Programming Assignments

All six projects for this course are programming assignments. You may write and test the code on any Unix system, **but** you must make sure your code is copied to, compiles on, and runs on the machines at the science center. You are encouraged to use C, the language of Unix system programming, but you may use C++.

Grading

Homework assignments are graded on a 100 point scale. The 100 points are divided between Function (70 points) and Design (30 points). In software engineering, getting a program that works is only part of the problem. The rest of the problem involves updating, correcting, and reusing the code.

Thus, if your program works right but is poorly designed, you get a C. Design is further divided into three categories each worth 10 points: Documentation, Readability, and Modularity. Documentation refers to commenting; files, functions, variables, and chunks of code should be documented. Readability means two things: how well code is arranged on the screen and how well objects are named. Modularity refers to how the program is broken into separate, well-defined, and insulated units. A detailed description of what we want will be distributed with the assignments.

Academic Honesty

The work you submit must be your own work. You may build your code on samples from class or examples from texts, and we encourage students to discuss problems and techniques. Your homework should be your own work or a combination of your own work and your synthesis and extension of examples. Please state the sources of any piece of code you use.

Submitting Homework

Homework is due by midnight on Saturday evenings. There is a 10 point penalty for each day late you turn in an assignment. You will submit code and text electronically. Please see the course website for an explanation of submitting your work by computer.

Creating Sample Runs

For most projects you will include a sample run of your program. Use the script command to capture the sample runs of your program. script will record everything that appears on the screen and saves it all to a file.

To make a script, type in `script`. The computer will print a message and give you the "%" prompt. Now run your program. Type "exit" at the "%" prompt when you wish to stop recording. Unless you specify some other name, script will save everything in a file called "typescript", so include thst file. A sample session is shown below:

```
$ script
Script started, file is typescript
$ cat foo.c
 . . .
$ ./a.out
 . . .
$ exit
Script done, file is typescript
```