Университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Программирование»

**Отчет**

По лабораторной работе №4

Вариант 130130.5

Студент:

Казаев М. П.

*P3111*

Преподаватель:

Харитонова А. Е.

Санкт-Петербург, 2023 г.

**Текст задания:**

Они вошли в дом, и когда Крошка Ру сказал "Здравствуй, Пух", и "Здравствуй, Пятачок" (по одному разу), и "Здравствуй, Тигра" (два раза, потому что это очень забавно звучало и, кроме того, он ведь никогда еще так не здоровался), они рассказали Кенге, зачем они пришли, и Кенга очень ласково сказала: "Ну что ж, милый Тигра, загляни в мой буфет и посмотри-- что тебе там понравится". Ведь Кенга сразу поняла, что, хотя с виду Тигра очень большой, он так же нуждается в ласке, как и Крошка Ру. Но чем больше Тигра совал то свой нос, то лапу, то в одну, то в другую банку, тем больше он находил вещей, которые Тигры не любят. И когда он перерыл весь буфет и нашел все, что там было, и оказалось, что он ничего этого есть не может, он спросил Кенгу: Но Кенга, и Кристофер Робин, и Пятачок-- все стояли вокруг Крошки Ру, уговаривая его принять рыбий жир. И Ру говорил: "Может, не надо?"-- а Кенга говорила: "Ну-ну, милый Ру, вспомни, что ты мне обещал". Тигра подошел поближе и наклонился над спинкой кресла Ру. И вдруг он высунул язык, послышалось громкое "буль-буль", и, подскочив от удивления, Кенга вскрикнула: "Ох!"-- и ухватила ложку как раз в ту секунду, когда она уже исчезала в пасти Тигры. Ложку она спасла, но рыбий жир исчез. Тут Тигра посмотрел на потолок, закрыл глаза, и язык его пошел ходить кругами вокруг мордочки, на тот случай, если что-нибудь осталось снаружи. Затем его озарила умиротворенная улыбка, и он сказал: Теперь нас не удивит, что он поселился в доме у Кенги и всегда получал рыбий жир на завтрак, обед и ужин. Иногда (когда Кенга считала, что ему нужно подкрепиться) он вместо лекарства принимал ложку-другую кашки, которой завтракал Ру.

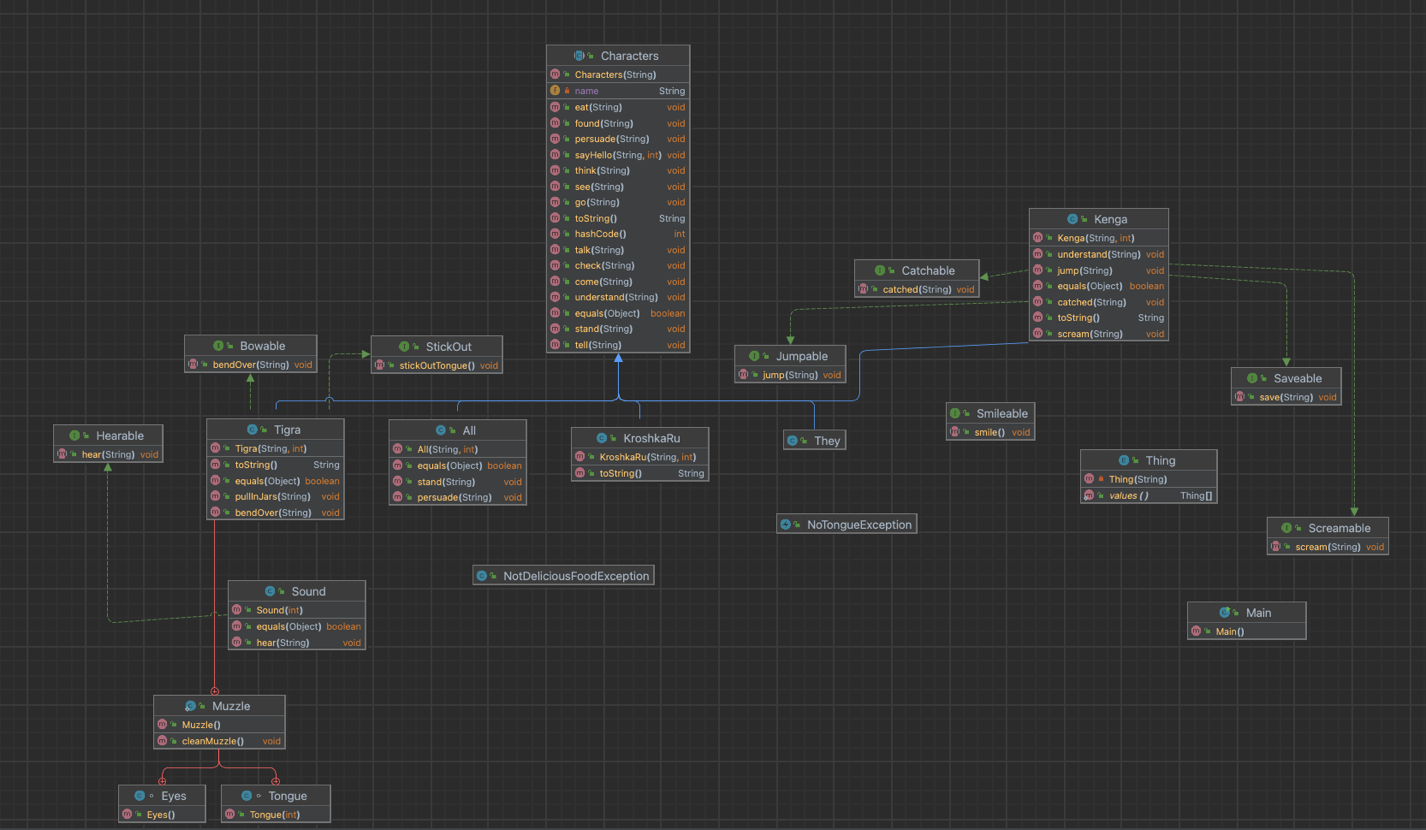
**Программа должна удовлетворять следующим требованиям:**

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

**Порядок выполнения работы:**

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

**UML-диаграмма классов**



**Программа:**

public class All extends Characters {

private int cnt;

public All(String name, int cnt) {

super(name);

}

@Override

public void stand(String place) {

System.out.println("Персонажи " + super.getName() + " стояли " + place);

}

@Override

public void persuade(String things) {

System.out.println("Персонажи " + getName() + " уговаривали " + things);

}

@Override

public String toString() {

return "Кенга, Кристофер Робин и Пятачок" + cnt;

}

@Override

public boolean equals(Object obj) {

if (obj == this) {

return true;

}

if (obj == null || obj.getClass() != getClass()) {

return false;

}

All all = (All) obj;

return cnt == all.cnt;

}

}

public interface Bowable {

void bendOver(String place);

}

public class They extends Characters{

public They(String name) {

super(name);

}

}

public interface Catchable {

void catched(String something);

}

public interface Hearable {

void hear(String sound);

}

public interface Jumpable {

void jump(String reason);

}

public class Kenga extends Characters implements Jumpable, Saveable, Screamable, Catchable {

private int age;

public Kenga(String name, int age) {

super(name);

}

@Override

public void catched(String something) {

System.*out*.println("Персонаж Кенга схватил " + something);

}

@Override

public void jump(String reason) {

System.*out*.println("Персонаж Кенга подскочил от " + reason);

}

@Override

public void save(String thing) {

System.*out*.println("Персонаж Кенга спас " + thing);

}

@Override

public void scream(String word) {

System.*out*.println("Персонаж Кенга вскрикнул " + word);

}

@Override

public void understand(String thing) {

System.*out*.println("Персонаж Кенга понял " + thing);

}

@Override

public String toString() {

return "Персонаж Кенга" + age;

}

@Override

public boolean equals(Object obj) {

if (obj == this) {

return true;

}

if (obj == null || obj.getClass() != getClass()) {

return false;

}

Kenga kenga = (Kenga) obj;

return age == kenga.age;

}

}

public class KroshkaRu extends Characters {

private int age;

public KroshkaRu(String name, int age) {

super(name);

}

@Override

public String toString() {

return "Крошка РУ" + age;

}

@Override

public boolean equals(Object obj) {

if (obj == this) {

return true;

}

if (obj == null || obj.getClass() != getClass()) {

return false;

}

KroshkaRu kroshkaRu = (KroshkaRu) obj;

return age == kroshkaRu.age;

}

}

public interface Saveable {

void save(String thing);

}

public interface Screamable {

public void scream(String word);

}

public interface Smileable {

void smile();

}

public interface StickOut {

void stickOutTongue();

}

public enum Thing {

*SPOON*("Объект ложка исчез"),

*FISHFAT*("Объект рыбий жир исчез");

Thing(String s) {

System.*out*.println(s);

}

}

    public NotDeliciousFoodException(String massage) {

        super(massage);

    }

}

public class NoTongueException extends RuntimeException{

    public NoTongueException(String massage){super(massage);}

}

public class Tigra extends Characters implements Bowable, StickOut{

private int age;

public Tigra(String name, int age) {

super(name);

}

@Override

public void bendOver(String place) {

System.*out*.println("Персонаж Тигра наклонился над " + place);

}

@Override

public void stickOutTongue() {

System.*out*.println("Персонаж Тигра высунул язык");

}

@Override

public String toString() {

return "Тигра" + age;

}

@Override

public boolean equals(Object obj) {

if (obj == this) {

return true;

}

if (obj == null || obj.getClass() != getClass()) {

return false;

}

Tigra tigra = (Tigra) obj;

return age == tigra.age;

}

public void pullInJars(String what) {

class Jars{

@Override

public String toString() {

return "в банки";

}

}

Jars jars = new Jars();

System.*out*.println("Персонаж Тигра совал " + what + jars.toString());

}

public static class Muzzle {

class Eyes {

public void close() {

System.*out*.println("Персонаж Тигра закрыл свои глаза");

}

}

Eyes eyes = new Eyes();

class Tongue {

private int len;

public Tongue(int len){

this.len = len;

}

public void lickMuzzle() throws NoTongueException {

if (len > 5) {

System.*out*.println("Язык Персонажа Тигры пошел ходить кругами по его мордочке");

} else {

throw new NoTongueException("Слишком короткий язык(");

}

}

}

Tongue tongue = new Tongue(7);

public void cleanMuzzle() {

eyes.close();

tongue.lickMuzzle();

}

public void satisfiedmuzzle() {

new Smileable() {

@Override

public void smile() {

System.*out*.println("Персонажа Тигру озарила умиротворенная улыбка");

}

}.smile();

}

}

}

public class Main {

public static void main(String[] args) {

Kenga kenga = new Kenga("Кенга", 15);

Tigra tigra = new Tigra("Тигра", 13);

KroshkaRu krosh = new KroshkaRu("Крошка Ру", 6);

Sound sound = new Sound(40);

All all = new All("Кенга, Кристофер Робин и Пятачок", 3);

They they = new They("Они");

they.come("в дом");

krosh.sayHello("Пух", 1);

krosh.sayHello("Пятачок", 1);

krosh.sayHello("Тигра", 2);

they.tell("Кенге зачем они пришли");

kenga.talk("Ну что ж, милый Тигра, загляни в мой буфет и посмотри-- что тебе там понравится");

kenga.understand("хоть Тигра и большой, но он все равно нуждается в ласке");

tigra.pullInJars(" свой нос и лапы ");

tigra.found("все что не нравится тиграм");

tigra.check("все что было в буфете");

all.stand("вокруг");

all.persuade("принять рыбий жир");

krosh.talk("Может не надо");

kenga.talk("Ну-ну, милый Ру, вспомни, что ты мне обещал");

tigra.go("ближе");

tigra.bendOver("спинкой кресла");

tigra.stickOutTongue();

sound.hear("буль-буль");

kenga.jump("удивления");

kenga.scream("Ox!");

kenga.catched("ложку");

Thing spoon = Thing.*SPOON*;

Thing fishFat = Thing.*FISHFAT*;

Tigra.Muzzle muzzle = new Tigra.Muzzle();

tigra.see("на потолок");

try{

muzzle.cleanMuzzle();}

catch (NoTongueException e){

System.*out*.println(e.getMessage());

}

muzzle.satisfiedmuzzle();

tigra.talk("Теперь нас не удивит, что он поселился в доме у Кенги и всегда получал рыбий жир");

kenga.think("нужно подкрепиться");

try {

kenga.eat("каша");

} catch (NotDeliciousFoodException e) {

System.*out*.println(e.getMessage());

}

}

}

**Результат работы программы:**

Персонаж Они пришел в дом

Персонаж Крошка Ру говорит: Персонаж Крошка Руговорит: Здравствуй, Пух

Персонаж Крошка Ру говорит: Персонаж Крошка Руговорит: Здравствуй, Пятачок

Персонаж Крошка Ру говорит: Персонаж Крошка Руговорит: Здравствуй, Тигра

Персонаж Крошка Ру говорит: Персонаж Крошка Руговорит: Здравствуй, Тигра

Персонаж Они рассказал Кенге зачем они пришли

Персонаж Кенга говорит: Ну что ж, милый Тигра, загляни в мой буфет и посмотри-- что тебе там понравится

Персонаж Кенга понял хоть Тигра и большой, но он все равно нуждается в ласке

Персонаж Тигра совал свой нос и лапы в банки

Персонаж Тигра нашел все что не нравится тиграм

Персонаж Тигра обыскал все что было в буфете

Персонажи Кенга, Кристофер Робин и Пятачок стояли вокруг

Персонажи Кенга, Кристофер Робин и Пятачок уговаривали принять рыбий жир

Персонаж Крошка Ру говорит: Может не надо

Персонаж Кенга говорит: Ну-ну, милый Ру, вспомни, что ты мне обещал

Персонаж Тигра пошел ближе

Персонаж Тигра наклонился над спинкой кресла

Персонаж Тигра высунул язык

Послышался звук буль-буль

Персонаж Кенга подскочил от удивления

Персонаж Кенга вскрикнул Ox!

Персонаж Кенга схватил ложку

Объект ложка исчез

Объект рыбий жир исчез

Персонаж Тигра посмотрел на потолок

Персонаж Тигра закрыл свои глаза

Язык Персонажа Тигры пошел ходить кругами по его мордочке

Персонажа Тигру озарила умиротворенная улыбка

Персонаж Тигра говорит: Теперь нас не удивит, что он поселился в доме у Кенги и всегда получал рыбий жир

Персонаж Кенга думает нужно подкрепиться

Персонаж Кенга ест каша

**Вывод:**

Во время выполнения лабораторной работы я научился работать с локальными, анонимными и вложенными классами. Освоил работу с исключениями и создал пару своих собственных исключений.