Mariam Ben-Neticha
CS 162: Intro to Computer Science II
11.8.2015
Assignment 3: Inheritance

Program Design:

<u>Problem(s) to be Solved:</u>

- Create an abstract Creature class
    - Solution: include at least one pure virtual function
- Create a subclass for the following creatures: Goblin, Barbarian, Reptile People, Blue Men, The Shadow.
    - include attack() and defense() functions in each class
    - make these functions inherited
    - include specific creature characteristics
        - Characteristics include:
            - number of attack dice
            - number of attack sides
            - number of defense dice
            - number of defense sides
                - Dice will be created within each subclass' attack() and defense() functions
            - armor value
            - strength points value
- Create a Combat Class
    - this class will:
        - calculate the damage
        - apply the damage
        - determine whether two of the same character are fighting each other (or not)
        - have combat between two characters
- Create a Die Class
    - this class will roll an instantiated die one time
- Apply the SPECIAL armor to the Shadow
- Apply the ACHILLES attack to the Goblin
- Determine if two Goblins are combatting and make the ACHILLES attack void.

<u>Inputs:</u>

- What the user wishes to do
    - Combat or Exit
- Which creatures user wants to fight each other

<u>Outputs:</u>

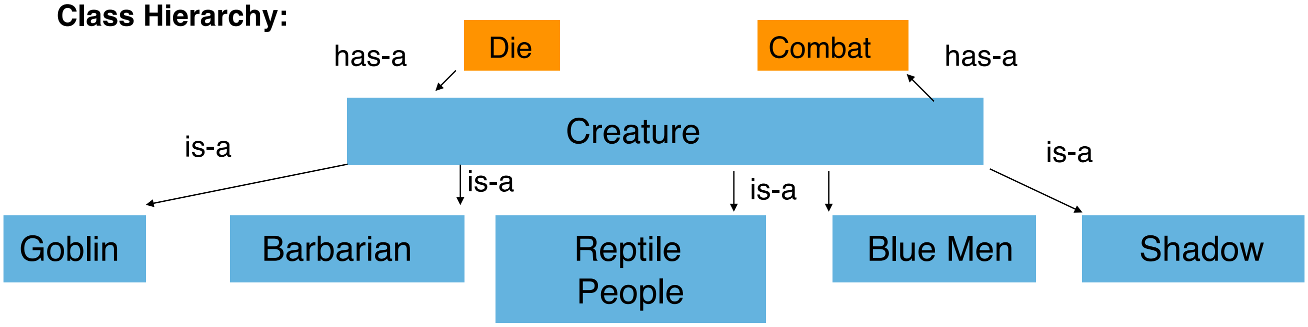
- Instructions for user to choose action or exit program (repeats after every action until user Exits)
- Instructions for user to choose two creatures to combat
- Displaying the round #, attacking creature_type, defending creature_type, attack roll, defense roll, damage, armor, damage dealt, and adjusted strength points.
- Displaying winner at end of combat.

<u>Algorithms:</u>

- Calculating the damage

    - Subtract the defense roll from the attack roll
        - damage = attack_roll - defense_roll
- Calculating the damage dealt
    - Subtract the armor from the damage
        - damage_dealt = damage - armor
- Calculating the adjusted strength points
    - Subtract the damage dealt from the initial strength points
        - strength_points = strength - damage_dealt

Class Hierarchy:



Test Plan:

1. Does the main menu display properly the first time vs the following times?
    a. Yes: The menu displays properly and the same menu displays following user actions/ inputs. The menu also functions properly.
    b. No: The menu causes issues or continuously loops without allowing any user input. Or the menu does not display all the options properly and is difficult to understand.
2. Does the creature menu display properly the first time vs the following times?
    a. Yes: The menu displays properly and the same menu displays following user actions/ inputs. The menu also functions properly.
    b. No: The menu causes issues or continuously loops without allowing any user input. Or the menu does not display all the options properly and is difficult to understand.
3. Is the User-Input received properly? And does the Combat reflect the changes?
    c. Yes: The input is read properly into the appropriate members. All actions reflect the appropriate changes.
        - e.g. - choosing to combat leads user to next menu to select creatures for combat.
        - e.g. - choosing the creatures leads to programming calling the combat of creatures chosen.
    d. No: The input is not read properly. No changes are reflected in the Shopping List or the changes that occur do not make sense.
4. Are the rounds displayed and updated after every action?
    e. Yes: Adjustments to a creature's strength points are reflected in the displayed information for each round. A creature's strength points either decrease or stay the same.
    f. No: The round displays no changes expected to be seen to the creature(s) strength points.
5. Do the values for attack_roll, defense_roll, damage, armor, damage_dealt, and strength_points make sense?
    g. Yes: The initial values and adjusted values make sense.

   h. No: The initial values and adjusted values do not make sense.

6. Does the Shadow creature exhibit its SPECIAL armor? Is it exhibited properly?
   i. Yes: The Shadow creature sometimes exhibits SPECIAL armor. It is exhibited properly and Shadow’s strength points remain the same for that round.
   j. No: The Shadow creature never exhibits SPECIAL armor. Or it is exhibited improperly and Shadow’s strength points are decreased for that round when they should not be.
7. Does the Goblin creature exhibit its ACHILLES attack? Is it exhibited properly?
   k. Yes: The Goblin creature sometimes exhibits ACHILLES attack. It is exhibited properly and Goblin’s foe’s attacks are halved for remainder of the fight/combat.
   l. No: The Goblin creature never exhibits ACHILLES attack. Or it is exhibited improperly and Goblin’s foe’s attacks are not halved for remainder of the fight/combat.
8. Is the proper creature declared a winner at the end of combat?
   m. Yes: The winning creature is declared the winner.
   n. No: The creature with zero strength points is declared winner.

Combat Test Plan:

1. Does the Goblin vs Goblin combat implement correctly?
   a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. No ACHILLES attack is implemented here.
   b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or ACHILLES attack is implemented here.
2. Does the Goblin vs Barbarian combat implement correctly?
   a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. ACHILLES attack may be implemented here.
   b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or no ACHILLES attack can implemented here.
3. Does the Goblin vs Reptile People combat implement correctly?
   a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. ACHILLES attack may be implemented here.
   b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or no ACHILLES attack can implemented here.
4. Does the Goblin vs Blue Men combat implement correctly?
   a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. ACHILLES attack may be implemented here.
   b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or no ACHILLES attack can implemented here.
5. Does the Goblin vs Shadow combat implement correctly?
   a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. ACHILLES attack may be implemented here. SPECIAL armor is implemented here 50% of the time.
   b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or no ACHILLES attack can implemented here. Or SPECIAL armor is never implemented.
6. Does the Barbarian vs Barbarian combat implement correctly?
   a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed.
   b. No: The combat improperly calculates the change in strength points and decreases strength points as needed.

7. Does the Barbarian vs Reptile People combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed.
8. Does the Barbarian vs Blue Men combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed.
9. Does the Barbarian vs Shadow combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. SPECIAL armor is implemented here 50% of the time.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or SPECIAL armor is never implemented.
10. Does the Reptile People vs Reptile People combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed.
11. Does the Reptile People vs Blue Men combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed.
12. Does the Reptile People vs Shadow combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. SPECIAL armor is implemented here 50% of the time.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or SPECIAL armor is never implemented.
13. Does the Blue Men vs Blue Men combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed.
14. Does the Blue Men vs Shadow combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. SPECIAL armor is implemented here 50% of the time.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or SPECIAL armor is never implemented.
15. Does the Shadow vs Shadow combat implement correctly?
    a. Yes: The combat properly calculates the change in strength points and decreases strength points as needed. SPECIAL armor is implemented here 50% of the time.
    b. No: The combat improperly calculates the change in strength points and decreases strength points as needed. Or SPECIAL armor is never implemented.

Test Results:

Test Number	Test Case	Input Values	Expected Outcome	Observed Outcome
1. Main Menu Display	Input is an option	Input = 1	Prompt user to combat.	Prompt user to combat.
		Input = 2	Exit program with “Program End” message.	Exit program with “Program End” message.
	Input is not an option	Input != (1 ll 2)	Prompt user for proper input.	Prompt user for proper input.
2. Creature Menu Display	Input is an option	Input = 1	1st/2nd creature = Goblin	1st/2nd creature = Goblin
		Input = 2	1st/2nd creature = Barbarian	1st/2nd creature = Barbarian
		Input = 3	1st/2nd creature = Reptile	1st/2nd creature = Reptile
		Input = 4	1st/2nd creature = Blue Men	1st/2nd creature = Blue Men
		Input = 5	1st/2nd creature = Shadow	1st/2nd creature = Shadow
	Input is not an option	Input != (1 ll 2 ll 3 ll 4 ll 5)	Prompt user for input.	Prompt user for input.
3. User-Input Read-In	Input is user-input	Input = values between (main menu)1-2 or (creature menu)1-5	main menu value is stored in menu; combat between creatures chosen is instantiated.	main menu value is stored in menu; combat between creatures chosen is instantiated.
4. Rounds Displayed and Updated Properly	N/A	N/A	Round number appears followed by attacking creature_type, defending creature_type, attack roll, defense roll, damage, armor, damage dealt, and adjusted strength points	Round number appears followed by attacking creature_type, defending creature_type, attack roll, defense roll, damage, armor, damage dealt, and adjusted strength points

Test Number	Test Case	Input Values	Expected Outcome	Observed Outcome
5. Set Values vs Adjusted Values and Algorithmic Calculations	N/A	N/A	Values for attack_roll, defense_roll, damage, armor, damage_dealt, and strength_points make sense based on mathematical calculations and operations.	Values for attack_roll, defense_roll, damage, armor, damage_dealt, and strength_points make sense based on mathematical calculations and operations.
6. Shadow exhibits SPECIAL armor	N/A	N/A	Shadow exhibits SPECIAL armor 50% of the time; Shadow's strength points remain the same for that round; no damage incurred.	Shadow exhibits SPECIAL armor 50% of the time; Shadow's strength points remain the same for that round; no damage incurred.
7. Goblin exhibits ACHILLES attack	N/A	N/A	Goblin exhibits ACHILLES attack when a 12 attack is rolled. Opponent's attack_rolls are halved for remainder of combat.	Goblin exhibits ACHILLES attack when a 12 attack is rolled. Opponent's attack_rolls are halved for remainder of combat.
8. Winner declared correctly	N/A	N/A	When one creature's strength points == 0, opponent creature is declared winner.	When one creature's strength points == 0, opponent creature is declared winner.

Test Number	Test Case	Input Values	Expected Outcome	Observed Outcome
1. Goblin vs Goblin	Input is an option	Input = 1 && = 1	When combat conducted, no ACHILLES attack; combat functions properly.	When combat conducted, no ACHILLES attack; combat functions properly.
2. Goblin vs Barbarian	Input is an option	Input = 1 && = 2	When combat conducted, 12 rolls ACHILLES attack; combat functions properly.	When combat conducted, 12 rolls ACHILLES attack; combat functions properly.

Test Number	Test Case	Input Values	Expected Outcome	Observed Outcome
3. Goblin vs Reptile People	Input is an option	Input = 1 && = 3	When combat conducted, 12 rolls ACHILLES attack; combat functions properly.	When combat conducted, 12 rolls ACHILLES attack; combat functions properly.
4. Goblin vs Blue Men	Input is an option	Input = 1 && = 4	When combat conducted, 12 rolls ACHILLES attack; combat functions properly.	When combat conducted, 12 rolls ACHILLES attack; combat functions properly.
5. Goblin vs Shadow	Input is an option	Input = 1 && = 5	When combat conducted, 12 rolls ACHILLES attack; SPECIAL armor 50% of time; combat functions properly.	When combat conducted, 12 rolls ACHILLES attack; SPECIAL armor 50% of time; combat functions properly.
6. Barbarian vs Barbarian	Input is an option	Input = 2 && = 2	When combat conducted, combat functions properly.	When combat conducted, combat functions properly.
7. Barbarian vs Reptile People	Input is an option	Input = 2 && = 3	When combat conducted, combat functions properly.	When combat conducted, combat functions properly.
8. Barbarian vs Blue Men	Input is an option	Input = 2 && = 4	When combat conducted, combat functions properly.	When combat conducted, combat functions properly.
9. Barbarian vs The Shadow	Input is an option	Input = 2 && = 5	When combat conducted, SPECIAL armor 50% of time; combat functions properly.	When combat conducted, SPECIAL armor 50% of time; combat functions properly.
10. Reptile People vs Reptile People	Input is an option	Input = 3 && = 3	When combat conducted, combat functions properly.	When combat conducted, combat functions properly.
Reptile People vs Blue Men	Input is an option	Input = 3 && = 4	When combat conducted, combat functions properly.	When combat conducted, combat functions properly.
Reptile People vs The Shadow	Input is an option	Input = 3 && = 5	When combat conducted, SPECIAL armor 50% of time; combat functions properly.	When combat conducted, SPECIAL armor 50% of time; combat functions properly.

Test Number	Test Case	Input Values	Expected Outcome	Observed Outcome
Blue Men vs Blue Men	Input is an option	Input = 4 && = 4	When combat conducted, combat functions properly.	When combat conducted, combat functions properly.
Blue Men vs The Shadow	Input is an option	Input = 4 && = 5	When combat conducted, SPECIAL armor 50% of time; combat functions properly.	When combat conducted, SPECIAL armor 50% of time; combat functions properly.
The Shadow vs The Shadow	Input is an option	Input = 5 && = 5	When combat conducted, SPECIAL armor 50% of time; combat functions properly.	When combat conducted, SPECIAL armor 50% of time; combat functions properly.

Reflection:

This program was a much more challenging program to create due to it's complexity in terms of both the amount of written code necessary and the functions that were necessary to accomplish the assignment's requirements. The amount of written code necessary in creation of this program made debugging the program much more time-consuming and more difficult than the shorter programs we have written for this course. I found that debugging was where I spent the most of my time. Implementing the Achilles and Special characteristics, along with the other requirements of the assignment were not nearly as difficult as debugging. After some thought, I would quickly be able to find a possible solution for coding the above characteristics or requirements. Most of the time, my initial solution worked and simple tweaking was necessary to correct any compile-time errors. An example of this is using a pointer versus a reference vs a value. Since the code within this program is so vast, I made some mistakes in correspondence to how I was calling my objects. These were quickly fixed, however.

First, I created my Creature class. I quickly determined that this class needed to have the attack() and defend() functions as pure virtual functions. This made the class abstract. I knew that I would need to get and/or set certain values and so created the corresponding functions in order to allow me to do so. The Creature class was also were the derived classes would instantiate their own objects. Therefore, it contained private data members relating to the six components that made up each creature: number of attack dice, number of attack sides, number of defense dice, number of defense sides, armor value, strength points value.

Next, I created my Die class, similar to the Die class from a previous lab, and all of the derived classes and defined their attack() and defense() functions within their class. I also input the creature's characteristic components that the creature would access when the derived class object was instantiated.

Following this, I created my Combat class. This class handled all the calculations for damage, strength point, and setting the strength point of the creature to it's new value. It also determined whether the object in combat was fighting one of it's own type (i.e. Goblin vs Goblin or Goblin vs Other). Finally, it set the two creatures in combat and looped through the combat until one creature's strength points decreased to zero.

I then returned to my Goblin class and implemented the Achilles attack within the attack() function. I also edited my combat() function so that the Goblin's opponent's future attacks would be halved until the combat was over. Following this, I returned to my Shadow class and added a redefined get_armor() function. Within this function, I implemented the Shadow's Special armor. I created a random number generator to have this ability appear 50% of the time. I then went to the calc_damage() function in my Combat class to ensure that the Shadow creature would not experience any damage if the Special armor was implemented during that round.

After getting my program working properly, I added two menus. The first menu allowed the user to choose between Combat or Exiting the program. The second menu let the user pick which two creatures should combat and instantiated combat between them. At the end of the combat, the user is returned to the first menu.

I tested my program regularly along the way, stopping to make changes to some code to better implement my design or to fix some mistakes. At this point, my program was able to pass all of the tests within my Test Plan.