ECS705/ECS717

Lab Sheet 6: : Inheritance, Abstract Classes, Polymorphism and Interfaces

---

Essential exercises:

This year's lab exercises are based on the game Jelly Defense (for IPhone/Andriod). Note you do not have to know anything about the game to do the assignment ☺.

Exercise 34.

Download the file `Jelly.java` from the course webpage. This code is not very object-oriented. It does not support information hiding. Rewrite the code so that it uses oo principles (including inheritance) to represent the different kinds of Jellies and their attack capabilities. Also provide a method: `boolean isAlive()`, that checks to see if a Jelly is alive or dead (health of 0 indicates a Jelly is dead).

Exercise 35.

Download the files `Tower.java, BlueRedTower.java, BlueTower.java,` RedTower.java and TowerTest.java from the course webpage. Compile and run the files. The output from printing the towers is not very user friendly. It prints something along the lines of:

```
Tower@50a9ae05
BlueRedTower@33dff3a2
BlueTower@33f42b49
BlueTower@6345e044
RedTower@86c347
```

Fix this so it prints the following:

```
Generic Tower : Level 1
Blue Red Tower : Level 1
Blue Tower : Level 2
Blue Tower : Level 4
Red Tower : Level 1
```

You may not alter `TowerTest.java`.
Hint: Investigate in the javadocs what println or print actually does when called with an object.

Exercise 36.

Download the files `Tower.java`, `BlueRedTower.java`, `BlueTower.java`, `RedTower.java` and `TowerTest.java` from the course webpage. In the game, there is no such thing as a "generic" tower. Fix the classes so that no client (i.e. `TowerTest.java` or any other test program) can make instances of Towers. Add a method to Tower : int attackJelly(). This method should have no implementation in Tower, but should force all child classes to implement it. Attack Jelly should have the following output:

```
Blue Red Tower attacks Jelly for 1 point damage.
Blue Tower attacks Jelly for 2 points damage.
Blue Tower attacks Jelly for 2 points damage.
Red Tower attacks Jelly for 2 points damage.
```

Desirable exercises:

Exercise 37.

Download the files `Tower.java`, `BlueRedTower.java`, `BlueTower.java`, `RedTower.java` and `TowerCompare.java` from the course webpage. Using the `Comparable` interface, fix the code so it compiles and runs. Towers should be sorted on the basis of Level: smallest to largest.

Optional exercises:

Exercise 38.

Using your classes from Exercise 34 and Exercise 36. Rewrite `int attackJelly()` so that it takes in a Jelly. All tower types can attack all Jellies with the following rules:

Tower Type	Blue Jelly	Red Jelly
Blue Tower	Level 1: 2-5 points damage Level 2: 5-9 points damage Level 3: 9-12 points damage Level 4: 12-15 points damage	Level 1: 0 points damage Level 2: 1 points damage Level 3: 2 points damage Level 4: 3 points damage
Red Tower	Level 1: 0 points damage Level 2: 1 points damage Level 3: 2 points damage Level 4: 3 points damage	Level 1: 2-5 points damage Level 2: 5-9 points damage Level 3: 9-12 points damage Level 4: 12-15 points damage
BlueRedTower	Level 1: 2 points damage Level 2: 2-4 points damage Level 3: 4-6 points damage Level 4: 6-8 points damage	Level 1: 2 points damage Level 2: 2-4 points damage Level 3: 4-6 points damage Level 4: 6-8 points damage

Create a new test file TowerAttack.java that creates an ArrayList of 10 different types of Jellies, an ArrayList of 5 different towers of random powerlevels. After that you allow all the towers to attack the Jellies until they are dead.