import java.util.ArrayList;

import java.util.List;

import java.util.Map;

import java.util.stream.Collectors;

class Employee implements Comparable<Employee> {

private String name;

private double salary;

public Employee(String name, double salary) {

this.name = name;

this.salary = salary;

}

public String getName() {

return name;

}

public double getSalary() {

return salary;

}

@Override

public String toString() {

return "{" +

"name='" + name + '\'' +

", salary=" + salary +

'}';

}

@Override

public int compareTo(Employee other) {

return Double.compare(this.salary, other.salary);

}

}

public class lab7 {

public static void main(String[] args) {

List<Employee> employeeList = new ArrayList<>();

employeeList.add(new Employee("John", 50000));

employeeList.add(new Employee("Alice", 60000));

employeeList.add(new Employee("Bob", 55000));

employeeList.add(new Employee("Charlie", 70000));

employeeList.add(new Employee("David", 48000));

employeeList.add(new Employee("Eva", 62000));

System.out.println("Employee List:");

employeeList.forEach(employee -> System.out.println(employee));

employeeList.sort(null);

System.out.println("\nSorted Employee List (based on salary):");

employeeList.forEach(employee -> System.out.println(employee));

double salaryThreshold = 60000;

List<Employee> highSalaryEmployees = employeeList.stream()

.filter(employee -> employee.getSalary() > salaryThreshold)

.collect(Collectors.toList());

System.out.println("\nEmployees with Salary > " + salaryThreshold + ":");

highSalaryEmployees.forEach(employee -> System.out.println(employee));

double averageSalary = employeeList.stream().mapToDouble(Employee::getSalary).average().orElse(0);

System.out.println("\nAverage Salary: " + averageSalary);

Map<String, List<Employee>> employeesBySalaryRange = employeeList.stream()

.collect(Collectors.groupingBy(employee -> {

if (employee.getSalary() < 50000) {

return "Low";

} else if (employee.getSalary() < 60000) {

return "Moderate";

} else {

return "High";

}

}));

System.out.println("\nEmployees Grouped by Salary Range:");

employeesBySalaryRange.forEach((range, employees) -> {

System.out.println(range + ": " + employees);

});

Employee lowestPaidEmployee = employeeList.stream().min(Employee::compareTo).orElse(null);

System.out.println("\nEmployee with the Lowest Salary: " + lowestPaidEmployee);

}

}

