CSE 430 - Operating Systems
Fall 2015
Project #1
Due Date: Oct 8, 2015 (soft deadline, see comment below)\

Groups: Groups of 2 are mandatory, and there will be a penalty of 40% for not having a group. Only 1 submission per group, i.e. there will be two names on each submission.

Due Date: **This project is NOT due, and not graded** – as it is essential that it works perfectly for the next projects to work. Hence the proper implementation is mandatory. The due date is a target date so that you are on track.

Bugs in the Q routines have been the #1 cause for strange errors in the project, always. Careful that you get it right, else things will go bump later.

Overview:

For this project you are to write routines that perform standard queuing functions. The functions work on multiple queues, and structure each queue as a doubly linked, circular list.

Data Structures:

A queue consists of a head-pointer and a set of q-elements.

A q-element is a structure, consisting of a prev and next pointer, and a payload consisting of 1 integer. The header is a pointer to the first element of the queue. *The head pointer is "null" if the q is empty.*

Functions:

The functions that you implement are:

1. **item = NewItem**(); // returns a pointer to a new q-element
2. **InitQueue**(&head) // creates a empty queue, pointed to by the variable head.
3. **AddQueue**(&head, item) // adds a queue item, pointed to by "item", to the queue pointed to by head.
4. item = **DelQueue**(&head) // deletes an item from head and returns a pointer to the deleted item
5. **RotateQ**(&head) // Moves the header pointer to the next element in the queue. This is equivalent to AddQ(&head, DeleteQ(&head)), but is simpler to use and more efficient to implement.

Note: All the routines work on pointers. They do ***not*** copy q-elements. Also they to not allocate/deallocate space (except NewItem()). You may choose to implement a **FreeItem**(item) function – optional.

Implementation The routines should me implemented in C under the Linux operating system. If you are not familiar with Linux and/or do not have it installed. Please use a Ubuntu Virtual Machine. For more details/questions post requests on the discussion board.

All the above routines and data structures are to be placed in 1 file, called "q.h". Do not include other files into this file. The test routines can be in "proj-1.c" and this will include q.h and other standard header files.

Testing

Write test routines that thoroughly test the queue implementation. Use multiple queues. Pay special attention to deleting the last element of a q. Also make sure "RotateQ" is behaves properly (strange behavior of this routine means the insert/delete routines have bugs.)

One suggested test case: Add three elements to queue Q1, and then add 3 more to queue Q2. Rotate each Q twice.

Delete elements from each q, one by one and print values till the queues are empty. Repeat the above test again.

Further warning: Bugs in the Q routines have been the #1 cause for strange errors in the project, always.

Submission and Grading: (do not submit this project)

SUBMIT (not for this project…., but this is the technique that will be used for later projects):
Your project must consist of 2 files. (Usage of C is encouraged but if you use C++ the instructions would be similar)

1. q.h -- the functions defined above should be here, both definitions and body of the functions.
2. proj_1.c (to test q.h)
   (make sure the compile command, "gcc proj_1.c" does the correct compilation).

The two files are to be ZIPPED into one zip or gzip file. The name of this file should reflect the name(s) of the people submitting (abbreviated, do not make it too long).
Email the zip file to 430<dot>proj at gmail.
Note: Grading will be done on Ubuntu. The entire project has already been tested for compatibility on Redhat (general), Ubuntu 11 and Ubuntu 12, and the same code works.