Федеральное государственное автономное образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский
университет информационных технологий, механики и оптики»

Факультет Программной Инженерии и Компьютерной Техники

Дисциплина: Программирование

Лабораторная работа по программированию №3

Выполнил: Конаныхина Антонина Александровна
Группа: P3115
Вариант: 311544
Преподаватель: Письмак Алексей Евгеньевич

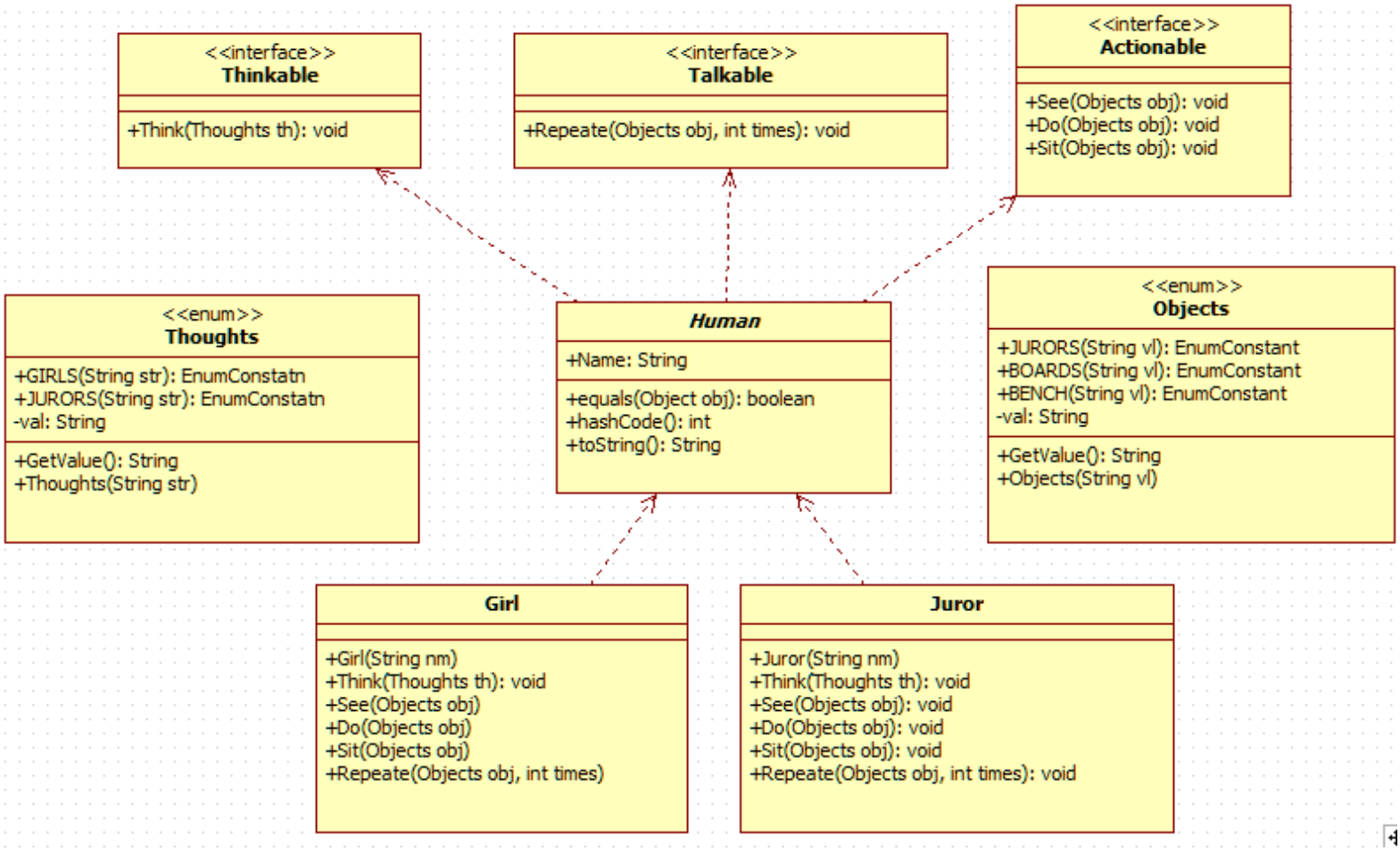
Санкт-Петербург, 2020г

На языке программирования Java разработать программу, реализующую объектную модель и последовательность действий в соответствии с заданным описанием предметной области. Результат работы программы должен выводиться в стандартный поток вывода.

Описание предметной области, по которой должна быть построена объектная модель:

Алиса не без гордости раза два-три повторила это слово. "Вряд ли много найдется девочек в моем возрасте, а то и старше,- подумала она,- которые слыхали такое слово и знают, что оно значит". Пожалуй, она была права, хотя слово "заседатели" было бы ничуть не хуже. Присяжные сидели на большой скамье, стоявшей на возвышении ("Это скамья присяжных. В скамейке то все и дело, кто на нее присел, тот и присяжный",- подумала Алиса). У всех у них были грифельные доски, и все они что-то деловито записывали.

UML-схема:



Код:

```
public class Main
{
    public static void main(String[] args)
    {
        Girl Alice = new Girl("Алиса");
        Juror Jurs = new Juror("Присяжные");
        Alice.Repeate(Objects.JURORS_WORDS, 3);
        Alice.Think(Thoughts.GIRLS);
        Alice.See(Objects.JURORS);
        Jurs.Sit(Objects.BENCH);
        Alice.Think(Thoughts.JURORS);
        Jurs.Do(Objects.BOARDS);
        //Girl act1 = new Girl("Алис", nar);
        //SeeAction act2 = new SeeAction();
        //System.out.println(nar.equals(Alice));
    }

    @Override
    public boolean equals(Object obj) {
        return super.equals(obj);
    }
}


public interface Actionable
{
    public void See(Objects obj);
    public void Do(Objects obj);
    public void Sit(Objects obj);
}

public interface Talkable
{
    public void Repeate(Objects obj, int times);
}

interface Thinkable
{
   public void Think(Thoughts th);
}


public abstract class Human implements Thinkable, Actionable, Talkable
{

   String Name;


   @Override
   public boolean equals(Object obj) {
      if (this.getClass() != obj.getClass())
         return false;
      Human others = (Human) obj;
      return this.Name == others.Name;
   }

   public int hashCode() {
      return this.Name.hashCode();
   }

   public String toString() {
      return this.Name;
```

```
    }
}




public class Juror extends Human
{
    public Juror(String nm)
    {
        Name = nm;
    }

    //String Name;

    @Override

    public void Think(Thoughts th)
    {
        System.out.println(Name + " подумали " + th.GetValue());
    }
    public void See(Objects obj)
    {
        System.out.println(Name + " увидели " + obj.GetValue());
    }
    public void Do(Objects obj)
    {
        System.out.println(Name + " " + obj.GetValue());
    }
    public void Sit(Objects obj)
    {
        System.out.println(Name + " сидели на " + obj.GetValue());
    }
    public void Repeate(Objects obj, int times)
    {
        for (int i = 0; i < times; i++) {
            System.out.println(Name + " повторили " + obj.GetValue());
        }
    }
}




public class Girl extends Human
{
    public Girl(String nm)
    {
        Name = nm;
    }

    //String Name;

    @Override

    public void Think(Thoughts th)
    {
        System.out.println(Name + " подумала " + th.GetValue());
    }
    public void See(Objects obj)
    {
        System.out.println(Name + " увидела " + obj.GetValue());
    }
```

```
    public void Do(Objects obj)
    {
        System.out.println(Name + " " + obj.GetValue());
    }
    public void Sit(Objects obj)
    {
        System.out.println(Name + " села на " + obj.GetValue());
    }
    public void Repeate(Objects obj, int times)
    {
        for (int i = 0; i < times; i++) {
            System.out.println(Name + " повторила слово " + obj.GetValue());
        }
    }
}


public enum Objects {
     JURORS("присяжных"),
     JURORS_WORDS("присяжные"),
     BOARDS("пишут на грифельных досках"),
     BENCH("скамейке");

     private String val;
     Objects(String vl)
     {
         val = vl;
     }
     public String GetValue()
     {
         return val;
     }
}



enum Thoughts
{
    GIRLS("Вряд ли много найдется девочек в моем возрасте, а то и старше,
которые слыхали такое слово и знают, что оно значит"),
    JURORS("Это скамья присяжных. В скамейке то все и дело, кто на нее присел,
тот и присяжный");
    private String val;
    Thoughts(String str)
    {
        val = str;
    }
    String GetValue()
    {
        return val;
    }
}
```

Вывод программы:

Алиса повторила слово присяжные

Алиса повторила слово присяжные

Алиса повторила слово присяжные

Алиса подумала Вряд ли много найдется девочек в моем возрасте, а то и старше, которые слыхали такое слово и знают, что оно значит

Алиса увидела присяжных

Присяжные сидели на скамейке

Алиса подумала Это скамья присяжных. В скамейке то все и дело, кто на нее присел, тот и присяжный

Присяжные пишут на грифельных досках

Вывод:

В процессе выполнения лабораторной работы были получены навыки использования объектно-ориентированного подхода программирования, изучены методы SOLID и STUPID. С каждой лабой всё больше убеждаюсь, что Java страшнее, чем кажется ☹