COMP2396B Object-oriented Programming and Java

Assignment 2 – State & Behavior of Class

Due date: 18th October, 2024 23:59

Story

After helping Handsome Jeff surviving his Big Northwest Motor Bus Company job crisis, you are curious on how he got passed in COMP2396. Jeff told you that the night before final exam, he broke into the general office to steal the examination paper and got 100 marks.

You are fascinated by this idea and wish to do the same. However, the general office has employed a security guard to secure the papers this year. One must take a combat to beat the guard before stealing anything. To ensure success, you want to develop a Java project (i.e. this assignment) to stimulate different scenario of the combat.

You are required to implement a few classes to model some characters and weapons, which will involve in the combat.

Part 1 – General Combat Scenario [50%]

To make things simple, you first stimulate a general combat scenario between two characters by implementing two classes:

- The ***Character*** class, which represents any game character.
- The ***Weapon*** class, which represents any game weapon.

The ***OfficeCombat1*** class is provided to simulate the fighting event. Both characters will hold a weapon and take turns to combat with each other. They have specific energy level at the beginning and lose energy upon being attacked by the weapon. The character loses if his energy level drops equal to or smaller than 0.

Implementation

The ***Character*** class:

Properties

`String characterName`
: Name of the character

`int skillLevel`
: Weapon skill level of the character

`int energyLevel`
: Remaining energy of the character. The character loses when this value drops to zero or below.

Methods (all methods must be public)

`Character(String name, int energyLevel, int skillLevel)`
: Constructor, which setup the three properties accordingly

`String getName()`
: Return the name of the character

`int getSkillLevel()`
: Return the skill level of the character

`int getEnergyLevel()`
: Return the remaining energy level of the character

`int hurt(int attackAmount)`
: Reduce the energy level of the character by the specific amount of attackAmount.
Returns the amount of energy reduced from the attack.

`int attack(Weapon w1)`

Return the total amount of attack generated by the character. It is the sum of power of the weapon generated from the shoot and the skill level of the character.

`boolean isLose()`

Return true if the character's energy level drops to zero or below.

The ***Weapon*** class:

Properties

`String weaponName`

Name of the weapon

`int power`

Power level of the weapon

Methods (all methods must be public)

`Weapon(String name, int power)`

Constructor, which setup the two properties accordingly

`String getName()`

Return the name of the weapon

`int shoot()`

Return the power generated when the weapon is shot/used, in which it equals to the power level of the weapon

You may add additional properties and methods to these two classes.

The Main Program

A sample main class, ***OfficeCombat1*** is given to test your program. Here is the code in the main method of the class:

```
import java.io.*;

public class OfficeCombat1 {
    public static void main(String[] args) throws IOException {
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader inData = new BufferedReader(isr);

        // Input combat data
        String c1_info[] = inData.readLine().split(" ");
        String c2_info[] = inData.readLine().split(" ");
        String w1_info[] = inData.readLine().split(" ");
        String w2_info[] = inData.readLine().split(" ");

        Character c1 = new Character(c1_info[0], Integer.valueOf(c1_info[1]),
                                     Integer.valueOf(c1_info[2]));
        Character c2 = new Character(c2_info[0], Integer.valueOf(c2_info[1]),
                                     Integer.valueOf(c2_info[2]));
        Weapon w1 = new Weapon(w1_info[0], Integer.valueOf(w1_info[1]));
        Weapon w2 = new Weapon(w2_info[0], Integer.valueOf(w2_info[1]));

        // Start fighting
        System.out.println("Now fighting: " + c1.getName() + " VS " + c2.getName());
        System.out.println("Skill level of " + c1.getName() + ": " + c1.getSkillLevel());
        System.out.println("Skill level of " + c2.getName() + ": " + c2.getSkillLevel());
        System.out.println("Energy level of " + c1.getName() + ": " + c1.getEnergyLevel());
        System.out.println("Energy level of " + c2.getName() + ": " + c2.getEnergyLevel());
        System.out.println("----------------------------");

        int round = 0;
        while (!c1.isLose() && !c2.isLose()) {
            if (round % 2 == 0) {
                int attackAmount = c1.attack(w1);
                int hurtAmount = c2.hurt(attackAmount);

                System.out.println(c1.getName() + " makes an attack by " + w1.getName() + "!");
                System.out.println(c2.getName() + " takes a hurt of " + hurtAmount + "!
                                   Remaining energy becomes " + c2.getEnergyLevel() + ".");
            }
            else {
                int attackAmount = c2.attack(w2);
                int hurtAmount = c1.hurt(attackAmount);

                System.out.println(c2.getName() + " makes an attack by " + w2.getName() + "!");
                System.out.println(c1.getName() + " takes a hurt of " + hurtAmount + "!
                                   Remaining energy becomes " + c1.getEnergyLevel() + ".");
            }
            round++;
        }

        if (c1.isLose()) {
            System.out.println(c2.getName() + " wins!");
        }
        else {
            System.out.println(c1.getName() + " wins!");
        }
    }
}
```

Sample Test Cases:

Test Case 1
Input:
TallRichHandsome 20 5
Jeppo 15 2
MP5 3
AK47 4

Output:
Now fighting: TallRichHandsome VS Jeppo
Skill level of TallRichHandsome: 5
Skill level of Jeppo: 2
Energy level of TallRichHandsome: 20
Energy level of Jeppo: 15

TallRichHandsome makes an attack by MP5!
Jeppo takes a hurt of 8! Remaining energy becomes 7.
Jeppo makes an attack by AK47!
TallRichHandsome takes a hurt of 6! Remaining energy becomes 14.
TallRichHandsome makes an attack by MP5!
Jeppo takes a hurt of 8! Remaining energy becomes -1.
TallRichHandsome wins!

Test Case 2
Input:
MrsHo 14 5
MrHo 9 3
Soup 4
Brick 3

Output:
Now fighting: MrsHo VS MrHo
Skill level of MrsHo: 5
Skill level of MrHo: 3
Energy level of MrsHo: 14
Energy level of MrHo: 9

MrsHo makes an attack by Soup!
MrHo takes a hurt of 9! Remaining energy becomes 0.
MrsHo wins!

Part 2 – General Office Combat Scenario [50%]

After the success from Part 1, you then stimulate the combat scenario that happens in the CS general office by implementing the following classes:

- The ***SecurityGuard*** class, which is a subclass of the ***Character*** class
- The ***Student*** class, which is a subclass of the ***Character*** class
- The ***BadGun*** class, which is a subclass of the ***Weapon*** class
- The ***SuperGun*** class, which is a subclass of the ***Weapon*** class

The ***SecurityGuard*** and ***Student*** have the following special features:

- The ***SecurityGuard*** is well-funded by the department and must use a ***SuperGun***. Only the guard knows how to boost the ***SuperGun***.
- The ***Student*** is poor and can only afford using a ***BadGun***. However, he/she can hide to dodge an attack and receive 0 hurt.

The ***BadGun*** and ***SuperGun*** have the following special features:

- The ***BadGun*** has poor quality such that at each shot, only 80% of it's original power can be produced (round down to the nearest integer).
- The ***SuperGun*** can produce a boosted shot, such that the shot produces two times power than the original.

The ***OfficeCombat2*** class is provided to simulate the fighting event. A ***SecurityGuard*** and ***Student*** will hold their weapon and take turns to combat with each other. They have specific energy level at the beginning and lose energy upon being attacked by the weapon. The character loses if his energy level drops equal to or smaller than 0.

Implementation

The ***SecurityGuard*** class, a subclass of the ***Character*** class:

Methods (all method must be public)

`SecurityGuard(String name, int energyLevel, int skillLevel)`
Constructor, which setup the three properties accordingly

`void boostWeapon(SuperGun w1)`
The guard boosts the gun for the next attack.

The ***Student*** class, a subclass of the ***Character*** class:

Methods (all method must be public)

`Student(String name, int energyLevel, int skillLevel)`
Constructor, which setup the three properties accordingly

`void hide()`
Hides himself / herself from the next attack.

`int hurt(int attackAmount)`, an overridden method
If hidden, he / she takes no hurt at all. Otherwise, reduce energy level with the amount specified as usual. Reset the hiding status afterwards. Returns the amount of energy reduced from the attack.

The ***BadGun*** class, a subclass of the ***Weapon*** class:

Methods (all method must be public)

`BadGun(String name, int power)`
Constructor, which setup the two properties accordingly

`int shoot()`, an overridden method
Return the actual amount of power generated by the gun at each shot, which is 80% of it's original power (round down to the nearest integer)

The ***SuperGun*** class, a subclass of the ***Weapon*** class:

Methods (all method must be public)

`SuperGun(String name, int power)`
Constructor, which setup the two properties accordingly.

`void boost()`
Boost the gun for the next attack.

`int shoot()`, an overridden method
If boosted, return twice of the power level if it is boosted. Otherwise, return the power level as usual. Reset the boosting effect afterwards.

You may add additional properties and methods to these three classes.

The Main Program

A sample main class, ***OfficeCombat2*** is given to test your program. Here is the code in the main method of the class:

```
import java.io.*;

public class OfficeCombat2 {
    public static void main(String[] args) throws IOException {
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader inData = new BufferedReader(isr);

        // Input combat data
        String c1_info[] = inData.readLine().split(" ");
        String c2_info[] = inData.readLine().split(" ");
        String w1_info[] = inData.readLine().split(" ");
        String w2_info[] = inData.readLine().split(" ");

        SecurityGuard c1 = new SecurityGuard(c1_info[0], Integer.valueOf(c1_info[1]),
                                             Integer.valueOf(c1_info[2]));
        Student c2 = new Student(c2_info[0], Integer.valueOf(c2_info[1]),
                                 Integer.valueOf(c2_info[2]));
        SuperGun w1 = new SuperGun(w1_info[0], Integer.valueOf(w1_info[1]));
        BadGun w2 = new BadGun(w2_info[0], Integer.valueOf(w2_info[1]));

        // Start fighting
        System.out.println("Now fighting: " + c1.getName() + " VS " + c2.getName());
        System.out.println("Skill level of " + c1.getName() + ": " + c1.getSkillLevel());
        System.out.println("Skill level of " + c2.getName() + ": " + c2.getSkillLevel());
        System.out.println("Energy level of " + c1.getName() + ": " + c1.getEnergyLevel());
        System.out.println("Energy level of " + c2.getName() + ": " + c2.getEnergyLevel());
        System.out.println("----------------------------");

        int round = 0;
        while (!c1.isLose() && !c2.isLose()) {
            if (round % 2 == 0) {
                int attackAmount = c1.attack(w1);
                System.out.println(c1.getName() + " makes an attack by " + w1.getName() + "!");

                int hurtAmount = c2.hurt(attackAmount);
                if (hurtAmount == 0) {
                     System.out.println(c2.getName() + " hides from the attack!");
                }
                else {
                     System.out.println(c2.getName() + " takes a hurt amount of " + hurtAmount
                     + "! Remaining energy becomes " + c2.getEnergyLevel() + ".");
                }

                if (round % 3 == 0) {
                     c2.hide();
                }
            }
            else {
                int attackAmount = c2.attack(w2);
                int hurtAmount = c1.hurt(attackAmount);

                System.out.println(c2.getName() + " makes an attack by " + w2.getName() + "!");
                System.out.println(c1.getName() + " takes a hurt amount of " + hurtAmount +
                                   "! Remaining energy becomes " + c1.getEnergyLevel() + ".");

                if (round % 3 == 0) {
                     c1.boostWeapon(w1);
                     System.out.println(c1.getName() + " boost the " + w1.getName() + "!");
                }
            }
            round++;
```

```
        }

        if (c1.isLose()) {
             System.out.println(c2.getName() + " wins! The examination paper is stolen!");
        }
        else {
             System.out.println(c1.getName() + " wins! The examination paper is secured!");
        }
    }
}
```

Sample Test Cases:

Test Case 1
Input:
Bob 15 3
Alice 8 4
MP5 3
AK47 4

Output:
Now fighting: Bob VS Alice
Skill level of Bob: 3
Skill level of Alice: 4
Energy level of Bob: 15
Energy level of Alice: 8

Bob makes an attack by MP5!
Alice takes a hurt amount of 6! Remaining energy becomes 2.
Alice makes an attack by AK47!
Bob takes a hurt amount of 7! Remaining energy becomes 8.
Bob makes an attack by MP5!
Alice hides from the attack!
Alice makes an attack by AK47!
Bob takes a hurt amount of 7! Remaining energy becomes 1.
Bob boost the MP5!
Bob makes an attack by MP5!
Alice takes a hurt amount of 9! Remaining energy becomes -7.
Bob wins! The examination paper is secured!

Test Case 2
Input:
Secure 20 4
Naughty 30 6
F15 3
MiG25 4

Output:
Now fighting: Secure VS Naughty
Skill level of Secure: 4
Skill level of Naughty: 6
Energy level of Secure: 20
Energy level of Naughty: 30

Secure makes an attack by F15!
Naughty takes a hurt amount of 7! Remaining energy becomes 23.
Naughty makes an attack by MiG25!
Secure takes a hurt amount of 9! Remaining energy becomes 11.
Secure makes an attack by F15!
Naughty hides from the attack!
Naughty makes an attack by MiG25!
Secure takes a hurt amount of 9! Remaining energy becomes 2.
Secure boost the F15!
Secure makes an attack by F15!
Naughty takes a hurt amount of 10! Remaining energy becomes 13.
Naughty makes an attack by MiG25!
Secure takes a hurt amount of 9! Remaining energy becomes -7.
Naughty wins! The examination paper is stolen!

Grading & Submission

Marking

For both parts,

80% marks are given by the auto-grader for the **correctness of program output**.

- You may assume that the program input is always valid.
- You program output must exactly match to what is described in this document. Any mismatch, such as additional or missing space character, will lead to 0 mark.

20% marks are given by the TAs manually to your **JavaDoc.**

- A complete JavaDoc includes documentation of every classes, member fields and methods that are not private. JavaDoc for the main method may be omitted.
- You don't need to submit JavaDoc htmls. Just write the doc comment blocks in your source program.

You will get 0 mark if:

- You submit .class files instead of .java source files, or
- You submit java source files that cannot be compiled, or
- You submit java source files that is not compliable or executable with the given **OfficeCombat1** and **OfficeCombat2** class.
- You submit java source files that are downloaded from the Internet, or
- You submit java source files from your classmates, or
- You submit java source files from friends taken this course last year.

You should never change the content in the **OfficeCombat1** and **OfficeCombat2** class. Or else, 50% mark penalty will be given to your assignment's total score!

Submission & Evaluation

For Part 1, please submit the following to Moodle VPL and evaluate:

- Character.java
- Weapon.java

For Part 2, please submit the following to Moodle VPL and evaluate:

- Character.java
- Weapon.java
- SecurityGuard.java
- Student.java
- BadGun.java
- SuperGun.java

Note the following:

- Double check your submission. Please check the assignment page again after submission to see all files you have submitted.
- You must evaluate your program before the assignment due date to get marks. **Late submission / evaluation is not allowed!**
- If you encounter any technical problems, please contact us as soon as possible
- We will use more test cases to test your program outputs against standard outputs after the submission deadline.