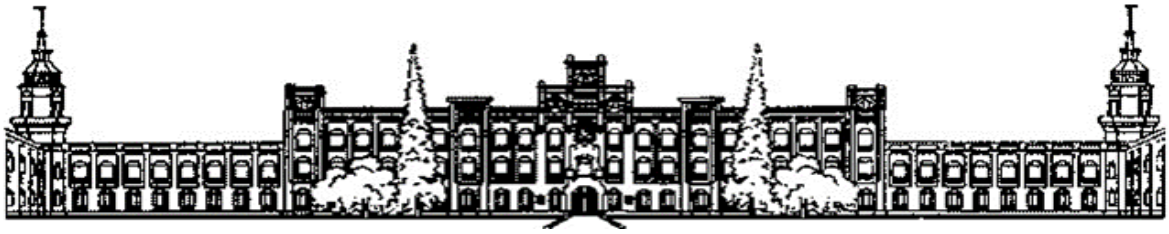


Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Лабораторна робота №10

з дисципліни «Основи програмування»
Тема: « Колекції. Списки»

Виконали:

студенти групи ІА-23

Каширов Д. О.

Проценко. В. І.

Шрубович Н. С.

Перевірив:

Колеснік Валерій

Київ 2022

Хід роботи:

1. Ознайомитись з javadoc для наступних інтерфейсів та класів:
   - Collection
   - List
   - ArrayList
   - LinkedList
   - Iterator
   - RandomAccess

2. Виконати завдання з таблиці 2 відповідно до свого варіанту у таблиці 1. Для цього:
   - проаналізувати завдання;
   - створити зазначенні класи;
   - для створення списків слід використовувати класи та інтерфейси з Collection Framework (заборонено використовувати масиви);
   - усі списки мають бути типізованими (наприклад, ArrayList<Student>, а не просто ArrayList);
   - при реалізації задач «1)», «2)», «3)» слід застосувати наступні методи перегляду колекцій у відповідності до свого варіанту (табл. 1):
     a) нетипізований ітератор;
     b) типізований ітератор;
     c) типізований цикл «for-each».

6	1	c	b	a

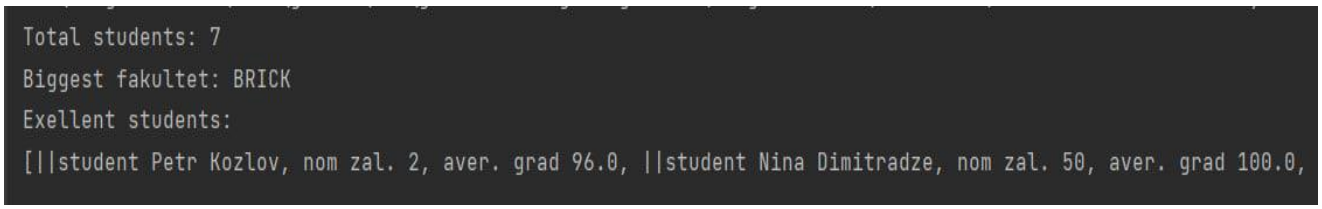
1	Класи: Інститут (назва, список факультетів) Факультет (назва, список студентів) Студент (ім'я, прізвище, номер залікової книжки, середній бал) Задача: 1) Знайти загальну кількість студентів, що навчається в інституті 2) Знайти факультет, на якому навчається найбільша кількість студентів 3) Скласти список студентів, у яких середній бал в діапазоні 95..100

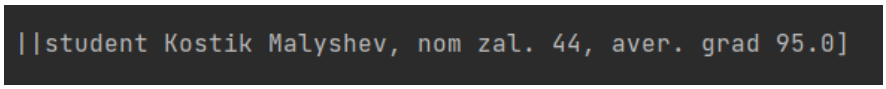
```
import java.util.ArrayList;
import java.util.Iterator;

//package lab10;

public class Task6_Lab10 {

    public static void main(String[] args) {
        Institut institut = new Institut("KPI", Algoritm.a);
        Fakultet fakultet = new Fakultet("ABCD");
        fakultet.addStudent(new Student("Ivan", "Ivanov", 1, 80));
        fakultet.addStudent(new Student("Petr", "Kozlov", 2, 96));
        institut.addFakultet(fakultet);
        fakultet = new Fakultet("TYRE");
        fakultet.addStudent(new Student("Alex", "Kovalenko", 3, 75));
        fakultet.addStudent(new Student("Olga", "Petrenko", 4, 88));
        institut.addFakultet(fakultet);
```

```
        fakultet = new Fakultet("BRICK");
        fakultet.addStudent(new Student("Nina", "Dimitradze", 50, 100));
        fakultet.addStudent(new Student("Kostik", "Malyshev", 44, 95));
        fakultet.addStudent(new Student("Dima", "Kabanov", 11, 65));
        institut.addFakultet(fakultet);
        System.out.println("Total students: " + institut.getStudQuontity());
        System.out.println("Biggest fakultet: " +
institut.getBiggestFakultet());
        System.out.println("Exellent students: " +
institut.getExellentStudents());
    }

}

// варіанти алгоритмів
// a - нетипізований ітератор
// b - типізований ітератор
// c - типізований цикл «for-each»
enum Algoritm {
    a, b, c;
}

class Student {
    private String firstName;
    private String lastName;
    private int nomZK;
    private double averGrad;

    Student(String firstName, String lastName, int nomZK, double averGrad) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.nomZK = nomZK;
        this.averGrad = averGrad;
    }

    @Override
    public String toString() {
        return "||student " + firstName + " " + lastName + ", nom zal. " +
nomZK + ", aver. grad " + averGrad;
    }

    public boolean isGreatStudent() {
        return averGrad >= 95;
    }
}

class Fakultet {
    private String name;
    private ArrayList<Student> students = new ArrayList<Student>();

    Fakultet(String name) {
        this.name = name;
    }

    public void addStudent(Student student) {
        students.add(student);
    }

    public int getStudQuontity() {
        return students.size();
    }

    @Override
    public String toString() {
```

```
        return name;
    }

    public ArrayList<Student> getExellentStudents(Algoritm algoritm) {
        ArrayList<Student> result = new ArrayList<Student>();
        if (algoritm == Algoritm.a) {
            for (Iterator i = students.iterator(); i.hasNext();) {
                Object o = i.next();
                Student student = (Student) o;
                if (student.isGreatStudent()) {
                    result.add(student);
                }
            }
        } else if (algoritm == Algoritm.b) {
            for (Iterator<Student> i = students.iterator(); i.hasNext();) {
                Student student = i.next();
                if (student.isGreatStudent()) {
                    result.add(student);
                }
            }
        } else if (algoritm == Algoritm.c) {
            for (Student student : students) {
                if (student.isGreatStudent()) {
                    result.add(student);
                }
            }
        }
        return result;
    }
}

class Institut {
    private String name;
    private ArrayList<Fakultet> fakultets = new ArrayList<Fakultet>();
    private Algoritm algoritm;

    Institut(String name, Algoritm algoritm) {
        this.name = name;
        this.algoritm = algoritm;
    }

    public void addFakultet(Fakultet fakultet) {
        fakultets.add(fakultet);
    }

    public int getStudQuontity() {
        int result = 0;
        if (algoritm == Algoritm.a) {
            for (Iterator i = fakultets.iterator(); i.hasNext(); ) {
                Object o = i.next();
                Fakultet fakultet = (Fakultet) o;
                result += fakultet.getStudQuontity();
            }
        } else if (algoritm == Algoritm.b) {
            for (Iterator<Fakultet> i = fakultets.iterator(); i.hasNext(); )
{
                Fakultet fakultet = i.next();
                result += fakultet.getStudQuontity();
            }
        } else if (algoritm == Algoritm.c) {
            for (Fakultet fakultet : fakultets) {
                result += fakultet.getStudQuontity();
            }
        }
```

```
        return result;
    }

    public Fakultet getBiggestFakultet() {
        Fakultet result = null;
        int maxTotal = 0;
        if (algoritm == Algoritm.a) {
            for (Iterator i = fakultets.iterator(); i.hasNext(); ) {
                Object o = i.next();
                Fakultet fakultet = (Fakultet) o;
                int tot = fakultet.getStudQuontity();
                if (tot > maxTotal) {
                    maxTotal = tot;
                    result = fakultet;
                }
            }
        } else if (algoritm == Algoritm.b) {
            for (Iterator<Fakultet> i = fakultets.iterator(); i.hasNext(); )
{
                Fakultet fakultet = i.next();
                int tot = fakultet.getStudQuontity();
                if (tot > maxTotal) {
                    maxTotal = tot;
                    result = fakultet;
                }
            }
        } else if (algoritm == Algoritm.c) {
            for (Fakultet fakultet : fakultets) {
                int tot = fakultet.getStudQuontity();
                if (tot > maxTotal) {
                    maxTotal = tot;
                    result = fakultet;
                }
            }
        }
        return result;
    }

    public ArrayList<Student> getExellentStudents() {
        ArrayList<Student> result = new ArrayList<Student>();
        if (algoritm == Algoritm.a) {
            for (Iterator i = fakultets.iterator(); i.hasNext(); ) {
                Object o = i.next();
                Fakultet fakultet = (Fakultet) o;
                result.addAll(fakultet.getExellentStudents(algoritm));
            }
        } else if (algoritm == Algoritm.b) {
            for (Iterator<Fakultet> i = fakultets.iterator(); i.hasNext(); )
{
                Fakultet fakultet = i.next();
                result.addAll(fakultet.getExellentStudents(algoritm));
            }
        } else if (algoritm == Algoritm.c) {
            for (Fakultet fakultet : fakultets) {
                result.addAll(fakultet.getExellentStudents(algoritm));
            }
        }
        return result;
    }
}
```





Висновок: Під час виконання цієї лабораторної роботи ми ознайомилися з основними інтерфейсами Collection Framework та використали їх на практиці, вдосконалили свої навички написання коду.