DUE: 10/16/2013

Project #1—Gini's Quilt Works

Gini Quilter is a quilt maker with aspiration to sell her work on the Internet. She has commissioned you to design a simple invoice form to be used as a prototype for her website sales. A sample of the invoice form is shown below. The prototype should show only three quilt items, although she hopes to have all of her creations listed for the final web project. She has only 6 Shining Star quilts (first image in sample form) that sell for $950, 10 of the Kat Soup (second image) that sell for $480, and 3 Birch Trees (third image) that sell for $315 each. The tax rate is 9.8%. Write a C# program that will calculate the invoice for a given order.

Your invoice GUI (shown to the right at start time) does not have to be identical to the one shown, but it should provide the user the ability to enter a valid four-digit invoice number, number of each quilt ordered, and it should calculate the extended totals for each item, the subtotal, tax and the total. Images of three quilts (find your free images on the Internet) are required as are the buttons for Calculate, Clear and Quit.

Make your buttons easy to use for keyboard entry. Make the Calculate button default and the Clear button the cancel button. Do not allow bad input data to cancel the program; instead display an error message to the user.

Input: Invoice number and quantity for each of the three quilts.
Output: Extended total for each quilt, subtotal, tax and total.

1. The project is worth 30 points and it will be graded on correctness, completeness, presentation, comments, punctuality, and the like.
2. Make sure that your name, project number, and due date appear as comments in your code (preferably on top). Use the Heading.txt file for this purpose.

3. Submit the zipped file containing all files and folders for the project Canvas before midnight of the due date.

EXTRA CREDIT (Optional—3 points). Functional and meaningful menus, printing and validating input for type and range will fetch a maximum of 3 extra credit points.

Project Guidelines

The following list reviews the GUI design and programming guidelines you have learned so far. You can use this list to verify that the applications and projects you create adhere to the standards outlined herein.

- Information should flow either vertically or horizontally, with the most important information always located in the upper-left corner of the screen.
- Maintain a consistent margin of two or three dots from the edge of the window.
- Related controls should be grouped together using either white space or a group box.
- Position and align related controls on succeeding dots.
- Buttons should either be centered along the bottom of the screen or stacked in either the upper-right or lower-right corner.
- If the buttons are centered along the bottom of the screen, then each button should be the same height; their widths, however, may vary.
- If the buttons are stacked in either the upper-right or lower-right corner of the screen, then each button should be the same height and the same width.
- Use no more than six buttons on a screen.
- The most commonly used button should be placed first.
- Button captions should:
  - ✓ be meaningful
  - ✓ be from one to three words appear on one line
  - ✓ be entered using book title capitalization
  - ✓ contain & for "hot" keys activation
- Use labels to identify the controls in the interface.
- Identifying labels should:
  - ✓ be from one to three words
  - ✓ appear on one line
  - ✓ be aligned by their left borders
  - ✓ be positioned either above or to the left of the control that they identify
  - ✓ end with a colon (:)
  - ✓ be entered using sentence capitalization
  - ✓ have their BackStyle property set to 0-Transparent
  - ✓ have their BorderStyle property set to 0-None
- Labels that display program output (for example, the results of calculations):
  - ✓ should have their BorderStyle set to 1-Fixed Single
  - ✓ should have their Appearance property set to 0-Flat
  - ✓ can have their BackStyle property set to either 0-Transparent or 1-Opaque
- Align labels and controls to minimize the number of different margins.

- If you use a graphic in the interface, use a small one and place it in a location that will not distract the user.
- Use no more than two different font sizes, which should be 10, 12 or 14 points.
- Use only one font type, which should be a sans serif font, in the interface.
- Avoid using italics and underlining.
- Use light colors such as white, off-white, light gray, pale blue, or pale yellow for an application's background, and very dark color such as black for the text. Alternately, use dark colors for the form's background with light colors for the text—contrast is the name of the game.
- Use color sparingly and don't use it as the only means of identification for an element in the interface.
- Set each control's TabIndex property to a number that represents the order in which you want that control to receive the focus (begin with 0).
- A text box's TabIndex value should be one more than the TabIndex value of its identifying label.
- Lock the controls in place on the form.
- Assign a unique access key to each essential element of the interface (text boxes, buttons, and so on).
- Document the program internally.
- Enter the Option Explicit and Option Strict statements in the General Declaration section of every form and module.
- Use variables to control the preciseness of numbers in the interface.
- Avoid using the Val function—provide meaningful validation of data.
- Set the form's BorderStyle, MinButton, MaxButton, and ControlBox properties appropriately:
  - ✓ Splash screens should not have a Minimize, Maximize, or Close button, and their borders should not be sizable.
  - ✓ Forms that are acting as dialog boxes should have a BorderStyle property of 3-Fixed Dialog.
  - ✓ Forms other than splash screens and dialog boxes should always have a Minimize button and a Close button, but disable the Maximize button.
  - ✓ Forms other than splash screens and dialog boxes typically have a BorderStyle property setting of either 1-Fixed Single or 2-Sizable.
- Test the application with both valid and invalid data (test the application without entering any data; also test it by entering letters where numbers are expected). Test the application for correct results with known data.
- In the InputBox function, use sentence capitalization for the prompt, and book title capitalization for the title.
- Center a form by including the appropriate formulas in the form's Load event.
- Display an hourglass or an arrow and an hourglass if a procedure will take a long period of time.
- Remove excess or duplicated code from the application.
- Provide ample comments for each form and each procedure.
- Avoid excessive use of class-level variables.

Grading Guidelines: Your project will be graded on the following:

- Correct solution to the proposed problem.
- GUI design (as outlined above).
- Elegance, simplicity, style, and readability of the algorithms and code.
- Comments:
  - ✓ Use of header.
  - ✓ Description of project.
  - ✓ Description of procedures.
  - ✓ Description of complicated code.

Hints for Project #1: *Given that your GUI resembles the given one, follow these guidelines:*

- ✓ Consider the steps for this project:
  - Read carefully and understand the problem.
  - Draw a flowchart or pseudo code and have an example completely worked out on a piece of paper following your flowchart. Make sure the calculations are correct.
  - Design the GUI. You do not have to copy the given design, but the functionality should match or exceed the requirements.
  - Write the code following the IPO (Input, Process, Output) protocol.
  - Make sure you have a declaration section for the constants and variables, an input section, a calculation section and an output section.
  - Test the program for correct results and invalid input.

- ✓ You should use your own quilt images but follow these restrictions:
  - o There must be exactly three quilt pieces.
  - o The images should be small in size, preferably included in an Images folder in the Debug subfolder.
  - o The images should be free—it takes a simple request with an email for permission to use if obtained through the Internet. Consider including a comment of where you obtained the images.
- ✓ The current date should appear as the form loads either in a label or disabled (or read-only) textbox. Consider declaring a Date variable and use the `FormatDateTime()`function to invoke the System's date. Alternatively you may use the `now()` function or a DateTimePicker control.
- ✓ The price of each quilt should be set in a label or read-only textbox and also at load-time using class-level constants. This allows for easy price change.
- ✓ Have at least three buttons (Clear, Calculate, and Quit) with Calculate as a default (so it is activated when you press the enter key). Endow them with accelerator "hot keys" capabilities (this means if you press Alt-Q, the button Quit should fire).
- ✓ Comments, comments, comments all over. Modify the lab heading that describes what the program does, I/O requirements, platform, time used, project name, your name, etc.
- ✓ Chose variable of appropriate type and name them using the established C# naming convention. Do the same with all controls and constants.
- ✓ Indent code and make it easy to ready.
- ✓ Make sure you use constant declarations for all constants.
- ✓ Use NumericUpDown text boxes for number of quilts ordered. Validation and range restriction becomes much easier with this control.
- ✓ Consider adding only one ToolTip with appropriate description for each quilt on the pictures. Include in the ToolTip description for the quantity the number of available quilts.
- ✓ Disable the output text boxes—the user should not be allowed to change the any of the currencies other than the dollars. Alternatively, you might want you use labels.

Pay attention to the following criteria for grading:

'~~~~~~

'Good effort. Here are suggestions for improving:

'1. Use comments through the program, not just at the top heading. Use comments to describe each procedure and complicated code.

'2. Convert entries in textboxes from strings to numbers with an appropriate Convert.ToYYY function.

'3. Make sure Option Explicit and Option Strict are turned-On.

'4. Make the Calculate button default.

'5. Correct the calculations—results are incorrect.

'6. Follow C#'s 3-letters prefix naming convention for variables and controls and capitalize all constants.

```
'7. Use const definitions for the prices of instruments and tax.
'8. It is good style to tab code inside a procedure.
'9. Provide Calculate, Clear, and Quit buttons with accelerator keys.
'10. Use Application.Exit()  to exit program.
'11. Set focus to txtInvoiceNumber after Clear.
'12. Change the tab index to indicate correct order of entry.
'13. Name the project appropriately; a generic WindowsApplication1 or Project1 or Form1 just won't do.
'14. Use the Heading.txt heading for the project.
'15. Clear should clear all text boxes and set focus on invoice number.
'16. New should clear everything and increment the current order number by one.
'17. Display $$ results as currency.
'18. Display current date in read-only date field.
'19. Avoid excessive number of class variables-declare variables inside procedures.
'20. Clean up the empty procedures code.
'21. Separate code in main procedure into 4 section: Declaration, Assignment, Calculation, and Display.
'22. Do not allow negative entries for number of items--provide range and type check.
'23. Provide free images for the quilts.
'24. Set Enabled property to False for all output text boxes.
'25. Disable form from maximization or resizing.
'26. Missing ToolTip.
'27. Use NumericUpDowns instead of TextBoxes for number of items ordered.
'28. Give the form an icon.
'29. User does not have to purchase from each item.
'30. Program crashes.
'31. Invoice number needs to be entered and is should be a four-digit number.
'32. Use appropriate type for variables; quantities should be Byte and all money variables should be
Decimal.
'
'The ones that apply to your project are:
'3, 6, 15, 21, 22, 26, 32
'
'24+1=25/30
'~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```