Name: Brendan Corazzin
Assignment: Project 3
Date: 5/4/2017

Reflection:

While writing the program, I had to make a couple of changes that deviated from the program design. These changes, I believe, made the program code simpler and easier to understand. I will outline these changes below.

- For the Vampire class I did not write charm as a function, but just included it in the defend function. Writing a function for such a simple procedure was unnecessary.

- For the BlueMen class I did not write mob function, I just included it in the defend function. I did this for the same reason as above.

- For the Medusa class I did the same as above and included the functionality of the glare in the attack function.

- For the HarryPotter class I did the same as above and included the Hogwarts code in the defend function.

- The class diagram was also inaccurate. Since the creature class was just a base class, I had to include the attack, defend, and getType functions in each of the derived classes. I also added functions and did not include functions that turned out to be unneeded. I didn't need the getArmor function because I only needed access to the armor variable from within the class in which it was a data member. I added a setData function which was used to set the data members of each class. I also had to add a getAttacker and setAttacker function to get and set a variable that was used to track which player should attack and which should defend.

- For the Die class, I added a function to get the number of dice being used by the character. I also renamed the updateDieSides.

- For the Game and Creature class, I added a destructor to delete the memory that was dynamically allocated.

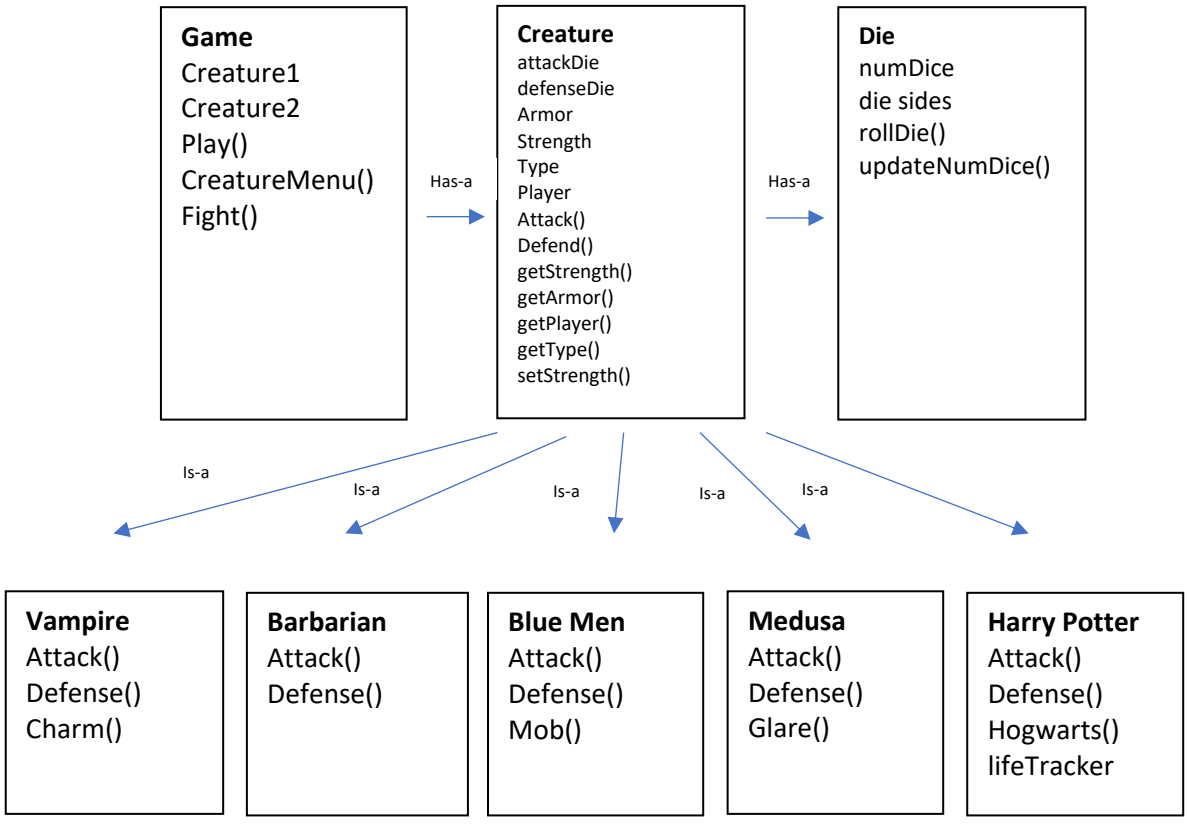
Overall, I feel like I am beginning to grasp Object Oriented design. I didn't struggle to come up with a design for this program – it happened much more intuitively. I still need plenty of practice and there are things I would like to change in this program (if I had more time to work on it). One thing I question is the use of the setData function. Initially, I tied to set all of the derived classes data members within a constructor function for each class, but I couldn't get the code to compile. So, I used the setData function and that worked, although I feel like the data should be set within in the class rather than outside of the class because the initial value for those data (strength, armor, die sides, number of die) never change. Other than that, I'm happy with the code I wrote for this assignment.

Program Design

- The problem to be solved: create a game with five different characters. Each character has different attributes for attach, defense, armor, and strength. The user will be able to select two characters to fight one another. Each fighter will take turns attacking until one of the character strength points equals 0 or negative. After the characters fight, the user will select if they would like to play another round, or quit the game.
- Identify the inputs:
    - Select from menu of players (input will be 1 – 5, each number corresponds with a character)
    - Select from menu to play again, or exit (input will be 1 or 2, each number corresponds to menu option)
- Desired output
    - Welcome user to the game
    - Prompt user to select two characters to fight
    - Display info for each round: round number, attack, defense, armor, final strength point, special characteristic for the character
    - Display winner
    - Prompt user to select from game menu
    - Error messages for invalid inputs

Class Diagram

There will be a game class that has-a creature class(s). There will be 5 types of characters and each of these is-a creature. There will also be a die class. Each creature has-a attack die and a defense die



High-Level Pseudocode

**Main()**
Print welcome message
Call mainMenu()
If play game
    Create game object
    Call gameObject.play()
Else
    Quit program

**Play()**
Print play message
Call fight()
Prompt user to play again
If yes
    Call play()
Else
    Quit Program

**Fight()**
Print fight message
Call creatureMenu()
Create creature objects  and var roundNum to track rounds
Randomly select fighter and defender
Print selection results
While fighters getStrength() > 0
    ++roundNum
    Print roundNum and Prompt user to press key to for x to attack x
    Call attack()
    Call defend()
Print winner

**Attack() : returns int**
rollDie()
call special feature function
print results of attack
return roll

**Defense(): takes int from attack()**
rollDie()
call special feature function
update creature data
print results of defense

**rollDie()**
generate random number for each die and add total roll
return total roll