**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**“НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО”**

Факультет Программной Инженерии и Компьютерной Техники

Дисциплина: «Программирование»

**ОТЧЁТ**

по лабораторной работе №3  
Вариант №2657

Выполнил:

Студент группы P3108

Грищенко Андрей Викторович

Проверила:

Харитонова Анастасия Евгеньевна

Санкт-Петербург

2023 г.

**Задание**

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

"ПРОПАЛА СУМКА МУМИ-МАМЫ! Никаких путеводных нитей! Розыски продолжаются. Неслыханное пиршество в вознаграждение за находку!" Как только новость облетела долину, по всему лесу, на горах и на морском побережье поднялась страшная беготня. Даже последняя лесная крыса и та сочла небезвыгодным принять участие в поисках. Дома остались лишь старые да недужные, и вся долина огласилась криками и топотом ног.

**Исходный код программы**

public abstract class Entity {

private final String **name**;

private Place **place**;

public Entity(String name, Place place) {

this.**name** = name;

this.**place** = place;

}

public String getName() {

return **name**;

}

public Place getPlace() {

return **place**;

}

}

import java.util.Objects;

public class ForestRat extends Young {

public ForestRat(String name) {

super(name, Place.Forest);

}

@Override

public void makeNoise() {

System.**out**.println("Даже лесная крыса " + this.getName() + " подняла шум в " + getPlace().toString());

super.**isNoising** = true;

}

@Override

public String toString() {

String data = "Лесная крыса " + this.getName();

return super.**isNoising** ? data : data + ", которая подняла шум";

}

@Override

public boolean equals(Object obj) {

if (this == obj) return true;

if (obj == null || obj.getClass() != this.getClass()) return false;

ForestRat other = (ForestRat) obj;

return Objects.equals(getName(), other.getName());

}

@Override

public int hashCode() {

int result = super.getName().hashCode();

result = 31 \* result + super.getPlace().toString().hashCode();

return result;

}

}

import java.util.Objects;

public class News implements *Spreadable* {

private String **title**;

private String **description**;

public News(String title, String description) {

this.**title** = title;

this.**description** = description;

}

public String getTitle() {

return **title**;

}

public String getDescription() {

return **description**;

}

@Override

public void spread() {

System.**out**.println("Новость \"" + getTitle() + ": " + getDescription() + "\" разлетелась");

}

@Override

public String toString() {

return "Новость \"" + getTitle() + "\": " + getDescription();

}

@Override

public boolean equals(Object obj) {

if (this == obj) return true;

if (obj == null || obj.getClass() != this.getClass()) return false;

News other = (News) obj;

return Objects.equals(getTitle(), other.getTitle());

}

@Override

public int hashCode() {

int result = getTitle().hashCode();

result = 31 \* result + (getDescription().hashCode());

return result;

}

}

public interface *Noisable* {

public void makeNoise();

}

import java.util.Objects;

public class Old extends Entity {

public Old(String name, Place place) {

super(name, place);

}

@Override

public String toString() {

return "Старый и недужный " + this.getName() + " находится " + getPlace();

}

@Override

public boolean equals(Object obj) {

if (this == obj) return true;

if (obj == null || obj.getClass() != this.getClass()) return false;

Old other = (Old) obj;

return Objects.equals(getName(), other.getName());

}

@Override

public int hashCode() {

int result = super.getName().hashCode();

result = 31 \* result + super.getPlace().toString().hashCode();

return result;

}

}

public enum Place {

Home("Дома"),

Valley("Долина"),

Forest("Лес"),

Hills("Горы"),

SeaCoast("Морское побережье");

private final String **title**;

Place(String title) {

this.**title** = title;

}

@Override

public String toString() {

return **title**;

}

}

public interface *Spreadable* {

public void spread();

}

import java.util.Objects;

public class Young extends Entity implements *Noisable* {

protected boolean **isNoising** = false;

public Young(String name, Place place) {

super(name, place);

}

@Override

public void makeNoise() {

System.**out**.println(this.getName() + " подняли шум в " + getPlace().toString());

**isNoising** = true;

}

@Override

public String toString() {

String data = "Молодой и здоровый " + this.getName();

data += ", находящийся в " + this.getPlace();

return **isNoising** ? data : data + ", поднял шум";

}

@Override

public boolean equals(Object obj) {

if (this == obj) return true;

if (obj == null || obj.getClass() != this.getClass()) return false;

Young other = (Young) obj;

return Objects.equals(getName(), other.getName());

}

@Override

public int hashCode() {

int result = super.getName().hashCode();

result = 31 \* result + super.getPlace().toString().hashCode();

return result;

}

}

**Результат работы программы**

Новость "ПРОПАЛА СУМКА МУМИ-МАМЫ!: Никаких путеводных нитей! Розыски продолжаются. Неслыханное пиршество в вознаграждение за находку!" разлетелась

Жители долины подняли шум в Долина

Жители леса подняли шум в Лес

Жители гор подняли шум в Горы

Жители морского побережья подняли шум в Морское побережье

Даже лесная крыса Лесная крыса подняла шум в Лес

Старый и недужный Старики находится Дома

**Выводы по работе**

В результате проделанной лабораторной работы, я изучил принципы SOLID, а также построил объектную модель, следуя этим принципам на языке Java