This program explores file I/O and object serialization and expands the GUI application developed in Program 2.

In this assignment you will build on your instructor's solution to Program 2 (find under Course Content, Program Solutions, Programs). Use this solution as your starting project and edit/add code as needed. There are two primary additions to the functionality provided in Program 2. First, the UserParcelView data that are entered need to be able to be saved to a file and loaded back into the application. Second, in addition to inserting new addresses, users should be able to edit existing addresses to update the fields with new information.

- The application must add two menu items, Open and Save As, to the File menu.
- The Open menu item allows the user to choose a file that contains the UserParcelView object's data. When opened, the data from the file will replace the addresses and parcels currently in the application.
- The Save As menu item allows the user to save the current data in the application's UserParcelView object to a file. You are required to use object serialization with binary formatting for this. This will require minor modifications to the existing business logic classes in the solution to make them [Serializable].
- The application must include appropriate exception handling, so that file-related errors do not crash the program.
- Since the user will now be able to load their own data, there is no longer a need to pre-populate the UserParcelView object with test data. However, before removing the test data, be sure to use the Save As function of the application to save a data file that may be used as a starting point for grading. Save the file as "Prog3Data.dat" and store in your project folder (at the same location as the project's .SLN file). By placing the file in this location, it should be included when your ZIP your project for submission.
- In addition, the application must add a new menu, Edit with a single menu item, Address.
- When selected, present the user with the list of addresses and allow the user to choose which one they'd like to edit. You might do this in several different ways, possibly using a combo box (as in the Letter form), through a ListBox or some other mechanism. The GUI design for this is up to you but it needs to be functional and attractive.
- Once an address from the list is selected, use the existing Address form dialog box to allow the user to edit the fields in the address. Remember, when editing, the existing data should be loaded into the form fields and when submitted, the existing Address object should have it's fields updated to match. It should not be necessary to modify the Address form code, so don't make any changes to this file without getting approval first. Note well, you are not adding a new object to the list of addresses when editing! You are modifying the attributes of an existing Address object.

As with Program 2, all menus and menu items must be able to be activated using Alt-key shortcuts. 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 3" 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 "Prog3.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.