ISM 3232-001 – Spring 2022 – Individual Assignment #3

DUE DATE: Friday 4/8/22

ASSIGNMENT OBJECTIVES

- To again meet the Objectives of Individual Assignments #1 and #2 listed here: ⇩
  - Use Visual C# 2017 to create a project. (*** See the **next page** for the **correct project name** to use. ***)
  - Observe professional programming style guidelines (e.g., comments, indentation, etc.).
  - Determine the most appropriate C# controls to use to accomplish the requirements of the program.
  - Declare all necessary constants and variables with appropriate data types and scopes.
  - Follow appropriate standardized naming conventions for variables, constants, and controls.
  - Write code to accept input from the user, perform calculations, and convey output to the user.
  - Utilize appropriate decision structures to branch code execution as required by the program.
  - Format output so that it displays appropriately for the situation involved (e.g., as currency).
  - Create custom message boxes that communicate with the user and respond appropriately.
- To properly utilize looping structures in your program code where necessary.
- To read data in from external files, and output data to a formatted message box.
- To write custom methods for appropriate use in your program code.

BUSINESS SCENARIO

Bonnie's Balloons is a small privately-owned balloon shop operated by Bonnie Leiter, who employees a small staff of assistants, including delivery personnel. The shop specializes in high-quality Mylar balloons decorated for special occasions. The price of a balloon order depends on the size of the balloon bundle, what types of "extras" are included with the order (e.g., flowers, candy, etc.), whether a message card is included, and whether the customer picks up the order at the shop or has it delivered to them. Balloon orders are predominantly taken by phone, although there are some walk-in customers. Bonnie is interested in converting the current paper-based order-taking system into an automated one. She has hired you to begin work on this project. The specific requirements of a first draft of this program are detailed below.

ASSIGNMENT REQUIREMENTS

- For decorative purposes, there is to be a picture box control placed at the top of the form. This control should display a graphic file related to balloons that is appropriate for this company. Below, or to the side of, this picture box should be a bordered label displaying the company name in a font style and font size other than the default.
- Four groups of controls (described immediately below) should be set apart by the use of group box controls.
- Customer Information – The title, first name, last name, street, city, state, zip, and phone number are to be entered for the customer. For the customer's title, a combo box should provide a pre-defined choice of: Dr., Mr., Mrs., Ms., and Rev. (Additionally, other titles not specified here are to be allowed to be entered by the user if desired.) When the program begins, this title combo box should not display any value until the user selects one from the list or types one in. For state, allow only two uppercase letters; for zip, allow only five digits; for phone, use (999) 000-0000 format.
- Delivery Information – A control is to be provided to hold the desired delivery date (00/00/0000 format) that each order is to be ready for either store pick-up or home delivery. This control should automatically default to displaying the current date when the program begins. Also, in this section, the delivery type (store pick-up or home delivery) is to be specified with radio buttons. At program startup, the store pick-up option should be selected by default. Whenever home delivery is specified, a $7.50 charge is added to the order total price. This $7.50 price should be displayed on the form next to the home delivery option. *(Hint: Utilize a constant to provide this value to the label to facilitate any future change. This approach should also be used for the other prices to be displayed – see below.)*
- Order Details – The balloon bundle size desired by the customer is to be specified by selecting between three radio buttons: single, half-dozen, and dozen. The prices of these three options – $9.95, $35.95, and $65.95 respectively – should display next to the names of the options. At program startup, the single option should be selected by default. The special occasion for the order is to be selected from a combo box. *Bonnie's Balloons* currently offers balloons for seven occasions, named as follows: Birthday, Congratulations, New Year's Day, Valentine's Day, Halloween, Graduation, and Anniversary. These occasion names are to be read in from an external text file named Occasions.txt. This file should be located in the default bin\Debug subfolder of the project. As the program begins, appropriate code should read these values from the text file into the list of the combo box. The occasions should always be displayed in the list in alphabetical order, regardless of how they are listed in the text file. Birthday should be displayed as the default choice at program startup. The program user should not be able to enter in an alternative to the existing choices. Next in this section is to be a list box that displays the available "extras" that can be added to a balloon order, at additional cost. The five current options are: Coffee Mug, Flower Arrangement, Potted Plant, Box of Chocolates, and Jar of Jelly Beans. A label should display the $9.50 price that applies to each option selected. These option names are to be read in from an external text file named Extras.txt. This file should be located in the default bin\Debug subfolder of the project. As the program begins, appropriate code should read

these values from the text file into the list of the list box. The options should always be displayed in the list in alphabetical order, regardless of how they are listed in the text file. As many of these options as desired (including none or all) may be selected by the user. No options should initially be selected, and no alternative to the existing choices is to be allowed. Finally in this section, a check box control should be included that allows the user to indicate whether the customer wishes to include a personalized message card in the order. The $2.50 price required for this option should be displayed in a label. Below this check box is to be a text box into which the custom message (to be printed on a card) may be entered. The message is to be limited to 30 characters, and a label near the text box should state this limitation. *(Hint: To enforce this limit, use the MaxLength property of the text box control.)* The text box and instruction label here should be disabled when the program starts, with the check box unchecked. Anytime the user checks the check box, the textbox and label should be enabled; when the check box is unchecked, the text box and label should be disabled (and any text in the text box deleted).

- Order Totals – Three total values for the order should display in this section: the subtotal (which is the total pre-tax order price that includes all charges based on the selections made by the user), the sales tax amount (which is the subtotal times the tax rate of 7.00%; the tax rate should be displayed on the form here), and the order total (which is the subtotal plus the sales tax amount). When the program begins, these controls should display the values associated with the default settings of the form (single balloon bundle size is selected and no other item charges are included).

- There are to be three buttons in a row at the bottom of the form. These are to be labeled: Display Summary, Clear Form, and Exit Program. There should be an Access Key and an appropriate ToolTip defined for each of these three buttons.

- Clicking the Display Summary button is <u>not</u> to actually save any data to an external file in this initial program prototype. Instead, the saving of data will be simulated by displaying the entire contents of the order in a message box. This message box should have appropriate text in its title bar and should include an Information icon. The message box should display the following data, <u>properly labeled</u>, with one data item per line: a header line (e.g., Bonnie's Balloons Order Summary); customer name (title, first name, and last name on one line); customer street; customer city, state, and zip (on one line); customer phone; delivery date; delivery type; bundle size; occasion; extras (with each extra on a separate line); message (left blank if there is no message); order subtotal; sales tax amount; and order total. After displaying the order summary message box, the form should be cleared and returned to its original startup state. <u>However</u>, when the Display Summary button is clicked, if the customer first name or last name fields have been left blank, or if the phone number entry is incomplete, the program should display an error message explaining that this specific information is required, <u>instead of displaying the order summary</u>. This error message box should have appropriate title bar text and an Error icon.

- Clicking the Clear Form button is to clear all data entry areas and return the form to its original state as it appears upon program startup (including that the focus is sent to the first data entry control on the form).

- Clicking the Exit Program button is to terminate the program using the Close method. However, the exact same procedure for asking the user to confirm their intent to exit the program that was used in Assignment #2 should be used here.

- <u>Three custom methods must be written in your code</u>. One, named PopulateBoxes, should include the code that reads the data from the two external files into the combo box and list box described above. A second, named ResetForm, should be used to perform the form reset actions required by use of the Display Summary and Clear Form buttons (see above). The third, named UpdateTotals, should be called to handle the update of the three totals whenever the user makes any selection on the form that affects the values of the totals. Examples of such actions include: selecting the delivery type (pick-up or delivery), selecting the bundle size, selecting extras from the list box, and selecting/deselecting an optional message card.

- Appropriate use of the Try-Catch structure should be made in your code to handle any potential input/output exceptions.

- Follow standard C# naming conventions in naming all form controls, and constants, variables, and methods in your code.

- Use appropriate labels for your controls to clearly identify them, and use appropriate text in the form's title bar.

- The alignment, spacing, and sizing of all screen controls should be appropriate, neat, and professional in appearance.

- When the program runs, the form should display in the center of the screen.

- The tab order should be set correctly so that focus flows logically through the controls on the form.

- Data that represents currency values should be displayed with the correct currency format.

- Numeric/currency values should be displayed right-aligned in controls, while text values should be displayed left-aligned.

- Use comments very liberally throughout your code, for each event handler and method, and each significant block of code.

- Include an initial four-line comment before all other code, using the same format specified for the previous assignments.

DELIVERABLES

- This assignment is due no later than Midnight (11:59pm) on Friday, April 8th. *No late work will be accepted.*
- Name your C# project as: ***YourLastName_3*** (for example, Smith_3, if your last name is Smith). ⇦
- Zip your <u>entire</u> project into a single .zip file. Make sure that all folders and files are included. (Refer to the handouts titled "Zipping and Unzipping a Visual C# Project" and "A Common Mistake When Zipping a Visual C# Project".)
- Submit your zipped file via the class Canvas site following the same steps that you followed for Individual Assignment #1.

> ***This is an <u>individual</u> assignment. You are <u>NOT</u> to discuss it with other students or collaborate with other students in <u>any</u> way. If you have questions about this assignment, contact your <u>instructor</u>.***