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 all your own work or a combination of your own work and your synthesis and extension of class examples. You must cite any sources.

You are responsible for understanding Harvard Extension School policies on academic integrity (www.extension.harvard.edu/resources-policies/student-conduct/academic-integrity) and how to use sources responsibly. Not knowing the rules, misunderstanding the rules, running out of time, submitting the wrong draft, or being overwhelmed with multiple demands are not acceptable excuses. There are no excuses for failure to uphold academic integrity. To support your learning about academic citation rules, please visit the Harvard Extension School Tips to Avoid Plagiarism (www.extension.harvard.edu/resources-policies/resources/tips-avoid-plagiarism), where you'll find links to the Harvard Guide to Using Sources and two free online 15-minute tutorials to test your knowledge of academic citation policy. The tutorials are anonymous open-learning tools.

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 designed poorly, you get a C. Design is divided into three categories each worth 10 points: Documentation, Modularity, and Clarity. *Documentation* refers to commenting; files, functions, variables, and chunks of code should be documented. *Modularity* refers to how the program is broken into separate, well-defined, and insulated units. *Clarity* refers to the physical presentation and logical design of the program. Have you ever read a book or essay that expressed an idea or technique so clearly that after just one reading, you saw exactly what the writer was getting at? Have you ever read a book or essay that, even after several readings still confused you? Some code is really clear, and some code is hard to figure out, hard to maintain, hard to modify. Strive for clarity; you'll be graded on it.

A detailed description of what we want will be distributed

TURNING HOMEWORK IN

Due on Saturday at 11:59PM

Homework is due by 11:59pm on Saturday. There is a 10 point penalty for each day late. You must turn in both a listing of all your code and sample runs of your program (how to do this is described below).

Electronic Version

We use an computerized system for submitting and returning homework. You do not have to print any paper for most of the assignments. Details for using the system will be provided with the assignment.

GENERATING SAMPLE RUNS

Use the `script` command to generate the sample runs you will hand in. When you run `script`, everything that appears on your terminal screen is saved in a file. To make a script, type in `script`. The computer will print a message and give you a new, regular prompt. Now list your program (using cat) and run it. 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 print that file. A sample session is shown below:

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

You can specify a different name for the script file by typing the command `script` *`filename`*. **Note:** Every time you run script the script file will be overwritten.