Design

Menu Class
Introduction function:
This function will have a greeting as well as the instructions presented to the user. Should be its own function as it will only need to be used once at the start of the game.

Fight choice function:
Present the user with the list of available fighters and have them choose whichever one they like. Create a new instance of the class of fighter that the user chose.

Fight function:
Function should have respective combatants battle each other until one of them is dead. Function should then declare the winner.

Play or quit function:
Ask the user if they would like to play again or quit.

Character class (base class for barbarian, bluemen, medusa, harry potter and vampire)
Have functions that allow for attack and defense as well as to get the characters attributes (armor and strength) and their name to declare the winner. Defense function needs to take both defensive die roll and base armor into consideration when calculating damage.

Barbarian
Has everything that is found in the character class with no special abilities.

Bluemen
Has all the same functions as a character class. Defense function to account for decrease in number of defensive dice bluemen have based on the damage they've taken.

Medusa
Has all the same functions as the character class with a modification to the attack function. Attack function needs to account for one hit kill using glare special ability.

Harry Potter
Has all the same functions as the character class with a modification to defense. Special ability hogwarts allows player to be revived with double their original strength (10 to 20). This can only happen once. Perhaps set a bool function which changes when Harry was revived so it doesn't loop.

Vampire
Has all the same functions as the character class with a modified defense function. Special ability charm prevents other fighter from attacking. Perhaps it can trick the other fighter while nullifying the damage the other fighter would have otherwise caused.

Input validation needed for fighter selection and to see if player would like to play another round or quit the game.

Main

Creates an instance of the menu class and calls it's functions to play through as many rounds as they player would like.

TESTING

Test	Description	Expected Result
Input validation for menus	Test to see if player typed acceptable input.	If inappropriate entry is made message will display indicating input was invalid and prompt to enter again.
Play or quit	Decision tree based on if the user wants to play or quit	If play take to main menu if quit, quit program
Bluemen special ability	Defensive die decrease as damage increases	2 die when between 8 and 4 health, less than 4 health 1 die.
Harry Potter special ability	Brings character back to life once.	If strength reaches 0 set it to 20. When strength reaches 0 again Harry dies.
Medusa special ability	Glare used	If attack equals 12 use glare which kills any player (except vampire with charm active) in one shot.
Vampire special ability	Activate charm	50% activation chance at which time vampire takes no damage.

Cause damage	Roll die and cause damage	Make sure damage is in accordance with number of die and rolls.
Correct block	Decrease block if block > damage	Have block equal damage to nullify it instead of having defender defend against more damage than would have been taken.
Winner	Exit combat when one fighter strength = 0.	Whoever has strength > 0 is the winner of the round.

Reflection

Out of all the other projects I thought that this was the easiest so far. There was a lot of copy and paste with inheritance and polymorphism functions for the various characters. However, that being said it was also tricky to implement some of the special abilities that were required. The attack and defend functions of the respective characters did most of the work. Out of all of them I'd have to say the vampire was the most difficult for me to wrap my mind around. I was hard pressed to come up with an idea on how to not get the other fighter to attack if there wasn't anyway for them to tell they were fighting a vampire. In the end, I let the fighter "believe" they attacked and used a bit of lore to let the audience know that they too may have been put under the vampires charm spell causing them to believe in something that didn't happen. After that the vampire simply doesn't take any damage and the game continues.

Overall I'd have to say that this has been the most fun I've had on a project. Partly because I enjoyed playing games like this growing up but also due to the freedom we were given. I feel like this program has been the loosest in terms of restrictions that we have had up until this point. This allows the game creators (myself in this case) a lot of wiggle room to deal with problems in unique ways.