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 №1

Колекції. Множина TreeSet

Виконали

студенти групи ІА-23:

Волошин Вадім Вікторович

Воронюк Євгеній Владиславович

Савонік Назар Анатолійович

Перевірив:

Колеснік Валерій Миколайович

Київ 2023

Хід роботи:

1. Ознайомитись з javadoc для наступних інтерфейсів та класів:
   - Set
   - TreeSet
   - Comparable
   - Comparator
   - SortedSet
   - NavigableSet

2. Виконати завдання попередньої лабораторної роботи, замінивши списки List (ArrayList та LinkedList) на множини Set (TreeSet). При цьому необхідно щонайменше один раз використати Comparable та щонайменше один раз – Comparator.

3. Відповісти на контрольні питання.

Company.java {}

```
import java.util.*;

public class Company {
    public static void main(String args[]) {
        System.out.println("Company name: GnomIT");
        System.out.println("Director name: Michael Reeves");
        TreeSet set = new TreeSet();
        set.add("MARKETING");
        set.add("LEGAL");
        set.add("HR");
        set.add("OPERATIONS");

        for (Iterator i = set.iterator(); i.hasNext();) {
            Object o = i.next();
            System.out.println(o);
        }
    }
}
```

```
Company name: GnomIT
Director name: Michael Reeves
HR
LEGAL
MARKETING
OPERATIONS

Process finished with exit code 0
```

Department.java {} *(допоміжний клас)*

```
import java.util.List;

public class Department {
    private String DepName;
    private List<String> EmployeeList;
```

```
    public Department(String DepName, List<String> EmployeeList) {
        this.DepName = DepName;
        this.EmployeeList = EmployeeList;
    }

    public void printEmployeeInfo() {
        System.out.print(this.DepName + ": ");
        for (String film: this.EmployeeList) {
            System.out.print(film + ", ");
        }
        System.out.println();
    }
}
```

DepList.java {}

```
import java.util.List;
import java.util.ArrayList;
import java.util.Collections;

public class DepList {
    public static void main(String[] args) {
        //Creating departments
        String[] departments = {"HR", "OPERATIONS", "MARKETING", "LEGAL"};
        //HR
        List<String> DepEmployeeList = new ArrayList<>();
        String[] EmployeeList = {"Tommy Versace", "Catherine Jones", "Tom
Ford"};
        Collections.addAll(DepEmployeeList, EmployeeList);
        Department hr = new Department(departments[0], DepEmployeeList);
        System.out.println("Head of HR: " + EmployeeList[0]);
        hr.printEmployeeInfo();
        //operations
        DepEmployeeList = new ArrayList<>();
        EmployeeList = new String[]{"James Elliot", "John White", "James
Alvaro"};
        Collections.addAll(DepEmployeeList, EmployeeList);
        Department operations = new Department(departments[1],
DepEmployeeList);
        System.out.println("\nHead of Operations: " + EmployeeList[0]);
        operations.printEmployeeInfo();
        //marketing
        DepEmployeeList = new ArrayList<>();
        EmployeeList = new String[]{"Tom Jones", "Nancy Smith", "Frank
Anthony"};
        Collections.addAll(DepEmployeeList, EmployeeList);
        Department marketing = new Department(departments[2],
DepEmployeeList);
        System.out.println("\nHead of Marketing: " + EmployeeList[0]);
        marketing.printEmployeeInfo();

        //legal
        DepEmployeeList = new ArrayList<>();
        EmployeeList = new String[]{"Ryan Gosling", "Harry Major", "Ethan
Hardy"};
        Collections.addAll(DepEmployeeList, EmployeeList);
        Department legal = new Department(departments[2], DepEmployeeList);
        System.out.println("\nHead of Legal: " + EmployeeList[0]);
        legal.printEmployeeInfo();
        }
}
```

```
Head of HR: Tommy Versace
HR: Tommy Versace, Catherine Jones, Tom Ford,

Head of Operations: James Elliot
OPERATIONS: James Elliot, John White, James Alvaro,

Head of Marketing: Tom Jones
MARKETING: Tom Jones, Nancy Smith, Frank Anthony,

Head of Legal: Ryan Gosling
MARKETING: Ryan Gosling, Harry Major, Ethan Hardy,

Process finished with exit code 0
```

EmployeeList.java {} *(task3 here)*

```
import java.util.*;
import java.util.Iterator;

public class EmployeeList {
    public static void main(String[] args) {

        TreeSet<Employee> employeeL = new TreeSet<>();
        employeeL.add(new Employee("Tom Jones", 18000, "Marketing", true,
false));
        employeeL.add(new Employee("Tommy Versace", 19000, "HR", true,
false));
        employeeL.add(new Employee("Tom Ford", 17000, "HR", false, false));
        employeeL.add(new Employee("Ryan Gosling", 34000, "Legal", true,
false));
        employeeL.add(new Employee("Harry Major", 20000, "Legal", false,
false));
        employeeL.add(new Employee("Ethan Hardy", 30000, "Legal", false,
false));
        employeeL.add(new Employee("Nancy Smith", 15000, "Marketing", false,
false));
        employeeL.add(new Employee("Catherine Jones", 18000, "HR", false,
false));
        employeeL.add(new Employee("James Elliot", 33000, "Operations", true,
false));
        employeeL.add(new Employee("James Alvaro", 31000, "Operations",
false, false));
        employeeL.add(new Employee("John White", 35000, "Operations", false,
false));
        employeeL.add(new Employee("Frank Anthony", 17000, "Marketing",
false, false));
        employeeL.add(new Employee("Michael Reeves", 45000, "Head of
company", false, true));

        //TASK3
        System.out.println("List of employees: " + employeeL);
    }
}
class Employee implements Comparable<Employee> {
     String name;
```

```
     Integer salary;
     String department;
     Boolean isDirector;
     Boolean isHead;

    public Employee(String n, Integer salary, String d, Boolean isHead,
Boolean isDirector) {
        name = n;
        this.salary = salary;
        department = d;
        this.isHead = isHead;
        this.isDirector = isDirector;
    }

    public double getSalary() {
        return salary;
    }

    public Boolean getisDirector() {
        return isDirector;
    }

    public String toString() {
        return "\n"+ "(" + name + ", " + salary + ", " + department + ", " +
isHead + ", " + isDirector + ")";
    }


    @Override
    public int compareTo(Employee empl) {
        return this.name.compareTo(empl.name);
    }
}
```

```
List of employees: [
(Catherine Jones, 18000, HR, false, false),
(Ethan Hardy, 30000, Legal, false, false),
(Frank Anthony, 17000, Marketing, false, false),
(Harry Major, 20000, Legal, false, false),
(James Alvaro, 31000, Operations, false, false),
(James Elliot, 33000, Operations, true, false),
(John White, 35000, Operations, false, false),
(Michael Reeves, 45000, Head of company, false, true),
(Nancy Smith, 15000, Marketing, false, false),
(Ryan Gosling, 34000, Legal, true, false),
(Tom Ford, 17000, HR, false, false),
(Tom Jones, 18000, Marketing, true, false),
(Tommy Versace, 19000, HR, true, false)]

Process finished with exit code 0
```

Task1.java {}

```java
import java.util.*;

public class Task1 {
    public static void main(String[] args) {
        Set<Integer> set = new TreeSet<>(new SalaryComparator());
        set.add(12000);
        set.add(19000);
        set.add(17000);
        set.add(34000);
        set.add(20000);
        set.add(30000);
        set.add(15000);
        set.add(18000);
        set.add(24000);
        set.add(31000);
        set.add(35000);
        set.add(32000);
        set.add(45000);

        System.out.println(set);
    }
}
class SalaryComparator implements Comparator<Integer> {

    @Override
    public int compare(Integer o1, Integer o2) {

        return o1.compareTo(o2);
    }
}
```

```
[12000, 15000, 17000, 18000, 19000, 20000, 24000, 30000, 31000, 32000, 34000, 35000, 45000]

Process finished with exit code 0
```

Task2.java {}

```java
import java.util.*;

public class Task2 {

    public static void main(String[] args) {

        TreeSet <Employee1> set = new TreeSet<>();
        set.add(new Employee1("Tom Jones", "Marketing", 18000));
        set.add(new Employee1("Frank Anthony", "Marketing", 17000));
        set.add(new Employee1("Ryan Gosling", "Legal", 34000));
        set.add(new Employee1("Catherine Jones", "HR", 18000));
        set.add(new Employee1("James Elliot", "Operations", 33000));
        set.add(new Employee1("Tommy Versace", "HR", 19000));
        set.add(new Employee1("Tom Ford", "HR", 17000));
        set.add(new Employee1("Harry Major", "Legal", 20000));
        set.add(new Employee1("Ethan Hardy", "Legal", 30000));
        set.add(new Employee1("James Alvaro", "Operations", 31000));
        set.add(new Employee1("John White", "Operations", 35000));
        set.add(new Employee1("Nancy Smith", "Marketing", 15000));
        System.out.println("\n" + "Sort elements in ascending order: " +
set);
```

```
    }
}

class Employee1 implements Comparable<Employee1> {

     String name;
     String dep;
     Integer salary;

    public Employee1(String name, String dep, int salary) {
        super();
        this.name = name;
        this.dep = dep;
        this.salary = salary;
    }

    public String toString() {
        return (this.name + "| " + this.dep + "| " + this.salary + "|");
    }

    @Override
    public int compareTo(Employee1 empl) {
        return this.salary.compareTo(empl.salary);
    }

}
```

```
Sort elements in ascending order: [
Nancy Smith| Marketing| 15000|,
Frank Anthony| Marketing| 17000|,
Tom Jones| Marketing| 18000|,
Tommy Versace| HR| 19000|,
Harry Major| Legal| 20000|,
Ethan Hardy| Legal| 30000|,
James Alvaro| Operations| 31000|,
James Elliot| Operations| 33000|,
Ryan Gosling| Legal| 34000|,
John White| Operations| 35000|]

Process finished with exit code 0
```

Висновок: на даній лабораторній роботі ми ознайомилися з наступними інтерфейсами та класами: Set, TreeSet, Comparable, Comparator, SortedSet, NavigableSet. Виконали завдання попередньої лабораторної роботи, замінивши списки List (ArrayList та LinkedList) на множини Set/TreeSet. При цьому використали Comparable та Comparator. Ознайомилися із відмінностями між множиною і списком, SortedSet та Set. Перевірили, чому в TreeSet не слід зберігати посилання на мутабельні об’єкти.