CS1050 – Lab 11

Fall 2021

Concepts to Practice

- File Processing
- Standard string functions

Submission Information

Submit this assignment by following the instructions given by your TA. SUBMIT ONLY the .c file (no a.out or executable file is required). All of the lab assignments must be submitted before the end of the lab using the lab code given by the TA.

Use the following submit command:

```
mucs submit <class> <assignment_name> <filename>
```

For example:

```
mucs submit 1050 lab11 lab11.c
```

Description

OK, I admit it. I am 55 years old, and I like Dungeons and Dragons. Still. My character (Ahote) is a barbarian with a flaming sword, and he likes nothing better than to fight. Since he isn't real, the best solution is for him to fight some (also not real) monsters or watch them fight each other (don't try this at home!).

For the lab assignment, you will read data from a file called monster.csv. Your goal will be to read data for the following two creatures from this file and to have them "fight" each other:

Ahote
FrostGiant

The first thing you need to do is get all the "starter code" for this lab. To get started on this lab, type the following while logged in to tc.rnet.missouri.edu:

```
cs1050start lab11
```

This command will create a directory called lab11. Go into that directory ("cd lab11") and get a list of the files there ("ls -la"). Notice that there is a file called lab11.c. You can start editing this file to do your lab. Notice that there is a commented-out line that says "SetSeed(0)". Later on, if you want to see different results each time you run the program, you can just uncomment this line to make the fights truly "random".

One more thing you should notice is that there is a header file called "fight.h". This file contains the names of two functions that I have provided for you. One is the SetSeed() function previously mentioned. The other is called Fight() and you should call that function when you have all of the information from the monster.csv file you need. You will need the following stats for each monster: AC (armor class), HP (hit points), hitbonus (To Hit Bonus), a string the represents the damage for the first attack, and a string that represents the damage for the second attack (some creatures get two attacks).

If all of that sounds confusing, all you need to do is read 3 integers (ac, hp, hitbonus), 2 strings (attack1damage, and attack2damage), and the name of the monster (also a string) for each monster. Pass these parameters (in the correct order) into the Fight() function, and you are good!

Instead of your usual command (compile lab11.c), you should just type in "make". This will compile your program and make sure it is linked to the library that has the Fight() function in it.

Hints

- If you are having trouble understanding how things work, just call the Fight() function with some hard-coded data to try it out. Example: Fight("JimR",21,900,10,"5d6+5","5d12+5","Lar",5,100,5,"1d6","N/A");
  You could literally just add this example line to the lab11.c and then type make and then run ./a.out to see it work.
- Hard-code your data file and the two creature types initially. You can change this later if you are doing the bonus. If you get things working with hard-coding, go ahead and submit your code in case you run out of time. You can always submit again.
- You might want to grab some of the code from your prelab. Reading the name of the creature and the attack damage strings is very similar to the way we read names in the prelab (since each of these ends when you hit a comma).
- Don't forget to check whether you have reached the end of the file.
- Don't forget the close the file when you are all done.
- If you get everything working, it is more fun to call SetSeed(0). That will set things to a random number, so you will get different results each time you run the program. You might want to wait and try this at the end, as it is easier to debug things if you get the same results each time you run it.

Bonus

Yep, this program includes the possibility of getting bonus points again! Hooray! All you have to do is remember my lecture about command-line processing (or look it up if you have time in lab). To get the bonus points, you must not "hard-code" the name of the data file or the names of the creatures that will participate in the fight. Instead, you must take each of these as a command-line argument. If you do the bonus, you must also check that the user has put in the correct number of arguments, that the specified file can be read, and that the creatures specified actually exist in the data file.

Honors

Nothing special for the honors folks for a change. Just do the regular assignment and try to do the bonus part if you can – you will still get bonus points like everyone else if you succeed.

Sample Output from 8 runs (shown using optional Bonus command-line parameters)

```
jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ make
cc -c lab11.c
cc lab11.o -lfight -L.
jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out
*** Syntax:
./a.out combatant_file combatant1_name combatant2_name

jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out asdfasdf
*** Syntax:
./a.out combatant_file combatant1_name combatant2_name

jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out asdf asdf asdf
** Error:  Could not open asdf
jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out monster.csv asdf asdf
*** Failed to read the right number of data elements from the file:-1 ***
*** Error: could not find combatant asdf
jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out monster.csv Ahote asdfasdf
*** Failed to read the right number of data elements from the file:-1 ***
*** Error: could not find combatant asdfasdf
jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out monster.csv asdfasdf Ahote
*** Failed to read the right number of data elements from the file:-1 ***
*** Error: could not find combatant asdfasdf
jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out monster.csv Ahote AirElemental
**** Combat between Ahote and AirElemental ****
**Ahote=90, AirElemental=90

        Ahote hits AirElemental for 25 damage
        Ahote hits AirElemental for 20 damage
        AirElemental misses Ahote
        AirElemental misses Ahote
**Ahote=90, AirElemental=45

        Ahote hits AirElemental for 22 damage
        Ahote hits AirElemental for 27 damage
**Ahote=90, AirElemental=-4

Ahote wins!
**** Combat ends ****

jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out monster.csv Ahote Zombie
**** Combat between Ahote and Zombie ****
**Ahote=90, Zombie=22

        Ahote hits Zombie for 25 damage
**Ahote=90, Zombie=-3

Ahote wins!
**** Combat ends ****

jimr@JimRXPS13:~/CS1050/FS2021/labs/lab11$ ./a.out monster.csv asdfasddf Ahote
*** Failed to read the right number of data elements from the file:-1 ***
*** Error: could not find combatant asdfasddf
```

Guidelines for Grading Lab 11
40 Points Possible (+5 Bonus)

General

If your program does not compile or produce any input/output (I/O) because most of the source code is commented out then your lab will receive a grade of ZERO POINTS. Further, if your program does not actually follow the specifications, but merely prints out lines that make it appear to follow the specifications, you will receive a grade of ZERO POINTS. For partial credit your C program must not only compile but also produce some valid I/O that meets the lab specifications.

You program is expected to have a comment header at the top that includes your name, pawprint, the course you are taking, and the lab that you are solved (e.g., "Lab 11"). Your code should be nicely indented. **You will lose up to 10 points if you do not meet these basic requirements**.

5 points: Your code opens the file correctly.
5 points: Your code correctly closes the file before the program ends.
5 points: Your code reads ac, hp, hitbonus correctly.
5 points: Your code reads each of the two attack damage strings correctly.
5 points: Your code reads create name correctly.
5 points: Your finds each of the specified creatures in the file correctly.
5 points: Your program is able to use the provided Fight() function, calling it with correct parameters.
5 points: Your output fairly closely matches the sample output.
BONUS 5 points: Your code accepts command-line parameters specifying which file to open and which two creatures should fight. It checks whether the user has input enough information on the command-line and whether the provided parameters are valid (e.g., can you actually find the given creatures in the data file).