Rob Navarro
CS 162

Assignment 4

Design:

This program will use several classes from previous assignments/labs. They are:

- Die.cpp
- Queue.cpp
- Stack.cpp
- Battle.cpp

Along with these classes I will create an additional class called Tourney.cpp
I think it makes the most sense to start this program off by having a prompt within the main that creates a queue for player 1 and 2.

- To begin this process I will prompt with an initial question as to how many players each player will have.
- I will then display an option for the possible characters and the corresponding number for their selection. (I will create a separate function inside of main in order to not repeat code).
- Once this number is determined I can use a for loop to fill the queues for both players.
    - Inside of this for loop each player will be prompted what type of character they would like to add. I will also ask them to give each character a name. This will allow us to distinguish the difference between creatures of the same type.

Once both queues have been created I will pass both queues into the tourney class.

- Since the tourney class is new I will need to start from scratch.
- I will have two stacks within this class
    - The first stack will be for the losers
        - If a character loses a battle they will be placed in the loser stack.
    - I will also include a top3 stack. This will make it easy to display the top3 fighters.
        - This stack will not be updated until the end of the match
- I will also include a parameter for the player 1 and 2 creatures that will be fighting.
    - In order to fill these I will call the remove function for the queue class.
        - This function will return the first creature for each player in their queue as well as removing it from the queue (pop).
- Each creature will then be passed into the battle class and the fight function will be called.
- Inside of the battle class I will display what creature won the battle.
- I will make simple get functions that will return the winner and loser (They will be placed in a variable within the battle class following the battle).
- Once this is completed the tourney section will return the winner to the appropriate queue and the loser will be placed into the loser pile.
    - Before the winner is placed back in the queue a heath function will be called.
        - This health will restore a certain amount of health to the winner.
        - A 6 sided die will be used and if the roll is a positive number that amount will be restored to the players health.
- The tourney class will also have two variables that will keep track of total points for p1 and p2.
    - For winning a battle a team will be awarded 1 point.
    - At the end of the match the top 3 performers will also be awarded points
    - 1st: 10 pts 2nd: 7 pts 3rd: 6 pts
    - The total at the end of the match will determine which team wins the tourney.

- This whole sequence will be placed in a do while loop and will continue until one of the queues is empty.
- Once a queue is empty we will determine the overall winner by who won the last match. So if player1's creature ends up winning the decisive battle, they will be in first place. Then, 2nd and 3rd place will be determined by who fought most recently. This will only be an issue if more than one creature is alive on the winning team.
- Once the do loop exits display the tourney winner team and the top3 (include team).
    - Give the user an option to display the losers pile.

Test Plan

Since this program is using several classes that I have already made I will not have to test everything extensively. The Die and Battle classes have no changes to them and since they worked correctly in assignment 3, I am assuming that they are still working correctly (Though if they are not I will be able to spot it testing other classes).

The Stack and Queue classes are staying the same for the most part. The one change that will be made inside of each will be that the int parameter will be changed to Creature*. Once these changes are made I will need to retest the Stack and Queue to make sure that items are being added and removed correctly.

The Creature class will experience some changes as well. I will add a heal() virtual function that will allow each character to heal itself if they win a battle. Since each creature has a different maximum health I will have to override the function for each creature. Once this function is created I will do a couple of test battles and then heal call the healing function to see if the character heals correctly. In order to make sure it's working correctly I will display the health before healing and after healing. Also, I will display what number was rolled for healing.

The Tourney class will need several test since it is brand new. I will first start by making sure that Creatures are successfully being added to the proper queue. It is important that this part works correctly since the game will not work without it. Once the queues are set correctly I can remove the first creature from each queue and put them in a battle class. To start with I will just have one set of creatures fight and then display the winner and loser. This will allow me to make sure that the creatures fought correctly and that I am getting the expected results.

Once I confirm that the queues are working correctly and the creatures were able to fight successfully I will test my do/while loop for the tourney. This loop will continue until one of the queues is empty. After each round of the tournament I will display the results and see if they make sense. I will also test the 1st, 2nd and 3rd place display at this time and see if it is working correctly.

If I am able to successfully get through a tournament and display the results correctly I will assume that things are working correctly. From this point I will start testing several different cases of the tournament (different # of creatures, different creature combos).

Rob Navarro
CS 162

Testing Results

I found the initial testing to go quite well. I was able to get the Stack and Queue classes working with Creatures and I was also able to use the battle class with creatures from the Queue. The healing function for each character also worked as expected and did not require much testing.

I did run into several issues with the Tourney class. The main issue that I was running into was the destructor for the Queue and Stack class was not working as expected and was causing errors. I tried to figure out what was wrong with it but I couldn't, so I decided to use my main function for the tourney. By having all of the actions take place inside of the main I wasn't having to pass stack and queue classes, which is what I think was causing issues.

The main ended up being designed closely to how I planned to set up the tourney class. I would grab a creature from the player 1 and player 2 queue and then place them inside of the battle class. One change I did make was to check if player 1 or player 2 were alive inside of the main, instead of inside the battle class. This made it easier to move the creature to the correct stack or queue, and it made it pretty simple to display the winner of the round and current score for each team. I also decided that passing creatures back from the battle class was not necessary since I could access the results from the main. The winner and loser get functions were also not needed since I was storing the losers stack inside of main and I ended up not using the winners stack at all. Instead, I decided to place everything into the losers stack at the end of the tournament and then grab the top 3 creatures and place them into 1^{st}, 2^{nd} and 3^{rd} place variables.

I started small with the initial testing of the tournament and things went fairly well once I started to use my main. Pretty much all of the errors I ran into were syntax and required no design changes. Once I had a few tournaments run correctly I added more prompts to the program. This included allowing the player to say if they want to see the results after each round and if they wanted the losers stack to be displayed.

At this point my program was complete and I only needed to do some testing. I played the game several times with different characters and different line up sizes. I would track by pencil and paper which player won and which lost. This helped me make sure that the players were being placed correctly in the results and it also allowed me to tally the score correctly. I was also able to test a case where the team that had the first place finisher did not win since the other team had the second and third place finisher. The scoring system probably could have been more in depth but for this program I decided to just keep it simple and use what the scoring system listed above in my design.

Testing between weaker and stronger creatures also went as expected. I once tested a tournament with one team have all barbarians and one team having all reptile people. The reptile people won every round, which is what I expected. I also tested a team of all reptile people vs all blue men and the result was a long tournament. With each team fairly evenly matched no one team was able to pull away quickly.