Joel Huffman
02/15/18
Project 3 Reflection

Design:
Character class:
- Needs to be abstract(pure virtual functions)
- Data members:
    - Int strength
    - Int armor
    - String name
    - Int attack die size
    - Int attack die number
    - Int defense die size
    - Int defense die number
- Member functions:
    - Int virtual attack()
        - Rolls dice equal to their attack dice and returns that value
    - Int virtual defense(int incomingAttack) = 0;
    - Int virtual defenseDieRoll() = 0;
    - Int roll(int die)
        - Generates and returns an int from 1-die

Barbarian class:
- Derived class from Character
- Member functions:
    - Barbarian() initialize the following:
        - Int strength
        - Int Armor
        - String Name
        - Int attack die size
        - Int attack die num
        - Int defense die size
        - Int defence die num
    - Int attack()
        - Rolls dice equal to their attack dice and returns that value
    - Int defense(int incomingAttack)
        - Takes an incoming attack value and subtracts the defense roll and armor to get the inflicted damage.
        - Reduces strength by inflicted damage
        - Returns the defense roll
    - Int defenseDieRoll()
        - Rolls the characters defense dice and returns the result

Vampire class:
- Derived class from Character
- Member functions:

- Vampire() initialize the following:
    - Int strength
    - Int Armor
    - String Name
    - Int attack die size
    - Int attack die num
    - Int defense die size
    - Int defence die num
- Int attack()
    - Rolls dice equal to their attack dice and returns that value
- Int defense(int incomingAttack)
    - Takes an incoming attack value and subtracts the defense roll and armor to get the inflicted damage.
    - Reduces strength by inflicted damage
    - Returns the defense roll
- Int defenseDieRoll()
    - Calls charmEnemy(), if true defense is equal to 999
    - Otherwise, rolls the characters defense dice and sets defense to the result
    - Returns defense value
- Bool charmEnemy()
    - Has a 50% chance to return true

Bue men class:

- Derived class from Character
- Member functions:
    - BlueMen() initialize the following:
        - Int strength
        - Int Armor
        - String Name
        - Int attack die size
        - Int attack die num
        - Int defense die size
        - Int defence die num
    - Int attack()
        - Rolls dice equal to their attack dice and returns that value
    - Int defense(int incomingAttack)
        - Takes an incoming attack value and subtracts the defense roll and armor to get the inflicted damage.
        - Reduces strength by inflicted damage
        - Call mob()
        - Returns the defense roll
    - Int defenseDieRoll()

        - Rolls the characters defense dice and returns the result
    - Void mob()
        - If strength > 8, set defensiveDieNumber to 3
        - Else if strength > 4, set defensiveDieNumber to 2
        - Else, set defensiveDieNumber to 1

Medusa class:
- Derived class from Character
- Member functions:
    - Medusa() initialize the following:
        - Int strength
        - Int Armor
        - String Name
        - Int attack die size
        - Int attack die num
        - Int defense die size
        - Int defence die num
    - Int attack()
        - Rolls 12 sided die
        - If 12, increase attack to 50
        - Else, attack is equal to rolled value
        - Returns attack value
    - Int defense(int incomingAttack)
        - Takes an incoming attack value and subtracts the defense roll and armor to get the inflicted damage.
        - Reduces strength by inflicted damage
        - Returns the defense roll
    - Int defenseDieRoll()
        - Rolls the characters defense dice and returns the result

Harry Potter class:
- Derived class from Character
- Data members:
    - Bool tearsOfDenial
- Member functions:
    - HarryPotter() initialize the following:
        - Int strength
        - Int Armor
        - String Name
        - Int attack die size
        - Int attack die num
        - Int defense die size

        - Int defence die num
        - Set hogwarts to true
    - Int attack()
        - Rolls dice equal to their attack dice and returns that value
    - Int defense(int incomingAttack)
        - Takes an incoming attack value and subtracts the defense roll and armor to get the inflicted damage.
        - Reduces strength by inflicted damage
        - If tears of denial is true and strengh < 1, call hogwarts()
        - Returns the defense roll
    - Int defenseDieRoll()
        - Rolls the characters defense dice and returns the result
    - Void hogwarts()
        - Sets strength to 20
        - Sets tearsOfDenial to false

Int roll(int die) function:
- Get random number from 1 - die
- Return that number
- Return total

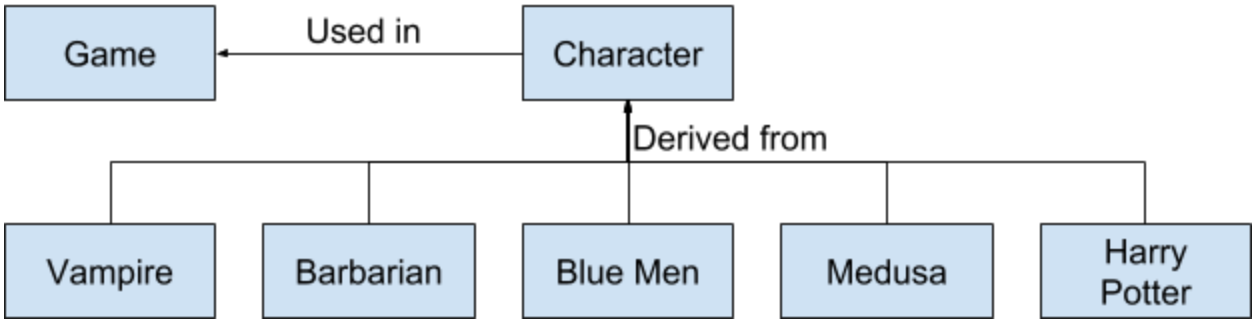
Menu:
- Display all character names for user to choose who fights
- Welcome prompt
- Define keepPlaying = true
- While keepPlaying is true:
    - Ask user to choose first combatant
    - Ask user to choose second combatant
    - Begin battle (while p1.hp > 0 && p2.hp > 0):
        - Play fight rounds
        - Each round must show:
            - Attacker type
            - Defender type, armor, hp
            - Attackers attack roll
            - Defenders defend roll
            - Total damage inflicted
            - Defenders updated hp after damage
        - Print victor
        - Free up any heap memory
    - After combat is over ask users:
        - Play again
        - Quit

Joel Huffman
02/15/18
Project 3 Reflection

Game class:

- Runs the game, plays out the battles until the
- Data members:
    - *character victor;
    - *character p1;
    - *character p2;
- Member functions:
    - Game(character* p1, character* p2)
        - Takes two pointers to characters for the combatants
    - Void battle(character* p1, character* p2):
        - Keeps playing rounds until a combatant dies
    - Void round()
        - Plays a single round of a battle
            - One player attacks
            - isDead()
            - Other player attacks
            - isDead()
    - Bool isDead(*character)
        - Checks if pointed to character has hp < 1
        - Returns a bool

Class hierarchy diagram:



Test Cases:

Test Case	Inputs	Expected outcome	Observed Outcome
Ensure Hogwarts is working	5 5	HarryPotter's health is set back to 20 after dropping below 1 once, and only once	HarryPotter's health is set back to 20 after dropping below 1 once, and only once
Ensure mob is working	3 3	BlueMen's max defense roll is 18 from 12-9 strength,	BlueMen's max defense roll is 18 from 12-9 strength,

Joel Huffman
02/15/18
Project 3 Reflection

		12 from 8-5 strength, and 6 from strength below 5	12 from 8-5 strength, and 6 from strength below 5
Ensure charm is working	1 1	Vampire's defense rolls should be 999 ~50% of the time	Vampire's defense rolls are 999 ~50% of the time
Ensure glare is working	4 4	Medusa's attacks should be 50 ~9% of the time	Medusa's attacks are infrequently 50, but they always kill the target (excluding pre-hogwarts Harry Potter)
BlueMen vs Harry Potter	3 5	BlueMen is victorious!	BlueMen is victorious!
Medusa vs Harry Potter	4 5	Medusa is victorious!	Medusa is victorious!
Vampire vs Medusa	1 4	Vampire is victorious!	Vampire is victorious!

Reflection:
Changes in design:

I originally didn't have a dedicated function for rolling dice. I soon added the roll(int die) function to the character class to reduce code repetition. Otherwise, my program pretty much adhered to the design document.

Problems encountered:

I experienced a "pure virtual function called" error that was due to my character objects resolving from going out of scope before they were called.

I also had some memory leak issues where I wasn't deallocating objects defined with the new keyword.

How problems were resolved:

I fixed the "pure virtual function called" problem by declaring the character objects with new to keep them on the heap until I was ready to delete them.

I resolved my memory leaks by using valgrind to identify the problem then properly deallocating that heap memory with the delete keyword