Project: Robot Arena RPG

Nicolas Gerard Mangilit

2120815

|  |
| --- |
| **Robot** |
| Private:  string name;  int health;  int maxHealth;  int energy;  int maxEnergy;  int xp;  int neededXp;  int level;  int damage; |
| Public:  Robot();  Robot(const string, int, int, int, int, int);  string getName();  void setHealth(int);  int getHealth();  void setMaxHealth(int);  int getMaxHealth();  void setEnergy(int);  int getEnergy();  void setMaxEnergyint(int);  int getMaxEnergy();  void setXp(int);  int getXp();  void setNeededXp(int);  int getNeededXp();  void setDamage(int);  int getDamage();  void setLevel(int);  int getLevel();  void saveProgressToFile(string filename);  void loadProgressFromFile(string filename);  virtual void repair() = 0;  virtual void charge() = 0;  virtual void gainXp() =0;  virtual void standardAttack(Robot& enemy) = 0;  virtual void gainLevel() = 0;  virtual void printInfo() =0; |

|  |  |  |
| --- | --- | --- |
| **Tank** | **Scrapper** | **Sniper** |
| Private:  int armorPlates; | private:  int berzerkCharges; | Private:  int critChance;  double critDamage; |
| public:  Tank(int, const string, int, int, int, int,int);  void repair();  void charge();  void gainXp(); //player gains xp  void setArmorPlates(int);  int getArmorPlates();  void standardAttack(Robot& enemy);  void gainLevel();  void plasterBlaster(Robot& target);  void harmerArmor(Robot& target);  void printInfo(); | public:  Scrapper(int, const string, int, int, int, int,int);  void setBerzerkCharges(int);  int getBerzerkCharges();  void repair();  void charge();  void gainXp();  void standardAttack(Robot& enemy);  void gainLevel();  void maniacsCharge(Robot& target);  void scrapperRapper(Robot& target);  void printInfo(); | Sniper(int, double, const string, int, int, int, int,int);  void setCritChance(int);  int getCritChance();  void setCritDamage(double);  double getCritDamage();  void repair();  void charge();  void gainXp(); //after battle player gets xp  void standardAttack(Robot& enemy);  void gainLevel();  void trickShot(Robot& enemy);  void doubleShot(Robot& enemy);  void printInfo(); |

Members of Robot Base Class:

Robot() //sets name to “”,health and max health to 1000, energy and max energy to 100, xp to 0, and damage to 100

Robot(const string, int, int, int, int, int) //sets attributes based class

getName() // gets name of robot

setHealth(int) // sets health of robot, cant be more than max health

int getHealth(); //gets robots health

void setMaxHealth(int); //sets the max health for robot

int getMaxHealth(); //gets the max health for robot

void setEnergy(int); //sets the current energy of the robot, cant be more than max energy

int getEnergy(); //gets current energy

void setMaxEnergyint(int); //sets the max energy

int getMaxEnergy(); //gets the max energy

void setXp(int); //sets the xp of robot

int getXp(); //gets the xp

void setNeededXp(int); //set the neededxp

int getNeededXp(); //gets the neededxp

void setDamage(int); //sets base damage value of robot

int getDamage(); //gets that damage value

void setLevel(int); //sets current level of the player

int getLevel(); //gets that level

void saveProgressToFile(string filename); //save the stats of the player into the string which is the files name

void loadProgressFromFile(string filename); //loads the progress from the save file

virtual void repair() = 0; // heals robot for certain percent of health based on class. Classes define this

virtual void charge() = 0; // allows robots to get more energy

virtual void gainXp() =0; //after battle, user gets xp and levels up if xp threshold is met

virtual void standardAttack(Robot& enemy) = 0; //all robots have a standard attack but they vary depending on what robot is chosen

virtual void gainLevel() = 0; //when xp equals max x, the robot levels up and allows for player to increase a stat

virtual void printInfo() =0; // prints info of robot

Members of Tank Derived Class:

Private:

int armorPlates;

public:

Tank(int, const string, int, int, int, int,int); //Tank overloaded constructor (plates,name,health,energy,level,damage,xp)

void repair(); //this function allows robot to gain health for a turn. parameter is health

void charge(); //allows for robot to gain energy

void gainXp(); //player gains xp

void setArmorPlates(int); //sets the number of armor plates that the player has

int getArmorPlates(); //gets the current number of plates a player has

void standardAttack(Robot& enemy); //nothing special with tank special attack

void gainLevel(); //when xp equals max x, the robot levels up and allows for player to increase a stat

void plasterBlaster(Robot& target); //deals slightly less damage than a normal attack but it allows the user to receive an armor plate

void harmerArmor(Robot& target); //can only use it if you have an armor plate. if you do, -1 plate and deal 1.5 damage

void printInfo(); //prints the name,health,energy,and number of plates

Members of Scrapper Derived Class:

private:

int berzerkCharges;

public:

Scrapper(int, const string, int, int, int, int,int); //(berzerkcharges, name, health,energy,damage,xp,level)

void setBerzerkCharges(int); //Sets the number of berserk charges the player gets, the player only gets 2 each battle

int getBerzerkCharges(); //gets the number of berserk charges that the user has

void repair(); //this function allows robot to gain health for a turn. parameter is health

void charge(); //allows for robot to gain energy

void gainXp(); //allows player to gain xp and level up if threshold is met

void standardAttack(Robot& enemy); //a normal attack but base damage is higher for this class

void gainLevel(); //when xp equals max x, the robot levels up and allows for player to increase a stat

void maniacsCharge(Robot& target); //deal double damage but take damage, uses berzerk charge

void scrapperRapper(Robot& target); //terrible singing causes 1.5 damage but at cost of a lot of energy

void printInfo(); //Prints name,health,energy,charges

Members of Sniper Derived Class:

Sniper(int, double, const string, int, int, int, int,int); //(critChance, critDamage , name, health, energy,)

void setCritChance(int); //sets crit chance of player

int getCritChance(); //gets crit chance variable

void setCritDamage(double); //sets a double to set crit damage multiplier

double getCritDamage(); //uses that multiplier

void repair(); //this function allows robot to gain health for a turn. parameter is health

void charge(); //allows for robot to gain energy

void gainXp(); //after battle player gets xp

void standardAttack(Robot& enemy); //all robots have a standard attack but they vary depending on what robot is chosen

void gainLevel(); //when xp equals max x, the robot levels up and allows for player to increase a stat

void trickShot(Robot& enemy); //have a 30 percent chance to do a lot of damage, 70 percet n to do none

void doubleShot(Robot& enemy); //attack twice for .75 damage but each has a chance to crit

void printInfo(); // Prints name, health, energy

The Robot class is the parent class. It holds all the functions and attributes that all the classes share. Things like Health, xp, energy, and name. This is also the class where pure virtual functions are declared that need to be defined in the derived classes. These are needed because they perform similar functions but are different in nature.

The Tank class is a derived class from the Robot Class. The special mechanic that the Tank class gets is the idea of armor plates. Having armor plates mitigates whenever any damage is done to the player. Every time that damage is mitigated, the user loses 1 armor plate. The tank also gets a higher base health and higher based healing. One ability allows for the player to deal damage and gain a plate at the same time. The other ability uses a plate to deal extra damage.

The Scrapper class is a derived class from the Robot class. The special mechanic that the Scrapper class gets is the idea of berserk charges. The scrapper is a class that focuses on attack and is more like a glass cannon. The berserk charges are mainly just used for the Scrapper first ability. It uses up a charge to increase damage output but it then deals damage to the player. The second ability is just a silly ability that does more damage at the cost of extra energy. The scrapper has higher base damage but the lowest base health.

The Sniper class is a derived class from the Robot class. The special mechanic that the Sniper class gets is the idea of critical hits. This is done with two variables with crit chance being an integer and crit damage being a double. The chance is determined with the random function. The double is a multiplier of damage dealt if the hit does a critical chance. The first ability is a game of chance with a big payout. You are more likely to lose but if you succeed, you deal triple damage. The second ability has the user shoot twice but for slightly reduced damage. Each shot has a chance to become a critical hit though.

Pseudocode for Main:

1. new character
   1. If loop to choose from 3 class
      1. set a variable that indicates the class. (1=class 1, 2=class 2, 3=class 3) It’ll help with battle system
      2. Tank
         1. Initialize player object
         2. Also initialize max xp needed to level up (100 at first)
         3. Special mechanic is armor plates
      3. Scrapper
         1. Initialize player object
         2. Also initialize max xp needed to level up (100 at first)
         3. special mechanic is berzerk
      4. Sniper
         1. Initialize player object
         2. Also initialize max xp needed to level up (100 at first)
         3. special mechanic is crit
2. load character
   1. call loadProgress function with
3. save game
   1. call save function
4. battle
   1. randomly generate an enemy made from the enemy class
   2. if loop that depends on the class chosen
      1. Class 1 battle loop
         1. normal attack
         2. heal
         3. charge
         4. special 1
         5. special 2
      2. Class 2 battle loop
         1. normal attack
         2. heal
         3. charge
         4. special 1
         5. special 2
      3. Class 3 battle loop
         1. normal attack
         2. heal
         3. charge
         4. special 1
         5. special 2
      4. Once done it now displays if you won or not
         1. if you won you get xp
      5. Statuses are then reverts like health, energy, and charges for certain classes
5. exit game
   1. use exit() function