CS 162: Assignment 3

Pseudocode:

Creature.hpp

```
    Include iostream
    Include time.h
    Create class creature
        Private
            Int damage is related to the attack //Attack returns damage done.
            Int outcome is related to the defense after damage is calculated
        Public
            Constructor creature that defines damage and outcome
            Have a virtual int attack() //returns damage
            Have a virtual int defense(damage)   //Takes in damage, reduces it
                                                 //Armor is added to defense roll
                                                 //returns outcome
            Have a virtual int strength(outcome) //attack output is damage
                                                 //Takes damage, reduces HP
                                                 //returns total health
```

Vampire.hpp

```
    Include creature.hpp
    Class vampire : public creature
        Private

        Public
            Have a virtual int attack() //returns damage
            Have a virtual int defense(damage) //returns outcome
            Have a virtual int strength(outcome) //returns health
```

Vampire.cpp

```
    Include creature.hpp
    Call creature constructor creature::creature(int damage1, int outcome1)
        Damage = damage1;
        Outcome = outcome1;


    Int vampire::attack()
    {
        Uses rand() to roll a 1d12, setting it to damage
        Returns damage which will be an int
    }
    Int vampire::defense(damage)
    {
        //Charm
        Uses rand() to roll a 1d2
        If roll is 1
            Display message that the vampire charmed the opponent to not attack.
            Outcome is 0
            Return outcome
        If roll is 2
            Uses rand() to roll a 1d6 and add + 3 for armor //enter seed for rand to be
different?
            Subtracts roll from damage to create outcome
            If outcome is < 0
                Outcome is 0
                Return outcome
            Else
                Return outcome
    }

    Int vampire::strength(outcome)
    {
        Int health;
        Health = 8 for vampire
        Outcome is subtracted from health
        If health is <= 0
            Display message saying that vampire has died
        Return health

    }
```

Medusa.hpp

```
    Include creature.hpp
```

```
Class medusa : public creature
    Private

    Public
        Have a virtual int attack() //returns damage
        Have a virtual int defense(damage) //returns outcome
        Have a virtual int strength(outcome) //returns health
```

Medusa.cpp

```
    Include creature.hpp
        Call creature constructor creature::creature(int damage1, int outcome1)
        {
            Damage = damage1;
            Outcome = outcome1;
        }

        Int medusa::attack()
        {
            Use rand() to roll 2 1d6 die, changing the seed to get 2 different numbers
            Add the two rolls together as damage
            If damage is = to 12
                Display message that medusa has turned her opponent to stone
                Set damage to 1000
                Return damage as an int
            else
                Return damage as an int
        }

        Int medusa::defense(damage)
        {
            Int defend
            Use rand() to roll a 1d6 die, set the number as defend
            Add 3 to defend to make up for armor value
            Subtract defend from damage to make outcome
            Return outcome
        }



        Int medusa::strength(outcome)
        {
            Int health;
            Set health = to 8 for medusa
            Subtract outcome from health
            If health = 0
                Display message stating that medusa has been slain
            Return health
        }
```

Bubba.hpp

```
    Include creature.hpp
    Class bubba : public creature
        Private

        Public
```

```
                Have a virtual int attack() //returns damage
                Have a virtual int defense(damage) //returns outcome
                Have a virtual int strength(outcome) //returns health
Bubba.cpp
    Include creature.hpp
    Call creature constructor creature::creature(int damage1, int outcome1)
    {
        Damage = damage1;
        Outcome = outcome1;
    }

    Int bubba::attack()
    {
        Use rand() to roll 2 1d6 die, changing the seed to get 2 different numbers
        Add the two rolls together as damage
        Return damage
    }

    Int bubba::defense(damage)
    {
        Int defense
        Use rand() to roll 2 1d6 die, changing the seed to get 2 different numbers
        Add the two rolls together as defense
        Subtract defense from damage and set as outcome
        If outcome is < 0
            Outcome is 0
            Return outcome
        Else
            Return outcome
    }

    Int bubba::Strength(outcome)
    {
        Int health
        Set health to be = to 12
        Subtract outcome from health
        If health <= 0
            Display message that bubba has died
        Return health;
    }
```

Main.cpp

```
    Include Creature.hpp
```

```
Include Vampire.hpp
Include Medusa.hpp
Include Bubba.hpp
srand(time(NULL)); //Maybe put in each function when you call rand?

Vampire *v1 = new vampire;
Creature *c1 = v1;
Vampire *v2 = new vampire; //for when vampire has to fight vampire
Creature *c2 = v2

Medusa *m1 = new medusa;
Creature *c3 = m1;
Medusa *m2 = new medusa; //for when medusa has to fight medusa
Creature *c4 = m2;

Bubba *b1 = new bubba;
Creature *c5 = b1;
Bubba *b2 = new bubba; //for when bubba has to fight bubba
Creature *c6 = b2;

Int choice = 0;
Do
{
        Display message stating this fisticuffs match between some of the world's most
fiercest creatures.
        Display menu as [1] Roster. [2] Fight. [3] Exit
        Cin >> choice

FF /*   If choice == 1
                Do
                {
                        Display combatant 1 as vampire, combatant 2 as medusa,
                                 combatant 3 as bubba, 4 to return to first menu
                        Display choose a monster to see it's details
                        Cin choice
                        If choice == 1
                                Display vampire stats. Vampire faster than medusa
                        If choice == 2
                                Display medusa stats. Medusa faster than bubba
                        If choice == 3
                                Display bubba stats. Bubba slowest
                        If choice > 4 || choice < 0
                                Display that that number wasn't an option.
```

```
                                        Choice = 4;
For Fun             */          } While (choice =! 4)




          If choice == 2
                    Int f1
                    Int f2
                    Int tick = 2;
                    Display message to choose your fighters. [1] Vampire [2] Medusa [3]
                              Bubba
                    Display message “Fighter 1: “
                    Cin << f1
                    Display message “Fighter 2: “
                    Cin << f2

                    If f1 == 1 && f2 == 1
                    ((Vampire vs Vampire
                    {
                              Do
                              {
                                        Display message vampire1 made his attack
                                        c1->attack();
                                        Display message vampire2 tried to defend
                                        c2->defense(damage); //takes in return value of attack
                                        c2->strength(outcome); //takes in return value of defense
                                                                      //returns health
                                        If health (return value of strength) <= 0
                                                  Break;

                                        Display message for vampire2 made his attack
                                        c2->attack();
                                        Display message vampire1 tried to defend
                                        c1->defense(damage); //takes in return value of attack
                                        c1->strength(outcome); //takes in return value of defense
                                                                      //returns health
                                        If health (return value of strength) <= 0
                                                  Tick == 1
                              } while(tick =! 1)
                    }


                    Else If f1 == 1 && f2 == 2
```

```
((Vampire vs Medusa))
{
        Do
        {
                Display message vampire1 made his attack
                c1->attack();
                Display message medusa1 tried to defend
                c3->defense(damage); //takes in return value of attack
                c3->strength(outcome); //takes in return value of defense
                                          //returns health
                If health (return value of strength) <= 0
                        Break;

                Display message for medusa1 made his attack
                c3->attack();
                Display message vampire1 tried to defend
                c1->defense(damage); //takes in return value of attack
                c1->strength(outcome); //takes in return value of defense
                                          //returns health
                If health (return value of strength) <= 0
                        Tick == 1
        } while(tick =! 1)
}

Else If f1 == 1 && f2 == 3
((Vampire vs Bubba))
{
        Do
        {
                Display message vampire1 made his attack
                c1->attack();
                Display message bubba1 tried to defend
                c5->defense(damage); //takes in return value of attack
                c5->strength(outcome); //takes in return value of defense
                                          //returns health
                If health (return value of strength) <= 0
                        Break;

                Display message for bubba1 made his attack
                c5->attack();
                Display message vampire1 tried to defend
                c1->defense(damage); //takes in return value of attack
                c1->strength(outcome); //takes in return value of defense
```

```
                                        //returns health
                If health (return value of strength) <= 0
                        Tick == 1
        } while(tick =! 1)
}
Else If f1 == 2 && f2 == 2
((Medusa vs Medusa))
{
        Do
        {
                Display message medusa1 made his attack
                c3->attack();
                Display message medusa2 tried to defend
                c4->defense(damage); //takes in return value of attack
                c4->strength(outcome); //takes in return value of defense
                                        //returns health
                If health (return value of strength) <= 0
                        Break;

                Display message for medusa2 made his attack
                c4->attack();
                Display message medusa1 tried to defend
                c3->defense(damage); //takes in return value of attack
                c3->strength(outcome); //takes in return value of defense
                                        //returns health
                If health (return value of strength) <= 0
                        Tick == 1
        } while(tick =! 1)
}

Else If f1 == 2 && f2 == 3
((Medusa vs Bubba))
{
        Do
        {
                Display message medusa1 made his attack
                c3->attack();
                Display message bubba1 tried to defend
                c5->defense(damage); //takes in return value of attack
                c5->strength(outcome); //takes in return value of defense
                                        //returns health
                If health (return value of strength) <= 0
                        Break;
```

```
                    Display message for bubba1  made his attack
                    c5->attack();
                    Display message medusa1 tried to defend
                    c3->defense(damage); //takes in return value of attack
                    c3->strength(outcome); //takes in return value of defense
                                            //returns health
                    If health (return value of strength) <= 0
                        Tick == 1
                } while(tick =! 1)
            }
            Else If f1 == 3 && f2 == 3
            ((Bubba vs Bubba))
            {
                Do
                {
                    Display message bubba1 made his attack
                    c5->attack();
                    Display message bubba2 tried to defend
                    c6->defense(damage); //takes in return value of attack
                    c6->strength(outcome); //takes in return value of defense
                                            //returns health
                    If health (return value of strength) <= 0
                        Break;

                    Display message for bubba2  made his attack
                    c6->attack();
                    Display message bubba1 tried to defend
                    c5->defense(damage); //takes in return value of attack
                    c5->strength(outcome); //takes in return value of defense
                                            //returns health
                    If health (return value of strength) <= 0
                        Tick == 1
                } while(tick =! 1)
            }
            Else
            {
                Cout << “Those were not choices. Returning to menu” << endl;
                Choice = 3;
            }
}While (choice =! 3)
Return 0;
```

//Then to call attacks you'd use c1->attack() to run vampire's attack(), c1->defense(attackInput) to run Vampire's defence where attackInput is the output of the other combatants attack, and c1->strength(defenseOutput) to run Vampire's health where defenseOutput is the output of Vampire's defence.

Makefile

```
test: Creature.cpp Vampire.cpp Medusa.cpp Bubba.cpp Main.cpp Vampire.hpp Medusa.hpp Bubba.hpp
	g++ Creature.cpp Vampire.cpp Medusa.cpp Bubba.cpp Main.cpp -o test
```

Reflection:

This assignment was one that I had a lot of fun with when creating. It is hands down the most fun I've had programing because I felt like I could combine my computer science skills with my creativity to make a well versed battle simulation game. On the surface, my goal was to make it an enjoyable and interesting experience for the player. I chose to create a set of displayed commands which described when and how the creatures were attacking and how the other creature reacted to the attack until one of them dies. I kept most of the coding behind the scenes to make the most enjoyable experience as possible. I also decided to add a Roster because I figured knowing the stats and tricks of the monsters would be good to know more about them before choosing to play as them. I decided to spend a lot more time than I usually do on my pseudocode this time around and I am very glad for it because when I went to code my program, it took me very little time to actually do. The only real problem I came up against was when I got an error I couldn't solve but through help of one of my fellow classmates who figured out how to solve the error I was able to figure it out. The error itself turned out to be very minor as I didn't put in an = 0 in the Creature cpp file.

Test Plan:

Input	Output
Vampire [1] vs Vampire [1]	Vampire and Vampire fight.
Vampires [1] vs Medusa [2]	Vampire and Medusa fight.

[7] vs [8]	ERROR:: Number was chosen that is out of the spectrum. Please choose 1, 2, or 3.
[4] vs Bubba [3]	ERROR:: Number was chosen that is out of the spectrum. Please choose 1, 2, or 3.
[a] vs [b]	ERROR:: Choose an int 1, 2, or 3
Medusa [2] vs [ ]	ERROR:: No choice detected for 2nd choice
[ ] vs Vampire [1]	ERROR:: No choice detected for 1st choice