Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение

высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №4

Вариант 32856.6

Группа: P3132

Выполнил: Овчаренко Александр Андреевич

г. Санкт-Петербург

2021 г.

Оглавление

[Задание 3](#_Toc90975283)

[Выполнение задания 4](#_Toc90975284)

[Korotishi 4](#_Toc90975285)

[Gender 5](#_Toc90975286)

[ThisObject 5](#_Toc90975287)

[Story 6](#_Toc90975288)

[Window 6](#_Toc90975289)

[WindowCloseException 6](#_Toc90975290)

[Eyes 7](#_Toc90975291)

[StatusOpenClose 7](#_Toc90975292)

[Book 8](#_Toc90975293)

[JumpInWindow 9](#_Toc90975294)

[OpenCloseWindowAction 9](#_Toc90975295)

[CarryAction 10](#_Toc90975296)

[CheckInterface 10](#_Toc90975297)

[TellStoryAction 11](#_Toc90975298)

[PutAction 11](#_Toc90975299)

[GoOnTrip 12](#_Toc90975300)

[ReadBookAction 13](#_Toc90975301)

[SeeAction 13](#_Toc90975302)

[ThinkAction 14](#_Toc90975303)

[Main 15](#_Toc90975304)

[Вывод программы 17](#_Toc90975305)

[Итоги 19](#_Toc90975306)

# Задание

Доработать программу из лабораторной работы #3, обновив реализацию объектной модели в соответствии с новой версией описания предметной области.

Программа должна удовлетворять следующим требованиям:

* В программе должны быть реализованы 2 собственных класса исключений (checked и unchecked), а также обработка исключений этих классов.
* В программу необходимо добавить использование локальных, анонимных и вложенных классов (static и non-static).

Порядок выполнения работы:

* Доработать объектную модель приложения.
* Перерисовать диаграмму классов в соответствии с внесёнными в модель изменениями.
* Согласовать с преподавателем изменения, внесённые в модель.
* Модифицировать программу в соответствии с внесёнными в модель изменениями.

Вариант 32856.6

# Выполнение задания

Korotishi

package lab\_4.kids;

public class Korotishi {

    private final String name;

    private final Gender gender;

    public static Korotishi malishi = new Korotishi("Malishi", Gender.Malishi);

    public static Korotishi malishki = new Korotishi("Malishki", Gender.Malishki);

    private Korotishi(String name, Gender gender) {

        this.name = name;

        this.gender = gender;

    }

    public String getGender() {

        return gender.getGender();

    }

    @Override

    public int hashCode() {

        return name.hashCode() + gender.hashCode();

    }

    @Override

    public String toString() {

        return name;

    }

    @Override

    public boolean equals(Object obj) {

        if (this == obj) return true; // сравнение ссылок, та же ли ссылка

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

        Korotishi that = (Korotishi) obj; // сужеющее приведение, чтобы взять поля

        return name.equals(that.name) && gender == that.gender;

    }

    public static Korotishi createMalish(String name) {

        if (name.equals("Malishi")) throw new IllegalArgumentException(name + " - wrong name");

        else {return new Korotishi(name, Gender.Malishi);}

    }

    public static Korotishi createMalishka(String name) {

        if (name.equals("Malishki")) throw new IllegalArgumentException(name + " - wrong name");

        else {return new Korotishi(name, Gender.Malishki);}

    }

}

Gender

package lab\_4.kids;

public enum Gender {

    Malishi("Malish"),

    Malishki("Malishki");

    private String gender;

    Gender(String gender) {

        this.gender = gender;

    }

    public String getGender() {

        return gender;

    }

}

ThisObject

package lab\_4.objects;

public abstract class ThisObject {

    private final String object;

    public ThisObject (String object) {

        this.object = object;

    }

    @Override

    public boolean equals(Object obj) {

        if (this == obj) return true;

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

        ThisObject that = (ThisObject) obj;

        return object == that.object;

    }

    @Override

    public String toString() {

        return object;

    }

}

Story

package lab\_4.objects;

public class Story extends ThisObject{

    private String topic;

    public Story(String topic){

        super("story");

        this.topic = topic;

    }

    public String getTopic(){

        return topic;

    }

}

Window

package lab\_4.objects;

public class Window extends ThisObject {

    private StatusOpenClose status;

    public Window(StatusOpenClose status){

        super("window");

        this.status = status;

    }

    public String getStatus(){

        return status.getStatus();

    }

    public void setStatus(StatusOpenClose status){

        this.status = status;

    }

}

WindowCloseException

package lab\_4.exception;

public class WindowCloseException extends Exception {

    public WindowCloseException(String message){

        super(message);

    }

}

Eyes

package lab\_4.objects;

import lab\_4.kids.Korotishi;

public class Eyes {

    private Korotishi who;

    private StatusOpenClose status;

    public Eyes(Korotishi who, StatusOpenClose status){

        this.who = who;

        this.status = status;

    }

    public Korotishi getKorotishi(){

        return who;

    }

    public String getStatus(){

        return status.getStatus();

    }

    public void OpenEyes(){

        status = StatusOpenClose.Open;

        System.out.println(who + "opens eyes.");

    }

    public void CloseEyes(){

        status = StatusOpenClose.Close;

        System.out.println(who + "closes eyes.");

    }

}

StatusOpenClose

package lab\_4.objects;

public enum StatusOpenClose {

    Open("open"),

    Close("close");

    private String status;

    StatusOpenClose(String status){

        this.status = status;

    }

    public String getStatus(){

        return status;

    }

}

Book

package lab\_4.objects;

//inner class

public class Book{

    private final String nameOfBook;

    private final String themeOfBook;

    public Book(String name, String theme){

        nameOfBook = name;

        themeOfBook = theme;

    }

    public String getNameOfBook(){

        return nameOfBook;

    }

    public String getThemeOfBook(){

        return themeOfBook;

    }

    public void printBook(){

        System.out.println("Name of Book: " + nameOfBook + "\n" + "Theme of Book: " + themeOfBook);

    }

    public class StoryOfBook extends ThisObject{

        private String topic;

        public StoryOfBook(String topic){

            super("storyOf" + getNameOfBook());

            this.topic = topic;

        }

        public String getTopic(){

            return topic;

        }

    }

}

JumpInWindow

package lab\_4.actions;

import lab\_4.exception.\*;

package lab\_4.actions;

import lab\_4.exception.\*;

import lab\_4.kids.Korotishi;

import lab\_4.objects.Window;

//Check exception

public class JumpInWindow {

    private Korotishi who;

    private Window window;

    public JumpInWindow(Korotishi who, Window window){

        this.who = who;

        this.window = window;

    }

    public void jumpInWindow() throws WindowCloseException{

        if(window.getStatus().equals("close")) throw new WindowCloseException("the window is closed.");

        else{System.out.println(who + " jumps out in window.");}

    }

}

OpenCloseWindowAction

package lab\_4.actions;

import lab\_4.kids.Korotishi;

import lab\_4.objects.StatusOpenClose;

import lab\_4.objects.Window;

public class OpenCloseWindowAction {

    private Korotishi who;

    private Window window;

    public OpenCloseWindowAction(Korotishi who, Window window){

        this.who = who;

        this.window = window;

    }

    public void openWindow(){

        window.setStatus(StatusOpenClose.Open);

        System.out.println(who + " opens window.");

    }

    public void closeWindow(){

        window.setStatus(StatusOpenClose.Close);

        System.out.println(who + " closes window.");

    }

}

CarryAction

package lab\_4.actions;

import lab\_4.kids.\*;

public class CarryAction {

    private final String name = "carry";

    private final Korotishi[] who;

    private final Korotishi whom;

    private final String place;

    public CarryAction(Korotishi[] who, Korotishi whom, String place) {

        this.who = who;

        this.whom = whom;

        this.place = place;

    }

    public Korotishi[] getWho(){

        return who;

    }

    public Korotishi getWhom(){

        return whom;

    }

    public void print() {

        System.out.print(who[0]);

        for(int i = 1; i<who.length; i++) System.out.print(", " + who[i]);

        System.out.println(" " + name + " " + whom + " " + place + ".");

    }

}

CheckInterface

package lab\_4.actions;

import lab\_4.objects.ThisObject;

public interface CheckInterface {

    public boolean check(ThisObject object);

}

TellStoryAction

package lab\_4.actions;

import lab\_4.kids.Korotishi;

import lab\_4.objects.\*;

import lab\_4.objects.Book.StoryOfBook;

//anon class

public class TellStoryAction {

    private Korotishi who;

    private Korotishi whom;

    private StoryOfBook[] stories;

    public TellStoryAction(Korotishi who, Korotishi whom, StoryOfBook[] stories){

        this.who = who;

        this.whom = whom;

        this.stories = stories;

    }

    public void tellStories(){

        CheckInterface check = (ThisObject object) -> {

            StoryOfBook story = (StoryOfBook) object;

            if (story.getTopic().equals("journey")

) return true;

            else{return false;}

    };

        for(StoryOfBook st: stories){

            System.out.println(who + " tells story about " + st.getTopic() + " to " + whom + ".");

        for(StoryOfBook st: stories){

            System.out.println(who + " tells story about " + st.getTopic() + " to " + whom + ".");

            if (check.check(st)) {

                System.out.println(whom + " likes story about " + st.getTopic());

            }

            else{

                System.out.println(whom + " doesn't like story about " + st.getTopic());

        }

        }

    }

}

PutAction

package lab\_4.actions;

import lab\_4.kids.Korotishi;

public class PutAction {

    private Korotishi whom;

    private Korotishi[] who;

    private String place;

    public PutAction(Korotishi[] who, Korotishi whom, String place){

        this.who = who;

        this.whom = whom;

        this.place = place;

    }

    public void print(){

        System.out.print(who[0]);

        for(int i = 1; i<who.length; i++) System.out.print(", " + who[i]);

        System.out.println(" put " + whom + " " + place + ".");

    }

}

GoOnTrip

package lab\_4.actions;

import lab\_4.kids.\*;

import lab\_4.objects.\*;

//nested class

public class GoOnTrip{

    private Korotishi[] who;

    private String place;

    public GoOnTrip(Korotishi[] who, String place){

        this.who = who;

        this.place = place;

    }

    public void goOnTrip(){

        System.out.print(who[0]);

        for(int i = 1; i<who.length; i++) System.out.print(", " + who[i]);

        System.out.println(" go on trip " + place + ".");

    }

    public static class SuggestionTransportForTrip extends ThisObject{

        private String name;

        SuggestionTransportForTrip(String name){

            super("suggestiontransportForTrip");

            this.name = name;

        }

        public void suggestsTransport(Korotishi who){

            System.out.println(who + "suggests using " + getName() + "to travel.");

        }

        public String getName(){

            return name;

        }

    }

}

ReadBookAction

package lab\_4.actions;

import lab\_4.kids.Korotishi;

import lab\_4.objects.Book;

public class ReadBookAction {

    private final Korotishi who;

    private Book book;

    public ReadBookAction(Korotishi who, Book book){

        this.who = who;

        this.book = book;

    }

    public void changeBook(Book book){

        this.book = book;

    }

    public void printReadAction(){

        System.out.println(who + " reads book \"" + book.getNameOfBook() + "\".");

    }

}

SeeAction

package lab\_4.actions;

import java.util.List;

import lab\_4.kids.\*;

import lab\_4.objects.Eyes;

import lab\_4.exception.\*;

//Un-check exception

public class SeeAction {

    private final Korotishi who;

    private final Eyes eyes;

    public SeeAction(Korotishi who, Eyes eyes) throws AlienEyesException{

        if(who.equals(eyes.getKorotishi())){

            this.who = who;

            this.eyes = eyes;

        }

        else{

            throw new AlienEyesException("These eyes is not " + who + "'s eyes.");

        }

    }

    public void printSeeAction(List<Object> objects) throws EyesCloseException{

        if(eyes.getStatus().equals("open"){

            for(int i=0; i<objects.size(); i++){

                System.out.println(who + " see " + objects.get(i) + ".");

            }

        }

        else{

            throw new EyesCloseException("Eyes is closed.");

        }

    }

}

ThinkAction

package lab\_4.actions;

import lab\_4.kids.Korotishi;

public class ThinkAction {

    private final String name = "think";

    private final Korotishi who;

    private final int num\_day;

    private final int num\_night;

    private final String result;

    public ThinkAction(Korotishi who, int day, int night, String result) {

        this.who = who;

        num\_day = day;

        num\_night = night;

        this.result = result;

    }

    public Korotishi getWho(){

        return who;

    }

    public void printDays(){

        System.out.println("num\_days = " + num\_day + ", num\_night = " + num\_night);

    }

    public String getResult(){

        return result;

    }

    public void print() {

        System.out.println(who + " " + name + ". He has been thinking for " + num\_day +

        " days and " + num\_night + " night. Result - " + result + ".");

    }

}

Main

package lab\_4;

import java.util.ArrayList;

import java.util.List;

import lab\_4.actions.\*;

import lab\_4.exception.EyesCloseException;

import lab\_4.exception.WindowCloseException;

import lab\_4.kids.Korotishi;

import lab\_4.objects.\*;

import lab\_4.objects.Book.StoryOfBook;

public class Main{

    public static void main(String[] args) throws WindowCloseException, EyesCloseException{

        //creat heros

        Korotishi znayka = Korotishi.createMalish("Znayka");

        Korotishi neznayka = Korotishi.createMalish("Neznayka");

        Korotishi avoska = Korotishi.createMalish("Avoska");

        Korotishi vintik = Korotishi.createMalish("Vintik");

        Korotishi pilylkin = Korotishi.createMalish("Pilylkin");

        Korotishi[] korotArray = new Korotishi[]{znayka, avoska, vintik, pilylkin};

        //creat Actions

        CarryAction carryNeznayka = new CarryAction(korotArray, neznayka, "home");

        PutAction putNeznayka = new PutAction(korotArray, neznayka, "on the bed");

        List<Object> objectsOfSeeAction = new ArrayList<Object>();

        objectsOfSeeAction.add("on the left");

        objectsOfSeeAction.add("on the right");

        Window window1 = new Window(StatusOpenClose.Open);

        JumpInWindow neznaykaJumpsInWindow = new JumpInWindow(neznayka, window1);

        Book bookZnayka = new Book("first", "journey");

        StoryOfBook storyFirst = bookZnayka.new StoryOfBook("journey");

        StoryOfBook storySecond = bookZnayka.new StoryOfBook("love");

        StoryOfBook storyThird = bookZnayka.new StoryOfBook("journey");

        ThinkAction znaykaThinkFirst = new ThinkAction(znayka, 3, 3, "make a big ball");

        StoryOfBook[] arrayOfStory = new StoryOfBook[]{storyFirst, storySecond, storyThird};

        ReadBookAction znaykaRead = new ReadBookAction(znayka, bookZnayka);

        TellStoryAction znaykaTellStory = new TellStoryAction(znayka, Korotishi.malishi, arrayOfStory);

        //print story

        carryNeznayka.print();

        putNeznayka.print();

        Eyes eyesOfNeznayka = new Eyes(neznayka, StatusOpenClose.Open);

        try{

            SeeAction neznaykaSee = new SeeAction(neznayka, eyesOfNeznayka);

            neznaykaSee.printSeeAction(objectsOfSeeAction);

        }

        catch(RuntimeException e){

            System.out.println("RuntimeException: " + e.getMessage());

            System.out.println("Please, check the correspondence of Korotish and Eyes");

        }

        catch(EyesCloseException e){

            System.out.println("Exception: " + e.getMessage());

            System.out.println("Eyes is closed");

        }

        try{

            neznaykaJumpsInWindow.jumpInWindow();

        }

        catch(WindowCloseException e){

            System.out.println("The window is closed");

            OpenCloseWindowAction openWindow = new OpenCloseWindowAction(neznayka, window1);

            openWindow.openWindow();

            neznaykaJumpsInWindow.jumpInWindow();

        }

        znaykaRead.printReadAction();

        znaykaTellStory.tellStories();

        znaykaThinkFirst.print();

    }

}

Вывод программы

Znayka, Avoska, Vintik, Pilylkin carry Neznayka home.

Znayka, Avoska, Vintik, Pilylkin put Neznayka on the bed.

Neznayka see on the left.

Neznayka see on the right.

Neznayka jumps out in window.

Znayka reads book "first".

Znayka tells story about journey to Malishi.

Malishi likes story about journey

Znayka tells story about love to Malishi.

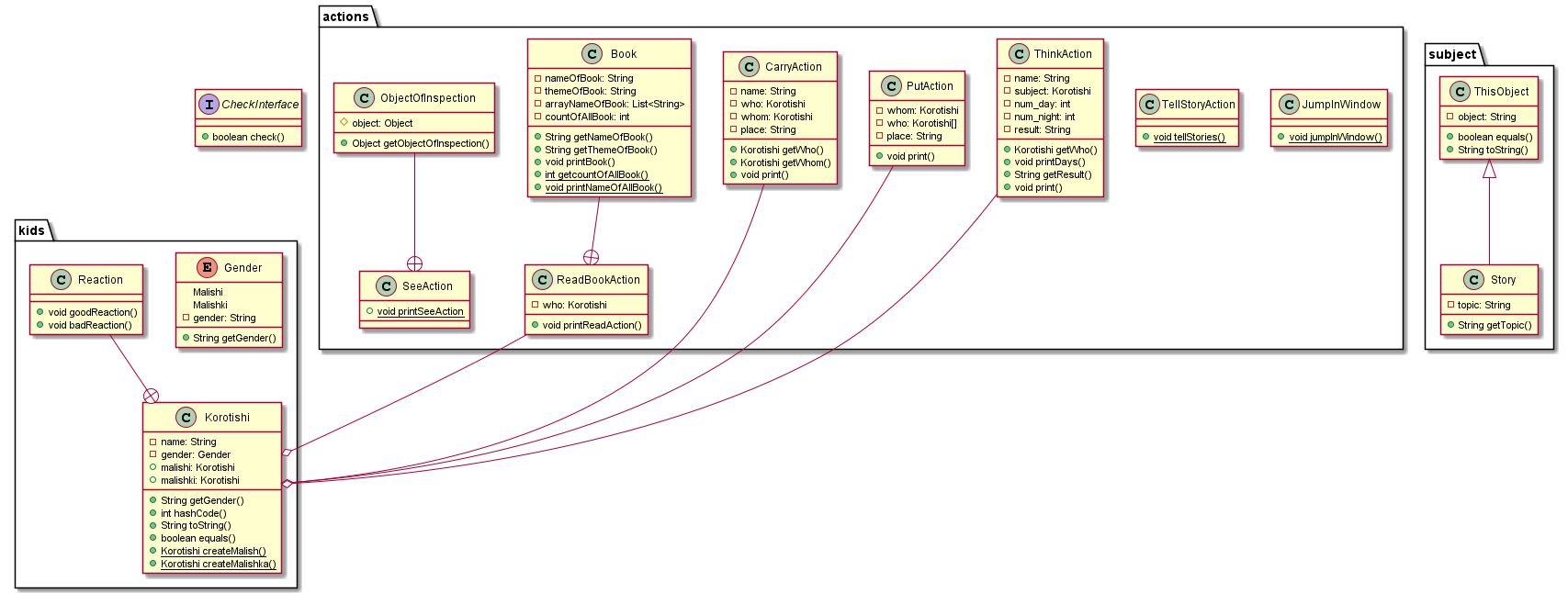
Malishi doesn't like story about love

Znayka tells story about journey to Malishi.

Malishi likes story about journey

Znayka think. He has been thinking for 3 days and 3 night. Result - make a big ball.

UML



# Итоги

В результате выполнения лабораторной работы были изучены вложенный, внутренний, локальный и анонимный классы. Была доработанная программа, в которую были включены эти классы.