Lab Statement

Aviral Agrawal, Sarthak Moorjani, Shubhi Jain

September 2018

Disclaimer: It is all purely nonfictional. Any resemblance to anything or anyone real or imagined must be taken as a real failure of imagination.

1 Introduction

As mentioned in the prelab statement, we want you to construct a colony of houses. Although the javadoc has sufficient information to complete the lab, this document will help you understand the design better. So before making the colony of houses, we will construct a house.

Our house is a combination of a rectangular base pyramid over a base. The base is a collection of rooms(cuboids). Further a cuboid is made up of 6 rectangular walls and the pyramid has 4 triangular walls. Hence this defines the structure of the classes. We will have something like this -

```
class House{
    // Field and Methods
    ... ... ...
    ... ... ...
    class Cuboid{
        //Fields and Methods
        ... ... ...
        ... ... ...
        class Rectangle{
        //Fields and Methods
        }
    }
    class Pyramid{
        //Fields and Methods
        ... ... ...
        class Triangle{
        //Fields and Methods
        }
    }
}
```

2 House Class

Now let us talk something about the internal design. We have the following attributes in the House Class.

- `baseLength, baseWidth and height` denote the dimensions of the base and will be provided in the constructor. Note that the height is same for all the rooms.
- `pyramidHeight` denotes the height of the pyramid. This is passed in the constructor of the `House` class only.
- `houseType` is an important attribute. Houses can be of 2 types, either residential or commercial. Depending on the `houseType`, the revenue from the house will be generated. 'r' denotes residential House and 'c' denotes commercial house. If the house is commercial then, the revenue generated by that house is the sum of costs of each room(given in the Cuboid Constructor) otherwise it will be the total surface area of the house.

Method Details are elaborated in the java doc. ALso, it is possible that the base of the house might not be constructed fully. That is the volume of the base may not equal the sum of volume of the rooms. Infact we want you to create a method which actually calculates the freeSpace in the house. Also, you will need to read about the inner classes also to understand completely what the methods are supposed to do.

3 Cuboid Class

A cuboid essentially represents a room. A room is made up of 6 walls(Rectangle). Also, a room may have a cost associated with it which depends on whether the house is commercial or residential. Remember that `roomHeight` is same as the `height` of the House which is passed in the constructor of the `House` class and not the `Cuboid` class. There are a few trivial methods in which not much has to be done. Other information is explicitly mentioned in the javadoc.

4 Rectangle Class

A rectangle essentially represents a wall. A wall has a `wallLength` and a `wallWidth` whose description is very clear in the javadoc provided. Since we are overwhelmed with the rainy season in Goa, we introduce a parameter `wallState` which denotes the current state of the wall. It may be 'g' (good) or 'b' (bad) depending on the `isRaining` parameter passed in the constructor. If `isRaining` is true, then the wall is damaged and is bad, and if `isRaining` is false, then the wall is good. **The most important part about the this class is the initialization of the parameters in the constructor. Read the javadoc properly.**

5 Pyramid Class and Triangle Class

The Pyramid Class and Triangle class are similar to the Cuboid and the Rectangle classes as defined above. Note that the base dimensions of the pyramid are same as the base dimensions of the house (obviously) and the height of the pyramid was passed as a parameter in the house class. (Mentioned above). Again for the Triangle class, **initialization of parameters in the constructors is the most important part. The details are there in the javadoc.**

NOTE: Every calculation is an integer calculation, DO NOT USE `float` OR `double` FOR CALCULATIONS.

6 About the test cases

We have tests which tell the following methods(In random order) -

- `getBadWalls()` ······ 1 mark
- `getCountRooms() and volume of the room` ······ 1 mark
- `freeSpace()` ······ 1 mark
- `getRent()` ······ 1 mark
- `getSurfaceArea() of the house` ······ 1 mark
- `addHouse()` ······ 1 mark
- `goodWalls() in Colony Class` ······ 2 marks
- `getRevenue() in Colony Class` ······ 2 marks

******ALL THE BEST******