Programming Project 4

Due: Friday, 9/19/14 at 11:59 pm

Restaurant Waiting List System

For this lab, write a C program that will implement a customer waiting list that might be used by a restaurant. The program must use **a linked list** to implement the queue-like data structure.

The linked list is to maintain the following information for each group that is waiting:

- name (we assume a maximum name length of 30 characters)
- group size
- in-restaurant status: whether the group has called ahead or is waiting in the restaurant

The system does not take reservations for a specific time and date (i.e. table of 4 for 7pm on Saturday), but it will allow for a group to call ahead and get their name on the waiting list before they arrive. Note: these call-ahead groups will still need to check in when they arrive so the staff knows they are waiting in the restaurant.

Groups are added to the wait list when they call-ahead or when they arrive at the restaurant. Groups are always added to the end of the wait list. The system will require that each name used be unique. So when a group is added to the wait list, the system must make sure that no other group is already using that name.

When a table becomes available in the restaurant, the system returns the name of the group that is in the restaurant and can sit at that table (i.e. the number of seats at the table is greater than or equal to the number of people in that group). Note that this may not be the first (or even the second or third) group on the wait list.

The commands used by this system are listed below and are to come from standard input. Your program is to prompt the user for input and display error messages for unknown commands or improperly formatted commands. Note that the name of the group when given will be given as the last item on the input line. The name of the group may contain white space characters in the middle of the name but not at the beginning or end of the name. Each command given must display some information about the command being performed.

Command	Description
q	Quit the program.
?	List the commands used by this program and a brief description of how to use each one.
a <size> <name>	Add the group to the wait list using the given group size and name specifying the group is waiting in the restaurant. The group's information is added to the end of the list. If the name already exists in the wait list, give an error message and do not add the information.
c <size> <name>	Add the group to the wait list using the given group size and name specifying the group as a call ahead group. The group's information is added to the end of the list. If the name already exists in the wait list, give an error message and do not add the information.

w <name>	Mark the call ahead group using the given name as waiting in the restaurant. If the name does not exist is the wait list or is not a call ahead group, give an error message.
r <table-size>	Retrieve and remove the first group on the wait list that is waiting in the restaurant and is less than or equal to the table size. Note that "first" is the group that has been in the wait list the longest.
l <name>	List total number of groups that are in the wait list in front of the group specified by the given name. Also list the size of each group that are in the wait list ahead of the group specified by the given name. If the name does not exist, give an error message.
d	Display the total number of groups in the wait list. Also display the names, group size and in-restaurant status of all groups in the wait list in order from first to last.

Note that **<size>** and **<table-size>** are to be integer values and **<name>** is a list of characters.

Use of C struct and C functions

When writing your code, you MUST create a C struct for the nodes in the linked list of the wait list. These data items must include the following (and may include others if needed).

- the name of the group
- the integer variable specifying the size of the group (number of people in the group)
- the in-restaurant status (5 extra points for using an enum!)
- a pointer to the next node in the list

The pointer for the head of the linked list MUST be declared in main(). **It may NOT be global.** Each operation performed on the linked list MUST be done in its own function. These function must take the head of the linked list as its FIRST parameter.

Linked List Operations/Functions

You must write C functions for the following 7 operations. These functions must be called when the specified commands are given as input.

addToList ( ) – This operation is to add a new node to the end of the linked list. This is to be used when the a and c commands are given as input.

doesNameExist ( ) – This operation is to return a Boolean value indicating whether a name already exists in the linked list. This is to be used when the a, c, w and l commands are given as input.

updateStatus ( ) – This operation is to change the in-restaurant status when a call-ahead group arrives at the restaurant. This operation will return a FALSE value if that group is already marked as being in the restaurant. This is to be used when the w command is given as input.

retreiveAndRomove ( ) – This operation is to find the first in-restaurant group that can fit at a given table. This operation is to return the name of group. This group is to be removed from the linked list. This is to be used when the r command is given as input.

countGroupsAhead ( ) – This operation is to return the number of groups waiting ahead of a group with a specific name. This is to be used when the l command is given as input.

displayGroupSizeAhead ( ) – This operation traverses down the list until a specific group name is encountered. As each node is traversed, print out that node's group size. This command is to be used when the l command is given.

displayListInformation ( ) – This operation to traverse down the entire list from beginning to end. As each node is traversed, print out that node's group name, group size and in-restaurant status. This command is to be used when the d command is given as input.

Command Line Argument: Debug Mode

Your program is to be able to take one optional command line argument, the -d flag. When this flag is given, your program is to run in "debug" mode. When in this mode, your program is to display each group's information as you traverse through the linked list of the wait list. Note that for the w, r and l commands, the entire list may not be traversed, so you only display the part of the list that is needed to be traversed to complete the command.

When the flag is not given, this debugging information should not be displayed. One simple way to set up a "debugging" mode is to use a boolean variable which is set to true when debugging mode is turned on but false otherwise. This variable may be a global variable. Then using a simple if statement controls whether information should be output or not.

```
if ( debugMode == TRUE )
        printf (" Debugging Information \n");
```

Provided Code for the User Inteface

The code given in proj4.c should properly provide for the user interface for this program including all command error checking. This program has no code for the linked list. It is your job to write the functions for the specified operations and make the appropriate calls. Most of the changes to the existing proj4.c program need to be made in each of the **doXXXX ( )** functions. Look for the comments of:

```
// add code to perform this operation here
```

Note: the head of the linked list is required to be a local variable in main and you are required to pass the head of the linked to the operation functions. All of the **doXXXX ( )** functions currently have no parameters. It will then be expected that you modify the function signature to allow for this information to be passed as required.

Program Submission

You are to submit the programs for this lab via the Assignments Page in Blackboard.

To help the TA, name your file with your net-id and the assignment name, like:

- ptroy1ProjectX.c