Национальный исследовательский университет ИТМО

Мегафакультет компьютерных технологий и управления

Факультет программной инженерии и компьютерной техники



Лабораторная работа №3 предмет

«**Программирование**»

Вариант: 315305

Группа: P3114

Студент: Дау Конг Туан Ань

Преподаватель: Письмак Алексей Евгеньевич

г. Санкт-Петербург

Ноябрь, 2021

**1. Текст задания.**

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

|  |
| --- |
| Незнайка тем временем безмятежно спал в пищевом отсеке и даже не чувствовал, что космический полет начался. Среди ночи он, однако, проснулся и никак не мог понять, почему находится здесь, а не дома в постели. Постепенно он вспомнил, что нарочно забрался в ракету. Почувствовав невесомость и обратив внимание на мерный шум реактивного двигателя, Незнайка понял, что космический корабль находится в полете. "Значит, пока я спал, Знайка и остальные коротышки погрузились на корабль и отправились на Луну. Все получилось точно, как я рассчитал!" -- подумал Незнайка. Лицо его расплылось в счастливой улыбке, а внутри словно что-то затрепетало, заметалось от радости. Он уже хотел вылезти из своего убежища и, разыскав Знайку, признаться ему, что без спросу залез в ракету. Поразмыслив немного, он решил все же подождать, когда ракета отлетит от Земли подальше. |

**Программа должна удовлетворять следующим требованиям:**

1. Доработанная модель должна соответствовать [принципам SOLID](https://en.wikipedia.org/wiki/SOLID_(object-oriented_design)).
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).

**2. Диаграмма классов объектной модели.**

**Diagram, schematic

Description automatically generated**

**3. Исходный код программы.**

**Story.java**

package Main;

import List.Objects;

import Object.Boy;

public class Story {

public static void main(String[] args) {

Boy H = new Boy("Neznayka");

H.sleep(Objects.SLEEP);

H.setFeelMood(false);

H.feel(Objects.DIDNTFEEL);

H.aWake();

H.understand(Objects.UNDERSTAND);

H.remember(Objects.REMEMBER);

H.climb(Objects.CLIMB);

H.setFeelMood(true);

H.feel(Objects.FEEL);

H.notice(Objects.NOTICE);

H.realize(Objects.REALIZE);

H.think(Objects.THINK);

H.smile();

H.want(Objects.WANT);

H.confess(Objects.CONFESS);

H.decide(Objects.DECIDE);

}

}

**Boy.java**

package Object;

import Ability.Human;

import List.Objects;

public class Boy extends Human {

private Boolean feelMood;

public void setFeelMood(Boolean x) {

this.feelMood = x;

}

public Boolean getFeelMood() {

return this.feelMood;

}

public Boy(String name) {

this.setName(name);

}

@Override

public void sleep(Objects obj) {

System.out.println(this.getName() + " is sleeping at the " + obj.getValue());

}

@Override

public void feel(Objects obj) {

if(this.feelMood) {

System.out.println(this.getName() + " feeling " + obj.getValue());

} else {

System.out.println(this.getName() + " did not even feel that " + obj.getValue());

}

}

@Override

public void aWake() {

System.out.println(this.getName() + " woke up ");

}

@Override

public void smile() {

System.out.println(this.getName() + " smile ");

}

@Override

public void climb(Objects obj) {

System.out.println(this.getName() + " had deliberately climbed " + obj.getValue());

}

@Override

public void confess(Objects obj) {

System.out.println(this.getName() + " confess that " + obj.getValue());

}

@Override

public void understand(Objects obj) {

System.out.println(this.getName() + " could not understand " + obj.getValue());

}

@Override

public void remember(Objects obj) {

System.out.println(obj.getValue() + " " + this.getName() + " remembered that ");

}

@Override

public void notice(Objects obj) {

System.out.println(this.getName() + " pay attention " + obj.getValue());

}

@Override

public void realize(Objects obj) {

System.out.println(this.getName() + " realized " + obj.getValue());

}

@Override

public void want(Objects obj) {

System.out.println(this.getName() + " already wanted " + obj.getValue());

}

@Override

public void think(Objects obj) {

System.out.println(this.getName() + " thought " + obj.getValue());

}

@Override

public void decide(Objects obj) {

System.out.println(this.getName() + " decided " + obj.getValue());

}

}

**Human.java**

package Ability;

public abstract class Human implements Human\_Action, Human\_Mental {

String Name;

public void setName(String name) {

this.Name = name;

}

public String getName() {

return this.Name;

}

@Override

public boolean equals(Object obj) {

if (this.getClass() != obj.getClass())

return false;

Human others = (Human) obj;

return this.Name == others.Name;

}

@Override

public int hashCode() {

return this.Name.hashCode();

}

@Override

public String toString() {

return this.Name;

}

}

**Human\_Action.java**

package Ability;

import List.Objects;

public interface Human\_Action {

public void sleep(Objects obj);

public void aWake();

public void smile();

public void climb(Objects obj);

public void confess(Objects obj);

}

**Human\_Mental.java**

package Ability;

import List.Objects;

public interface Human\_Mental{

public void understand(Objects obj);

public void remember(Objects obj);

public void feel(Objects obj);

public void notice(Objects obj);

public void realize(Objects obj);

public void want(Objects obj);

public void think(Objects obj);

public void decide(Objects obj);

}

**Objects.java**

package List;

public enum Objects {

SLEEP("food compartment"),

DIDNTFEEL("the space flight had begun"),

AWAKE("In the middle of the night"),

REMEMBER("Gradually"),

CLIMB("into the rocket"),

UNDERSTAND("why he was here and not at home in bed"),

FEEL("weightlessness"),

NOTICE("to the dimensional noise of the jet engine"),

REALIZE("the spacecraft was in flight"),

THINK("So, while I was sleeping, Znayka and the rest of the little ones boarded the ship and went to the " + PlanetType.MOON + ". Everything turned out exactly as I calculated"),

WANT("to get out of his shelter"),

CONFESS("he had climbed into the rocket without asking"),

DECIDE("to wait for the rocket to fly away from " + PlanetType.EARTH);

private String val;

Objects(String val) {

this.val = val;

}

public String getValue() {

return this.val;

}

}

**PlanetType.java**

package List;

public enum PlanetType {

EARTH,

MOON,

SUN,

MAR,

MERCURY,

NEPTUNE

}

**4. Результат работы программы.**

Neznayka is sleeping at the food compartment

Neznayka did not even feel that the space flight had begun

Neznayka woke up

Neznayka could not understand why he was here and not at home in bed

Gradually Neznayka remembered that

Neznayka had deliberately climbed into the rocket

Neznayka feeling weightlessness

Neznayka pay attention to the dimensional noise of the jet engine

Neznayka realized the spacecraft was in flight

Neznayka thought So, while I was sleeping, Znayka and the rest of the little ones boarded the ship and went to the MOON. Everything turned out exactly as I calculated

Neznayka smile

Neznayka already wanted to get out of his shelter

Neznayka confess that he had climbed into the rocket without asking

Neznayka decided to wait for the rocket to fly away from EARTH

**5. Выводы по работе.**

Выполнив эту работу, я освоил правила объектно-ориентированного программирования SOLID и STUPID, как использовать интерфейс, абстрактный класс и тип ENUM. Я считаю, что мне еще нужно многому научиться, чтобы овладеть ООП