Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №3

Exceptions.

Виконали:
студенти групи ІА-23
Пожар Д. Ю.
Хохол М. В.
Тюска А. Ю.

Перевів:
Колеснік В. М.

Київ 2023

Хід роботи:
1. Ознайомитись з javadoc для наступних класів:
- Throwable;
- Error;
- Exception;
- RuntimeException;

а також повторити конструкції try-catch-finally, throw, throws.

2. Проаналізувати предметну область з л/р 10-12. Створити свій власний тип checked exception (підклас класу Exception), що описує порушення одного з бізнес-правил предметної області (наприклад, спробу додати студента у групу, в якій вже навчається максимально допустима кількість студентів, або пуста "" назва фільму). Додати throws-декларацію цього exception до відповідних методів, у яких потрібно «викидати» його у разі порушення бізнес-правил. Додати блок try-catch-finally для демонстрації виклику цього метода. Продемонструвати «викидання» та обробку цього exception. Також додати та продемонструвати код для «викидання» кількох стандартних RuntimeException (IllegalArgumentException, NullPointerException, ...).

3. Відповісти на контрольні питання.

Class Main

```
import java.util.*;
import java.util.Arrays;


public class Main {


    public static void main(String[] args) {
        Institute inst = new Institute();



        HashMap<String, String> fict = new HashMap<>();
        HashMap<String, String> fbmi = new HashMap<>();
        HashMap<String, String> fsp = new HashMap<>();



        Student student1 = new Student("Andrii", "Tiuska",
"11111_24", 80.3);
        Student student2 = new Student("Maxim", "Khokhol",
"11111_23", 98.7);
        Student student3 = new Student("David", "Pozhar", "22222_25",
96.5);
        Student student4 = new Student("Yevhenii", "Voroniuk",
"22222_4", 95);
        Student student5 = new Student("Vadim", "Voloshyn",
"22222_23", 100);
        Student student6 = new Student("Serhii", "Kalyna", "33333_2",
87);
        Student student11 = new Student("Vova", "Ragnarok",
```

```
"22222_4", 5);
        inst.addStudents(Arrays.asList(student1, student2, student3,
student4, student5, student6, student11));

        try {
            putStudents(inst, fict, "fict");
            putStudents(inst, fbmi, "fbmi");
            putStudents(inst, fsp, "fsp");
        } catch (NumberOfStudentsException ex) {
            System.out.println(ex.getMessage());
        } finally {
            System.out.println("I am code that is executed in any
case");
        }







        System.out.println("Students with an average mark of 95-
100:");
        task3(inst);

        System.out.print("Total amount of all students in institute:
");
        System.out.println(task1(inst));
        System.out.print("The largest faculty: ");
        System.out.println(whatIsTheLargestFaculty(fict, fbmi, fsp));

    }


    public static void putStudents (Institute institute,
HashMap<String, String> faculty, String name)
            throws NumberOfStudentsException{

        Set<Object> keyAndValue = new HashSet<>();
        for (Student student : institute.students) {
            String key = student.id;
            String value = student.name + " " + student.surname;
            if(institute.students.size() > 5) {
                throw new NumberOfStudentsException("The number of
students is too big, our groups cannot accommodate this count!!");
            } else {
                keyAndValue.add(key.split("_")[0]);
                if (keyAndValue.contains("11111") &&
name.equals("fbmi")) faculty.put(key, value);
                if (keyAndValue.contains("22222") &&
name.equals("fict")) faculty.put(key, value);
                if (keyAndValue.contains("33333") &&
name.equals("fsp")) faculty.put(key, value);
                keyAndValue.clear();
            }
```

```
        }
    }


    public static String whatIsTheLargestFaculty(HashMap<String,
String> fict, HashMap<String, String> fbmi, HashMap<String, String>
fsp)
            throws IllegalArgumentException{
        TreeSet<Integer> treeSet = new TreeSet<>();
        Faculty facultyFict = new Faculty("22222", "FICT");
        Faculty facultyFbmi = new Faculty("11111", "FBMI");
        Faculty facultyFsp = new Faculty("33333", "FSP");
        int countOfStudentsInFict = fict.size();
        int countOfStudentsInFbmi = fbmi.size();
        int countOfStudentsInFsp = fsp.size();
        treeSet.add(countOfStudentsInFict);
        treeSet.add(countOfStudentsInFbmi);
        treeSet.add(countOfStudentsInFsp);
        if(treeSet.last() == 0) {
            throw new IllegalArgumentException("faculties don't
accept students");
        }
        if (treeSet.last() == countOfStudentsInFict) return
facultyFict + ". Count Of Students: " + fict.size();
        if (treeSet.last() == countOfStudentsInFbmi) return
facultyFbmi + ". Count Of Students: " + fict.size();
        if (treeSet.last() == countOfStudentsInFsp) return facultyFsp
+ ". Count Of Students: " + fict.size();
        return "The largest faculty does not exist";
    }
    public static void task3(Institute institute) {
        for (Student x : institute.students) {
            if (x.averageMark >= 95) {
                System.out.print(x.name + " " + x.surname);
                System.out.println(" ");
            }
        }
    }

    public static int task1(Institute institute) throws
UnsupportedOperationException{
        int count = 0;
        Iterator iterator = institute.students.iterator();

        while (iterator.hasNext()) {
            count++;
            iterator.next();
        }
        if(count == 0){
            throw new UnsupportedOperationException("Oops, reconsider
the number of students at the university");
        }
        return count;
    }
```

```
}
class NumberOfStudentsException extends Exception{
    public NumberOfStudentsException(String message){
        super(message);
    }
}
```

Class Student

```
import java.util.Objects;

public class Student{
    String name;
    String surname;
    String id;
    double averageMark;

    public Student(String name, String surname, String id, double
averageMark){
        this.name = name;
        this.surname = surname;
        this.id = id;
        this.averageMark = averageMark;

    }

    @Override
    public boolean equals(Object anotherObject){
        if(this == anotherObject) return true;
        if (anotherObject == null || getClass() !=
anotherObject.getClass()) return false;
        Student student = (Student) anotherObject;
        return id.equals(student.id);
    }
    @Override
    public int hashCode(){
        return Objects.hash(id);
    }
}
```

Class Faculty

```
import java.util.*;

public class Faculty  extends  Institute{
    String idFaculty;
    String nameFaculty;

    public Faculty(String idFaculty, String nameFaculty){
        this.idFaculty = idFaculty;
        this.nameFaculty = nameFaculty;
    }
```

```
    public void addStudents(Collection<Student> studentCollection){
        this.students.addAll(studentCollection);
    }

    @Override
    public String toString(){
        return "Faculty name is " + nameFaculty + ". The Faculty ID:
" + idFaculty;
    }
}
```

Class Institute

```
import java.util.*;

public class Institute {

    HashSet<Student> students = new HashSet<>();


    public void addStudents(Collection<Student> studentCollection){
        this.students.addAll(studentCollection);
    }
}
```

Результат роботи програми:

```
The number of students is too big, our groups cannot accommodate this count!!
I am code that is executed in any case
Students with an average mark of 95-100:
Vadim Voloshyn
David Pozhar
Yevhenii Voroniuk
Maxim Khokhol
Total amount of all students in institute: 6
The largest faculty: Exception in thread "main" java.lang.IllegalArgumentException: faculties don't accept students
    at Main.whatIsTheLargestFaculty(Main.java:87)
    at Main.main(Main.java:49)
```

Висновок

Виконавши лабораторну роботу ми створили свій власний тип checked exception. Також пригадали та додали блок try-catch-finally до нашої попередньої роботи.