НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет программная инженерия и компьютерные технологии

Дисциплина Программирование

Лабораторная работа № 3

Выполнил студент

Черневская Карина Андреевна

Группа № Р3124

Преподаватель: Харитонова Анастасия Евгеньевна

г. Санкт-Петербург

2022

Оглавление

[Задание 3](#_Toc120109051)

[Ход работы 3](#_Toc120109052)

[Class LittleBoy 3](#_Toc120109053)

[class Bosse 5](#_Toc120109054)

[class Bimpo 6](#_Toc120109055)

[enum Food 6](#_Toc120109056)

[abstract class PeopleAtTheTable 7](#_Toc120109057)

[interface BoysAtTheTable 7](#_Toc120109058)

[interface Cuties 7](#_Toc120109059)

[class Main 7](#_Toc120109060)

[Варианты вывода программы 8](#_Toc120109061)

[UML-модель 11](#_Toc120109062)

[Вывод по работе 11](#_Toc120109063)

# Задание

Вариант 4258

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

И Малыш решил остаться дома и охранять Карлсона. Он тут же, еще сидя за столом и уплетая телячью отбивную, объяснил всем, почему он не может никуда ехать. Боссе расхохотался. На этот раз сидеть за обеденным столом оказалось не так весело, как обычно. И хотя Малыш был мальчиком милым и добрым, он мог иногда вдруг удивительно заупрямиться. Вот и сейчас он стал тверд как кремень и не проявлял никакой склонности вести переговоры.

# Ход работы

## Class LittleBoy

import java.util.Objects;  
import java.util.Random;  
  
public final class LittleBoy extends PeopleAtTheTable implements BoysAtTheTable {  
 private final String name = "Малыш";  
  
 LittleBoy(double percentagePointsOfHappiness) {  
 super(percentagePointsOfHappiness);  
 System.*out*.println();  
 if (this.getPointsOfHapiness() == 4) {  
 System.*out*.println(name + " разрывался между тем, чтобы остаться дома и охранять Карлосона, или все же поехать.");  
 } else if (this.getPointsOfHapiness() == 20) {  
 System.*out*.println(name + " преисполнился решимостью защитить летающего друга.");  
 } else {  
 System.*out*.println(name + " был в сметениях. Ему все еще предстояло выбрать: остаться дома и охранять нового друга или все же отправиться в долгожданную поездку.");  
 }  
 }  
  
 double pointsOfHappiness = this.getPointsOfHapiness();  
  
 int pick = new Random().nextInt(Food.*values*().length);  
 private final Food food = Food.*values*()[pick];  
  
  
 public void lunchTime() {  
 System.*out*.println("Зайдя на кухню, мальчик сразу же сел напротив Боссе, любопытно рассматривая журнальчик в руках брата");  
 switch (food) {  
 case *MEAT* -> {  
 System.*out*.println("На его месте уже стояла тарелка с теплыми телячьими отбивными"

);  
 pointsOfHappiness += 3;  
 }  
 case *SOUP* -> {  
 System.*out*.println("Перед ним уже стояла порция еще тепленького супа");  
 pointsOfHappiness += 1;  
 }  
 case *PASTRY* -> {  
 System.*out*.println("На середине стола стояла накрытая салфеткой тарелка - вчерашние пирожки с повидлом. Пока брат не смотрел, Малыш подвинул посудину чуть ближе к себе");  
 pointsOfHappiness += 8;  
 }  
 case *NOTHING* -> {  
 System.*out*.println("Малыш помнил, что в холодильнике еще были остатки вчерашнего ужина, но есть было нельзя - свою порцию, хоть и в виде целой банки варенья, он отдал своему новому другу");  
 pointsOfHappiness -= 3;  
 }  
 }  
 }  
 private boolean isSmitten;  
 public void smitten() {  
 if (pointsOfHappiness <= 10) {  
  
 System.*out*.println(name + " не знал, стоит ли говорить семье говорить про свои переживания. Поборов себя, он все же объяснился, почему поездку он все же пропустит");  
 isSmitten = true;  
 } else {  
 System.*out*.println("Оконачательно решив, что никуда не едет, " + name + " сказал всем за столов о своих намерениях");  
 isSmitten = false;  
 }  
 }  
 public boolean isSmitten() {  
 return isSmitten;  
 }  
 public void temp() {  
 System.*out*.println(pointsOfHappiness);  
 }  
 public Food dish() {  
 return food;  
 }  
 private boolean ifTheDishWasTaken;  
 public boolean isEmptyness() {  
 return food == Food.*NOTHING*;  
 }  
 public void takenFood(double pointsOfHappiness, boolean pastryIsStolen) {  
 if (food == Food.*MEAT* && pointsOfHappiness < 10) {  
 System.*out*.println(name + " совсем расхотел есть от переживаний. Идея еды не прельщала, но порция все же не отправиться обратно в холодильник - тарелка целиком была отдана собаке"+"\n");  
 ifTheDishWasTaken = true;  
  
 return;  
  
 } else if (food == Food.*PASTRY* && !(Math.*random*() <= 0.75)) {  
 System.*out*.println(name + " не успел и пирожка взять, как юркий щенок уже в прыжке с открытой пастью пытался стащить крайнюю булочку." +"\n" +"Вот только стол был чуток высоковат для песика, и никакой вкусняшки стащить не получилось");

System.*out*.println("Меньшее, что мог предложить Малыш в этой ситуации, это почесывания песика за ушком, но тот уже умчался в коридор"+"\n");

return;  
 } else if (food == Food.*PASTRY* && pastryIsStolen) {  
 System.*out*.println(name + " не успел и булочки взять, как юркий щенок не тлько одним прыжком умудрился достать до стола, "+"\n"+"но и извернулся так, что самый крайник пирожок ровно лег в открытую пасть зверя");  
 System.*out*.println("Малыш с улыбкой посмотрел на место, куда только что метнулся Бимпо и принялся за свою порцию, все еще думая над ситуацией, точнее, как ее объяснить семье"+"\n");  
 ifTheDishWasTaken = true;  
 return;  
 }else{  
 System.*out*.println("Вот только до посудины было совсем не достать, а хозяин не показывал намерения дать что-то со стола"+"\n");  
 }  
 ifTheDishWasTaken = false;  
 }  
 public boolean isIfTheDishWasTaken() {  
 return ifTheDishWasTaken;  
 }  
 public void character(boolean isSmitten){  
 if (isSmitten){  
 System.*out*.println(name + "был мальчиком милым и добрым, но уже взрослым не по годам. Он не знал, как поступить, разрываясь между моральным долгом и личным желанияем не пропустить поездку.");  
 System.*out*.println("Смог бы кто помочье ему в этом нелегком деле...");  
 }else{  
 System.*out*.println("И хотя Малыш был мальчиком милым и добрым, он мог иногда вдруг удивительно заупрямиться.");  
 System.*out*.println("Вот и сейчас он стал тверд как кремень и не проявлял никакой склонности вести переговоры.");  
 }  
 }  
 @Override  
 public void getScared(boolean barkedDog) {  
 System.*out*.println(barkedDog ? "Ой! Бимпо! Не пугай так!" : "");  
 }  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 LittleBoy littleBoy = (LittleBoy) o;  
 return Double.*compare*(littleBoy.pointsOfHappiness, pointsOfHappiness) == 0 && pick == littleBoy.pick && isSmitten == littleBoy.isSmitten && ifTheDishWasTaken == littleBoy.ifTheDishWasTaken && Objects.*equals*(name, littleBoy.name) && food == littleBoy.food;  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(name, pointsOfHappiness, pick, food, isSmitten, ifTheDishWasTaken);  
 }  
}

## class Bosse

public final class Bosse extends PeopleAtTheTable implements BoysAtTheTable{  
 private final String name = "Боссе";  
 public Bosse(double percentagePointsOfHappiness) {  
 super(percentagePointsOfHappiness);  
 }  
 public void laugh(boolean isSmitten){  
 if (isSmitten){  
 System.*out*.println(name + " рассхохотался ");  
 }else{  
 System.*out*.println(name + " рассхохотался от серьезного детского личика брата");  
 }  
 }  
 @Override  
 public void getScared(boolean barkedDog) {  
 if (barkedDog){  
 System.*out*.println("'Ты че разгавкался?!' - крикнул на собаку Боссе");  
 }  
 }  
}

## class Bimpo

public class Bimpo implements Cuties{  
 @Override  
 public void voice(boolean isEmptyness) {  
 System.*out*.println("Голодное существо выскочило из-под стола, уставившись на мальчика большими глазами в надежде на получение еды");  
 if (isEmptyness){  
 System.*out*.println("Увидев, что на столе ни крошки, песик метнулся в коридао, пропадая так же быстро, как и появившись");  
 }else {  
 System.*out*.println((Math.*random*() <= 0.5) ? "Тяв-тяв" : "Вуф-вуф");  
 }  
 }  
 public void thoughts(boolean isEmpty){  
 if (!isEmpty){  
 System.*out*.println("Бимпо полностью проигнорировал Боссе.");  
 System.*out*.println("Хоть с ракурса собаки и не было много видно, песик явно понимал, что где-то там еда явно есть, это главнее");  
 }  
 }  
  
 @Override  
 public boolean thiefSkill() {  
 return (Math.*random*()<=0.75);  
 }  
  
 @Override  
 public void catchingZZZ(boolean givenMeat) {  
 System.*out*.println((givenMeat) ? "В это же время Бимпо счастливо посапывает в ногах Малыша после вкусного обеда, свернувшись калачиком" :  
 "В это же время, так и не получив ничего со стола, Бимпо плюхается в ногах Малыша и медленно погружается в дрему, все еще мечтая хотя бы о куске мяса");  
 }  
  
}

## enum Food

public enum Food {  
 *MEAT*("телячья отбивная"),  
 *PASTRY* ("пирожки с повидлом"),  
 *NOTHING* ("ничего"),  
 *SOUP* ("суп");  
  
 private String lunch;  
 Food(String lunch){  
 this.lunch=lunch;  
 }  
 @Override  
 public String toString(){  
 return lunch;  
 }  
}

## abstract class PeopleAtTheTable

public abstract class PeopleAtTheTable {  
 double pointsOfHappiness;  
 PeopleAtTheTable(double percentagePointsOfHappiness){  
 if (percentagePointsOfHappiness<=0.2){  
 this.pointsOfHappiness = 20;  
 } else if (percentagePointsOfHappiness>=0.8){  
 this.pointsOfHappiness = 4;  
 } else {  
 this.pointsOfHappiness = 10;  
 }  
 }  
 public double getPointsOfHapiness(){  
 return pointsOfHappiness;  
 }  
}

## interface BoysAtTheTable

public interface BoysAtTheTable {  
 void getScared(boolean barkedDog);  
}

## interface Cuties

public interface Cuties {  
 void voice(boolean isEmptyness);  
  
 boolean thiefSkill();  
 void catchingZZZ(boolean something);  
}

## class Main

public class Main {  
 public static void main(String[] args){  
 LittleBoy littleBoy = new LittleBoy(Math.*random*());  
 Bosse bosse = new Bosse(Math.*random*());  
 Bimpo bimpo = new Bimpo();  
  
 littleBoy.lunchTime();  
  
  
 bimpo.voice(littleBoy.isEmptyness());  
 bosse.getScared(!littleBoy.isEmptyness());  
 bimpo.thoughts(littleBoy.isEmptyness());  
 littleBoy.takenFood(littleBoy.getPointsOfHapiness(), bimpo.thiefSkill());  
 littleBoy.smitten();  
 bosse.laugh(littleBoy.isSmitten());  
 bimpo.catchingZZZ(littleBoy.isIfTheDishWasTaken());  
 littleBoy.character(littleBoy.isSmitten());  
  
  
 }  
}

## Варианты вывода программы

№1

Малыш преисполнился решимостью защитить летающего друга.

Зайдя на кухню, мальчик сразу же сел напротив Боссе, любопытно рассматривая журнальчик в руках брата

На середине стола стояла накрытая салфеткой тарелка - вчерашние пирожки с повидлом. Пока брат не смотрел, Малыш подвинул посудину чуть ближе к себе

Голодное существо выскочило из-под стола, уставившись на мальчика большими глазами в надежде на получение еды

Вуф-вуф

'Ты че разгавкался?!' - крикнул на собаку Боссе

Бимпо полностью проигнорировал Боссе.

Хоть с ракурса собаки и не было много видно, песик явно понимал, что где-то там еда явно есть, это главнее

Малыш не успел и булочки взять, как юркий щенок не тлько одним прыжком умудрился достать до стола,

но и извернулся так, что самый крайник пирожок ровно лег в открытую пасть зверя

Малыш с улыбкой посмотрел на место, куда только что метнулся Бимпо и принялся за свою порцию, все еще думая над ситуацией, точнее, как ее объяснить семье

Оконачательно решив, что никуда не едет, Малыш сказал всем за столов о своих намерениях

Боссе рассхохотался от серьезного детского личика брата

В это же время Бимпо счастливо посапывает в ногах Малыша после вкусного обеда, свернувшись калачиком

И хотя Малыш был мальчиком милым и добрым, он мог иногда вдруг удивительно заупрямиться.

Вот и сейчас он стал тверд как кремень и не проявлял никакой склонности вести переговоры.

№2

Малыш был в сметениях. Ему все еще предстояло выбрать: остаться дома и охранять нового друга или все же отправиться в долгожданную поездку.

Зайдя на кухню, мальчик сразу же сел напротив Боссе, любопытно рассматривая журнальчик в руках брата

На его месте уже стояла тарелка с теплыми телячьими отбивными

Голодное существо выскочило из-под стола, уставившись на мальчика большими глазами в надежде на получение еды

Тяв-тяв

'Ты че разгавкался?!' - крикнул на собаку Боссе

Бимпо полностью проигнорировал Боссе.

Хоть с ракурса собаки и не было много видно, песик явно понимал, что где-то там еда явно есть, это главнее

Вот только до посудины было совсем не достать, а хозяин не показывал намерения дать что-то со стола

Оконачательно решив, что никуда не едет, Малыш сказал всем за столов о своих намерениях

Боссе рассхохотался от серьезного детского личика брата

В это же время, так и не получив ничего со стола, Бимпо плюхается в ногах Малыша и медленно погружается в дрему, все еще мечтая хотя бы о куске мяса

И хотя Малыш был мальчиком милым и добрым, он мог иногда вдруг удивительно заупрямиться.

Вот и сейчас он стал тверд как кремень и не проявлял никакой склонности вести переговоры.

№3

Малыш был в сметениях. Ему все еще предстояло выбрать: остаться дома и охранять нового друга или все же отправиться в долгожданную поездку.

Зайдя на кухню, мальчик сразу же сел напротив Боссе, любопытно рассматривая журнальчик в руках брата

Перед ним уже стояла порция еще тепленького супа

Голодное существо выскочило из-под стола, уставившись на мальчика большими глазами в надежде на получение еды

Тяв-тяв

'Ты че разгавкался?!' - крикнул на собаку Боссе

Бимпо полностью проигнорировал Боссе.

Хоть с ракурса собаки и не было много видно, песик явно понимал, что где-то там еда явно есть, это главнее

Вот только до посудины было совсем не достать, а хозяин не показывал намерения дать что-то со стола

Оконачательно решив, что никуда не едет, Малыш сказал всем за столов о своих намерениях

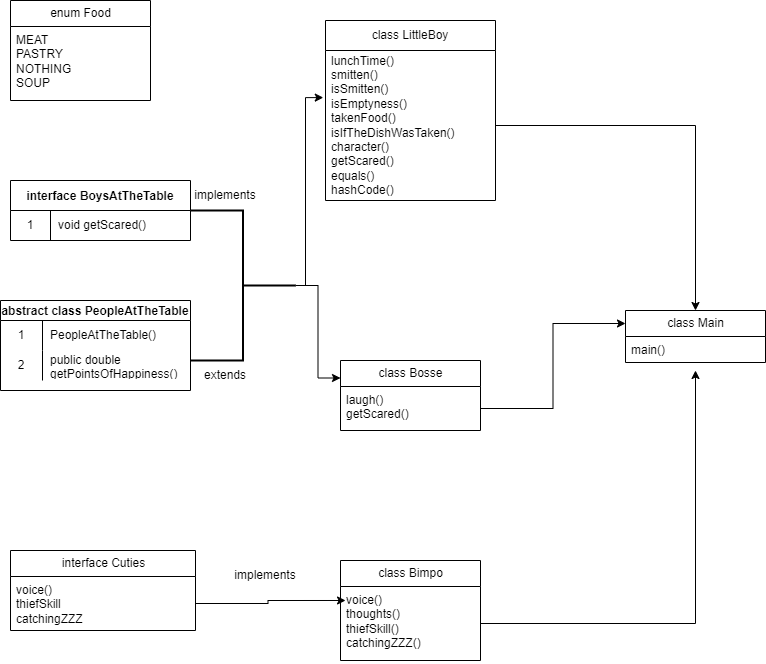
Боссе рассхохотался от серьезного детского личика брата

В это же время, так и не получив ничего со стола, Бимпо плюхается в ногах Малыша и медленно погружается в дрему, все еще мечтая хотя бы о куске мяса

И хотя Малыш был мальчиком милым и добрым, он мог иногда вдруг удивительно заупрямиться.

Вот и сейчас он стал тверд как кремень и не проявлял никакой склонности вести переговоры.

## UML-модель



# Вывод по работе

В ходе данной работы были применены знания по использованию абстрактных классов и интерфейсов, а также перечисляемых типов данных(enum).

По описанию предментов области была создана мини-история с несколькими ветвями развития.

# Список литературы

Васильев, А.Н. Java. Объектно-ориентированное программирование: Учебное пособие: для магистров и бакалавров. Базовый курс по объектно-ориентированному программированию / А.Н. Васильев. - СПб.: Питер, 2013. - 400 c.

Блох, Д. Java Эффективное программирование / Д. Блох. - М.: Лори, 2016. - 440 c.