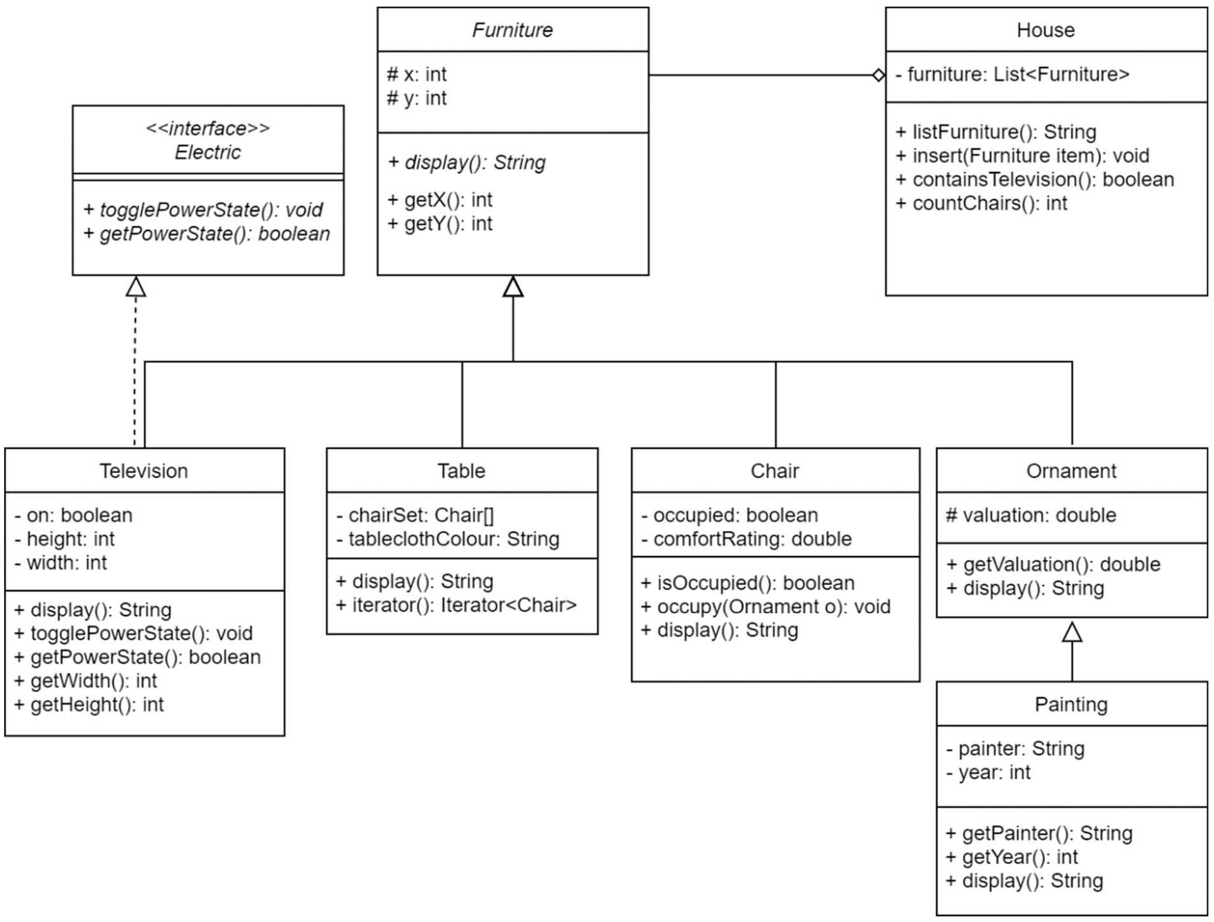
Task 3 - Furniture

Implement the classes based on the UML diagram:



There are 8 classes, here are the implementation details of each.

House

Constructor: `House()`

`listFurniture()`: Return a String that displays all of the furniture items in the house, for example:

```
This house contains:
A table with a white tablecloth
 - A chair with comfort rating 39.0
 - A chair with comfort rating 38.0
 - A chair with comfort rating 37.0
 - A chair with comfort rating 36.0
A 1920x1080 television in the off state.
A chair with comfort rating 36.0
A chair with comfort rating 36.0
A chair with comfort rating 36.0
An ornament worth $0.53
A painting by Van Gough painted in 1888, worth $4,699.00
```

If the house is empty, only return the first line

`insert(Furniture item):` Insert a furniture item into the list. You don't have to check if it already exists. If the item is null, don't add it.

`containsTelevision():` Return true if there is a Television in this house, otherwise false.

`countChairs()`: return the number of chairs in the house. Include chairs part of a table set.

Chair

Constructor: `Chair(int x, int y, double comfort)` - the comfort rating must be within the range [0,100]. If it's not, set it to be the closest value that is within the range, either 0 or 100.

`occupy(Ornament o)`: If the ornament's valuation is >= $100, then the chair is occupied by this ornament.

Table

Constructor: `Table(int x, int y, String colour, int chairs)` - the table comes with a set of Chairs that are created at the same coordinates as the table, with a comfort rating equal to the absolute value of the hashCode() of the String (colour+i), modulo 101, for the ith index chair in the table's set. For example if the table's colour is 'white', then for the first chair, take the absolute value of the hashCode() of 'white0' modulo 101. For the second chair, it would be the absolute value of the hashCode() of 'white1' modulo 101. hashCode() is a method of the String class that returns an integer that will be the same for equal string values.

`iterator()`: You need to make the Table class implement the Iterable<Chair> interface, and create an Iterator<Chair> that will iterate through the chairs in this table's set, like was shown in the week 7 tutorial.

Ornament

Constructor: `Ornament(int x, int y, double value)`

Painting

Constructor: `Painting(int x, int y, double value, String painter, int year)`

Television

Constructor: `Television(int x, int y, int width, int height)`

The television is initially off. If the `togglePowerState()` method is performed, it will change the power state to off if on, and on if off. In the display message, the width is displayed before the height in "<width>x<height>".

<u>Furniture</u>

This is an abstract class, you cannot create objects of it. You still need a constructor though, to pass the inherited variables to from subclass constructors. See week 6 abstract classes.

`display()` is an abstract method that returns a String with details about the furniture item. See the example of the furniture listing for what each furniture item should return. It should not have a newline at the end.

<u>Electric</u>

This is an interface, you cannot create objects of it.