Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное учреждение высшего образования

Санкт-Петербургский национальный исследовательский университет ИТМО

Факультет информатики и вычислительной техники

По дисциплине:

Программирование

Лабораторная работа № 4

Вариант 96.10

**Выполнили:**

**студент гр. Р3132**

Вали Насибулла

**Преподаватель:**

Блохина Е. Н.

2020

1 семестр

**Задание:**

Снусмумрик показал трубкой на маленькую кастрюльку с горошком, попыхивающую над костром. Поблизости стояла другая кастрюлька с горячим кофе.

-- Но ты небось пьешь только молоко? -- спросил он. Малышка Мю презрительно засмеялась. Не моргнув глазом она проглотила целых две чайных ложечки кофе и потом съела в придачу по крайней мере четыре горошины. Залив огонь водой, Снусмумрик протянул:

-- Ну и ну!

-- А теперь мне снова хочется спать, -- сказала малышка Мю.

-- Но мне больше нравится спать в карманах.

-- Ладно, -- сказал Снусмумрик и сунул ее в карман. --

Главное, что ты знаешь, чего хочешь. Клубок ангорской шерсти она прихватила с собой. А Снусмумрик отправился дальше по прибрежным лугам. Девятый вал, обессилев, присмирел в бухте. Здесь во всей своей красе царило лето. От извержения вулкана остались лишь облака пепла да чудесные темно-багровые закаты, которыми частенько любовался Снусмумрик. Он не имел ни малейшего представления о том, что случилось с его друзьями в Муми-доле, и полагал, что они все также мирно сидят на своей веранде и празднуют день летнего солнцестояния. Иногда ему приходила в голову мысль, что, наверное, Муми-тролль заждался его... Но прежде чем возвратиться, Снусмумрик должен был свести счеты со Сторожем парка. И дело это можно было уладить лишь в день летнего солнцестояния. Завтра все уже будет в порядке.

**Task:**

Snusmumrik pointed with his pipe to a small saucepan with peas puffing over the fire. There was another pot of hot coffee nearby.

“But I suppose you only drink milk? -- he asked. Little Mu laughed scornfully. Without batting an eye, she swallowed two whole teaspoons of coffee and then ate at least four peas in addition. Having flooded the fire with water, Snusmumrik stretched out:

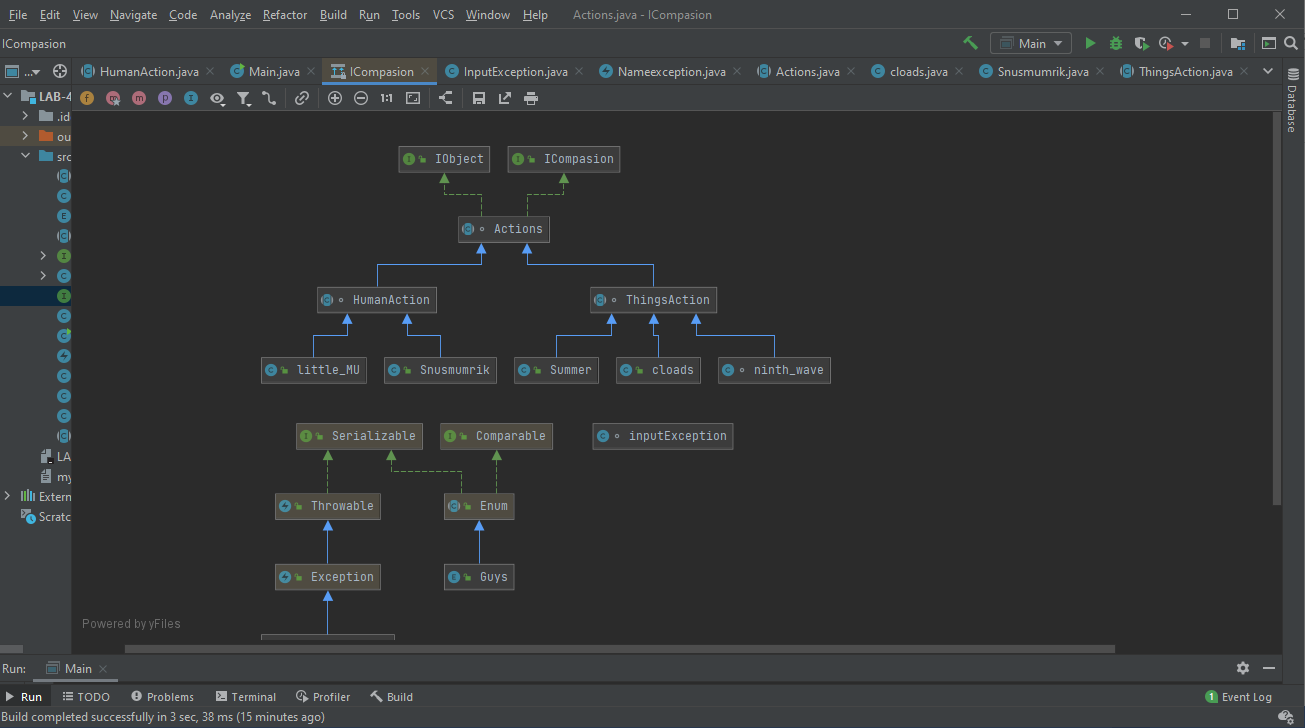
-- Well well!

“And now I want to sleep again,” said little Mu.

“But I like sleeping in my pockets better.

- Okay, - said Snusmumrik and put it in his pocket. -

The main thing is that you know what you want. She took the Angora ball with her. And Snusmumrik went further along the coastal meadows. The ninth wave, exhausted, calmed down in the bay. Summer reigned here in all its glory. From the volcanic eruption, only ash clouds and wonderful dark-purple sunsets, which Snusmumrik often admired, remained. He had no idea what happened to his friends in Moomin, and believed that they were all sitting peacefully on their veranda and celebrating the summer solstice as well. Sometimes it occurred to him that, probably, Moomin was waiting for him ... But before returning, Snusmumrik had to settle scores with the Park Guard. And this matter could be settled only on the day of the summer solstice. Everything will be all right tomorrow.

**Diagram**:

**Codes:**

**Main:**

public class Main {  
 public static void main(String[] args){  
 Snusmumrik snns = new Snusmumrik(Guys.*Humen*, " Snusmumrik ");  
 cloads cload= new cloads(Guys.*Things*," Ash clouds and dark crimson sunsets");  
 ninth\_wave wav= new ninth\_wave(Guys.*Things*,"The\_ninth wave");  
 Summer sum = new Summer(Guys.*Things*,"Summer");  
 little\_MU me= new little\_MU(Guys.*Humen*," Little MU");  
 inputException data= new inputException();  
  
 snns.pointed();  
 snns.asked();  
 me.laughed();  
 me.Swallowed();  
 me.ate();  
 snns.stretched();  
 me.want();  
 me.like();  
 snns.put();  
 me.took();  
 snns.went();  
 wav.exhausted();  
 wav.calmed\_down();  
 sum.regined();  
 cload.eruption();  
 snns.admired();  
 snns.believed();  
 snns.idea();  
 snns.Sattle();  
 data.call();  
 }  
}

**Action:**

abstract class Actions implements IObject, ICompasion {  
 Guys type;  
 String name;  
  
 Actions(Guys p,String n){  
 type=p;  
 name=n;  
 }  
 public Guys getType(){  
  
 return type;  
 }  
 public String getName(){  
 return name;  
 }  
 public String haveName() throws Nameexception{  
 if (name==""){  
 throw new Nameexception("Without name");  
 } else{  
 return name;  
 }  
 }  
 public String toString(){  
 return getName()+type;  
 }  
  
  
  
 public boolean equals(Object obj){  
 return (this.hashCode()==obj.hashCode());  
 }  
  
 public int hashCode() {  
 return getName().hashCode()+type.hashCode();  
  
 }  
}

**HumanAction:**

abstract class HumanAction extends Actions{  
 HumanAction(Guys p, String n){  
 super(p, n);  
 }  
 public void getTypeAndName(){  
 System.*out*.print(getType()+"");  
 System.*out*.print(getName());  
 }  
  
 public void admired(String Admired){  
 this.getTypeAndName();  
 System.*out*.println(Admired);  
 }  
 public void idea(String ide){  
 this.getTypeAndName();  
 System.*out*.println(ide);  
 }  
 public void believed(String belive){  
 this.getTypeAndName();  
 System.*out*.println(belive);  
 }  
 public void Sattle(String sitt){  
 this.getTypeAndName();  
 System.*out*.println(sitt);  
  
 }  
 public void point(String po){  
 this.getTypeAndName();  
 System.*out*.println(po);  
 }  
 public void asked(String ask){  
 this.getTypeAndName();  
 System.*out*.println(ask);  
 }  
 public void laughed(String lau){  
 this.getTypeAndName();  
 System.*out*.println(lau);  
 }  
 public void Swallowed(String Swal){  
 this.getTypeAndName();  
 System.*out*.println(Swal);  
 }  
 public void ate(String eat){  
 this.getTypeAndName();  
 System.*out*.println(eat);  
 }  
 public void stretched(String stre){  
 this.getTypeAndName();  
 System.*out*.println(stre);  
 }  
 public void want(String wan){  
 this.getTypeAndName();  
 System.*out*.println(wan);  
 }  
 public void like(String lak){  
 this.getTypeAndName();  
 System.*out*.println(lak);  
 }  
 public void put(String pu){  
 this.getTypeAndName();  
 System.*out*.println(pu);  
 }  
 public void took(String take){  
 this.getTypeAndName();  
 System.*out*.println(take);  
 }  
 public void went(String go){  
 this.getTypeAndName();  
 System.*out*.println(go);  
 }  
}

**ThingsAction:**

abstract class ThingsAction extends Actions {  
 ThingsAction(Guys p, String n){  
 super(p, n);  
  
 }  
 public void getTypeAndName(){  
 System.*out*.print(getType()+ " ");  
 System.*out*.print(getName());  
 }  
  
  
 public void exhausted(String exhausted){  
 this.getTypeAndName();  
 System.*out*.println(exhausted);  
 }  
  
 public void calmed\_down(String clam){  
 this.getTypeAndName();  
 System.*out*.println(clam);  
 }  
  
 public void reigned(String reign){  
 this.getTypeAndName();  
 System.*out*.println(reign);  
 }  
  
 public void eruption(String erup){  
 this.getTypeAndName();  
 System.*out*.println(erup);  
 }  
  
  
  
  
  
  
  
}

**Guys:**

public enum Guys {  
  
 *Humen* ("HUMAN"),  
 *Things* ("Things");  
 private final String Type;  
 Guys(String t){  
 Type =t;  
 }  
 public String tostring(){  
 return Type;  
 }  
  
  
}

**Icompasion:**

public interface ICompasion {  
 public int hashCode();  
 public String toString();  
 public boolean equals(Object obj);  
}

**IObject:**

public interface IObject {  
 Guys getType();  
 String getName();  
  
 }

**cloads:**

public class cloads extends ThingsAction{  
  
 cloads(Guys p, String n) {  
 super(p, n);  
 }  
 public void eruption(){  
 super.eruption(" created From the vocanic eruption");  
 }  
}

**IOexception:**

import java.io.FileOutputStream;  
import java.io.PrintStream;  
  
class inputException {  
  
 public void call() {  
 FileOutputStream out; // declare a file output object  
 PrintStream p; // declare a print stream object  
  
 try  
 {  
 // Create a new file output stream  
 // connected to "myfile.txt"  
 out = new FileOutputStream("myfile.txt");  
  
 // Connect print stream to the output stream  
 p = new PrintStream( out );  
  
 p.append ("this is my file");  
  
 p.close();  
 }  
 catch (Exception e)  
 {  
 System.*out*.println ("Error writing to file");  
 }  
 }  
}

**little\_MU**

public class little\_MU extends HumanAction{  
 little\_MU(Guys p, String n){  
 super(p, n);  
 }  
   
 public void laughed(){  
 super.laughed(" laughed scornfolly ");  
 }  
 public void Swallowed(){  
 super.Swallowed(" Swallowed two whole teaspoons of coffee without batting an eye");  
 }  
 public void ate(){  
 super.ate(" ate at least four peas");  
 }  
 public void want(){  
 super.want(" want to sleep again");  
 }  
 public void like(){  
 super.like(" like sleeping in pocket");  
 }  
 public void took(){  
 super.took(" took the Angora ball with her");  
 }  
}

**Nameexception:**

public class Nameexception extends Exception {  
 public Nameexception (String description) {  
 super(description);  
 }  
}

**ninth\_wave:**

class ninth\_wave extends ThingsAction{  
 ninth\_wave(Guys p, String n){  
  
 super(p, n);  
 }  
 public void exhausted(){  
 super.exhausted(" exhausted in the bay");  
 }  
 public void calmed\_down(){  
 super.calmed\_down(" clamed down in the bay");  
 }  
}

**Snusmumrik:**

public class Snusmumrik extends HumanAction{  
 Snusmumrik(Guys p, String n){  
 super (p,n);  
 }  
  
 public void admired(){  
 super.admired("often admired and remaind ash clouds and dark crimsons");  
 }  
 public void idea(){  
 super.idea("had no idea what happend to his friends in Moomin");  
 }  
 public void believed(){  
 super.believed("believed that his friends are all sitting peacefully on their veranda and celebrating summer");  
 }  
 public void Sattle(){  
 super.Sattle(" had to settle scores with the Park Guard");  
 }  
 public void pointed(){  
 super.point("pointed with a pipe to a small saucepan.");  
 }  
 public void asked(){  
 super.asked(" asked, Mu if she supposed to drink milk");  
 }  
 public void stretched(){  
 super.stretched("Streteched and said WELL WELL");  
 }  
 public void put(){  
 super.put("put little MU in his pocket");  
 }  
 public void went(){  
 super.went("went further along the coastel meadows");  
 }  
}

**Summer:**

public class Summer extends ThingsAction{  
 Summer (Guys p, String n){  
 super(p,n);  
 }  
 public void regined(){  
 super.reigned(" reigned here in all its glory");  
 }  
}

**Вывод программы**

Humen Snusmumrik pointed with a pipe to a small saucepan.

Humen Snusmumrik asked, Mu if she supposed to drink milk

Humen Little MU laughed scornfolly

Humen Little MU Swallowed two whole teaspoons of coffee without batting an eye

Humen Little MU ate at least four peas

Humen Snusmumrik Streteched and said WELL WELL

Humen Little MU want to sleep again

Humen Little MU like sleeping in pocket

Humen Snusmumrik put little MU in his pocket

Humen Little MU took the Angora ball with her

Humen Snusmumrik went further along the coastel meadows

Things The\_ninth wave exhausted in the bay

Things The\_ninth wave clamed down in the bay

Things Summer reigned here in all its glory

Things Ash clouds and dark crimson sunsets created From the vocanic eruption

Humen Snusmumrik often admired and remaind ash clouds and dark crimsons

Humen Snusmumrik believed that his friends are all sitting peacefully on their veranda and celebrating summer

Humen Snusmumrik had no idea what happend to his friends in Moomin

Humen Snusmumrik had to settle scores with the Park Guard

**Conclusion**

We've learned how the checked and unchecked exception classes work and are used, and how they are implemented. And we practice and enriched our knowledge in OOP concept of java programming.