Name: Orestes Bringas
Date: 04/22/2021

Project title: DND Console Simulator

Summary of Project: This is meant to be a DND simulator. The user will make a character based around their desired class (Mage, Warrior, Rogue, etc.) and roll for the stats. It will take a player down different paths with the chosen paths having their own enemies to fight. It is meant to simulate the initial power level of a player based on the stats and chosen class.

Classes:

1. Character: This class will be the main class which will hold the pure virtual functions and pass data to the Rogue, Mage, and Warrior subclasses. It creates the parent class/abstract class that will be used by the subclasses.
2. Rogue: This will be a subclass of Character which will have its own set of increased stats such as Charisma or Dexterity for example.
3. Mage: This will be a subclass of Character with its own set of increased stats such as Intelligence.
4. Warrior: This will likely be the final subclass of Character and will also house increased stats corresponding to this class, such as increased strength.
5. Enemy: This will be the enemy class which will set up the structure for the general enemy build.

Character Class

Abstract Class: Yes
Subclass of: N/A
Composed of: N/A

Data Members

Variable Name	**Data Type**	**Static**	**Description**
Name	String	No	Holds character name
baseHealth	Double	No	Holds base health for all characters
baseMana	Double	No	Holds base mana for the three classes
Race	String	No	Holds the race

Member Functions

Prototype	**Static**	**Virtual**	**Overloading Operator**	**Friend of this class**	**Description**
Char();	No	No	No	No	Constructor
~Char();	No	Yes	No	No	Destructor
`virtual void` setBaseHealth() = 0;	No	Yes	No	No	Sets base health numbers. Can be adjusted by subclasses.
`virtual void` setBaseMana() = 0;	No	Yes	No	No	Sets the base mana numbers. Can be adjusted by subclasses.

Rogue Class

Abstract Class: No
Subclass of: Character
Composed of: N/A
Data Members

Variable Name	Data Type	Static	Description
Strength	Int	No	Holds strength
Intelligence	Int	No	Holds intelligence
Wisdom	Int	No	Holds wisdom
Dexterity	Int	No	Hold dexterity
Constitution	Int	No	Holds constitution
Charisma	Int	No	Holds charisma
DamageNum	int	No	Holds damage numbers

Member Functions

Prototype	Static	Virtual	Overloading Operator	Friend of this class	Description
void setStrength()	No	No	Yes	No	Rolls for strength
void setIntel()	No	No	Yes	No	Rolls for intelligence
void setWis()	No	No	Yes	No	Rolls for wisdom
void setCon()	No	No	Yes	No	Rolls for constitution
void setCharis()	No	No	Yes	No	Rolls for charisma
void setDex()	No	No	Yes	No	Rolls for dexterity
void getDmgNum()	No	No	No	Yes	Returns damage numbers based on stats
void setBaseMana()	No	No	No	No	Adjusts mana rogue
void setBaseHealth()	No	No	No	No	Adjusts health
Void backstab();	No	No	No	No	Adjusts damage
Friend void printInfo();	No	No	Yes	No	Prints player info
Friend void printBase();	No	No	No	No	Sets and prints stats

Mage Class

Abstract Class: No
Subclass of: Character
Composed of: N/A
Data Members

Variable Name	**Data Type**	**Static**	**Description**
Strength	Int	No	Holds strength
Intelligence	Int	No	Holds intelligence
Wisdom	Int	No	Holds wisdom
Dexterity	Int	No	Hold dexterity
Constitution	Int	No	Holds constitution
Charisma	Int	No	Holds charisma
DamageNum	int	No	Holds damage numbers

Member Functions

Prototype	**Static**	**Virtual**	**Overloading Operator**	**Friend of this class**	**Description**
void setStrength()	No	No	Yes	No	Rolls for strength
void setIntel()	No	No	Yes	No	Rolls for intelligence
void setWis()	No	No	Yes	No	Rolls for wisdom
void setCon()	No	No	Yes	No	Rolls for constitution
void setCharis()	No	No	Yes	No	Rolls for charisma
void setDex()	No	No	Yes	No	Rolls for dexterity
void getDmgNum()	No	No	No	Yes	Returns damage numbers based on stats
void setBaseMana()	No	No	No	No	Adjusts mana rogue
void setBaseHealth()	No	No	No	No	Adjusts health
Void backstab();	No	No	No	No	Adjusts damage
Friend void printInfo();	No	No	Yes	No	Prints player info
Friend void printBase();	No	No	No	No	Sets and prints stats

Warrior Class

Abstract Class: No
Subclass of: Character
Composed of: N/A
Data Members

Variable Name	**Data Type**	**Static**	**Description**
Strength	Int	No	Holds strength
Intelligence	Int	No	Holds intelligence
Wisdom	Int	No	Holds wisdom
Dexterity	Int	No	Hold dexterity
Constitution	Int	No	Holds constitution
Charisma	Int	No	Holds charisma
DamageNum	int	No	Holds damage numbers

Member Functions

Prototype	**Static**	**Virtual**	**Overloading Operator**	**Friend of this class**	**Description**
void setStrength()	No	No	Yes	No	Rolls for strength
void setIntel()	No	No	Yes	No	Rolls for intelligence
void setWis()	No	No	Yes	No	Rolls for wisdom
void setCon()	No	No	Yes	No	Rolls for constitution
void setCharis()	No	No	Yes	No	Rolls for charisma
void setDex()	No	No	Yes	No	Rolls for dexterity
void getDmgNum()	No	No	No	Yes	Returns damage numbers based on stats
void setBaseMana()	No	No	No	No	Adjusts mana rogue
void setBaseHealth()	No	No	No	No	Adjusts health
Void backstab();	No	No	No	No	Adjusts damage
Friend void printInfo();	No	No	Yes	No	Prints player info
Friend void printBase();	No	No	No	No	Sets and prints stats

Enemy Class

Abstract Class: No
Subclass of: N/A
Composed of: N/A

Data Members

Variable Name	**Data Type**	**Static**	**Description**
baseHealth	Int	Yes	Sets base health for all enemies
DmgNum	Double	No	Sets the damage for the enemies
Race	String	Yes	Sets the enemy type
Resist	Int	No	Holds resistance numbers to damage

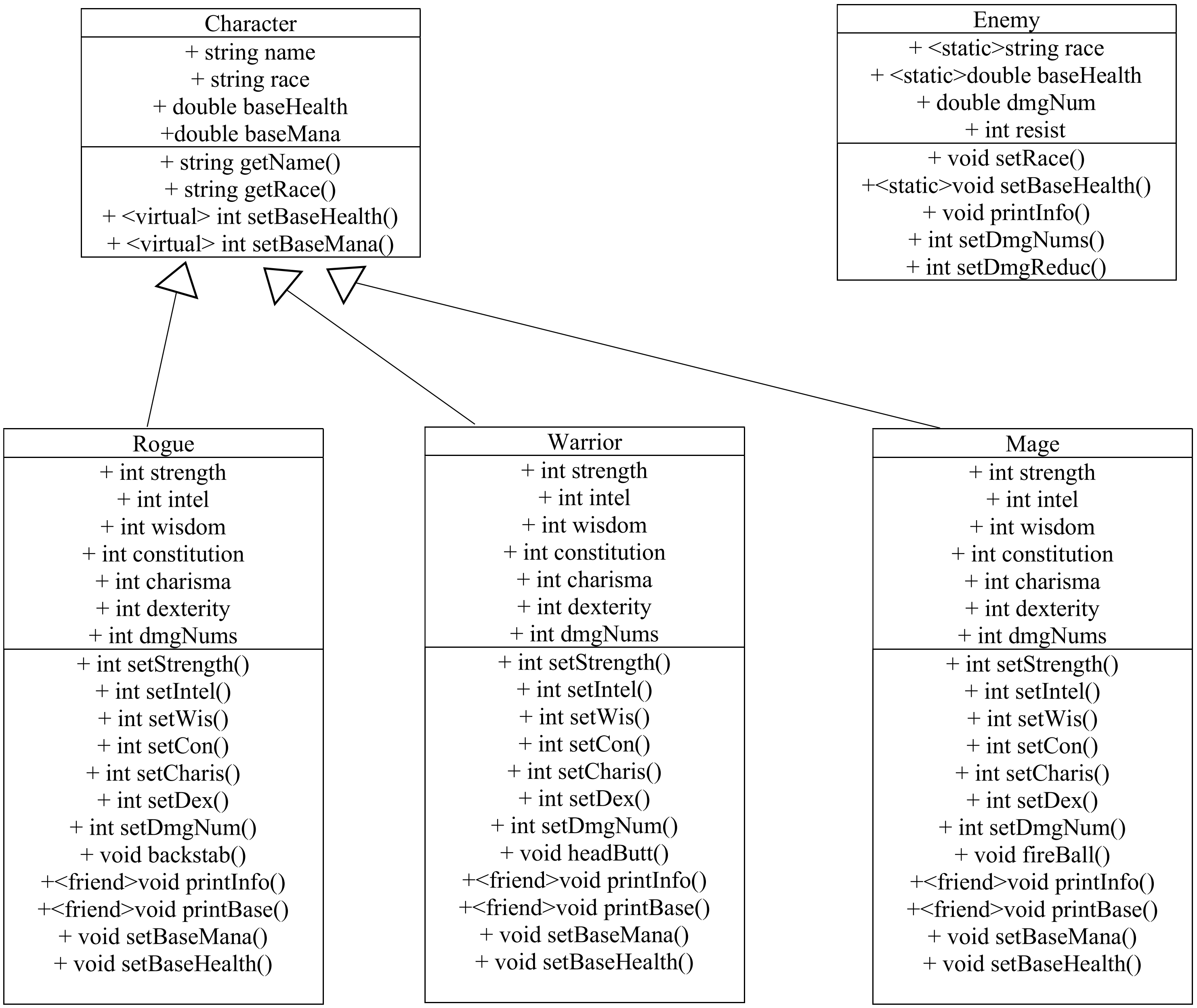
Member Functions

Prototype	**Static**	**Virtual**	**Overloading Operator**	**Friend of this class**	**Description**
void setRace();	No	No	No	No	sets the enemy type
Static void setBaseHealth();	Yes	No	No	No	Sets the base health for the enemies
Void printInfo();	No	No	No	No	Prints enemy info (health and race)
void setDmgNum();	No	No	No	No	Sets damage numbers
void setDmgReduc();	No	No	No	No	Sets the damage reduction due to resistances

Demonstration of OOD Concepts

1. Encapsulation:
    a. I made the Character class have public data members with public functions. (char.h, lines 12 and 17)
    b. The subclasses will have public data members with public functions but the data members will only be affected by the subclass and main function. (rog.h war.h mage.h, lines 9 and 17)
2. Inheritance:
    a. The subclasses for the different classes will all inherit the base stats from the Character class. This will include the "baseHealth" and "baseMana" data members which will then be adjusted by the subclasses according to which class it is and the stats the player rolls. (rog.h war.h mage.h, lines 30 and 31)
3. Polymorphism:
    a. This will take the form of base health and base mana. I will have the health and mana related functions called and readjusted per class. Damage will cause reductions in health moving it away from the base numbers. Healing will cause increases in health again changing it from base numbers. This causes the functions for health and mana to have multiple different uses. (char.h, lines 20 and 21)
4. Static Members/Functions:
    a. This will only be for the Enemy class as the enemy will be a simple object. The enemy will not benefit from any stats so the health will remain unchanged depending on the enemy type. Each enemy type will have its own static health and race. (enemy.h, lines 13,14, and 21)
5. Friend functions:
    a. I used friend functions to print the information about the player health, class, and mana. The printInfo() and printBase() functions both are friends to the subclasses and accept an object as an argument in order to output the information for the player's current condition and attributes. (rog.h, mage.h, war.h, lines 28 and 29)
6. Overloaded Operators:
    a. My overloading was done mainly to create new enemies upon the death of the original enemy. I used the new operator to create a new enemy and continue combat simulation within the game. (main.cpp, lines 102, 245, 387)
7. Text File(s):
    a. I used this to handle the inventory for the characters. I was able to use it to write to, read from, and remove items from the inventory.txt file. (main.cpp, lines 496 – 532)

UML Diagram



Character

+ string name
+ string race
+ double baseHealth
+double baseMana

+ string getName()
+ string getRace()
+ <virtual> int setBaseHealth()
+ <virtual> int setBaseMana()

Enemy

+ <static>string race
+ <static>double baseHealth
+ double dmgNum
+ int resist

+ void setRace()
+<static>void setBaseHealth()
+ void printInfo()
+ int setDmgNums()
+ int setDmgReduc()

Rogue

+ int strength
+ int intel
+ int wisdom
+ int constitution
+ int charisma
+ int dexterity
+ int dmgNums

+ int setStrength()
+ int setIntel()
+ int setWis()
+ int setCon()
+ int setCharis()
+ int setDex()
+ int setDmgNum()
+ void backstab()
+<friend>void printInfo()
+<friend>void printBase()
+ void setBaseMana()
+ void setBaseHealth()

Warrior

+ int strength
+ int intel
+ int wisdom
+ int constitution
+ int charisma
+ int dexterity
+ int dmgNums

+ int setStrength()
+ int setIntel()
+ int setWis()
+ int setCon()
+ int setCharis()
+ int setDex()
+ int setDmgNum()
+ void headButt()
+<friend>void printInfo()
+<friend>void printBase()
+ void setBaseMana()
+ void setBaseHealth()

Mage

+ int strength
+ int intel
+ int wisdom
+ int constitution
+ int charisma
+ int dexterity
+ int dmgNums

+ int setStrength()
+ int setIntel()
+ int setWis()
+ int setCon()
+ int setCharis()
+ int setDex()
+ int setDmgNum()
+ void fireBall()
+<friend>void printInfo()
+<friend>void printBase()
+ void setBaseMana()
+ void setBaseHealth()