Monster.cpp

User manual

Program: main.cpp

- This programs is a combat game between player and it's enemy.
- The name of the game is Adventure Battle.
- The user will first be asked to enter his/ her hero's name.
- Then they will have option to choose number of enemy they want to fight against.
- The selection of enemy will be random. The enemy would be selected among 5 different types of enemy.
- The hero will have option to choose different command. These includes Sword, Shield, Fireball, Potion.
- Player has 10 fireball and 6 potion.
- Fireball causes greater damage to the enemy. And potion increases the health of the hero.
- Shield limits the damage caused by the enemy to the hero by half.
- Sword is the basic attack of the player.
- The player can even exit the game by typing "exit" during the game.
- At the end, if layer defeats all the enemy he/she selects, the player will be declared the winner else will be declared defeated.

System Manual

Program: main.cpp

- This programs is a combat game between player and it's enemy.
- The name of the game is Adventure Battle.
- This program contains 7 user defined class.
- These classes are : bat_1, enemy, hero, dragon, wolf, bear, lizard.
-

Main function:

int main()

- This is the main function of the game.
- It first displays the name of the game.
- Then the player is asked to enter a name for his/ her hero.
- The name is stored in string variable name.
- The setname() method of hero class is called to set the name of the hero.
- They the player is asked to enter the number of enemy he or she wants to fight against.
- Then a for loop is used to push back enemies to the vector enem of enemy class type. This for loop is executed for number of enemies the payer wants to fight against. And also decides the enemy to fight against randomly.
- Then the next for loop is used to carry out battle.
- It first displays the enemy that the hero will encounter.
- Then a while loop is executed until the hero or the player dies (runs out of health).
- The player can choose the attack, the attack is stored in string variable command.
- If the value of command is "exit" then the program is terminated.
- Else if the command is valid then the enemy_attack pure virtual method of enemy class is called.
- The attack is carried out and the health to be decreased from the enemy is returned.
- This health is then passed to the enemy_damage pure virtual method of the hero's opponent.
- If the player dies, "Player is defeated" is displayed.
- Else if the enemy dies then combat between hero and next enemy begins and the same loop goes on till the last enemy is defeated.
- If the hero defeats all the enemy, then the "Enemy is defeated" is printed to the console and the game ends.
- At the end all the dynamically allocated memory is deleted.

Base Class

enemy header file

enemy.h

- This header file first checks if _ENEMY_H_ has been defined or not. If it is not defined, then the header file _ ENEMY_H_ is defined.
- This file then contains the enemy class definition.
- The class declared in this header file contains only pure virtual methods that are essential for all the enemy and hero that inherits this class.
- All the methods are defined to be public and the variables/ attributes are defined to be private.

Accessibility	Type of method	Prototype	Use
Public	Constructor	`enemy()`	
	Methods	`virtual string getname() = 0;`	
		`virtual int enemy_health() = 0;`	
		`virtual void enemy_damage(int) = 0;`	
		`virtual int enemy_attack(string) = 0;`	
		`virtual bool enemy_isdefeated() = 0;`	
		`virtual string getdescription() const = 0;`	
Private	Attributes	`int health_;`	To store health of the hero/ enemy
		`string name_;`	To store name of the enemy/ hero

Enemy class implementation

enemy.cpp

- This .cpp file has body of constructor of enemy class.
- Since all other methods in enemy.h file is purely virtual, their implementation is not required.

Derived Class

bat_1 Class header file

bat.h

- This header file contains definition of bat class
- This header file first checks if _BAT_1_H_ has been defined or not. If it is not defined, then the header file _ BAT_1_H_ is defined.
- The class declared in this header file contains only the prototype of the methods and the attributes required for this class.
- This class publicly inherits the enemy class
- All the methods are defined to be public and the variables/ attributes are defined to be private.

Accessibility	Type of method	Prototype	Use
Public	Constructor	`bat();`	
	Methods	`string getname();`	Returns name of the enemy
		`int enemy_health();`	Returns health of the enemy
		`void enemy_damage(int dam);`	To reduce health of the enemy when attacked
		`int enemy_attack(string a);`	Decides which attack should be used
		`bool enemy_isdefeated();`	Checks if the enemy health is above 0 or not
		`string getdescription() const;`	Gives description of the enemy

Card class implementation

bat_1.cpp

- This file has body of all the methods included in bat_1.h.
- The constructor sets health_ to 300 and name to "Bat".
- The getname() method returns the name of the enemy. Which is "bat" for this enemy.
- The enemy_health() method returns health of the enemy.
- The enemy_damage() method takes a integer parameter. This integer value is then reduced from the current health of the object.
- enemy_attack() method is used by the object to attack it's opponent. There are two attacks available for the object of this class. The attack is selected randomly. If the time is even, then "Bat screams with UltraSonic Waves" and causes damage of 50 health point to the opponent. Else if the time is odd, then "Bat Bites the opponent" and causes damage of 75 health point to the opponent.
- The enemy_isdefeated() method checks if the enemy is defeated or not. If the health is 0 or below, then the enemy is defeated and true is returned else false is returned.
- The getdescription() function returns the description of the class object.

lizard Class header file
lizard.h

- This header file contains definition of lizard class
- This header file first checks if _LIZARD_H_ has been defined or not. If it is not defined, then the header file _ LIZARD_H_ is defined.
- The class declared in this header file contains only the prototype of the methods and the attributes required for this class.
- This class publicly inherits the enemy class
- All the methods are defined to be public and the variables/ attributes are defined to be private.

Accessibility	Type of method	Prototype	Use
Public	Constructor	`lizard();`	
	Methods	`string getname();`	Returns name of the enemy
		`int enemy_health();`	Returns health of the enemy
		`void enemy_damage(int dam);`	To reduce health of the enemy when attacked
		`int enemy_attack(string a);`	Decides which attack should be used
		`bool enemy_isdefeated();`	Checks if the enemy health is above 0 or not
		`string getdescription() const;`	Gives description of the enemy

Card class implementation
lizard.cpp

- This file has body of all the methods included in lizard.h.
- The constructor sets health_ to 400 and name to "lizard".
- The getname() method returns the name of the enemy. Which is "lizard" for this enemy.
- The enemy_health() method returns health of the enemy.
- The enemy_damage() method takes a integer parameter. This integer value is then reduced from the current health of the object.
- enemy_attack() method is used by the object to attack it's opponent. There are two attacks available for the object of this class. The attack is selected randomly. If the time is even, then "Lizard Jumps" and causes damage of 50 health point to the opponent. Else if the time is odd, then "Lizard Flash Attack" and causes damage of 75 health point to the opponent.
- The enemy_isdefeated() method checks if the enemy is defeated or not. If the health is 0 or below, then the enemy is defeated and true is returned else false is returned.
- The getdescription() function returns the description of the class object.

bear Class header file
bear.h

- This header file contains definition of bear class
- This header file first checks if _BEAR_H_ has been defined or not. If it is not defined, then the header file _ BEAR_H_ is defined.
- The class declared in this header file contains only the prototype of the methods and the attributes required for this class.
- This class publicly inherits the enemy class
- All the methods are defined to be public and the variables/ attributes are defined to be private.

Accessibility	Type of method	Prototype	Use
Public	Constructor	`bear();`	
	Methods	`string getname();`	Returns name of the enemy
		`int enemy_health();`	Returns health of the enemy
		`void enemy_damage(int dam);`	To reduce health of the enemy when attacked
		`int enemy_attack(string a);`	Decides which attack should be used
		`bool enemy_isdefeated();`	Checks if the enemy health is above 0 or not
		`string getdescription() const;`	Gives description of the enemy

Card class implementation
bear.cpp

- This file has body of all the methods included in bear.h.
- The constructor sets health_ to 300 and name to "bear".
- The getname() method returns the name of the enemy. Which is "bear" for this enemy.
- The enemy_health() method returns health of the enemy.
- The enemy_damage() method takes a integer parameter. This integer value is then reduced from the current health of the object.
- enemy_attack() method is used by the object to attack it's opponent. There are two attacks available for the object of this class. The attack is selected randomly. If the time is even, then "Bear Jumps" and causes damage of 100 health point to the opponent. Else if the time is odd, then "Bear Flash Attack" and causes damage of 75 health point to the opponent.
- The enemy_isdefeated() method checks if the enemy is defeated or not. If the health is 0 or below, then the enemy is defeated and true is returned else false is returned.
- The getdescription() function returns the description of the class object.

dragon Class header file
dragon.h

- This header file contains definition of dragon class
- This header file first checks if _DRAGON_H_ has been defined or not. If it is not defined, then the header file _ DRAGON_H_ is defined.
- The class declared in this header file contains only the prototype of the methods and the attributes required for this class.
- This class publicly inherits the enemy class
- All the methods are defined to be public and the variables/ attributes are defined to be private.

Accessibility	Type of method	Prototype	Use
Public	Constructor	`dragon();`	
	Methods	`string getname();`	Returns name of the enemy
		`int enemy_health();`	Returns health of the enemy
		`void enemy_damage(int dam);`	To reduce health of the enemy when attacked
		`int enemy_attack(string a);`	Decides which attack should be used
		`bool enemy_isdefeated();`	Checks if the enemy health is above 0 or not
		`string getdescription() const;`	Gives description of the enemy

Card class implementation
dragon.cpp

- This file has body of all the methods included in dragon.h.
- The constructor sets health_ to 300 and name to "dragon".
- The getname() method returns the name of the enemy. Which is "dragon" for this enemy.
- The enemy_health() method returns health of the enemy.
- The enemy_damage() method takes a integer parameter. This integer value is then reduced from the current health of the object.
- enemy_attack() method is used by the object to attack it's opponent. There are two attacks available for the object of this class. The attack is selected randomly. If the time is even, then "Dragon Jumps" and causes damage of 100 health point to the opponent. Else if the time is odd, then "Dragon Flash Attack" and causes damage of 75 health point to the opponent.
- The enemy_isdefeated() method checks if the enemy is defeated or not. If the health is 0 or below, then the enemy is defeated and true is returned else false is returned.
- The getdescription() function returns the description of the class object.

wolf Class header file
wolf.h

- This header file contains definition of wolf class
- This header file first checks if _WOLF_H_ has been defined or not. If it is not defined, then the header file _ WOLF_H_ is defined.
- The class declared in this header file contains only the prototype of the methods and the attributes required for this class.
- This class publicly inherits the enemy class
- All the methods are defined to be public and the variables/ attributes are defined to be private.

Accessibility	Type of method	Prototype	Use
Public	Constructor	`wolf();`	
	Methods	`string getname();`	Returns name of the enemy
		`int enemy_health();`	Returns health of the enemy
		`void enemy_damage(int dam);`	To reduce health of the enemy when attacked
		`int enemy_attack(string a);`	Decides which attack should be used
		`bool enemy_isdefeated();`	Checks if the enemy health is above 0 or not
		`string getdescription() const;`	Gives description of the enemy

Card class implementation
wolf.cpp

- This file has body of all the methods included in wolf.h.
- The constructor sets health_ to 300 and name to "wolf".
- The getname() method returns the name of the enemy. Which is "wolf" for this enemy.
- The enemy_health() method returns health of the enemy.
- The enemy_damage() method takes a integer parameter. This integer value is then reduced from the current health of the object.
- enemy_attack() method is used by the object to attack it's opponent. There are two attacks available for the object of this class. The attack is selected randomly. If the time is even, then "Wolf Jumps" and causes damage of 100 health point to the opponent. Else if the time is odd, then "Wolf Flash Attack" and causes damage of 75 health point to the opponent.
- The enemy_isdefeated() method checks if the enemy is defeated or not. If the health is 0 or below, then the enemy is defeated and true is returned else false is returned.
- The getdescription() function returns the description of the class object.

Hero Class header file
hero.h

- This header file contains definition of hero class
- This header file first checks if _HERO_H_ has been defined or not. If it is not defined, then the header file _ HERO_H_ is defined.
- The class declared in this header file contains only the prototype of the methods and the attributes required for this class.
- This class publicly inherits the enemy class
- All the methods are defined to be public and the variables/ attributes are defined to be private.

Accessibility	Type of method	Prototype	Use
Public	Constructor	`hero();`	
	Methods	`string getname();`	Returns name of the enemy
		`void setname(string temp);`	Set's name of the hero
		`int enemy_health();`	Returns health of the enemy
		`int get_fireball();`	Returns number of fireball remaining.
		`int get_potion();`	Returns number of potion remaining.
		`void enemy_damage(int dam);`	To reduce health of the enemy when attacked
		`int enemy_attack(string a);`	Decides which attack should be used
		`bool enemy_isdefeated();`	Checks if the enemy health is above 0 or not
		`string get_status();`	Returns status of the hero.
		`string getdescription() const;`	Gives description of the enemy
Private	Attribute	`int potion_;`	Stores number of potion remaining
		`int fireball_;`	Stores number of fireball remaining
		`bool defend_mode;`	To store weather the shield is activated or not

Hero class implementation
hero.cpp

- This file has body of all the methods included in hero.h.
- The constructor sets health_ to 1000, potion_ to 6, fireball_ to 10 and defend_mode to false.
- The getname() method returns the name of the enemy. Which is "hero" for this enemy.
- The setname() method takes a string as its parameter and sets it to name.
- The get_fireball() returns the number of fireball remaining with the player.
- The get_potion() returns the number of potion remaining with the player.
- The enemy_health() method returns health of the enemy.
- The enemy_damage() method takes a integer parameter. If the defense mode is false then the integer value is reduced from the current health of the object. Else if the defense mode is set to false, then the health is reduced by half the value of integer.
- enemy_attack() method is used by the object to attack it's opponent. There are two attacks available for the object of this class. The attack is selected by the user.
  If the user selects "sword" or "fireball" then the description of these attack is printed to console and the value of damage it can cause to opponent is returned.
  If the user selects "potion" then the health of the player is increased by a certain amount.
  If the player selects shield, then the defence_mode is set to true.
- The enemy_isdefeated() method checks if the enemy is defeated or not. If the health is 0 or below, then the enemy is defeated and true is returned else false is returned.
- The getdescription() function returns the description of the class object.
- The get_status() method returns the status of the hero. The status contains, current health, number of fireball and number of potion remaining with the hero.