BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

K. K. BIRLA GOA CAMPUS

FIRST SEMESTER 2023-2024

CS F213 (Object Oriented Programming) - Lab 7 (Take Home)

Important Guidelines:

- Do not modify Driver.java. Uncomment the corresponding section in Driver.java after completing each class.
- Compilation of Driver: javac Driver.java
- Execution/running: java Driver
- Follow the sequence of completion as specified below.
- **Avoid hardcoding values**: your code will be evaluated using different versions of the driver code.

Problem Statement

The provided ZIP folder contains 13 Java files. The Driver.java file is fully implemented. You are required to complete the remaining 12 files. The specifications for each class are outlined below:

1. **Point2D**
    - Represents a coordinate point in a 2D plane.
    - Implement:
        - Getters, setters, a constructor, and a `move` method.
2. **Shape**
    - Represents a general shape.
    - Implement:
        - A constructor, getters, and a `move` method (overloading required).
3. **Circle**
    - Represents a circle in a 2D plane.

- Implement:
  - `withinCircle` method: Returns true for points inside or on the circle and false for points outside it.
  - Getters, setters, constructors, `getArea`, and `getPerimeter` methods.

4. **Card**
   - Represents a general object/card in the game.
   - Fields:
     - Circle range: Defines the object's range. The object is located at the center of the circle and can interact with other objects only within this range.
     - `health`: The health level may increase or decrease based on interactions. An object is considered dead when health reaches zero.
     - `level`: Indicates the object's level, starting from 0. The level can increase over time.
     - `damage`: An integer array representing the damage caused at each level.
   - Methods:
     - `incLevel`: Increases the object's level, ensuring it does not exceed the maximum level.
     - `withinRange(Card obj)`: Returns true if `obj` is within range; otherwise, returns false.
     - `isDead`: Returns true if the object is dead; otherwise, returns false.
     - `getDamage`: Returns the damage inflicted by the object at its current level.
     - `takeDamage`: Handles damage when the object is attacked.
5. **HogRider**
   - A character in the game.

    - Complete the `attack` method:
        - The HogRider can attack any object within its range.
        - The HogRider's level should increase after every 5 successful attacks.
        - Print "Out of Range" if an object is not within range.
6. **Prince**
    - A character in the game.
    - Complete the `attack` method:
        - The Prince can attack any object within its range.
        - The Prince's level should increase after every 2 successful attacks.
        - Print "Out of Range" if an object is not within range.
7. **DarkPrince**
    - A special type of Prince that deals twice the damage per attack as a normal Prince.
    - All other attributes and behaviors are the same as the standard Prince.
8. **Wizard**
    - A character in the game.
    - Complete the `attack` method:
        - The Wizard can attack any object within its range.
        - The Wizard's level should increase after every 2 successful attacks.
        - Print "Out of Range" if an object is not within range.
9. **IceWizard**
    - A special type of Wizard that deals twice the damage per attack as a normal Wizard.
    - All other attributes and behaviors are the same as the standard Wizard.
10. **Fireball**
    - A defense building in the game.
    - Complete the `attack(Card obj)` method:

        - The Fireball can attack any object except another Fireball. If it attempts to do so, print "Attack Failed".
        - If the object is out of range, print "Out of Range".
        - Only one message should be printed for each attack attempt, starting with the "Attack Failed" condition.
        - The Fireball's level should increase after every 4 successful attacks.
11. **WizardFireball**
    - Represents a defense structure in the game that combines the abilities of a Wizard and a Fireball.
    - Characteristics:
        - Both the Wizard and the Fireball attack separately. The Wizard behaves like a normal Wizard.
        - Use the `WizardAttack` method to attack other objects using the Wizard.
        - The number of attacks and the level of the Fireball and the Wizard are separate.
    - Methods:
        - `isWizardDead()`: Should return true if the Wizard is dead. The Wizard may die as a normal Wizard dies or when the Fireball dies. However, while the Fireball can live when the Wizard is dead, the Wizard cannot live without the Fireball.
        - Complete the `attack(Card obj)` method for the WizardFireball:
            - The WizardFireball can attack any object except another WizardFireball. If it attempts to do so, print "Attack Failed".
            - If the object is out of range, print "Out of Range".
            - Only one message should be printed for each attack attempt, starting with the "Attack Failed" condition.

        - The Fireball's level should increase after every 4 successful attacks, while the Wizard's level increases according to its own rules.

12. **Healer**
    - An object in the game that heals other objects (e.g., Wizard).
    - The damage array represents negative values used for healing.
    - Implement the `attack` method:
        - If the Healer attempts to heal a Fireball or another Healer, print "Heal Failed".
        - If the Wizard is out of range, print "Out of Range".
        - The Healer's level should increase after every 3 successful heals.

Expected Output for given Driver:

```
^[aditya@Aditya:/mnt/c/Users/adity/Downloads/Aditya Bagla/OOP-LAB/Solution Key$ javac Driver.java
aditya@Aditya:/mnt/c/Users/adity/Downloads/Aditya Bagla/OOP-LAB/Solution Key$ java Driver

Hello! Welcome to First Take Home Lab.
This is a test driver for the classes you have to implement in this lab.
Implement each class and uncomment and check your marks.
Please complete it step by step, the marks will be calculated sequentially.
Point2D is correct, Proceed to the Shape class.
Shape is correct, Proceed to the Circle class.
Circle is correct, Proceed to the Card class.
Card is correct, Proceed to the HogRider class.
Out of Range
Max attacks done
Max attacks done
Max attacks done
Max attacks done
Max attacks done
HogRider is correct, Proceed to the Prince class.
If your marks are 5 till here then you are on the right track.
Out of Range
Prince is correct, Proceed to the DarkPrince class.
Out of Range
DarkPrince is correct, Proceed to the Wizard class.
Out of Range
Wizard is correct, Proceed to the IceWizard class.
Out of Range
IceWizard is correct, Proceed to the Fireball class.
Attack Failed
Out of Range
Out of Range
Fireball is correct, Proceed to the Healer class.
Out of Range
Out of Range
Healer is correct. Congratulations Lab is over.


Congratulations on completing the lab
Marks: 15/15
Make sure that you have not hardcoded any value as we will be evaluating on a different driver code.
aditya@Aditya:/mnt/c/Users/adity/Downloads/Aditya Bagla/OOP-LAB/Solution Key$
```

Create submission zip file:

- Place all files in a single folder.
- Zip the folder and name it <ID_No.>_Lab7.zip

Ensure you follow these instructions carefully to receive full credit