Name:	Taimur Kamran
UID:	23BCS10578
Session:	622-A

PRATICE QUESTION

Part A –

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;
    }

    @Override
```

```
    public String toString() {
        return name + " (" + age + ", " + salary + ")";
    }
}

public class EmployeeSortDemo {
    public static void main(String[] args) {
        List<Employee> employees = new ArrayList<>();
        employees.add(new Employee("Alice", 30, 50000));
        employees.add(new Employee("Bob", 25, 60000));
        employees.add(new Employee("Charlie", 28, 55000));

        // Sort by name
        employees.sort((e1, e2) -> e1.name.compareTo(e2.name));
        System.out.println("Sorted by name: " + employees);
        // Sort by age
        employees.sort(Comparator.comparingInt(e -> e.age));
        System.out.println("Sorted by age: " + employees);
        // Sort by salary descending
        employees.sort((e1, e2) -> Double.compare(e2.salary, e1.salary));
        System.out.println("Sorted by salary (desc): " + employees);
    }
}
```

Output:

```
Sorted by name: [Alice (30, 50000.0), Bob (25, 60000.0), Charlie (28, 55000.0)]
Sorted by age: [Bob (25, 60000.0), Charlie (28, 55000.0), Alice (30, 50000.0)]
Sorted by salary (desc): [Bob (25, 60000.0), Charlie (28, 55000.0), Alice (30, 50000.0)]
```

PART B –

Code:

```
import java.util.*;
import java.util.stream.*;

class Student {
    String name;
    double marks;

    Student(String name, double marks) {
        this.name = name;
        this.marks = marks;
    }

    @Override
    public String toString() {
```

```
        return name + "(" + marks + ")";
    }
}

public class StudentStreamDemo {
    public static void main(String[] args) {
        List<Student> students = Arrays.asList(
            new Student("Alice", 80),
            new Student("Bob", 70),
            new Student("Charlie", 90),
            new Student("David", 60)
        );

        List<String> filteredNames = students.stream()
                .filter(s -> s.marks > 75)
                .sorted(Comparator.comparingDouble(s -> s.marks))
                .map(s -> s.name)
                .collect(Collectors.toList());

        System.out.println("Students with marks > 75 sorted by marks: " +
filteredNames);
    }
}
```

Output:

```
csharp

Students with marks > 75 sorted by marks: [Alice, Charlie]
```

PART C –

Code:

```
import java.util.*;

import java.util.stream.*;

import java.util.Map.Entry;


class Product {

    String name;

    double price;

    String category;


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

        this.name = name;

        this.price = price;

        this.category = category;

    }


    @Override

    public String toString() {
```

```
        return name + "(" + price + ")";
    }
}

public class ProductStreamDemo {
    public static void main(String[] args) {
        List<Product> products = Arrays.asList(
            new Product("Laptop", 60000, "Electronics"),
            new Product("Headphones", 1500, "Electronics"),
            new Product("Shirt", 1200, "Clothing"),
            new Product("Jeans", 2000, "Clothing"),
            new Product("Book", 500, "Stationery")
        );

        // Group by category
        Map<String, List<Product>> grouped = products.stream()
                .collect(Collectors.groupingBy(p -> p.category));
        System.out.println("Grouped by category: " + grouped);

        // Most expensive product per category
        Map<String, Optional<Product>> maxPrice = products.stream()
                .collect(Collectors.groupingBy(p -> p.category,
                        Collectors.maxBy(Comparator.comparingDouble(p -> p.price))));
```

```
        System.out.println("Most expensive product per category:");
        for (Entry<String, Optional<Product>> e : maxPrice.entrySet()) {
            System.out.println(e.getKey() + " -> " + e.getValue().get());
        }

        // Average price
        double avgPrice = products.stream()
                .collect(Collectors.averagingDouble(p -> p.price));
        System.out.println("Average price of all products: " + avgPrice);
    }
}
```

Output:

```
Grouped by category: {Electronics=[Laptop(60000.0), Headphones(1500.0)], Clothing=[Shirt(1200.0),
Most expensive product per category:
Electronics -> Laptop(60000.0)
Clothing -> Jeans(2000.0)
Stationery -> Book(500.0)
Average price of all products: 14140.0
```