**Experiment - 6**

**Name: Armaan UID: 22BCS12007**

**Aim:** To solve JAVA problems

1. Problem : EASY-LEVEL

Code:

import java.util.\*;

class Employee {

String name;

int age;

double salary;

Employee(String name, int age, double salary) {

this.name = name;

this.age = age;

this.salary = salary;

}

public String toString() {

return name + " - Age: " + age + ", Salary: $" + salary;

}

}

public class EmployeeSorter {

public static void main(String[] args) {

List<Employee> employees = Arrays.asList(

new Employee("Alice", 30, 50000),

new Employee("Bob", 25, 60000),

new Employee("Charlie", 28, 55000)

);

System.out.println("🔤 Sort by Name:");

employees.stream()

.sorted(Comparator.comparing(e -> e.name))

.forEach(System.out::println);

System.out.println("\n🎂 Sort by Age:");

employees.stream()

.sorted(Comparator.comparingInt(e -> e.age))

.forEach(System.out::println);

System.out.println("\n💰 Sort by Salary:");

employees.stream()

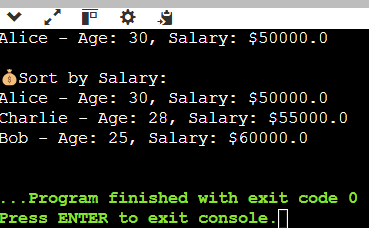
.sorted(Comparator.comparingDouble(e -> e.salary))

.forEach(System.out::println);

}

}

Output :



Problem : MEDIUM-LEVEL

Code:

import java.util.\*;

import java.util.stream.\*;

class Student {

String name;

double percentage;

Student(String name, double percentage) {

this.name = name;

this.percentage = percentage;

}

}

public class StudentFilterSort {

public static void main(String[] args) {

List<Student> students = Arrays.asList(

new Student("Annie", 82.5),

new Student("Raj", 74.0),

new Student("Meera", 91.0),

new Student("John", 76.0)

);

System.out.println("🏅 Students scoring above 75% (sorted):");

students.stream()

.filter(s -> s.percentage > 75)

.sorted(Comparator.comparingDouble(s -> -s.percentage))

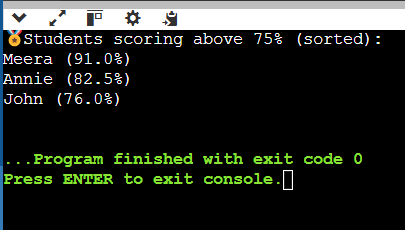
.map(s -> s.name + " (" + s.percentage + "%)")

.forEach(System.out::println);

}

}

Output :



Problem: HARD\_LEVEL

Code:

import java.util.\*;

import java.util.stream.\*;

class Product {

String name;

String category;

double price;

Product(String name, String category, double price) {

this.name = name;

this.category = category;

this.price = price;

}

public String toString() {

return name + " ($" + price + ")";

}

}

public class ProductProcessor {

public static void main(String[] args) {

List<Product> products = Arrays.asList(

new Product("Laptop", "Electronics", 1200),

new Product("Smartphone", "Electronics", 900),

new Product("T-shirt", "Clothing", 25),

new Product("Jeans", "Clothing", 45),

new Product("Book", "Stationery", 15),

new Product("Pen", "Stationery", 2)

);

// Group by Category

System.out.println("📦 Products grouped by category:");

Map<String, List<Product>> grouped = products.stream()

.collect(Collectors.groupingBy(p -> p.category));

grouped.forEach((cat, prods) -> {

System.out.println(cat + ": " + prods);

});

// Most Expensive Product per Category

System.out.println("\n💸 Most expensive product in each category:");

Map<String, Optional<Product>> mostExpensive = products.stream()

.collect(Collectors.groupingBy(

p -> p.category,

Collectors.maxBy(Comparator.comparingDouble(p -> p.price))

));

mostExpensive.forEach((cat, prod) ->

System.out.println(cat + ": " + prod.orElse(null)));

// Average Price of All Products

double avgPrice = products.stream()

.collect(Collectors.averagingDouble(p -> p.price));

System.out.println("\n📊 Average price of all products: $" + avgPrice);

}

}

Output:

