This program explores the creation of a simple GUI, use of dialog boxes, validation, and exception handling.

In this assignment you will build on your instructor's solution to Program 1A/B (find under Course Documents). These files have been added to a Windows Forms project I've created for Program 2 (see attached file *Prog2Start.zip*).

Next, you are asked to create a simple GUI that will serve as a front end for an application similar to MyUPS that we are calling a **UserParcelView**. I have written the class that will manage the business logic for the **UserParcelView** and it will be used by your Windows Forms application to accomplish the following. The application will be demonstrated in class on October 3 (review Panopto recording, if absent) and the general functionality of your program must match.

- Specifically, there must be a *File* menu with two items, *About* and *Exit*.
- The *About* item should display a dialog box with your Grading ID, section number, etc. The *Exit* item will exit the application.
- There must be an *Insert* menu with two items, *Address* and *Letter*.
- The *Address* item will display a modal dialog box with the form used to enter address information.
- All the fields will be text boxes except the state, which will use a combo box.
- Populate the list of states with at least four states.
- The address form must perform validation.
- Specifically, all text fields except address line 2 must contain text of some sort, a state must be selected, and the zip code must be valid (integer between 00000 - 99999).
- You may use the **ErrorProvider** component and the *Validating* and *Validated* events for the fields or some other approach.
- If the user enters valid information and submits the form, the address should be added to the list of addresses maintained by Program 2's UPV object.
- The *Letter* item will display a modal dialog box with the form used to enter letter information. The current list of addresses will be used to populate combo boxes used to select the origin and destination addresses for the letter.
- You should use the address' *Name* property as the string listed in the combo boxes.
- As with the address form, you must perform validation on the letter form, requiring that different addresses be selected for origin and destination, and ensuring that the fixed cost field is non-negative.
- Again, you may use the **ErrorProvider** component and the *Validating* and *Validated* events or some other validation.
- The final menu is the *Report* menu and must have two items, *List Addresses* and *List Parcels*.
- `These items will produce the list of addresses and parcels, respectively, using a multiline text box on the Program 2 GUI.
- The parcel list should also display the total cost for shipping all the entered parcels.
- You should preload some addresses and parcels into the **UserParcelView** object being maintained by the main form, such as the lists of test addresses and parcels used in your

instructor's solution to Program 1B. Use methods *AddAddress*, *AddLetter*, *AddGroundPackage*, etc.

All menus and menu items must be able to be activated using Alt-key shortcuts (as described on p. 598). Be sure to follow standard conventions for these when appropriate. No additional shortcut key combinations (such as CTRL-C, etc.) are necessary.

Be sure to add appropriate comments in your code for each file, including your **Grading ID**, program number, due date, course section, and description of each file's class. Each variable used in your program needs a comment describing its purpose. These requirements are expected for every program and are listed in the syllabus. Preconditions and postconditions are now required, as well. So, for each constructor, method, get property, and set property a pair of comments describing the precondition and postcondition must be provided. Please review the PowerPoint presentation (under Course Documents) for further details about preconditions and postconditions. Since the GUI components are created in the .Designer.cs file, you do not have to comment these variables. However, you must give the GUI components meaningful names, such as zipCodeTxt instead of textbox1, etc.

As with our labs, I'm asking you to upload a compressed ZIP archive of the entire project. The steps for doing this will vary somewhat based on the ZIP utility being used. Before you upload this .ZIP file, it's a good idea to make sure that everything was properly zipped. Make sure your code is present and you can run your file.

Once you have verified everything, return to the *Assignments, Programs* area of Blackboard. Click on "Program 2" and then click on *View Assessment* and the assignment description will appear. Click *Add Content* button to browse the system for your file. Click on *Insert Content* icon (+) at right-end of nav-bar, and then choose *Insert Local Files* from the submenu.

Browse to the location of your .ZIP file and select it. Note, multiple files may be attached by repeating the Insert Content sequence. For this assignment, we just need the "Prog2.zip" file. Make sure everything is correct in the form and then click *Submit* to complete the assignment and upload your file to be graded.