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

Лабораторна робота №13

із дисципліни «*Основи програмування*»

Тема: «Exceptions**»**

Виконали:
Студенти групи ІА-24
(бригада №1)
Котлярчук М.С.
Коханчук М.М.
Чайка А.П.

Перевірив:
Колеснік Валерій Миколайович

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.

Результати+код:

```
import java.util.*;

public class Main {
    public static void main(String[] args) throws FacultyException {
        dataBase();
        task1();
        task2(Institute.getAllFaculties());
        task3();
    }

    1 usage
    static void task1() {
        System.out.println("Number of students in institute:");
        int allSt = 0;
        for (Iterator<Student> i = Faculty.getAllStudents().iterator(); i.hasNext();) {
            i.next();
            allSt++;
        }
        System.out.println(allSt);
    }

    1 usage
    static void task2(HashMap<Integer, String> map) {
        System.out.println("-----------------------------------");
        System.out.println("The most popular faculty:");
        Integer val = 0;
        HashSet<Integer> num = new HashSet<>();
```

```
        for (Map.Entry<Integer, String> entryKey : Institute.getAllFaculties().entrySet()) {
            num.add(entryKey.getKey());
            for (Integer i : num) {
                if (val < i) {
                    val = i;
                }
            }
        }
        System.out.println(map.remove(val));
    }

    1 usage
    static void task3() {
        System.out.println("-------------------------------------");
        System.out.println("Students with average mark in range of 95-100");
        HashSet<Student> stud = new HashSet<>();
        for (Student i : Faculty.getAllStudents()) {
            if (i.getAverageMark() >= 95 && i.getAverageMark() <= 100) {
                stud.add(i);
            }
        }
        for (Student i : stud) {
            System.out.println(i.fullName());
        }
    }
```

```
    public static void dataBase() throws FacultyException {
        Student student1 = new Student("Anton", "Chayka", "10001", 96);
        Student student2 = new Student("Daria", "Sidenko", "10002", 95);
        Student student3 = new Student("Maxim", "Kotlyarchuk", "10003", 98);
        Student student4 = new Student("Anna", "Orlovska", "10004", 90);
        Student student5 = new Student("Yulia", "Meleshko", "10005", 89);
        Student student6 = new Student("Danya", "Yablonskiy", "10006", 99.5);
        Student student7 = new Student("Sanya", null, "10007", 67);
        Student student8 = new Student("Artem", "Kliminskiy", "10008", 58);
        Student student9 = new Student("Ivan", "Paliychuk", "10009", 27);
        Student student10 = new Student("Sergiy", "Ivanov", "100010", 105);
        Student student11 = new Student("Kostya", "Isaev", "10011", 83);
        Student student12 = new Student("Roman", "Krasnoshapka", "10012", 19);

        HashSet<Student> fictSpace = new HashSet<>();
        HashSet<Student> fpmSpace = new HashSet<>();

        Faculty.addStudent(student1, fictSpace);
        Faculty.addStudent(student2, fictSpace);
        Faculty.addStudent(student3, fictSpace);
        Faculty.addStudent(student4, fictSpace);
        Faculty.addStudent(student5, fictSpace);
        Faculty.addStudent(student11, fictSpace);
        Faculty.addStudent(student6, fpmSpace);
        Faculty.addStudent(student7, fpmSpace);
        Faculty.addStudent(student8, fpmSpace);
        Faculty.addStudent(student9, fpmSpace);
```

```
        Faculty.addStudent(student9, fpmSpace);
        Faculty.addStudent(student10, fictSpace);
        Faculty.addStudent(student12, fictSpace);

        System.out.println("-------------------------------------");

        Faculty.getAllStudents().addAll(fictSpace);
        Faculty.getAllStudents().addAll(fpmSpace);
        Institute.getAllFaculties().put(fpmSpace.size(), "FPM");
        Institute.getAllFaculties().put(fpmSpace.size(), "FIOT");

    }
}
```

```java
import java.util.*;

4 usages
public class Institute {
    2 usages
    String name;
    1 usage
    static HashMap<Integer, String> putFaculties = new HashMap<>();

    public Institute(String name) { this.name = name; }

    public String getName() { return name; }
    4 usages
    public static HashMap<Integer, String> getAllFaculties() { return putFaculties; }

}
```

```java
6 usages
public class FacultyException extends Exception {
    2 usages
    public FacultyException(String message) {
        super(message);
    }
}
```

```java
import java.util.*;
16 usages
public class Faculty{
    1 usage
    HashSet<Student> space;
    1 usage
    Integer quantity;
    1 usage
    static HashSet<Student> allStudents = new HashSet<>();

    public Faculty(HashSet<Student> space, Integer quantity){
        this.space = space;
        this.quantity = quantity;
    }
    4 usages
    public static HashSet<Student> getAllStudents() { return allStudents; }
    12 usages
    public static void addStudent(Student student, HashSet<Student> faculty) {
        try{
            if (faculty.size() >= 6) {
                throw new ArrayIndexOutOfBoundsException("Can't add " +student.fullName()+ " - faculty is full");
            } else {
                if (student.getAverageMark() < 30) {
                    throw new FacultyException(student.fullName() + " failed the session.");
                } else if (student.getAverageMark() > 30 & student.getAverageMark() < 60) {
                    throw new FacultyException(student.fullName() + " needs to pass the exam.");
                } else if (student.getAverageMark() > 100 || student.getAverageMark() < 0) {
                    throw new IllegalArgumentException("Invalid data given.");
                } else if (student.getSurname() == null || student.getName() == null) {
                    throw new IllegalArgumentException("Not all data given.");
                } else {
                    faculty.add(student);
                }
            }
        } catch (FacultyException | IllegalArgumentException | ArrayIndexOutOfBoundsException e) {
            System.out.println("EXCEPTION! " + e.getMessage());
        }
    }
}
```

```
java × Institute.java × FacultyException.java × Faculty.java × Student.java ×
import java.util.*;

38 usages
public class Student {
    3 usages
    private String name;
    3 usages
    private String surname;
    5 usages
    private String number;
    5 usages
    private double averageMark;
    12 usages
    public Student(String name, String surname, String number, double averageMark){
        this.name = name;
        this.surname = surname;
        this.number = number;
        this.averageMark = averageMark;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null) {
            return false;
        if (getClass() != obj.getClass()) {
            return false;
        }
        final Student other = (Student) obj;
        if (!Objects.equals(this.number, other.number)) {
            return false;
        }
        if (!Objects.equals(this.averageMark, other.averageMark)) {
            return true;
        }
        return true;
    }

    @Override
    public int hashCode() {
        int hash = 13;
        hash = 31 * hash * Objects.hashCode(this.number);
        hash = 31 * hash * Objects.hashCode((int) this.averageMark);
        return hash;
    }

    public String getName() { return  name; }
    1 usage
    public String getSurname() { return surname; }
    public String getNumber() { return number; }
    7 usages
    public double getAverageMark() { return averageMark; }
    4 usages
    public String fullName() { return surname + " " + name; }
}
```

```
EXCEPTION! Not all data given.
EXCEPTION! Kliminskiy Artem needs to pass the exam.
EXCEPTION! Paliychuk Ivan failed the session.
EXCEPTION! Can't add Ivanov Sergiy - faculty is full
EXCEPTION! Can't add Krasnoshapka Roman - faculty is full
-------------------------------------
Number of students in institute:
7
-------------------------------------
The most popular faculty:
FIOT
-------------------------------------
Students with average mark in range of 95-100
Sidenko Daria
Kotlyarchuk Maxim
Yablonskiy Danya
Chayka Anton
```

Висновок:

У цій роботі ми ознайомились з класами Throwable, Error, Exception, RuntimeException та конструкціями try-catch-finally, throw, throws. Узагальнили знання із створення об’єктів, конструкторів та методів, реалізації інтерфейсів у Java.