**Grade: 20/20**

**Homework 1**

**Focus**

* Functions
* Structs
* Vectors
* I/O

**Problem:**

We will model a game of medieval times. Our world is filled with warriors. Naturally what warriors like to do is fight. To the death. So we happily let them.

Each warrior starts out with a name and a certain amount of strength. Each time he fights, he loses some strength. (He gets to keep his name.) If his opponent is stronger than he is, then he loses *all* of his strength, in which case he is dead, or at the very least pretty useless as a fighter. Otherwise he loses as much strength as his opponent had. Of course, if he and his opponent had the same strength then they are both losers.

Even losers are allowed to pick a fight. It doesn't require having any strength in order to do battle with someone else. Not that you stand much of a chance of winning anything, but perhaps it's worth getting beaten (again) just to have those 15 seconds of fame.

Your program will read in a file of commands. There are three types of commands:

* Warrior creates a new warrior with the specified name and strength.
* Battle causes a battle to occur between two warriors.
* Status lists all warriors, alive or dead, and their strengths.

A sample input file looks like:

Warrior Jim 42  
Warrior Lancelot 15  
Warrior Arthur 15  
Warrior Torvalds 20  
Warrior Gates 8  
Status  
Battle Arthur Lancelot  
Battle Jim Lancelot  
Battle Torvalds Gates  
Battle Gates Lancelot  
Status

The name of the input file will be "warriors.txt". Note that the commands do not have to appear in that order. The only requirement is that a Battle command cannot appear until the specified warriors have been seen.

The Status command displays how many warriors there, then displays each one with his strength. The Warrior command does not display anything. The Battle command displays one line to says who is fighting whom and a second line to report the results, as shown below.

The output (which would display on the screen) for this file should look like:

There are: 5 warriors  
Warrior: Jim, strength: 42  
Warrior: Lancelot, strength: 15  
Warrior: Arthur, strength: 15  
Warrior: Torvalds, strength: 20  
Warrior: Gates, strength: 8  
Arthur battles Lancelot  
Mutual Annihilation: Arthur and Lancelot die at each other's hands  
Jim battles Lancelot  
He's dead Jim  
Torvalds battles Gates  
Torvalds defeats Gates  
Gates battles Lancelot  
Oh, NO! They're both dead! Yuck!  
There are: 5 warriors  
Warrior: Jim, strength: 42  
Warrior: Lancelot, strength: 0  
Warrior: Arthur, strength: 0  
Warrior: Torvalds, strength: 12  
Warrior: Gates, strength: 0

**Programming guidelines**

* For all assignments, please use the aspects of the language that we are focusing on in the assignment. For example, in this assignment, one of the topics is the use of struct. That means we do *not* want you to define classes or methods, or to use data hiding &emdash; that'll all be in the next assignment. (Obviously, you shouldn't worry if you don't know what I was just talking about there.) Why the restrictions? The purpose of each assignment is to have you excercise specific technical skills. If you want to wow us with the cool stuff you have figured out that goes beyond the current syllabus, great! Come by the office and I will be happy to look at and discuss your work.
* Make *good use* of functions. In this program, for example, when you determine that you are looking at a Warrior command then you should call a function to handle that command. Similarly with the other commands.
* Begin your file with a comment identifying who you are and what this program is for.
* If there are any *known* problems with your program include a comment explaining them. You *have* carefully tested your program, right?
* The program should have a "reasonable" amount of *useful* comments.
* The code must be well formatted. *Never* turn in code with bad indentation.
* All structs, functions and variables should have *good* names.
  + Function names should clearly identify what the function does.
  + A variable name that is a single letter, such as i, should only be used within the scope of a loop as a loop index. Ok, yes, x and y might make excellent names for referring to the coordinate of a point in 2D space.
  + temp is almost never a good name.
* Follow the convention that structs begin with an uppercase letter, constants are all uppercase and functions and variables begin with a lowercase letter.

**Turn in**

Hand in a single cpp file, hw01.cpp, containing your program.