COEN 44 Applied Programming **Fall 2016**

Lab 8 Revised

Reservation systems are a commonplace computer application – think restaurants, movies, sporting events, concerts, … This lab requires you to implement a simplified hotel reservation system that accepts and deletes reservations, maintains a waiting list of people who try to make a reservation when there's no room available, and lists the reservations that have been made.

This is a group project, to be conducted in groups of two. The members of a group must be from the same lab section.

Your group should work on the architecture of your solution and the design of the various functions together.

Objectives:

1. Develop an outline for the structure of a program from scratch.
2. Convert the outline to a C program.
3. Demonstrate good programming style: commenting, indenting blocks appropriately, good identifier names, etc.
4. Become more familiar with control structures, programmer-defined functions, arrays (including parallel arrays), and elementary string operations.

Part 1: Design and implement the reservation program

Your program will deal with reservations for a single night in a hotel with **3** rooms, numbered 1 to **3**. It must use an infinite loop to read commands from the keyboard and quit the program (return) when a quit command is entered. Use a switch statement to choose the code to execute for a valid command. The valid commands are:

R or r: reserve a room
C or c: cancel a reservation
W or w: remove a request from the waiting list
L or l: list the current reservations for the night
Q or q: quit the program
Any other input: print an error message and prompt for another command

You must use a one-dimensional array to represent the room, a parallel one-dimensional array of character strings to hold the name associated with the reservation, a one-dimensional array to hold waiting list entries, and a parallel one-dimensional array to hold the name associated with the waiting list entry. Actions taken in response to a valid command (r, c, w, or l) must be implemented using programmer-defined functions, one per command. Any needed data must be passed to the functions, not declared globally. Implement reservation ids using a simple integer counter. Names will have fewer than 15 characters. Actions for each command are:

Reservation: If there is a free room, reserve a room by assigning the next reservation id to the array element representing that room, print the reservation id for the person at the keyboard, and prompt for and read the name associated with the reservation. If there are no rooms, print an appropriate message and ask if the person wants to be entered on the waiting list. If they do, place the reservation id in the waiting list array, print the reservation id for the person at the keyboard, and prompt for and read the name associated with the waiting list entry.

Cancellation: If there is a room reserved under that reservation id, cancel the reservation. Otherwise print a message that the id is not valid. If a room is cancelled and there are entries on the waiting list, remove the first entry on the waiting list (array element [0]) assign that reservation id and the associated waiting list name to the room, then print a message indicating that reservation id is now confirmed. Shuffle any other entries on the waiting list up one place.

Wait cancellation: If there is a waiting list entry with that reservation id, delete it and the associated name, and shuffle all waiting list entries below the one deleted up one place. Otherwise print a message indicating that id is not on the waiting list.

List reservations: Print the reservation ids and associated names of all rooms that are reserved, along with the room number that reservation occupies. Do not print anything for rooms that are vacant. If there are no rooms reserved, print a message indicating that. **If there are any entries on the waiting list you should also print the reservation number and name of all elements on the waiting list.**

Quit: end the program by returning from the main function.

Any other command: print an error message and prompt for another command.

Your solution must use four arrays (two for the rooms and two for the waiting list). The waiting list should have as many entries as there are rooms. Use an integer counter for reservation ids that starts at 1. Reservation ids are not reused. Use another integer to keep track of the number of rooms reserved.

Your solution will be for a boutique hotel with only 10 (very expensive) rooms. But make liberal use of #define statements so it would be trivial to adapt your solution to a larger hotel.

Lab Steps:
1. Open up VS and Create a new **Project**
2. This will be a **console program** in C
3. Type in your program using good style practices
4. Test and debug your program using your test data
5. Demonstrate the execution for the TA
6. Submit your final program, with the outline of your solution as a comment at the beginning of the source code, to Camino.

Part 2: Possible enhancements

How would your program have to be modified to deal with reservations for a week, instead of a day? What about a month? A year? How should the program be modified to deal with some rooms having different prices? How would the interaction have to change to deal with that?

Submit answers to these questions to Camino as a document file.