SIENA*college*
Computer Science

Computer Science 330
Operating Systems
Siena College
Spring 2012

Lab 3: "Object-Oriented" C Programming

Due: 10:00 AM, Tuesday, February 14, 2012

For your second programming project, you will be developing a discrete event simulation of a CPU scheduler. Such a program requires a number of data structures and objects. You are all familiar with how to do this in a language like Java, but how can it be done in a language like C, which does not support objects.

A Very Simple Example

Recall an example you saw in CSIS 220 and during our first lab this semester.

See Example:
`~jteresco/shared/cs330/examples/ratios`

Re-read the section from the CSIS 220 MIPS decoding lab that describes this example. In particular make sure you understand how the data structure is defined, which function plays the role of the constructor, and how the other functions access and/or modify instances of the structure. Also, make sure you understand what belongs in a header (`.h`) file and what belongs in the implementation (`.c`) file.

A C Q

Develop a queue structure and corresponding functions to operate on queues in C that holds `int` values. Include an approrpriate header file, implementation file, and a separate file with a `main` function that tests your implementation. Also include a `Makefile` that compiles your queue implementation and your testing code.

Queue of Ratios

Next, create a queue structure and corresponding functions to operate on queues in C that holds `ratio` values. You may use the `ratio` structure from the `ratios` examples. Again, include an appropriate header file, implementation file and a file containing a `main` function that tests your implementation. Also include a `Makefile` that compiles your queue implementation and your testing code.

Priority Queue of Ratios

Next, create a priority queue structure and corresponding functions to operate on priority queues in C that holds `ratio` values. Your priority queue should remove the smallest ratio when an item is removed. Again, include an appropriate header file, implementation file and a file containing

a `main` function that tests your implementation. Also include a `Makefile` that compiles your priority queue implementation and your testing code.

Submission and Evaluation

This lab is graded out of 30 points.

To submit this lab, place all of the files that you are to turn in (and nothing else) into a directory containing three other directories, one for each structure implementation, its testing code and `Makefile`. In the parent directory of these three directories, create a "tar file" to submit using a command similar to:

```
tar cvf lab3.tar intqueue ratioqueue ratiopq
```

This will create a file `lab3.tar` in your directory. Send this tar file as an attachment to *jteresco@siena.edu* by 10:00 AM, Tuesday, February 14, 2012.

Please include a meaningful subject line (something like "CS330 Lab 3 Submission") and use the exact filenames specified (for this lab and all semester) to make my job easier when gathering your submissions together for grading. You don't want to annoy your grader with misnamed or missing files just before he grades your assignment. Please do not include any additional files, such as emacs backup files, object files, or executable programs.

Grading Breakdown	
`int` queue correctness	5 points
`ratio` queue correctness	5 points
`ratio` priority queue correctness	5 points
Testing functions	6 points
Code organization	3 points
Documentation	5 points
`Makefile`s to build executables	1 point