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 № 10

Колекції. Списки

Виконали

студенти групи ІА-23:

Волошин Вадім

Воронюк Євгеній

Савонік Назар

Перевірив:

Колеснік В. М.

Київ 2022

Хід роботи:

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».

Варіант: 2, Завдання: 2, Тип інтератора: 1 - a, 2 – c, 3 – b;

Виконання завдання:

Клас: 1. Фірма (назва, директор, список відділів).

class Company {}

```
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");
        List<String> list = new ArrayList<String>();
        list.add("MARKETING");
        list.add("LEGAL");
        list.add("HR");
        list.add("OPERATIONS");

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

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

Process finished with exit code 0
```

Класс: 2. Відділ (назва, начальник відділу, список працівників).

class Department {} *(допоміжний)*

```
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();
    }
}
```

class DepList {}

```
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
```

Клас: 3. Працівник (ім'я, прізвище, заробітна платня);

Задача: 1. Знайти значення максимальної заробітної платні з усіх працівників, включаючи начальників та директора. 3. Скласти список усіх співробітників фірми, включаючи начальників та директора. *(у класі продемонстровано дві задачі)*

class Employee {}

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

public class Employee {
    public String name;
    public double salary;
    public String department;
    private Boolean isDirector;
    private Boolean isHead;

    public Employee(String n, double 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 "(" + name + ", " + ", " + salary + ", " + department + ", " +
isHead + ", " + isDirector + ")";
    }

    public static void main(String[] args) {

    List<Employee> employeeList = new ArrayList<>();
            employeeList.add(new Employee("Tom Jones", 18000.00, "Marketing",
true, false));
            employeeList.add(new Employee("Tommy Versace", 19000.00, "HR",
true, false));
            employeeList.add(new Employee("Tom Ford", 17000.00, "HR", false,
false));
            employeeList.add(new Employee("Ryan Gosling", 34000.00, "Legal",
true, false));
            employeeList.add(new Employee("Harry Major", 20000.00, "Legal",
false, false));
            employeeList.add(new Employee("Ethan Hardy", 30000.00, "Legal",
false, false));
            employeeList.add(new Employee("Nancy Smith", 15000.00,
"Marketing", false, false));
            employeeList.add(new Employee("Catherine Jones", 18000.00, "HR",
false, false));
            employeeList.add(new Employee("James Elliot", 33000.00,
"Operations", true, false));
            employeeList.add(new Employee("James Alvaro", 31000.00,
"Operations", false, false));
            employeeList.add(new Employee("John White", 35000.00,
"Operations", false, false));
            employeeList.add(new Employee("Frank Anthony", 17000.00,
"Marketing", false, false));
            employeeList.add(new Employee("Michael Reeves", 45000.00, "Head
of company", false, true));

            //TASK1
        List<Integer> Salarylist = new ArrayList<>();
        Salarylist.add(12000);
        Salarylist.add(19000);
        Salarylist.add(17000);
        Salarylist.add(34000);
        Salarylist.add(20000);
        Salarylist.add(30000);
        Salarylist.add(15000);
        Salarylist.add(18000);
        Salarylist.add(24000);
        Salarylist.add(31000);
        Salarylist.add(35000);
        Salarylist.add(32000);
        Salarylist.add(45000);
        Iterator listIterator = Salarylist.iterator();
        int max = 0;
        while(listIterator.hasNext()) {
            Integer i = (Integer)listIterator.next();
            if(i > max)
                max = i;
        }
```

```
        System.out.println("Max salary is " + max);

        //TASK3
        System.out.println("\nList of Employees");
        Iterator<Employee> iter = employeeList.iterator();
        while(iter.hasNext()) {
            System.out.println(iter.next());
        }
    }
}
```

```
Max salary is 45000

List of Employees
(Tom Jones, , 18000.0, Marketing, true, false)
(Tommy Versace, , 19000.0, HR, true, false)
(Tom Ford, , 17000.0, HR, false, false)
(Ryan Gosling, , 34000.0, Legal, true, false)
(Harry Major, , 20000.0, Legal, false, false)
(Ethan Hardy, , 30000.0, Legal, false, false)
(Nancy Smith, , 15000.0, Marketing, false, false)
(Catherine Jones, , 18000.0, HR, false, false)
(James Elliot, , 33000.0, Operations, true, false)
(James Alvaro, , 31000.0, Operations, false, false)
(John White, , 35000.0, Operations, false, false)
(Frank Anthony, , 17000.0, Marketing, false, false)
(Michael Reeves, , 45000.0, Head of company, false, true)

Process finished with exit code 0
```

Задача: 2. Визначити, відділ, в якому хоча б один з співробітників отримує заробітну платню вищу за платню свого начальника.

Task2.java

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

public class Task2 {

    public static void main(String[] args) {

        List<Employee1> listOfEmployee = new ArrayList<>();
        listOfEmployee.add(new Employee1("Tom Jones", "Marketing", 18000));
        listOfEmployee.add(new Employee1("Frank Anthony", "Marketing",
17000));
        listOfEmployee.add(new Employee1("Ryan Gosling", "Legal", 34000));
        listOfEmployee.add(new Employee1("Catherine Jones", "HR", 18000));
        listOfEmployee.add(new Employee1("James Elliot", "Operations",
33000));
        listOfEmployee.add(new Employee1("Tommy Versace", "HR", 19000));
        listOfEmployee.add(new Employee1("Tom Ford", "HR", 17000));
```

```
        listOfEmployee.add(new Employee1("Harry Major", "Legal", 20000));
        listOfEmployee.add(new Employee1("Ethan Hardy", "Legal", 30000));
        listOfEmployee.add(new Employee1("James Alvaro", "Operations",
31000));
        listOfEmployee.add(new Employee1("John White", "Operations", 35000));
        listOfEmployee.add(new Employee1("Nancy Smith", "Marketing", 15000));


        int maxSalary = 0;
        for (int i = 0; i < listOfEmployee.size(); i++) {
            if (listOfEmployee.get(i).getSalary() > maxSalary)
                maxSalary = listOfEmployee.get(i).getSalary();
        }
        System.out.println("The maximum salary is "
                + maxSalary);
        Employee1 max = Collections.max(listOfEmployee);
        System.out.println("Department: " + max.getDep());

    }
}

class Employee1 implements Comparable<Employee1> {

    private String name;
    private String dep;
    private int salary;

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

    public String getName() {
        return name;
    }

    public String getDep() {
        return dep;
    }

    public int getSalary() {
        return salary;
    }

    @Override
    public int compareTo(Employee1 o) {
        if (this.getSalary() > o.getSalary()) {
            return 1;
        } else if (this.getSalary() < o.getSalary()) {
            return -1;
        }
        return 0;
    }

}
```

```
The maximum salary is 35000
Department: Operations

Process finished with exit code 0
```

Висновок: на даній лабораторній роботі ми ознайомилися із javadoc для наступних інтерфейсів та класів: Collection, List, ArrayList, LinkedList, Iterator, RandomAccess. Виконали завдання з таблиці 2 відповідно до свого варіанту у таблиці 1: проаналізували завдання, створили зазначенні класи, застосовували наступні методи для перегляду колекцій: нетипізований ітератор, типізований ітератор, типізований цикл «for-each». Проаналізували, чим масиви відрізняються від колекцій, чим списки відрізняються від масивів, чим ArrayList відрізняється від LinkedList. Зрозуміли, що таке Iterator, що таке типізовані та нетипізовані колекції, для чого потрібен інтерфейс RandomAccess, та чим Collection відрізняється від Collections.