|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| *Федеральное государственное бюджетное образовательное учреждение высшего образования* ***«МИРЭА – Российский технологический университет»***  **РТУ МИРЭА** |

**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**Дисциплина «Программирование на языке Джава»**

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №9-10**

Выполнил студент группы ИНБО-02-20 Судариков Д.А.

Принял Степанов П.В.

Практические работы выполнены «\_\_\_»\_\_\_\_\_\_\_2021г.

«\_\_\_\_\_\_\_\_\_\_\_\_\_\_» «\_\_\_»\_\_\_\_\_\_\_2021г.

Отметка о выполнении

**Москва – 2021 г.**

## **Задание**

Научиться создавать собственные исключения

Научиться применять порождающие паттерны при разработке программ на Java

## **Ход Работы**

В ходе выполнения работы были получены следующие исходные коды:

package ru.mirea.lab.lab9.ex1;  
  
  
import java.util.HashMap;  
  
  
  
public class INN {  
 static java.util.Scanner *IN* = new java.util.Scanner(System.*in*);  
  
 public static void main(String[] args) {  
  
 var INNBase = new HashMap<String, String>(5);  
  
 INNBase.put("5367687281", "Иванов Иван Иванович");  
 INNBase.put("0567475944", "Петров Василий Петрович");  
 INNBase.put("7992222888", "Пулин Владимир Владимирович ");  
 INNBase.put("3401662695", "Александров Александр Александрович ");  
 INNBase.put("0204012893", "Кулин Евгений Евгеньевич");  
  
 System.*out*.print("Введите ФИО: ");  
 String name = *IN*.nextLine();  
  
 System.*out*.print("Введите ИНН: ");  
 String inn = *IN*.next();  
  
 try {  
 String baseName = INNBase.get(inn);  
  
 if (!name.equals(baseName)) {  
 throw new InvalidINNException("[ERROR]: ИНН \"" + inn + "\" не действителен для пользователя " + name);  
 }  
  
 System.*out*.println("ИНН \"" + inn + "\" действителен для пользователя " + name);  
 }  
 catch (InvalidINNException exc) {  
 System.*err*.println(exc.getMessage());  
 }  
 }  
}

package ru.mirea.lab.lab9.ex1;  
  
public class InvalidINNException extends Exception {  
 InvalidINNException(String errMsg) {  
 super(errMsg);  
 }  
}

package ru.mirea.lab.lab9.ex2;  
  
import java.util.Arrays;  
import java.util.Comparator;  
  
public class Menu {  
 static java.util.Scanner *IN* = new java.util.Scanner(System.*in*);  
  
 public static void main(String[] args) throws StudentNotFoundException {  
 System.*out*.print("Введите количество студентов: ");  
 int length = *IN*.nextInt();  
  
 Student[] Students = new Student[length];  
  
 for (int i = 0; i < length; ++i) {  
 System.*out*.println("Студент №" + (i + 1) + ":");  
 System.*out*.println();  
 System.*out*.print("Введите ФИО студента: ");  
 *IN*.nextLine();  
 String fio = *IN*.nextLine();  
 System.*out*.print("Введите средний балл студента: ");  
 int gpa = *IN*.nextInt();  
 Students[i] = new Student(i, gpa, fio);  
 }  
  
 System.*out*.println("========================================");  
  
 while (true) {  
 System.*out*.println(" 1) Получить информацию о студенте;");  
 System.*out*.println(" 2) Вывести список студентов;");  
 System.*out*.println(" 3) Отсортировать список студентов;");  
 System.*out*.println(" 4) Завершить работу.");  
  
 int choice = *IN*.nextInt();  
  
 *IN*.nextLine();  
 System.*out*.println();  
  
 switch (choice) {  
 case 1:  
 *getStudent*(Students);  
 System.*out*.println();  
 break;  
  
 case 2:  
 *print*(Students);  
 System.*out*.println();  
 break;  
  
 case 3:  
 *sort*(Students);  
 *print*(Students);  
  
 System.*out*.println();  
 break;  
  
 case 4:  
 return;  
 }  
 }  
 }  
  
 public static int find(Student[] students, String name) throws StudentNotFoundException {  
 for (int i = 0; i < students.length; ++i) {  
 if (students[i].getName().equals(name))  
 return i;  
 }  
  
 throw new StudentNotFoundException("!ERROR!: Студента с ФИО " + name + " нет в списке");  
 }  
 private static void getStudent(Student[] students) throws StudentNotFoundException {  
 System.*out*.print("Введите ФИО студента: ");  
 String name = *IN*.nextLine();  
 System.*out*.println(students[*find*(students, name)]);  
 }  
  
 private static void print(Student[] students) {  
 System.*out*.printf("%3s|%50s|%14s\n", "ID", "ФИО", "Средний балл");  
 System.*out*.println("---------------------------------------------------------------------");  
  
 for (Student stud : students) {  
 System.*out*.printf("%3d|%50s|%14f\n", stud.getiD(), stud.getName(), stud.getGPA());  
 }  
 }  
 public static void sort(Student[] students){  
 Comparator<Student> scomp = new StudentGPAComparator();  
 Arrays.*sort*(students, scomp);  
 }  
}

package ru.mirea.lab.lab9.ex2;  
  
import java.util.Arrays;  
import java.util.Comparator;  
  
  
public class SortingStudentsByGPA extends Student {  
 public static void print(Student[] students){  
 for (int i = 0; i < 5; i++) {  
 System.*out*.println(students[i].getiD() + " " + students[i].getGPA());  
 }  
 }  
  
 public static void sort(Student[] students){  
 Comparator<Student> scomp = new StudentGPAComparator();  
 students[0] = new Student(64352, 245);  
 students[1] = new Student(5234, 223);  
 students[2] = new Student(1, 221);  
 students[3] = new Student(666, 290);  
 students[4] = new Student(777, 255);  
 String[] strs = {"gffgd","sgdfdgfdff","agggg"};  
 int[] x = {3434,5,2,1,-9999};  
 Arrays.*sort*(students, scomp);  
 *print*(students);  
 }  
  
 public static void main(String[] args) {  
 Student[] students = new Student[5];  
 *sort*(students);  
  
  
  
 }  
}  
  
class StudentGPAComparator implements Comparator<Student>{  
 public int compare(Student a, Student b){  
 if(a.getGPA()>b.getGPA())  
 return 1;  
 else if(a.getGPA()<b.getGPA())  
 return -1;  
 else return 0;  
 }  
}

package ru.mirea.lab.lab9.ex2;  
  
import java.util.Arrays;  
import java.util.Comparator;  
  
  
public class SortingStudentsByGPA extends Student {  
 public static void print(Student[] students){  
 for (int i = 0; i < 5; i++) {  
 System.*out*.println(students[i].getiD() + " " + students[i].getGPA());  
 }  
 }  
  
 public static void sort(Student[] students){  
 Comparator<Student> scomp = new StudentGPAComparator();  
 students[0] = new Student(64352, 245);  
 students[1] = new Student(5234, 223);  
 students[2] = new Student(1, 221);  
 students[3] = new Student(666, 290);  
 students[4] = new Student(777, 255);  
 String[] strs = {"gffgd","sgdfdgfdff","agggg"};  
 int[] x = {3434,5,2,1,-9999};  
 Arrays.*sort*(students, scomp);  
 *print*(students);  
 }  
  
 public static void main(String[] args) {  
 Student[] students = new Student[5];  
 *sort*(students);  
  
  
  
 }  
}  
  
class StudentGPAComparator implements Comparator<Student>{  
 public int compare(Student a, Student b){  
 if(a.getGPA()>b.getGPA())  
 return 1;  
 else if(a.getGPA()<b.getGPA())  
 return -1;  
 else return 0;  
 }  
}

}

package ru.mirea.lab.lab9.ex2;  
  
  
public class Student {  
 private int iD;  
 private double GPA;  
 private String name;  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public Student(int iD, int GPA, String name) {  
 this.iD = iD;  
 this.GPA=GPA;  
 this.name = name;  
  
 }  
  
 public Student(int iD, int GPA) {  
 this.iD = iD;  
 this.GPA = GPA;  
 }  
  
 public Student() {  
  
 }  
  
 public int getiD() {  
 return iD;  
 }  
  
 public void setiD(int iD) {  
 this.iD = iD;  
 }  
  
 public double getGPA() {  
 return GPA;  
 }  
  
 public void setGPA(int GPA) {  
 this.GPA = GPA;  
 }  
  
  
 @Override  
 public String toString() {  
 return "Student{" +  
 "iD=" + iD +  
 ", GPA=" + GPA +  
 ", name='" + name + '\'' +  
 '}';  
 }  
}

package ru.mirea.lab.lab9.ex2;  
  
public class StudentNotFoundException extends Exception {  
 public StudentNotFoundException(String errMsg) {  
 super(errMsg);  
 }  
}

**10 практика.** В ходе выполнения работы были получены следующие исходные коды:

package ru.mirea.lab.lab10.ex1;  
  
public class Complex {  
 private int re; // вещественная часть  
 private int im; // мнимая часть  
  
 public Complex(int re, int im) {  
 this.re = re;  
 this.im = im;  
 }  
  
 public int getRe() {  
 return this.re;  
 }  
  
 public int getIm() {  
 return this.im;  
 }  
  
 public void setRe(int re) {  
 this.re = re;  
 }  
  
 public void setIm(int im) {  
 this.im = im;  
 }  
  
 @Override  
 public String toString() {  
 return this.re + " + " + this.im + "i";  
 }  
}

package ru.mirea.lab.lab10.ex1;  
  
  
public interface ComplexAbstractFactory {  
  
 Complex createComplex();  
 Complex createComplex(int real, int image);  
}

package ru.mirea.lab.lab10.ex1;  
  
public class ConcreteFactory implements ComplexAbstractFactory {  
  
 public Complex createComplex() {  
 return new Complex(0, 0);  
 }  
  
 public Complex createComplex(int real, int image) {  
 return new Complex(real, image);  
 }  
}

package ru.mirea.lab.lab10.ex1;  
  
import java.util.Scanner;  
public class Test {  
 private static final Scanner *IN* = new Scanner(System.*in*);  
 private static final ConcreteFactory *FACTORY* = new ConcreteFactory();  
  
 public static void main(String[] args) {  
 System.*out*.print("Введите вещественную часть: ");  
 int real = *IN*.nextInt();  
  
 System.*out*.print("Введите мнимую часть: ");  
 int image = *IN*.nextInt();  
  
 Complex ComplexNumber11 = *FACTORY*.createComplex();  
 Complex ComplexNumber12 = *FACTORY*.createComplex(real, image);  
  
 System.*out*.println("ComplexNumber11 = " + ComplexNumber11);  
 System.*out*.println("ComplexNumber12 = " + ComplexNumber12);  
 }  
}

package ru.mirea.lab.lab10.ex2;  
  
public interface AbstractChairFactory {  
  
 VictorianChair createVictorianChair(int age);  
  
 MagicalChair createMagicalChair();  
  
 FunctionalChair createFunctionalChair();  
}

package ru.mirea.lab.lab10.ex2;  
  
public interface Chair {  
 String toString();  
}

package ru.mirea.lab.lab10.ex2;  
  
public class ChairFactory implements AbstractChairFactory {  
  
 public VictorianChair createVictorianChair(int age) {  
 return new VictorianChair(10);  
 }  
  
  
 public MagicalChair createMagicalChair() {  
 return new MagicalChair();  
 }  
  
  
 public FunctionalChair createFunctionalChair() {  
 return new FunctionalChair();  
 }  
}

package ru.mirea.lab.lab10.ex2;  
  
import java.util.Scanner;  
  
public class Client {  
  
 public Chair chair;  
  
 public void sit() throws Exception {  
 var scan = new Scanner(System.*in*);  
 var fact = new ChairFactory();  
  
 System.*out*.println("Какой стул создать?");  
 System.*out*.println(" 1) Викторианский стул");  
 System.*out*.println(" 2) Магический стул");  
 System.*out*.println(" 3) Функциональный стул");  
  
 int k = scan.nextInt();  
  
 System.*out*.println();  
  
 switch (k) {  
 case 1:  
 System.*out*.print("Введите возраст стула: ");  
 chair = fact.createVictorianChair(scan.nextInt());  
 System.*out*.println(chair);  
 break;  
  
 case 2:  
 chair = fact.createMagicalChair();  
 ((MagicalChair) chair).doMagic();  
 break;  
  
 case 3:  
 chair = fact.createFunctionalChair();  
 ((FunctionalChair) chair).doFunction();  
 break;  
 }  
  
 }  
}

package ru.mirea.lab.lab10.ex2;  
  
public class FunctionalChair implements Chair {  
  
 public void doFunction() throws Exception {  
 System.*out*.println("Функциональный стул очень удобен\*");  
 throw new Exception("[ERROR]: Функциональный стул сломался");  
 }  
  
 @Override  
 public String toString() {  
 return "FunctionalChair очень удобен";  
 }  
}

package ru.mirea.lab.lab10.ex2;  
  
public class MagicalChair implements Chair {  
  
 public void doMagic(){  
 System.*out*.println("Магия: \*происходит\*");  
 }  
  
 @Override  
 public String toString() {  
 return """  
 MagicalChair {  
 Вы телепортированы в подземелье  
 }""";  
 }  
}

package ru.mirea.lab.lab10.ex2;  
  
public class Test {  
 public static void main(String[] args) {  
 Client cl = new Client();  
 try {  
 cl.sit();  
 } catch (Exception exc) {  
 System.*out*.println(exc.getMessage());  
 }  
 }  
}

package ru.mirea.lab.lab10.ex2;  
  
public class VictorianChair implements Chair {  
 private final int age;  
  
 public VictorianChair(int age) {  
 this.age = age;  
 }  
  
 public int getAge() {  
 return this.age;  
 }  
  
 @Override  
 public String toString() {  
 return (  
 "VictorianChair {\n"  
 + " age: " + this.age + "\n"  
 + "}"  
 );  
 }  
}

## **Вывод**

В данной практической работе мной были получены знания по созданию собственных исключений. Также мной были получены знания по работе AbstractFactory.