**ФЕДЕРАЛЬОНЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗАВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**“НАЦИОНАЛЬНЫЙ ИСЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО”**

Факультет Программной Инженерии и Компьютерной Техники

Дисциплина: «Программирование»

**ОТЧЁТ**

по лабораторной работе №3  
Вариант №777

Выполнил:

Студент группы P3111

Дорохин Сергей Константинович

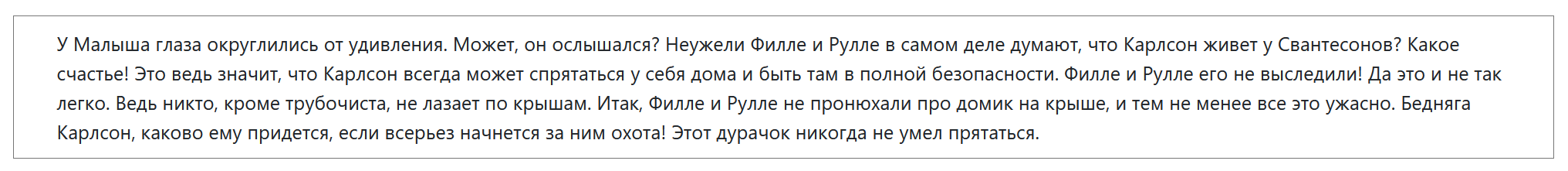
Проверил:

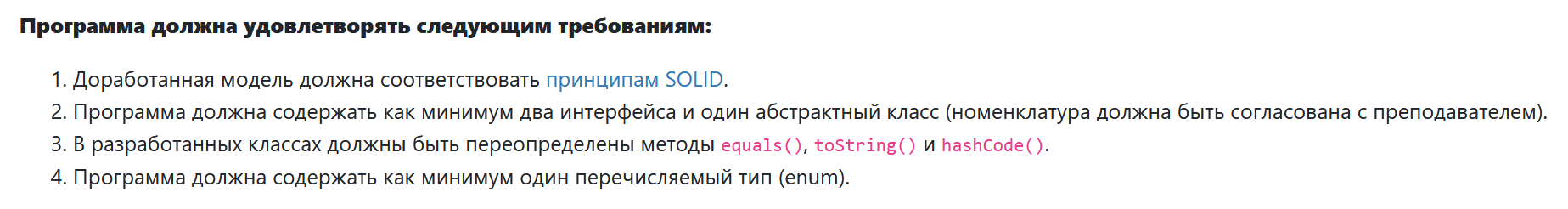
Письмак Алексей Евгеньевич

Санкт-Петербург

2023 г.

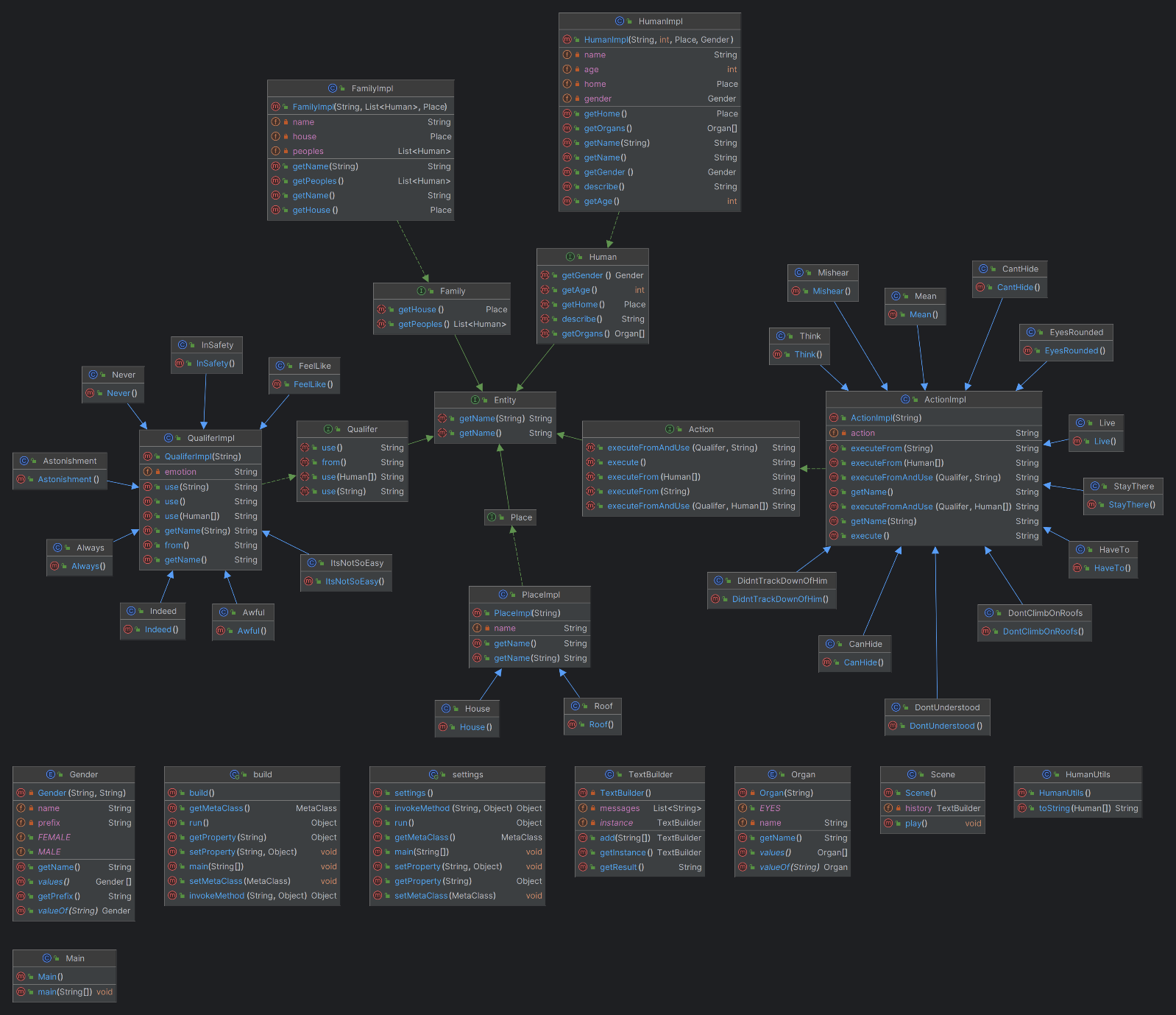
**Задание**

****

****

**Код Программы**

**Диаграмма**



**MyScene.java**

package com.serezka.scene;  
  
import com.serezka.Scene;  
import com.serezka.scene.entities.action.list.\*;  
import com.serezka.scene.entities.family.list.Svantesoni;  
import com.serezka.scene.entities.human.Organ;  
import com.serezka.scene.entities.human.list.\*;  
import com.serezka.scene.entities.place.list.CarlsonRoof;  
import com.serezka.scene.entities.qualifers.list.\*;  
  
import java.util.Objects;  
  
public class MyScene extends Scene {  
 private final TextBuilder history = TextBuilder.getInstance();  
  
 @Override  
 public String build() {  
 history.add(Kid.getInstance().getName("У"))  
 .add(new EyesRounded().executeFrom(Organ.EYES.getName()))  
 .add(new Astonishment().getName("от"))  
 .add(".");  
  
 history.add("Может,")  
 .add(new Mishear().executeFrom(Kid.getInstance().getGender().getPrefix()))  
 .add("?");  
  
 history.add("Неужели")  
 .add(new Think().executeFromAndUse(new Indeed(), Fille.getInstance(), Rulle.getInstance()))  
 .add(",").add("что")  
 .add(new Live().executeFrom(Carlson.getInstance())).add(Svantesoni.getInstance().getName("у"))  
 .add("?");  
  
 history.add("Какое счастье")  
 .add("!");  
  
 history.add(new Mean().executeFrom("Это"))  
 .add(",").add("что")  
 .add(new CanHide().executeFromAndUse(new Always(), Carlson.getInstance())).add("у себя дома")  
 .add("и")  
 .add(new StayThere().execute()).add(new InSafety().use())  
 .add(".");  
  
 history.add(new DidntTrackDownOfHim().executeFrom(Fille.getInstance(), Rulle.getInstance()))  
 .add("!");  
  
 history.add(new ItsNotSoEasy().use(), ".");  
  
 history.add("Ведь никто,")  
 .add(Sweep.getInstance().getName("кроме"))  
 .add(",", new DontClimbOnRoofs().execute(), ".");  
  
 history.add("Итак,")  
 .add(new DontUnderstood().executeFrom(Fille.getInstance(), Rulle.getInstance()))  
 .add(CarlsonRoof.getInstance().getName("про"), ",")  
 .add("и тем не менее").add(new Awful().getName("все это"))  
 .add(".");  
  
 history.add(Carlson.getInstance().getName("Бедняга"))  
 .add(",").add(new FeelLike().use()).add(new HaveTo().executeFrom("ему")).add(",")  
 .add("если всерьез начнется за ним охота").add("!");  
  
 history.add(new CantHide().executeFromAndUse(new Never(), "Этот дурачок"))  
 .add(".");  
  
 return history.getResult();  
 }  
  
 @Override  
 public String toString() {  
 return "Сцена {" +  
 "history=" + history +  
 '}';  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 MyScene myScene = (MyScene) o;  
 return Objects.equals(history, myScene.history);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(history);  
 }  
}

**Entity.java**

package com.serezka.scene.entities;  
  
public interface Entity {  
 String getName();  
 String getName(String prefix);  
}

**Human.java**

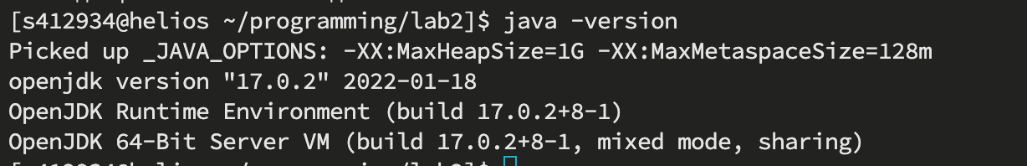
package com.serezka.scene.entities.human;  
  
import com.serezka.scene.entities.Entity;  
import com.serezka.scene.entities.place.Place;  
  
public interface Human extends Entity {  
 String describe();  
  
 Place getHome();  
 Gender getGender();  
 int getAge();  
  
 Organ[] getOrgans();  
}

**HumanImpl.java**

package com.serezka.scene.entities.human;  
  
import com.serezka.scene.entities.place.Place;  
import lombok.AccessLevel;  
import lombok.RequiredArgsConstructor;  
import lombok.experimental.FieldDefaults;  
  
import java.util.Objects;  
  
@FieldDefaults(level = AccessLevel.PRIVATE, makeFinal = true)  
@RequiredArgsConstructor  
public class HumanImpl implements Human {  
 String name;  
 int age;  
  
 Place home;  
 Gender gender;  
  
 @Override  
 public String describe() {  
 return String.format("%s, %s, %d лет, живет в %s",  
 this.name, this.gender.getName(), this.age, this.home.getName());  
 }  
  
 @Override  
 public Place getHome() {  
 return this.home;  
 }  
  
 @Override  
 public Gender getGender() {  
 return this.gender;  
 }  
  
 @Override  
 public int getAge() {  
 return this.age;  
 }  
  
 @Override  
 public Organ[] getOrgans() {  
 return Organ.values();  
 }  
  
 @Override  
 public String getName() {  
 return this.name;  
 }  
  
 @Override  
 public String getName(String prefix) {  
 return String.join(" ", prefix, this.name);  
 }  
  
 @Override  
 public String toString() {  
 return "Человек {" +  
 "name='" + name + '\'' +  
 ", age=" + age +  
 ", home=" + home +  
 ", gender=" + gender +  
 '}';  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 HumanImpl human = (HumanImpl) o;  
 return age == human.age && Objects.equals(name, human.name) && Objects.equals(home, human.home) && gender == human.gender;  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.hash(name, age, home, gender);  
 }  
}

**Остальной код:**  [**{github}**](https://github.com/serezk4/ITMO.labs/tree/master/programming/lab3)

**Результат работы**

Программа запускалась на Helios 

> Task :Main.main()

У Малыша глаза округлились от удивления. Может, он ослышался? Неужели Филле и Рулле в самом деле думают, что Карлсон живет у Свантесонов? Какое счастье! Это ведь значит, что Карлсон всегда может спрятаться у себя дома и быть там в полной безопасности. Филле и Рулле его не выследили! Да это и не так легко. Ведь никто, кроме трубочиста, не лазает по крышам. Итак, Филле и Рулле не пронюхали про домик на крыше, и тем не менее все это ужасно. Бедняга Карлсон, каково ему придется, если всерьез начнется за ним охота! Этот дурачок никогда не умел прятаться.

**Вывод**

Использовал принципы SOLID, познакомился с интерфейсами и абстрактными классами, много чего узнал