МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

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

ЛАБОРАТОРНАЯ РАБОТА № 3

по дисциплине

“ПРОГРАММИРОВАНИЕ”

Вариант № 49050

Выполнила:
Студент группы R3138
Аракчиева Ева
Дмитриевна
Преподаватели:
Харитонова Анастасия
Евгеньевна
Письмак Алексей
Евгеньевич

Университет ИТМО
Санкт-Петербург
2021

Задание:

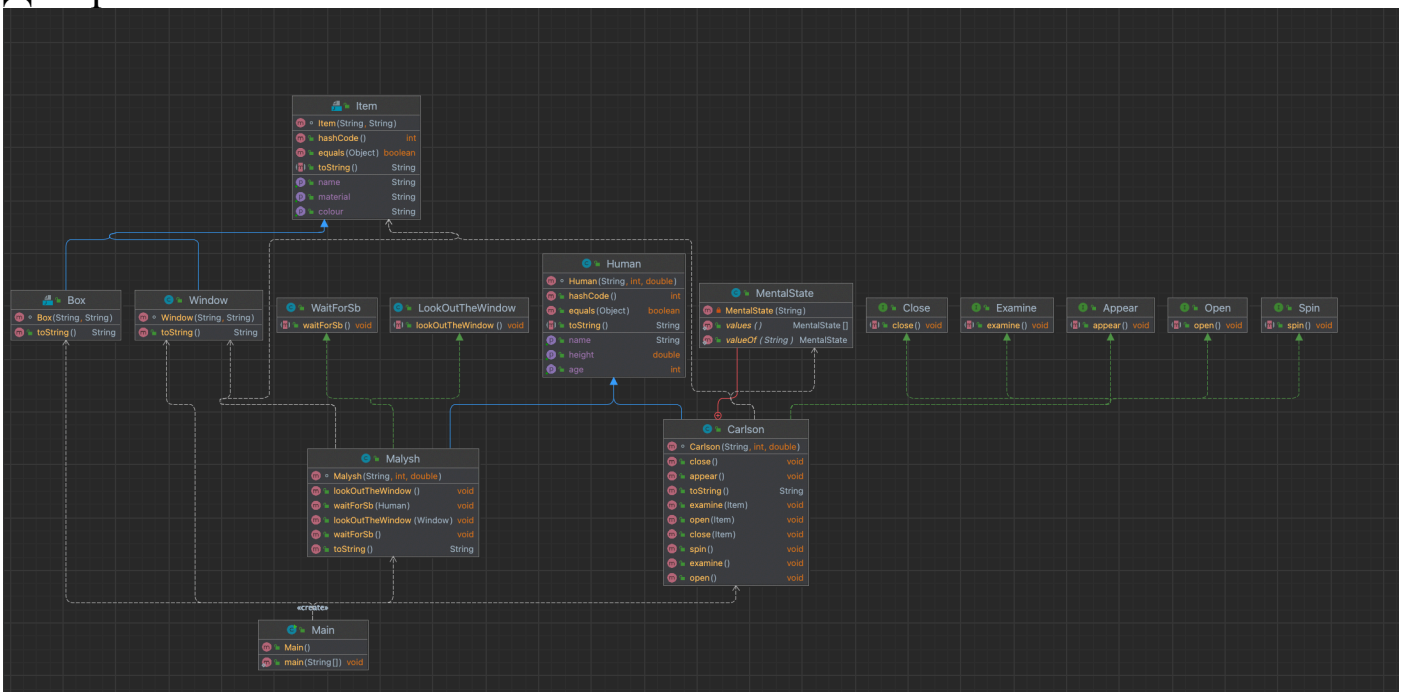
Описание предметной области, по которой должна быть построена объектная модель:

> Должно быть, Карлсон почувствовал, что Малыш его ждет: едва Малыш высунул нос в окошко, как Карлсон уже был тут как тут. Карлсон ни минуты не стоял на месте. Разговаривая, он все время кружил по комнате, трогал все, что попадалось под руку, с любопытством открывал и закрывал ящики и разглядывал каждую вещь с большим интересом.

Программа должна удовлетворять следующим требованиям:

1. Доработанная модель должна соответствовать принципам SOLID.
2. Программа должна содержать как минимум два интерфейса и один абстрактный класс (номенклатура должна быть согласована с преподавателем).
3. В разработанных классах должны быть переопределены методы `equals()`, `toString()` и `hashCode()`.
4. Программа должна содержать как минимум один перечисляемый тип (enum).

Диаграмма:


Item
Item(String, String)
hashCode() int
equals(Object) boolean
toString() String
name String
material String
colour String
Box
Box(String, String)
toString() String
Window
Window(String, String)
toString() String
WaitForSb
waitForSb() void
LookOutTheWindow
lookOutTheWindow() void
Human
Human(String, int, double)
hashCode() int
equals(Object) boolean
toString() String
name String
height double
age int
MentalState
MentalState(String)
values() MentalState[]
valueOf(String) MentalState
Close
close() void
Examine
examine() void
Appear
appear() void
Open
open() void
Spin
spin() void
Malysh
Malysh(String, int, double)
lookOutTheWindow() void
waitForSb(Human) void
lookOutTheWindow(Window) void
waitForSb() void
toString() String
Carlson
Carlson(String, int, double)
close() void
appear() void
toString() String
examine(Item) void
open(Item) void
close(Item) void
spin() void
examine() void
open() void
«create»
Main
Main()
main(String[]) void

Исходный код:

Main.java | Human.java

```java
package Story;

public class Main {
    public static void main(String[] args) {
        Malysh baby = new Malysh( name: "Малыш", age: 7, height: 140.0D);
        Carlson carl = new Carlson( name: "Карлсон", age: 36, height: 100.0D);
        Window window1 = new Window ( name: "окошко", material: "пластмасса");
        Box box = new Box( name: "ящик", material: "дерево");
        baby.waitForSb(carl);
        baby.lookOutTheWindow(window1);
        carl.appear();
        carl.spin();
        carl.open(box);
        carl.examine(box);
        carl.close(box);
    }
}
```

Human › equals

Main.java | Appear.java | Box.java | Malysh.java | Human.java

```java
import java.util.Objects;

public abstract class Human {
    private String name;
    private int age;
    private double height;

    Human(String name, int age, double height) {
        this.name = name;
        this.age = age;
        this.height = height;
    }

    public String getName() { return this.name; }

    public int getAge() { return this.age; }

    public double getHeight() { return this.height; }

    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        } else if (obj != null && obj.getClass() == this.getClass()) {
            Human person = (Human)obj;
            return Objects.equals(this.name, person.name) && Objects.equals(this.age, person.age) && Objects.equals(this.height, person.height);
        } else {
            return false;
        }
    }

    public abstract String toString();

    public int hashCode() {
        return Objects.hash(new Object[]{this.name, this.age, this.height});
    }
}
```

Story 〉 Item

Main.java | Appear.java | Box.java | Malysh.java | Human.java | Item.java

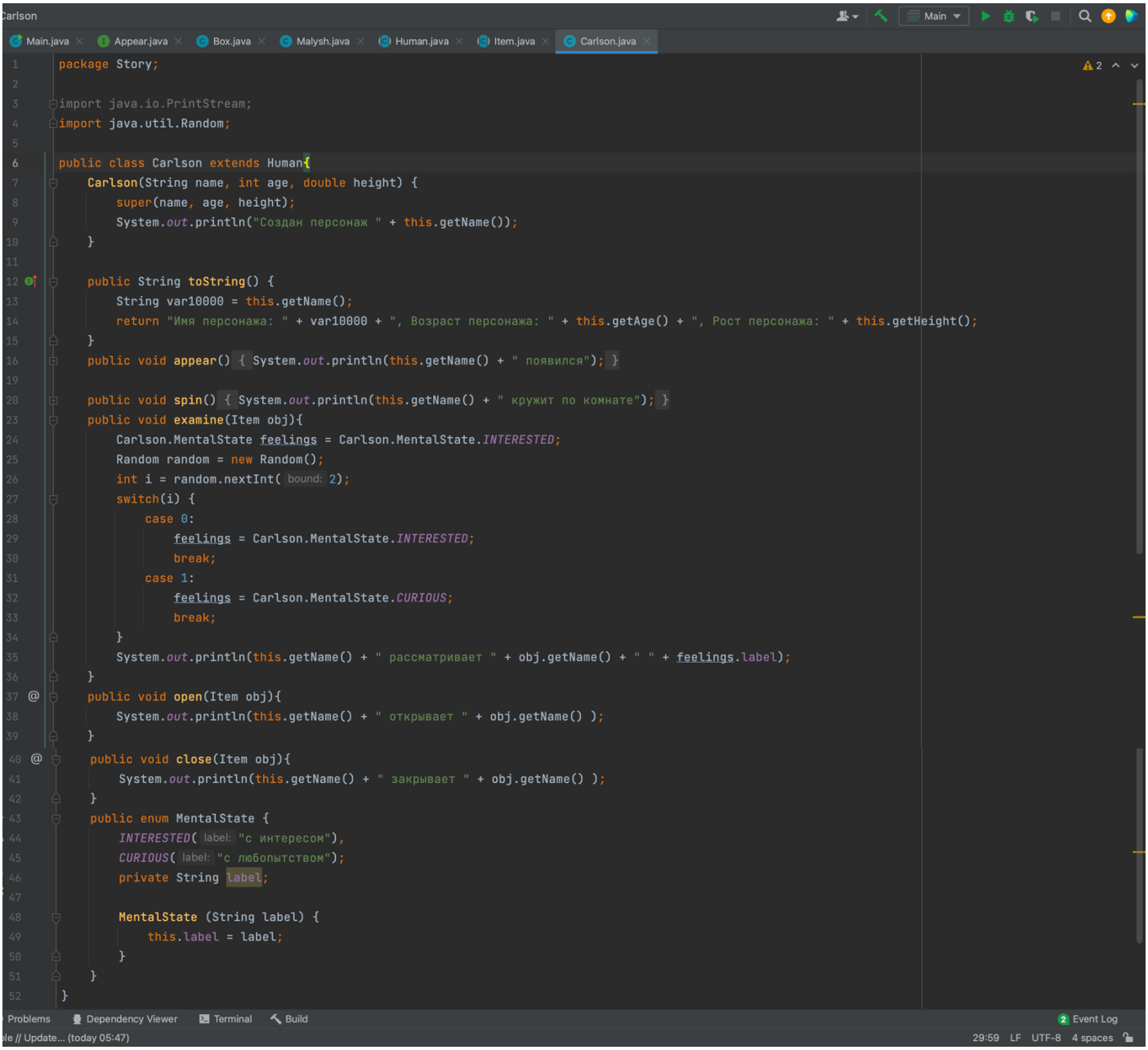
```java
package Story;

import java.util.Objects;

public abstract class Item {
    private String name;
    private String material;
    private String colour;

    Item(String name, String material) { this.name = name; }

    public String getName() { return this.name; }

    public String getMaterial() { return this.material; }

    public String getColour() { return this.colour; }



    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        } else if (obj != null && obj.getClass() == this.getClass()) {
            Item thing = (Item)obj;
            return Objects.equals(this.name, thing.name) && Objects.equals(this.material, thing.material) && Objects.equals(this.colour, thing.colour);
        } else {
            return false;
        }
    }

    public abstract String toString();

    public int hashCode() { return Objects.hash(new Object[]{this.name, this.material, this.colour}); }
}
```

Lab33 – Malysh.java

Malysh 〉 lookOutTheWindow

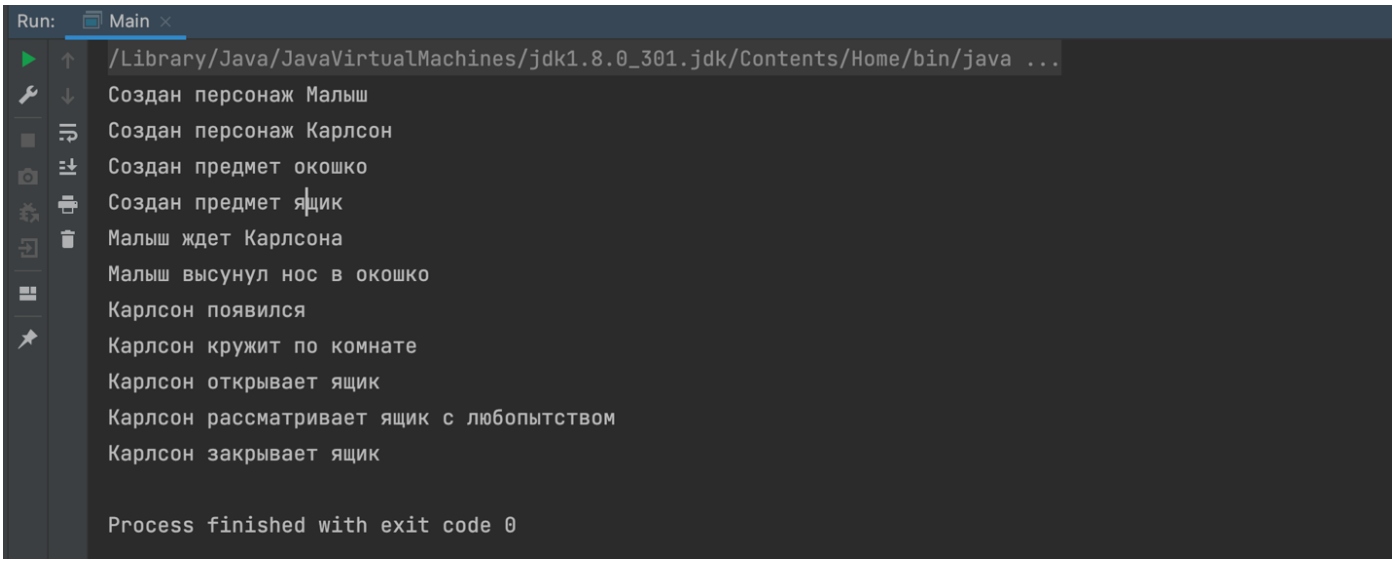
Main.java | Appear.java | Box.java | Malysh.java | Human.java | Item.java | Carlson.java

```java
package Story;

import java.io.PrintStream;

public class Malysh extends Human implements WaitForSb, LookOutTheWindow {
    Malysh(String name, int age, double height) {
        super(name, age, height);
        System.out.println("Создан персонаж " + this.getName());
    }

    public String toString() {
        String var10000 = this.getName();
        return "Имя персонажа: " + var10000 + ", Возраст персонажа: " + this.getAge() + ", Рост персонажа: " + this.getHeight();
    }

    public void waitForSb (Human obj) { System.out.println(this.getName() + " ждет " + obj.getName() + "а"); }
    public void lookOutTheWindow(Window obj) { System.out.println(this.getName() + " высунул нос в " + obj.getName()); }

    @Override
    public void waitForSb() { }

    @Override
    public void lookOutTheWindow() { }
}
```



```java
package Story;

import java.io.PrintStream;
import java.util.Random;

public class Carlson extends Human{
    Carlson(String name, int age, double height) {
        super(name, age, height);
        System.out.println("Создан персонаж " + this.getName());
    }

    public String toString() {
        String var10000 = this.getName();
        return "Имя персонажа: " + var10000 + ", Возраст персонажа: " + this.getAge() + ", Рост персонажа: " + this.getHeight();
    }
    public void appear() { System.out.println(this.getName() + " появился"); }

    public void spin() { System.out.println(this.getName() + " кружит по комнате"); }
    public void examine(Item obj){
        Carlson.MentalState feelings = Carlson.MentalState.INTERESTED;
        Random random = new Random();
        int i = random.nextInt( bound: 2);
        switch(i) {
            case 0:
                feelings = Carlson.MentalState.INTERESTED;
                break;
            case 1:
                feelings = Carlson.MentalState.CURIOUS;
                break;
        }
        System.out.println(this.getName() + " рассматривает " + obj.getName() + " " + feelings.label);
    }
    public void open(Item obj){
        System.out.println(this.getName() + " открывает " + obj.getName() );
    }
    public void close(Item obj){
        System.out.println(this.getName() + " закрывает " + obj.getName() );
    }
    public enum MentalState {
        INTERESTED( label: "с интересом"),
        CURIOUS( label: "с любопытством");
        private String label;

        MentalState (String label) {
            this.label = label;
        }
    }
}
```

Результат программы:



```
Run:  Main
/Library/Java/JavaVirtualMachines/jdk1.8.0_301.jdk/Contents/Home/bin/java ...
Создан персонаж Малыш
Создан персонаж Карлсон
Создан предмет окошко
Создан предмет ящик
Малыш ждет Карлсона
Малыш высунул нос в окошко
Карлсон появился
Карлсон кружит по комнате
Карлсон открывает ящик
Карлсон рассматривает ящик с любопытством
Карлсон закрывает ящик

Process finished with exit code 0
```

Вывод:
В результате выполнения лабораторной работы я узнала о принципах объектно-ориентированного программирования SOLID и STUPID, особенностях реализации наследования в Java, классе Object и его методах equals(), toString() и hashCode(), познакомилась с понятиями абстрактного класса и интерфейса.