Program 2: Enhanced Flight Reservations
CS 2574: Introduction to Data Structures & Software Engineering
DUE: August 5, 1998

Abstract
The objective of this assignment is to enhance the previous assignment by creating a graphical user interface for the flight reservation system. In addition, a waiting list for flight passengers is to be implemented instead of generating errors when the flight is full.

Description
Hokie Poke Airlines is back and requesting an enhanced version of the original flight reservation system. This enhanced system is to be used by travel agents and flight agents, so it needs to be user friendly. You are tasked with modifying the previous program and updating it to use a graphical user interface. In addition some improvements to the system for handling full planes.

Input
All input into the program will be made through a graphical user interface. The graphical user interface will be implemented using the GraphApp user interface library. GraphApp is installed in "C:\Program Files\DevStudio\GraphApp" on the McBryde 116/118 machines under Windows NT. We will cover using GraphApp shortly after the midterm. Take some time to go to the GraphApp web site and go through the tutorial slides and look over the reference manual available there.

The graphical user interface will encompass the three main input functions from Program 1. The specifications describe what is to be available to the user but not how it is to be presented. You are encouraged to be creative in creating your user interface.

Flight Specifications
Your program must allow for users to create new flight specifications with all of the same data in Program 1. A dialog should be presented to the user allowing them to enter the following data:

- flight number,
- destination,
- number of rows,
- number of seats per row,
- exit rows,
- aisle seats,
- window seats.

Seats and exit rows are labeled and specified as before. At any time after the flight specification is created, a user must be allowed to view the complete flight specification information.

**Flight Deletions** In addition to creating new flights, users must be allowed to delete flights from the system. Deleting a flight also deletes any reservations and the waiting list associated with the flight.

**Seat Reservations**

Your program must allow for users to create new reservations for a given flight. The user should be allowed to select from a list of available flights when making a reservation. After the flight is selected, the user is presented with a dialog allowing them to enter the following information:

- passenger’s first name,
- passenger’s last name,
- passenger’s middle initial,
- seating preference.

Note that the seating preference should be chosen from a list or using radio buttons, not using the cryptic codes used in Program 1.

A complete list of reservations for a flight should be available for viewing by the user at any time. Seats are assigned in the same fashion as in Program 1. However, when the plane is full, instead of reporting an error, the passenger is placed on a “waiting list” for the flight. The waiting list is a first come first served list that can hold up to 20 passengers. If both the flight and waiting list are full, an error reporting that reservation could not be made is presented to the user. The waiting list for any flight should be available for viewing by the user at any time.

**Cancellations**

Your program must allow for reservations to be canceled. To cancel a reservation, a user should only need to select the flight and the reservation to cancel from lists of flights and reservations. Cancellations may also be made for reservations in a flight’s waiting list.

When a cancellation is made, the waiting list for the flight should be searched to see if there is anyone waiting. If someone is waiting, they should be removed from the waiting list and placed in the newly available seat. EXCEPTION: If the newly available seat is in an exit row, then the first person in the waiting list who requested an exit row seat is placed in the newly available seat. If no one on the waiting list requested an exit row seat, no one is placed in the newly available seat.

**Errors**

The user interface will help to prevent errors that existed in the first program. The only errors that remain are

1. Flight specification already exists.
2. Reserving a seat on a full flight with a full waiting list.

When an error occurs, you should display a dialog box with the appropriate error message. A “Close” button should be on the dialog box which is close the window and place the user back at the main program window.
Graphical User Interface Guidelines

Although you are encouraged to be creative with your user interface, you need to follow some guidelines.

Labels

Every text entry field, list box, button, or other user interface component must be labeled. The label should indicate the appropriate use for the component. See the examples for ideas.

Dialog Boxes

If you use dialog boxes in your program, then every dialog box must have a “Close” button which closes the dialog box and does not perform any action such as creating a new specification, reservation, or cancellation.

Menus

Your program must have a menu bar and a “File” menu containing a “Quit” menu item. The “Quit” menu item exits the program immediately.

GUI Examples

Figures 1 and 2 are some sample screen shots to get ideas for implementing your own user interface.

Output

The program generates reports for flights as it did in program 1. The reports for each flight are generated in ascending order by flight number. Reports are generated when a user “saves” a file. If the user has not previously named the file to be saved, a dialog allowing the user to choose a file name must be presented. Once a file name as been chosen, all subsequent “saves” generate reports to the same file name, overwriting previous reports.

Also, the option to “save as” a report must be available to the user. A user choosing “save as” will always be presented with a dialog to choose a file name for the report file.

Design Requirements

This program must satisfy the same design requirements as Program 1. In addition, the waiting list should be implemented using an array implementation of a modified queue abstract data type (ADT).

Execution

The program, called guflight.exe should start with a “splash screen”, a window containing the class name, programming assignment name, and author(s) name(s). There should be a button on the splash screen that removes the splash screen and displays your main program screen.
Assumptions

You may assume that input strings will be no longer than 255 characters long, there will be no more than 20 exit rows per flight, and that there will be no more than 26 seats per row. You may assume that two or more passengers with the same name will not reserve seats on the same flight. Also you may assume that data of the correct format will be typed into input fields.

Structure Charts

An initial structure chart is due **Wednesday, July 29, 1998**. This should represent your understanding of the problem at this point, and must be reasonably complete to receive full credit. The initial structure chart must be given in the binder/folder used for the final program submission. A
final structure chart is to be turned in with the program submission. The final structure chart for this program must be generated using a computer drawing package.

**User’s Guide**

You are to turn in a user’s guide for your program. The user’s guide must be created on a computer and must describe the entire functionality of your program and how to perform the various functions for creating and deleting flight specifications, making reservations, and canceling reservations. Your user’s guide must have a table of contents and must be written using proper English sentences and grammar.

Screenshots can be included in the guide and are recommended. A screenshot can be obtained under Windows NT/95 by pressing the “Print Scrn” button, and then opening the Paint accessory and selecting “Paste” under the “Edit” menu. You should trim the screen shot to contain only your program windows.

**Grading**

The due date for this program is **Wednesday, August 5, 1998**. Turn in hard copies of the following: source code, sample output files, initial and final structure charts, and user’s guide. The hard copies should be placed together in one (three-ring) binder or folder, with each product clearly labeled and separated by a divider. Submit a diskette (system labeled: Windows 95 or Windows NT), with files containing: ASCII source code, MS C++ project files, executable image, I/O files, and a brief ASCII README file with execution instructions. The executable image must be called guiflight.exe. The files may be in a zip compressed self extracting archive. The disk should be placed in the binder.
or folder along with your hard copies. To receive partial credit for nonworking program, a brief one or two paragraph description of the problem(s) must be included in the assignment binder/folder. The location, routine minimum, must also be specified with possible corrections that need to be made.

In addition to the program submissions, you will be required to demonstrate your program to the teaching assistant or instructor on Wednesday, August 5, 1998 or Thursday, August 6, 1998. A sign up sheet for selecting a demonstration time will be available in class and on the instructor’s door starting Wednesday, July 29, 1998. More details about the demonstrations will be given in class.