НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Программирование

Лабораторная работа № 4

Выполнил студент

Урын Егор Антонович

Группа № P3120

Преподаватель: Письмак Алексей Евгеньевич

г. Санкт-Петербург

2022

Оглавление

[Вариант 3](#_Toc118406422)

[Задание 3](#_Toc118406423)

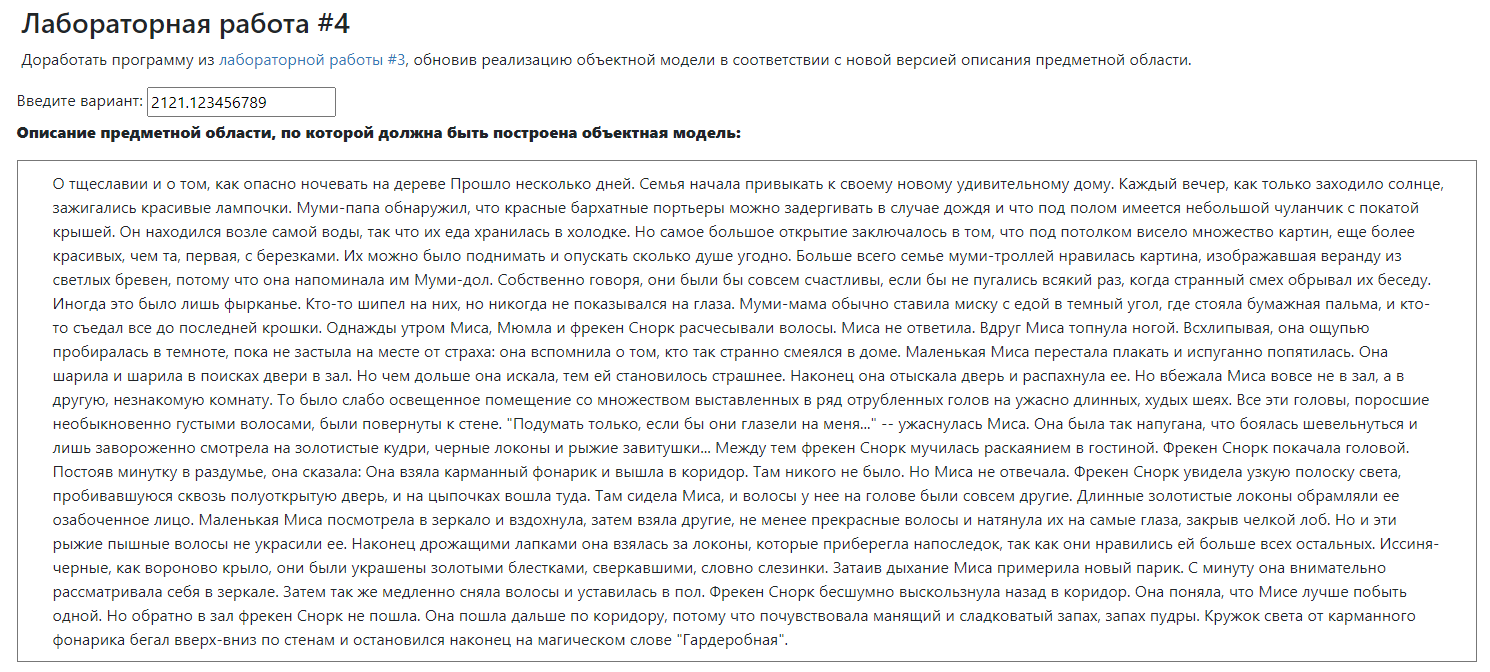
[Отчет: 4](#_Toc118406424)

[Результат работы 11](#_Toc118406425)

[Список литературы 15](#_Toc118406426)

**Вариант:** 2121.123456789

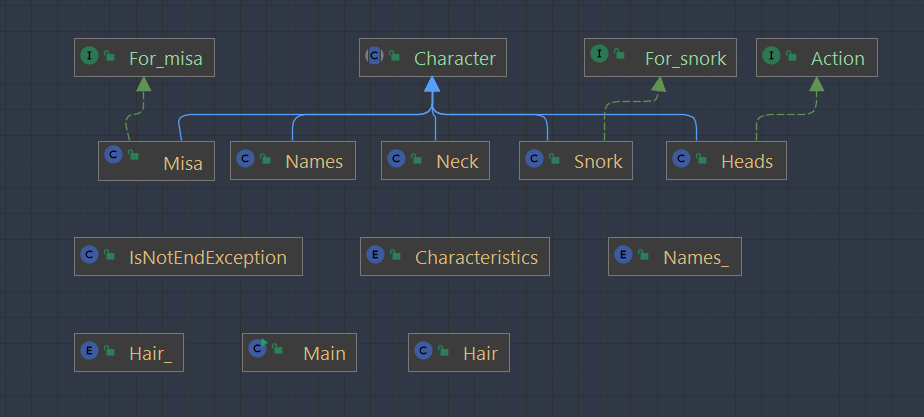
**Задание:**



# 

# **Отчет:**

Диаграмма классов проекта



**Исходный код:**

Main.java

import Exceptions.IsNotEndException;  
import class\_.\*;  
import interface\_.SnorkShake;  
  
public class Main {  
 public Main() {  
 }  
  
 public static void main(String[] args) {  
 Misa misa = new Misa();  
 misa.setName("Миса");  
 Names names = new Names();  
 Heads heads = new Heads();  
 heads.setName("головы");  
 Neck neck = new Neck();  
 neck.setName("шеях");  
 Hair hair = new Hair();  
 Snork snork = new Snork();  
 snork.setName("Снорк");  
 Snork.Out out\_ = new Snork.Out();  
 Snork.Take take\_ = snork.toTake();  
 SnorkShake shake\_ = new SnorkShake() {  
 @Override  
 public String shake() {  
 return "покачала головой";  
 }  
 };  
 *story*(misa, names, heads, neck, hair, snork, out\_, take\_, shake\_);  
 }  
  
 private static void story(Misa misa, Names names, Heads heads, Neck neck, Hair hair, Snork snork, Snork.Out out\_, Snork.Take take\_, SnorkShake shake\_) {  
 System.out.println(names.description(1) + ", " + names.description(2) + " и " + names.description(3) + " " + misa.combed());  
 System.out.println(misa + " " + misa.stomped());  
 System.out.println(misa + " " + misa.walking() + " и " + misa.frozen() + " " + misa.horror());  
 System.out.println(misa + " " + misa.get());  
 System.out.println(heads + " " + heads.was() + " на " + neck.description(1) + ", " + neck.description(2) + ", " + neck.description(3) + " " + neck);  
 System.out.println(heads + " " + heads.turn());  
 System.out.println(misa + " " + misa.fear() + ", " + misa.afraid() + " и " + misa.see() + " " + hair.description(1) + ", " + hair.description(2) + " и " + hair.description(3));  
 System.out.println(snork + " " + snork.suffered());  
 System.out.println(snork + " " + shake\_.shake());  
 System.out.println(snork + " " + take\_.take() + " и " + out\_.out()); *// snork.take() +* snork.setEnd(true);  
 try {  
 if (*endStory*(snork)) {  
 System.out.println("The end");  
 }  
 } catch (IsNotEndException e) {  
 System.out.println(e.getMessage());  
 }  
 }  
  
 public static boolean endStory(Snork snork) throws IsNotEndException {  
 if (snork.isEnd()) {  
 return true;  
 } else {  
 throw new IsNotEndException();  
 }  
 }  
}

Character.java

package abstract\_;  
  
public abstract class Character {  
 protected String name;  
  
 public Character() {  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getName() {  
 return this.name;  
 }  
  
 public String toString() {  
 return this.getName();  
 }  
}

Characteristics.java

package class\_;  
  
public enum Characteristics {  
 TERRIBLE,  
 LONG,  
 THIN;  
}

Hair.java

package class\_;  
  
public class Hair {  
 Hair\_ hair;  
 public Hair() {  
 }  
  
 public String description(int i) {  
 if (i == 1) {  
 this.hair = Hair\_.CURLS;  
 return "золотистые кудри";  
 } else if (i == 2) {  
 this.hair = Hair\_.RINGLET;  
 return "чёрные локоны";  
 } else {  
 this.hair = Hair\_.CURLICUES;  
 return "рыжие завитушки";  
 }  
 }  
}

Hair\_.java

package class\_;  
  
public enum Hair\_ {  
 CURLS,  
 RINGLET,  
 CURLICUES;  
}

Heads.java

package class\_;  
  
import abstract\_.Character;  
import interface\_.Action;  
  
public class Heads extends Character implements Action {  
 public Heads() {  
 }  
  
 @Override  
 public String was() {  
 return "были";  
 }  
  
 @Override  
 public String turn() {  
 return "повёрнуты к стене";  
 }  
}

Names\_.java

package class\_;  
  
public enum Names\_ {  
 MISA,  
 MYUMLA,  
 SNORK;  
 private Names\_() {  
 }  
}

Misa.java

package class\_;  
  
import abstract\_.Character;  
import interface\_.For\_misa;  
  
public class Misa extends Character implements For\_misa {  
 public Misa() {  
 }  
  
 public String horror() {  
 try {  
 if (Math.*random*() <= 0.5)  
 return "в ужасе";  
 else  
 throw new NullPointerException();  
 } catch (NullPointerException e) {  
 return "испытав страх";  
 }  
 }  
  
 @Override  
 public String combed() {  
 return "расчёсывались";  
 }  
  
 @Override  
 public String stomped() {  
 return "топнула ногой";  
 }  
  
 @Override  
 public String walking() {  
 return "шла";  
 }  
  
 @Override  
 public String frozen() {  
 return "застыла";  
 }  
  
 @Override  
 public String get() {  
 return "попала в комнату";  
 }  
  
 @Override  
 public String fear() {  
 return "испугалась";  
 }  
  
 @Override  
 public String afraid() {  
 return "боялась шевельнуться";  
 }  
  
 @Override  
 public String see() {  
 return "смотрела на";  
 }  
}

Names.java

package class\_;  
  
import abstract\_.Character;  
  
public class Names extends Character {  
 Names\_ name;  
 public Names() {  
 }  
  
 public String description(int i) {  
 if (i == 1) {  
 this.name = Names\_.MISA;  
 return "Миса";  
 } else if (i == 2) {  
 this.name = Names\_.MYUMLA;  
 return "Мюмла";  
 } else {  
 this.name = Names\_.SNORK;  
 return "Снорк";  
 }  
 }  
}

Neck.java

package class\_;  
  
import abstract\_.Character;  
  
public class Neck extends Character {  
 Characteristics characteristic;  
 public Neck() {  
 }  
  
 public String description(int i) {  
 if (i == 1) {  
 this.characteristic = Characteristics.TERRIBLE;  
 return "ужасных";  
 } else if (i == 2) {  
 this.characteristic = Characteristics.LONG;  
 return "длинных";  
 } else {  
 this.characteristic = Characteristics.THIN;  
 return "худых";  
 }  
 }  
}

Snork.java

package class\_;  
  
import abstract\_.Character;  
import interface\_.For\_snork;  
  
public class Snork extends Character implements For\_snork {  
 private boolean isEnd;  
  
 public Snork() {  
 isEnd = false;  
 }  
  
 @Override  
 public String suffered() {  
 return "мучалась в гостиной";  
 }  
  
 public interface Take {  
 String take();  
 }  
  
 public Take toTake() {  
 class SnorkTake implements Take {  
 final String take\_ = "взяла фонарик";  
 @Override  
 public String take() {  
 return this.take\_;  
 }  
 }  
 return new SnorkTake();  
 }  
  
 public static class Out {  
 public String out() {  
 return "вышла в коридор";  
 }  
 }  
  
 public boolean isEnd() {  
 return isEnd;  
 }  
  
 public void setEnd(boolean end) {  
 isEnd = end;  
 }  
}

IsNotEndException.java

package Exceptions;  
  
public class IsNotEndException extends Exception {  
 @Override  
 public String getMessage() {  
 return "грустно)";  
 }  
}

Action.java

package interface\_;  
  
public interface Action {  
 String was();  
 String turn();  
}

For\_misa.java

package interface\_;  
  
public interface For\_misa {  
 String combed();  
 String stomped();  
 String walking();  
 String frozen();  
 String get();  
 String fear();  
 String afraid();  
 String see();  
}

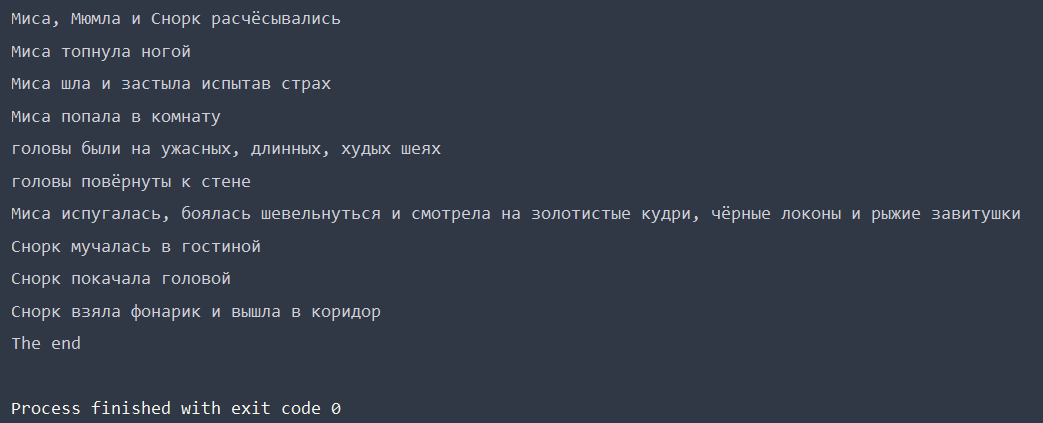
For\_snork.java

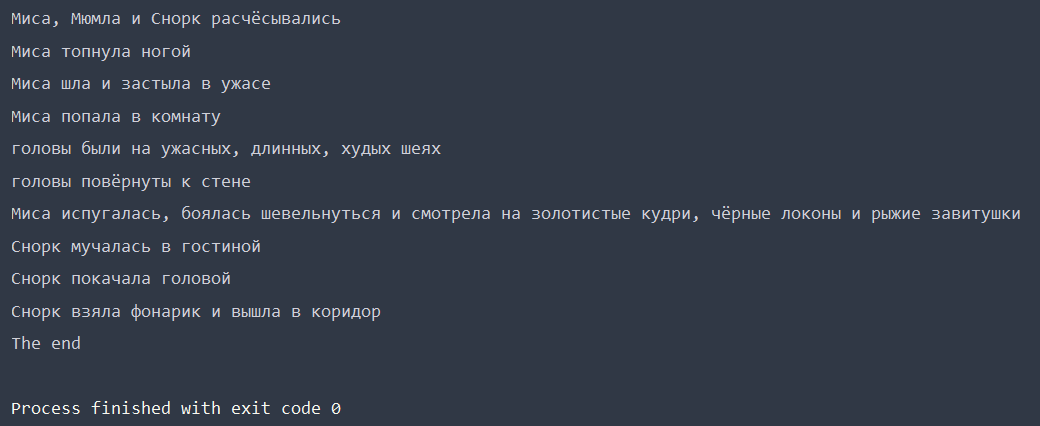
package interface\_;  
  
public interface For\_snork {  
 String suffered();  
 String shake();  
 String take();  
 String out();  
}

For\_snork.java

package interface\_;  
  
public interface SnorkShake {  
 public String shake();  
}

**Результат работы:**





**Вывод:**

Во время выполнения данной лабораторной работы я изучил принципы SOLID, научился работать с методами, классами, переопределять методы и узнал, что такое интерфейсы, абстрактные классы и enum.

**Список литературы:**

Гаврилов А.В., Клименков С.В., Королёва Ю.А., Харитонова А.Е., Цопа Е.А. Программирование на языке Java. Конспект лекций – СПб: Университет ИТМО, 2019. – 127 с.

Stackoverflow // https://stackoverflow.com