**Submitted By Eric Miller 08/23/2014**

**Final Project Plan and Analysis**

**Intro:**

-The final project is another iteration of the fantasy combat program we have been making over this past term.

-This time the twist is that it will be a battle between 2 teams of characters, instead of just one on one.

-Each team will be comprised of a list (contained in a structure), the head of which goes into combat.

-The character in combat will return to the back of their team’s list should they win, but join a list of dead characters should they lose.

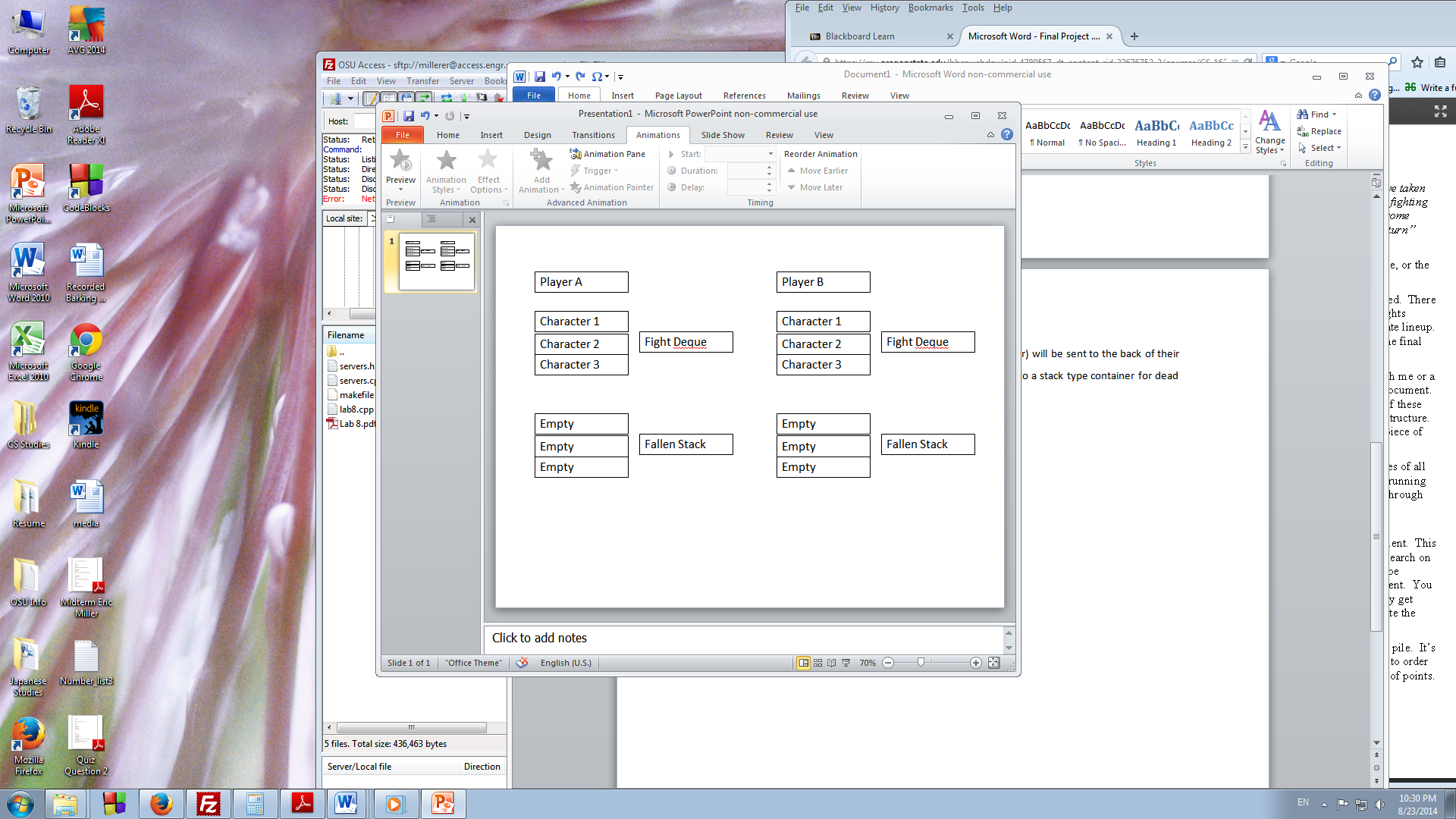
-At the end of the program, 1st, 2nd, and 3rd place characters are output along with their team.

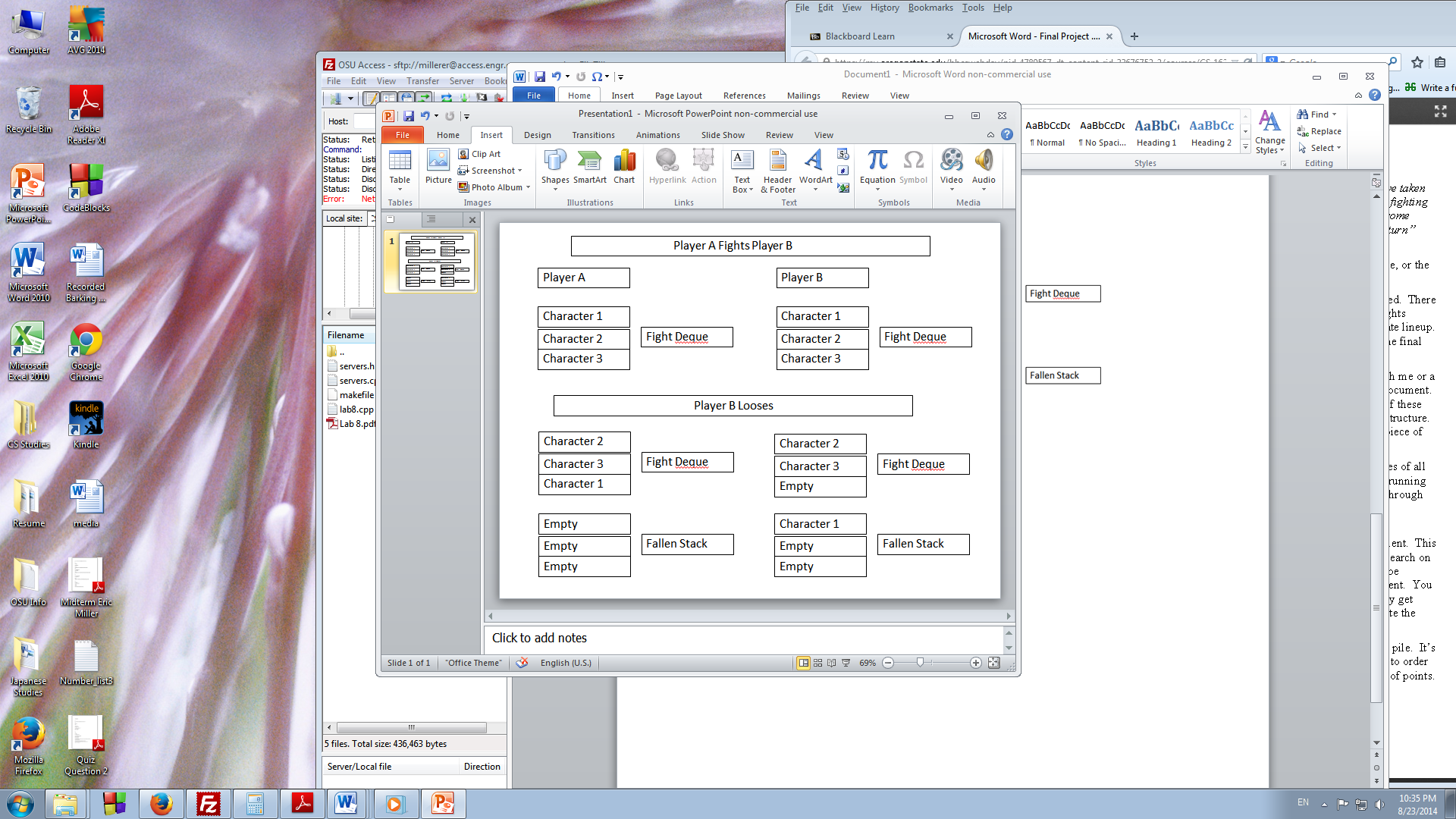
-The ranking system appears to be up to us, so I will outline one below.

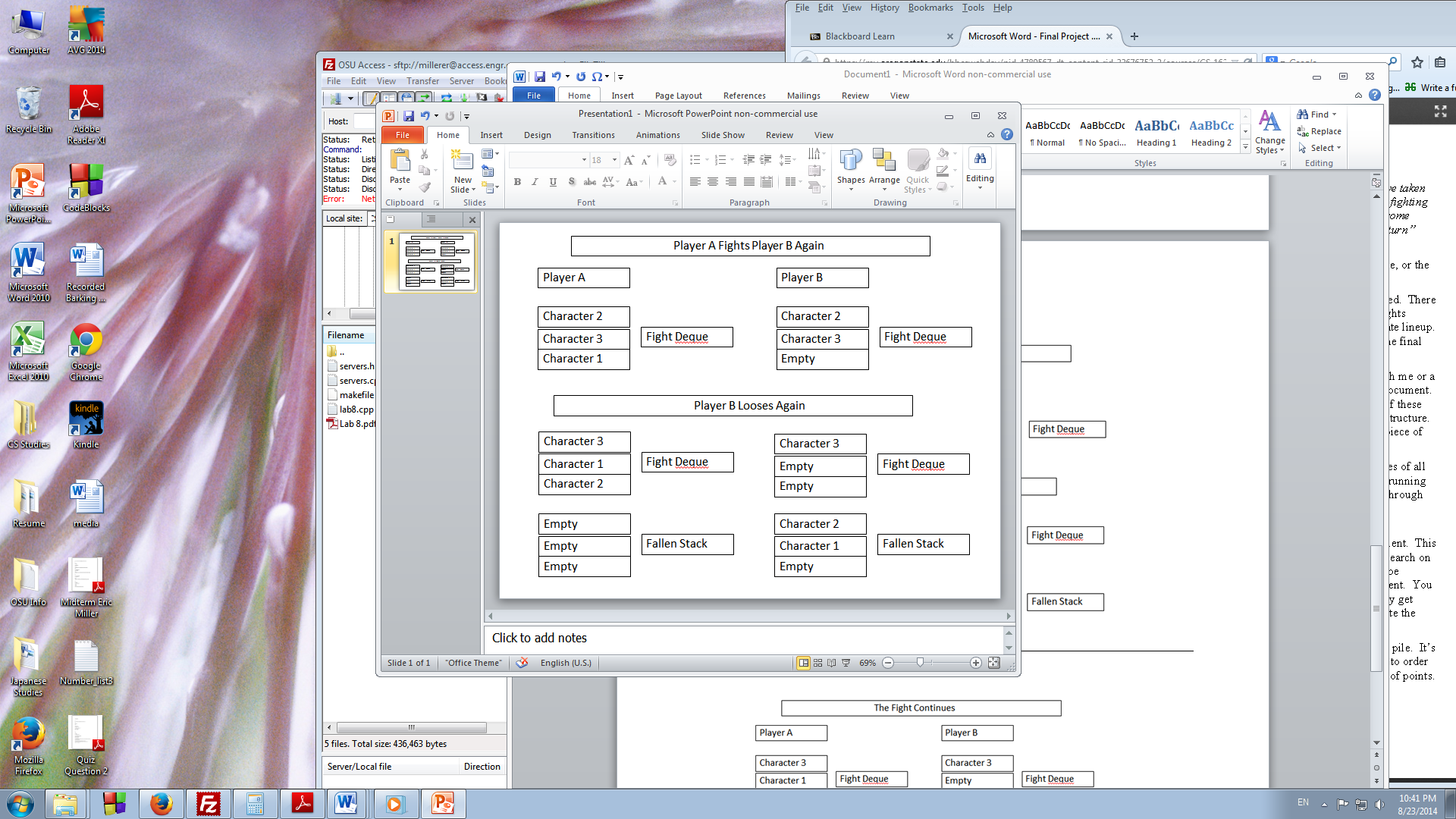
-More than one data structure type must be used.

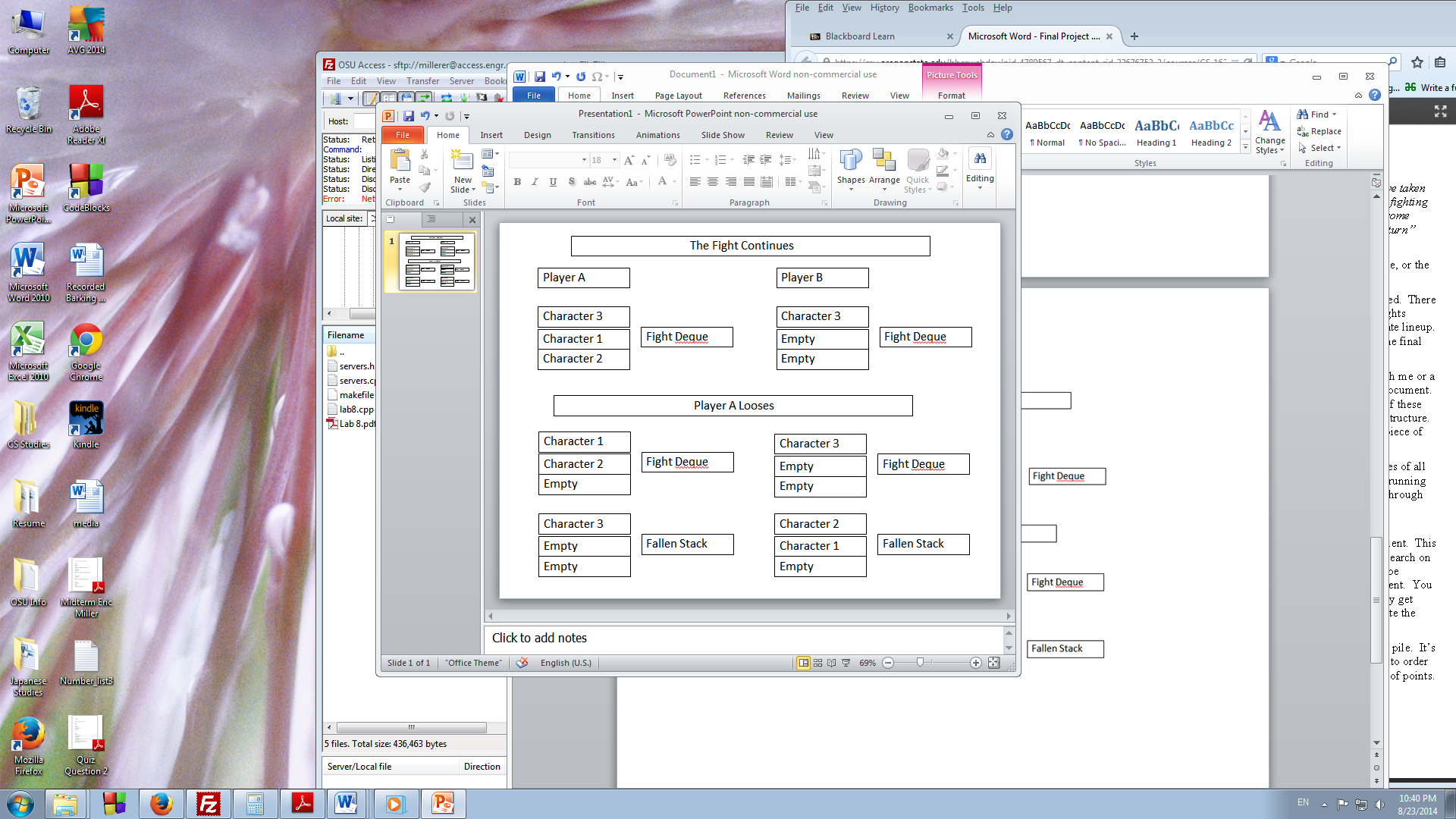
**General Program Outline:**

1. I will build this based on assignment 4, the samurai combat game that I made. Characters, abilities, and combat rules will remain the same as before.
2. The first things that I need to add are data structures to allow for a tournament format. Of the data structures so far, I have found STL deque containers to be pretty versatile, and for a second container type I will use a STL stack container.
3. To keep track of wins and losses, I’ll add two new int member variables to the samurai class so that I can calculate a rank.
4. I will also have an int variable called ‘victory’ to mark a character as the winner of a fight. This variable is set to ‘0’ whenever a character returns to the back of their list.
5. Teams will be lined up in deque containers. The front members will be placed into a battle function based on the one I made in assignment 4.
6. The looser of the battle will have +1 added to their losses variable and their victory variable set to 0.
7. Likewise the winner will have +1 added to their wins, and their ‘victory’ variable set to one.
8. The character with a victory variable equal to one (the winner) will be sent to the back of their deque container. The other character (looser) will be added to a stack type container for dead characters.
9. Winners will have their strength restored.
10. An example of these structures in action can be seen below:









1. Hopefully this gives a good idea of what I am hoping to do with containers in this program.
2. I should note that all containers will be of type samurai character class.
3. Also keep in mind that for each loss or win the character’s win/loss count is changed.
4. The game finishes once a player’s Fight Deque becomes empty.
5. To calculate rankings, losses are subtracts from total wins. The result will be the ranking for the character.
6. I expect there to be ties, so in that case a winner will be randomly chosen (50/50 draw).
7. The top three characters by win will then be output, including which team they were on. The team with the most ranked character wins will be considered the overall winner.

**Final Notes/Thoughts:**

Because we have been working on this combat program incrementally, I am hoping that incorporating the above systems will go smoothly. The fight mechanics are more or less already taken care of already, and it feels like more of an organizational problem than anything at this point. That is not to say that it won’t be challenging however, as I still don’t feel fully comfortable with lists and the other linked data structures we have been practicing over the past two weeks.