user clicks on the Exit button. The *pseudocode* for the event could be Terminate the project or Quit.

Programming

After you have completed the planning steps you are ready to begin the actual construction of the project. Use the same steps that you used for planning.

- ***Define the user interface***: When you define the user interface, you create the forms and controls that you designed in the planning stage. Think of this step as defining the objects you will use in your application.
- ***Set the properties***: When you set the properties of the objects (controls), you give each object (control) a name and define such attributes as the contents of a label, the size of the text, and the words that appear on top of a button and in the form's title bar. You might think of this step as describing each object.
- ***Write the Visual Basic code***: You will use Visual Basic programming statements, using the Visual Basic syntax to carry out the actions needed by your program. You will be surprised and pleased by how few statements you need to create a powerful Windows program. You can think of this third step as defining the actions of your program.

Naming Rule: A control's name that appears in the Code Editor should describe the control's purpose in the application. This make the control name very helpful to anyone reading the application's programming statements. it is a good programming practice to have each Name property of a control begin with a three-letter prefix that identifies the type of control used in the applications' Code view. For a Form control the prefix is "frm", for a Button control, the prefix is "btn", for a Label the prefix is "lbl", and for a TextBox control the prefix is "txt", and so forth. Additionally, change the name of Form1 in the Solution Explorer, and in the Properties window. Text property of each control, including the Form should be also appropriately set in the Designer window for more meaningful visual purpose.

Best Practices: Please note that descriptive naming convention is of the most important task in programming design. Take the initiative as early as possible when planning and programming all Visual Basic Assignments.

Failure to follow the Naming Rule, for each of the assignments in this programming project will result in a deduction of 10% (10 Percent).

Comments: A well prepared Visual Basic program should consist of plain English readable annotations (comments) that will document and describe to the reader of your program, the programmer's intent. **In-line Comments:** are placed within the program source code, summarizing and explaining the intent of single programming code, or block of codes within the program. In Visual Basic, all comments start with a comment delimiter ('). All comments are ignored by Compiler.

Introductory Comments: Unlike in-line comments that are placed within the source code, introductory comments are placed at the top of the Visual Basic *Code* Editor, documenting a general overview.

Please note: To enter introductory comments for every assignment in this programming project, while your user interface (Form) is displayed in the Design View, right click with the mouse on the form and select "View Code "and enter your introductory comments before the "Public Class ...." line of code.

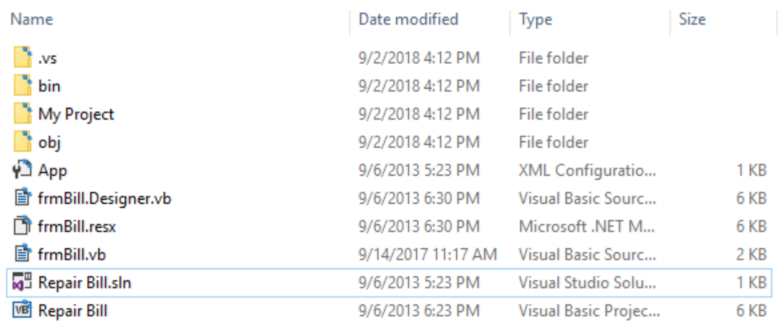
```
' Project:          Name of the project
' Programmer        Your Name
' Date:             Date completed
' Description:      Briefly describe what the project is desired to achieve.
```

Introductory comments are important for the instructor to identify the student and work s/he has completed.

Failure to provide introductory comments for each of the assignments in this programming project will result in a deduction of 10% (10 Percent).

The next step describes how Visual Basic generates files and folders for your program automatically, and how to save your program.

Program's files and folders: (Extremely Important). All Visual Basic programs generate numerous files and folders, for example for the **Repair Bill** program (see programming assignment 1), your Visual Basic program will automatically generate the following files and folders (See Below!)



Name	Date modified	Type	Size
.vs	9/2/2018 4:12 PM	File folder	
bin	9/2/2018 4:12 PM	File folder	
My Project	9/2/2018 4:12 PM	File folder	
obj	9/2/2018 4:12 PM	File folder	
App	9/6/2013 5:23 PM	XML Configuratio...	1 KB
frmBill.Designer.vb	9/6/2013 6:30 PM	Visual Basic Sourc...	6 KB
frmBill.resx	9/6/2013 6:30 PM	Microsoft .NET M...	6 KB
frmBill.vb	9/14/2017 11:17 AM	Visual Basic Sourc...	2 KB
Repair Bill.sln	9/6/2013 5:23 PM	Visual Studio Solu...	1 KB
Repair Bill	9/6/2013 6:23 PM	Visual Basic Projec...	6 KB

Once you save the above assignment in a folder by the name "**Repair Bill** ", all the above files and folders should be automatically placed in your program this folder.

Please note: At no time should you move or remove any of above files or folders, before they are saved properly in the "**Repair Bill** "folder.

Now perform the above step for the remaining assignments in this project, until you have three distinct folders, designed to each of the three programming assignments of this programming project. (see below!)

Repair Bill, **Change**, **Unit Price**

Run, and save your VB program. To ensure that all programing assignments works properly, providing you with desired output, run each, and check for possible **syntax and runtime, logic errors**, correct your program, and run the program until you have the satisfactory results. Save programing assignments as **Repair Bill**, **Change**, **Unit Price**

The next step describes how to upload the CS30_Project_2 to Canvas.

Compressed folder for submission of your project to Canvas; Gather all the three folders corresponding to the above programming assignments and place them in yet another folder. This is the final folder, and name it; **CS30_Project 2_You Full Name**. Please note that your full name must be part of the CS30_Project2 folder, otherwise, the course instructor cannot be distinguished one from the other one. Now, use the compression utility of Windows operating system and compress this new folder to make a zip file

"**CS30_ Project2_You Full Name.zip**"

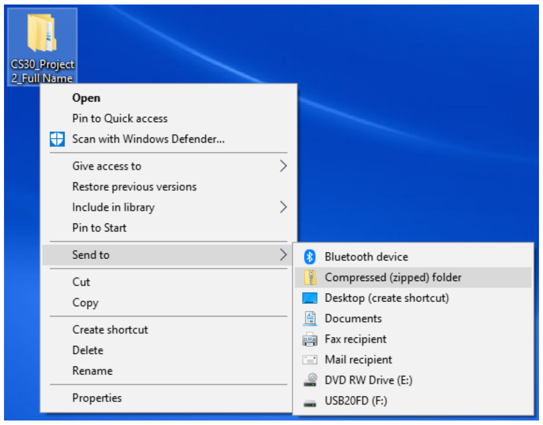
THIS LAST COMPRESSED FOLDER IS UPLOADED TO BLACKBAORD.

How to create and use compressed ("zipped") folders in Windows Operating Systems:
Please note that the only compression utility accepted for making compressed folder for this online course, is the **Windows Operating System compression utility**.

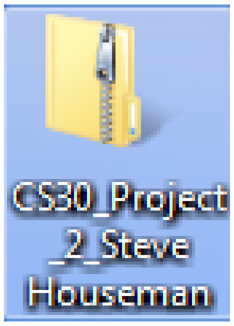
Note: If you use any other compression utility (third party compression utility), such as WinZip, 7-Zip, WinRaR, in order to upload your completed project to Canvas, you will be notified that the instructor of the course cannot open your compressed folder, in which case your project will not be graded or feedback provided.

The following steps demonstrates how to use **Windows Operating System compression utility**!

a. Right-click with the mouse on "**CS30_Project_2_Your Full-Name** "
b. On the drop-down menu, select "Send to"
c. Now, click on "Compressed (zip) folder



After a few second, the compressed folder is created. For example, using my own full name the following depicts the compressed folder that I have created.



This would be the final folder that you should upload it to Canvas.

IVC Computer Lab: During the entire semester computer lab access is available to all enrolled students. Further details are available at: http://academics.ivc.edu/successcenter/Pages/default.aspx. The Student Success Center is located at BSTIC110, First Floor. For more information, please call Irvine Valley Student Success Center

(Monday through Thursday: 9 am -7 pm, Friday 9 am -2 pm, Saturday 10 am – 2 pm) (Ph #: 949-451-5471).

Please Note: The Computer Lab is instructor supported. Please contact Student Success Center at IVC for further direction
Free Tutoring: All enrolled students are required to obtain and complete **Tutorial 301 Referral Form**. This form can be obtained at: http://academics.ivc.edu/successcenter/Pages/free.aspx. Since this is an online course, and the course instructor is not available at the campus, you are required to take the form to the Counselling Center in Student Services 210 for authorization from the counselor on duty. Then bring your signed form to the Student Center in BSTIC 110.

Completing Assignments at IVC Computer Lab: For administrative purposes, it is necessary to associate the computer lab with an existing lab course structure. The Computer Lab is instructor supported. Spring semester 2019, please contact professor **Youlin Shaw-Kingery** at yshawkinger@ivc.edu.

Professor Shaw-Kingery hours at the campus computer lab are;

Tuesday - Thursday, 12:30 - 7 pm.

Internet Connection & Canvas Compatible Browser: Always, while you are attending this online course, you must have Internet availability, and Canvas compatible browser.

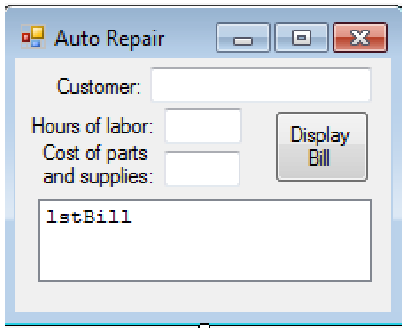
Assignment 1: (20 points): Problem Statement: (`Repair Bill`**):** Suppose automobile repair customers are billed at the rate of $35 per hour for labor. Also, suppose costs for parts and supplies are subject to a 5% sales tax. Write a Visual Basic program to display a simplified bill to perform following tasks.

Required: Use a descriptive named constant for the sales tax and assign 0.05 to it.

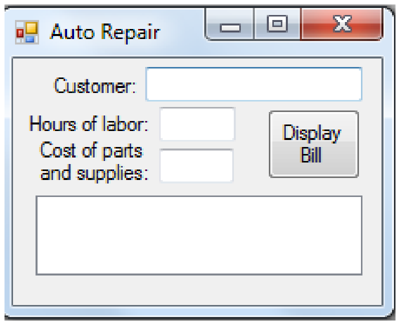
Required: Use a descriptive named constant for the labor/hour rate and assign 35 to it.

Required: The customer's name, the number of hours of labor, and the cost of parts should be entered in the program via text boxes. When the Display Bill button is clicked, the customer's name, and the three costs should be displayed in the list box control. (For more information on how to use a list box control for output, please refer to page 28, of section 2.2 of the text. Additionally, check Example 1 on pages 54-55 of section 3.1 of the text.)

This is how it work! Initially, when the program is loaded, the form" `Auto Repair`" should appear, as shown in the following display.



Once the program is run, the form should be ready for entry, as shown below



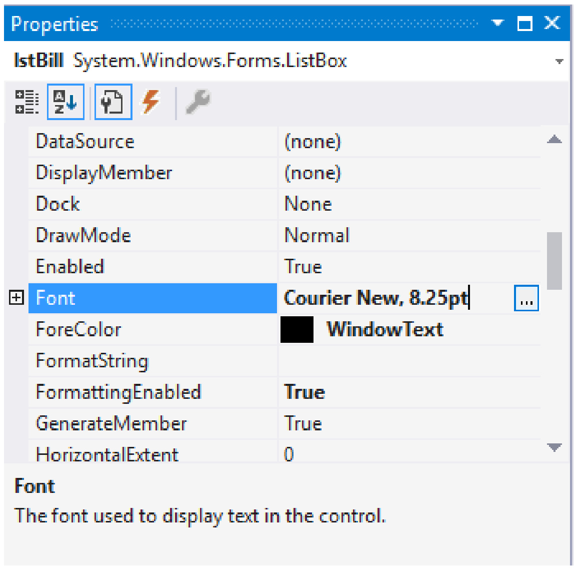
Required: When adding items to the list box, please make sure you are using "**Courier New** "font, otherwise your entries will not line-up (align) properly (see below!)

Here is how it works…

With you mouse pointer hover over the control object(s) on your form in Designer window, and select it. In this case the List Box control is named "lstBill".

Next, on the Properties window (see below), select the **Font** property of the List Box control object. Then with a single click of the mouse, in the right part of its Setting box click on the three (…) ellipsis button.

A window with three lists appears. Select the current name of the font as "**Courier New**", Choose "**Regular**". Next select any size you want, for example 8.25 point as shown in the following figure.


Properties

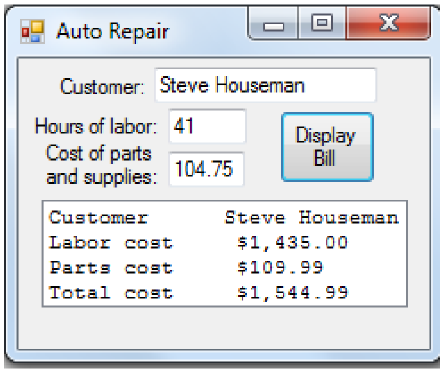
lstBill System.Windows.Forms.ListBox

DataSource	(none)
DisplayMember	(none)
Dock	None
DrawMode	Normal
Enabled	True
Font	Courier New, 8.25pt
ForeColor	WindowText
FormatString	
FormattingEnabled	True
GenerateMember	True
HorizontalExtent	0

Font

The font used to display text in the control.

After, the name of the customer, the hours of labor, and the cost of parts are all entered in the appropriate text boxes, the list box should provide you the following display, after the user clicks on the "`Display Bill`" button.



Customer: Steve Houseman

Hours of labor: 41

Cost of parts and supplies: 104.75

```
Customer        Steve Houseman
Labor cost       $1,435.00
Parts cost       $109.99
Total cost       $1,544.99
```

Recommended naming convention: For control objects appearing in the Code Editor.

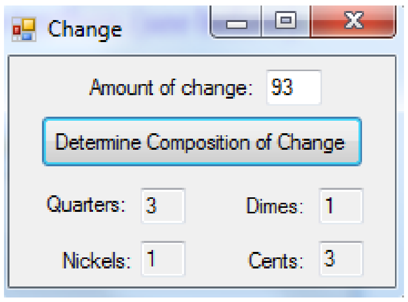
Name Property	Text Property	Control Type	Description
frmBill	Auto Repair	Form	Holds application's controls
txtCustomer		Text box	Captures customer's name entered
txtHours		Text box	Captures # of hours of labor entered
txtCost		Text box	Captures cost of parts and supplies entered
btnDisplay	Display Bill	Button	Triggers the event for displaying bill
lstBill		List Box	Displays customer's bill

Remember that in addition of the above, you are required to apply suitable naming convention to all variables/named constants that are declared and used (accessed in the application) in the program code.

Assignment 2: (15 points): Problem Statement: (`Change`**)** Write a Visual Basic program to perform the following tasks.

Required: The user inputs an amount of change of money from 0 to 99 cents, for instance, 93 cents (see below!), but it may be any other amount of change, entered by the user.

Required: The program converts the change amount in number of coins, quarters, dimes, nickels, and penny.

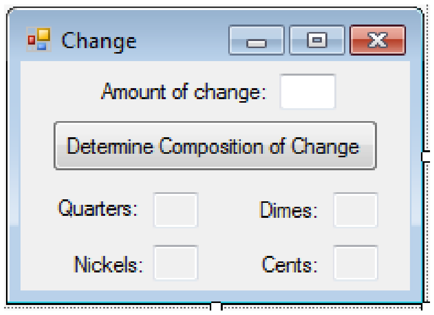


Example of final display

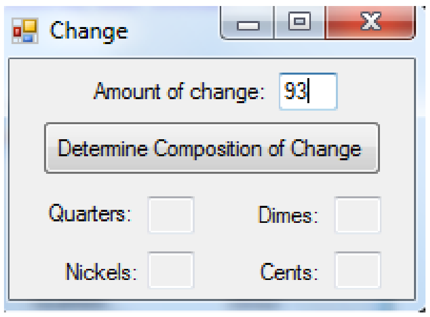
Recommended naming convention: For control objects appearing in the Code Editor.

Name Property	Text Property	ReadOnly property	Control Type	Description
frmChange	Change		Form	Holds application's controls
txtAmount			Text box	Captures the amount of change entered
btnDetermine	Determine Composition of Change		Button	Triggers the event for displaying composition of change
txtQuarters		True	Text Box	Displays # of Quarters (if any)
txtDimes		True	Text Box	Displays # of Dimes (if any)
txtNickels		True	Text Box	Displays # of Nickels (if any)
txtCents		True	Text Box	Displays # of Cents (if any)
Remember that in addition of the above, you are required to apply suitable naming convention to all variables that are declared and used (accessed in the application) in the program code.				

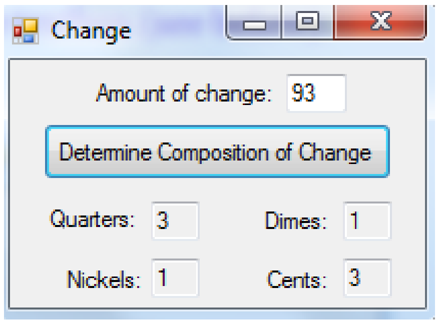
This is how it work! Initially, when the program is loaded, the form" Change" should appear, as shown in the following display.



Once the program is run, the form should be ready for entry, as shown below (suppose the user enter 93 cents in the appropriate textbox (see below)



Now, while the program is running, if the user clicks on the "`Determine Composition of Change` "button, the following display will show the change in various US coins composition.

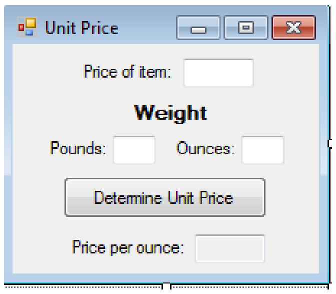


Assignment 3: (15 points): Problem Statement: (`Unit Price`**)** Write a Visual Basic program to perform following tasks. (**`One Pound is equal to 16 ounces`**).

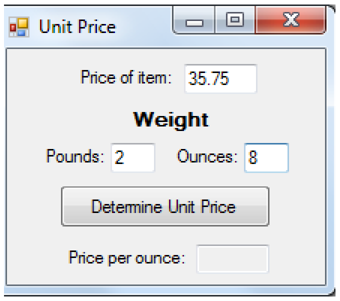
Required: Request the price and weight of an item in pounds and ounces, and then determine the price per ounce (user input)

Required: When the user clicks on the "`Determine the Unit Price` "button, the following display as Price per Ounce.

Initially, when the program is loaded, the form" `Unit Price`" should appear, as shown in the following display.



Once the program is run, the form should be ready for entry, as shown below (suppose the user enters 35.75 (as the price of the item) and 2 pounds and 8 ounces as the weight of the item in the appropriate text boxes (see below)



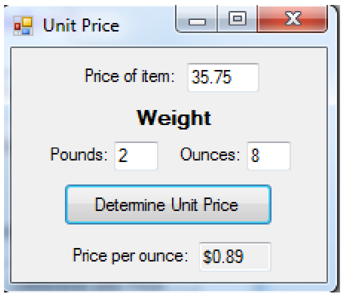
Price of item: 35.75

Weight

Pounds: 2 Ounces: 8

Price per ounce:

Now, while the program is running, and the user clicks on the "`Determine the Unit Price` "button, the following display as Price per Ounce.



Price of item: 35.75

Weight

Pounds: 2 Ounces: 8

Price per ounce: $0.89

Please note in order to display the Price Per Ounce in two decimal places, please use the appropriate function to convert, for example " `ToString("C").`

Recommended naming convention: For control objects appearing in the Code Editor.

Name Property	Text Property	ReadOnly property	Control Type	Description
frmUnitPrice	Unit Price		Form	Holds application's controls
txtPrice			Text Box	Captures the price of an item entered
txtPounds			Text Box	Captures the weight of the item (pounds portion) entered
txtOunces			Text Box	Captures the weight of the item (Ounces portion) entered
btnDetermine	Determine Unit Price		Button	Triggers the event for displaying price of an item per ounce
txtUnitPrice		True	Text Box	Displays price of an item per ounce

Remember that in addition of the above, you are required to apply suitable naming convention to all variables that are declared and used (accessed in the application) in the program code.