Chermaine Cheang
Assignment 4 Reflection

I made a lot of changes to my original design for this assignment. In my original creature class, I have a variable “name” which is used to store creature types (Barbarian, Blue Men, etc), this variable is changed to “type” to store these values. The variable “name” is now used to store name of the creature given by the user. After these changes were made, the constructor for all creature class were modified to take in the name as parameter. The program from assignment 3 was then run and tested to check if it still compiles. Other changes I did is that I did not include any mutator functions in my Creature class for the newly added variable “teamNum” and “name” because the value these variables hold will not change throughout the program.

In my original createTeam function, the function received two parameters, one for team lineup (Queue) and one for number of fighters (int). However, the Queue team parameter passed into this function was later passed into createCreature function as an integer team number, this caused an error during compilation because I asked the program to convert a Queue object into an integer which is not a valid conversion. As a result, I added another integer parameter for createTeam(). This parameter will pass in the team number, which is either 1 or 2, which is then passed on to createCreature. Thereby, solving the error. Another error occurred in this function was that even though my Queue object has elements in it, but once the function terminated, I was not able to access these elements due to limitation of scope. To resolve this issue, I had to pass in my Queue object by reference, so that all changes made to the Queue is preserved until the main program terminates.

The original formula used to calculate recoverPoint in randRecover was incorrect. The way it should be calculated is that the number randomly generated will represent the percentage of how much strength will be recovered. If the system generated a 2, then 20% of the strength will be recovered from current strength. Hence, the formula should be temp *= 10 to change the number generated into percentage. Then apply the formula originally workout recoverPoint = getStrength() * (temp/100). However, even after the formula is revised, I were still unable to get the correct result. Upon further investigation, I realized that the problem arise from the division of temp with 100. Because both denominator and numerator are integers, the result of this division will always be 0 due to the fact that all decimal points are dropped. Therefore, I changed the denominator to a double (100.0) to resolve this problem.

However, I decided to change how to calculate the recovery points for a creature. The original calculation was based on creature’s current strength, this was changed to based on creature’s points lost from a combat. This is calculated from the difference between creature’s max strength and current strength. The reason I changed the way recovery point should be calculated was that I figured it makes more sense to recover from what you lost as compared to recover from what you have now.

In recover(), the original plan to use switch and case statements failed because switching a string condition causes compilation error because string is not a primitive type. Therefore, instead of using switch and case, I used if statements for each creature. Since I do not need to recover a creature that

already has max strength, I added condition to check if creature's current strength is already at its max. If creature's strength is at max, then I do not need to call randRecover() for this creature. The order in which recover() is called was also revised, instead of calling recover() first followed by resetAbility(), I called resetAbility() first before calling recover() because some times my winning creatures do not need to recover because they did not sustain any damage from previous combat. Original resetAbility() forgot to reset Harry Potter's strength to 10 if his current strength is greater than 10 due to his special ability. This was added to take this into account.

Initial design to add loser in main is changed because winner and loser from each combat is not determined in main. Hence, the function startTournament(), which facilitate the tournament and used to determine the winner and loser of each combat, was changed from returning no value (void) to returning a stack object. Thus, returning the loser stack to main at the end of the tournament.

To avoid going into an infinity loop when two Baba Yaga attacks each other, I added a variable soulCounter which indicate how many times Baba Yaga's soul ability can be used. At the point of instantiating a Baba Yaga's object, soulCounter is set to 20. This is the number of times Baba Yaga's can use its soul ability. Every time BabaYaga gains strength point from her attack, soulCounter will decrease by 1. When soulCounter is 0, that is when Baba Yaga's used up all her soul ability, she will not gain any more strength points from her attack. If Baba Yaga won the combat, soulCounter will be reset to 20 in resetAbility().

Another error occurred when there is only one fighter for in each team. Since there is only a total two fighters in the whole program, there will not be a third place finishers. Hence, calling stack class remove() will return "Stack is empty", and when asked to print out top three finishers, the system printed out garbage results and caused error. To resolve this issue, when the system tries to call remove() for an empty Stack, instead of printing out "Stack is empty!", I have it returned a NULL value, and when this value is passed to print123() to print out top three finishers, the system will check if the pointer is NULL before printing out the first, second and third winner. If the pointer is NULL, the system does not print out the third winner because there is no third place finishers. Also, since there is only two fighters, the system will not display the option to display list of final order because there is no losers in the loser stack.

Not all memory is freed at the end of the program. Tried to check if all pointer to creature is currently pointing to NULL, if not, delete the pointer to free the memory, but did not solve the problem. Upon further investigation, realized that I did not free the memory allocated for all creature even though I removed them from their respective Queue or Stack. Therefore, when displaying the final order, after I removed a creature from the loser Stack and printing out the relevant information, I had to delete this creature to free the memory allocated for this creature. Since this is in a while loop, all the creatures in the loser stack will be freed accordingly.

Some functions that I added in addition to what I have in my original design were sameTeamBattle() and displayLoser(). The first is used to facilitate combats between fighters from the same team. This is

to determine the top three finishers at the end of the tournament between two teams. The later is used to display an option to display the list of final order according to user's wish.

Also, when asking user to select creature for their team lineup, there was no clear indication as to which team the user is selecting creatures for, and no indication as to where this creature will be in the lineup. Therefore, the wording when prompting user for these information was changed to make the instruction clearer. I also added a validation point to check if user enters right input for name to make sure user a name for all the creature.

Testing

Test Plan	Input value	Drive function	Expected outcome	Test result
1	Prompt user for number of fighters for each team and validate user's input. *User enters 0*	validate(int)	Invalid input. Prompt user to enter number of fighters for each team again.	Invalid input. Number must be greater than 0. Enter number of fighters for each team *waiting for user input*
2	Prompt user for number of fighters for each team and validate user's input. *User enters any characters*	validate(int)	Invalid input. Prompt user to enter number of fighters for each team again.	Invalid input. Input must be NUMERIC. Enter number of fighters for each team *waiting for user input*
3	Prompt user for number of fighters for each team and validate user's input. *User enters 5*	validate(int)	Input is valid.	Input is valid. Program terminated successfully.
4	Display the list of creature types	creatureType()	List of creature is displayed and waiting for user choice	System displayed Select your creature: 1 - Medusa 2 - Barbarian 3 - Baba Yaga 4 - Blue Men 5 - Harry Potter

				6 - Quit *Waiting for user input*
5	User wants to create a Barbarian for team 1 and name this Barbarian b1	creatureType() createCreature() Cout some information of this barbarian	New Barbarian is created with name b1 and added to team1 queue.	System displayed: Select your creature: 1 - Medusa 2 - Barbarian 3 - Baba Yaga 4 - Blue Men 5 - Harry Potter 6 - Quit *User enter 2* System printed out: Barbarian b1 team 1 strength 12
6	User created fighters for team 1	createTeam() getNumFighter() remove()	Team 1 has a total of 5 fighters	getNumFighter returned 5 System displayed: Removed: Barbarian b1 from team 1 Removed: Barbarian b2 from team 1 Removed: Medusa m1 from team 1 Removed: Harry Potter hp1 from team 1 Removed Harry Potter hp2 from team 1 Successfully removed all fighters from team1
7	User created fighters for team 2	createTeam() getNumItem()	Team 2 has a total of 5 fighters	getNumFighter returned 5 System displayed: Removed: Barbarian b1 from team 2 Removed: Barbarian b2

				from team 2 Removed: Medusa m1 from team 2 Removed: Harry Potter hp1 from team 2 Removed Harry Potter hp2 from team 2 Successfully removed all fighters from team2
8	Remove the first fighters from both team	remove() getNumFighter()	First fighter from both teams remove from their team lineup Each team now has a total of 4 fighters	Remove first fighter from team 1: b1 Remove first fighter from team 2: b1 Successfully removed the first fighter from both teams Number of fighter in team 1: 4 Number of fighter in team 2: 4
9	Battle between the two first fighters	combatBetwFighters()	Each round of attack and defend was displayed on screen.	Each rounds of attack and defend was displayed on screen. Stop when one fighter has 0 strength left.
10	Determine who won the battle by checking if their alive is true	startTournament()	Display the fighter who won the battle	Screen displayed: Team 1: Medusa medu1 vs Team 2: Harry Potter harry1 After rounds of attacking, screen displayed Harry Potter harry1 from team 2 won the combat!

11	Test if recoverPoint is calculated correctly from randRecover()	randRecover()	recoverPoint is calculated correctly	System displayed: Random recover roll: 6 Current strength: 12 Recover point: 7 recoverPoint calculated correctly
12	Recover the winner	recover() randRecover()	The fighter strength is recovered by some percentage depending on the value returned from randRecover()	Fighter strength successfully recovered by recoverPoint returned from randRecover.
13	Fighter is Harry Potter. Reset Hogwarts ability	resetAbility()	Reset Harry Potter's death count to 0	Harry Potter's death count is set to 0 correctly.
14	Fighter is Baba Yaga Reset Soul ability	resetAbility()	Reset Baba Yaga's strength to 12 if Baba Yaga's current strength is more than 12	Baba Yaga's strength is set to 12 correctly.
15	Fighter is Blue Men Reset Mob ability	resetAbility()	Reset Blue Men's defPoint to 3 if Blue Men's current defPoint is less than 3	Blue Men's defPoint is set to 3 correctly.
16	Return fighter to team lineup	add()	Fighter returned to back of team lineup. Team now has 5 fighters	Fighter added back to team lineup successfully.
17	Add loser to loser stack	add()	Loser is added to the loser pile	Loser is added to loser pile successfully.
18	Run the tournament until one team has no available fighter left	startTournament()	The while loop in startTournament kept iterating until either team1 has no fighter left or team2 has no fighter left.	System successfully repeated until one team has no fighter left.

			Exit the loop	
19	Team 1 has no available fighter left. Team 2 has one fighter left	main() print123()	Team 2 won the tournament. Displayed the top three finishers.	System displayed: Team 2 won the tournament. 1st - baba1 (Baba Yaga) team 2 2nd - hp1 (Harry Potter) team 2 3rd - barb1 (Barbarian) team 1
20	Team 1 has no available figther. Team 2 has 2 fighters left	main() combatBetwFighters() print123()	Display attacks and defends points for each round of attack between the two remaining fighters from team2. Print out team2 won the tournament Displayed the top three finishers in which the top two will be from team2	System displayed all attacks and defends points successfully. System displayed: Team 2 won the tournament. 1st - baba1 (Baba Yaga) team 2 2nd - hp1 (Harry Potter) team 2 3rd - barb1 (Barbarian) team 1
21	Prompt user if they want to display list of final order	main()	After displaying the top three finishers, system asked if user want to display list of final order.	Do you wish to display a list of final order? *Waiting for user input*
22	User wants to display a list of final order	main() displayLoser() remove()	A list of final order is displayed.	System displayed: 4th - medusa1 (Medusa) from team 1