

## LAB – 8 – HOME WORK

### Day – 1

**Problem 1 :** Consider the following lambda expression. Can this expression be correctly typed as a BiFunction?

```
(x,y) -> {  
    List<Double> list = new ArrayList<>();  
    list.add(Math.pow(x,y));  
    list.add(x * y);  
    return list;  
};
```

Demonstrate you are right by doing the following: In the main method of a Java class, assign this lambda expression to an appropriate BiFunction and call the apply method with arguments (2.0, 3.0), and print the result to console.

**Problem 2:** Get practice on Sorting.

```
class Product {  
    final String title;  
    final double price;  
    final int model;  
  
    public String getTitle() {  
        return title;  
    }  
  
    public double getPrice() {  
        return price;  
    }  
  
    public int getModel() {  
        return model;  
    }  
  
    public Product(String title, Double price, int model) {  
        this.title = title;  
        this.price = price;  
        this.model = model;  
    }  
  
    @Override  
    public String toString() {  
        return String.format("\n %s : %s : %s", title, price, model);  
    }  
}
```

}

- a. Sort by implementing a comparator for price attribute and print product list.
- b. Sort by implementing a comparator for title attribute and print product list.
- c. Implement the sort method so that only one type of Comparator is used for the task a & b in a Java 7 Way using closure.
- d. If the title is same use model as another attribute to sort. Do this by using lambdas.(Java 8 Way)

Task a & b – Using separate comparators – not closure (refer : comparator2 package )

Task c : Refer comparator3 package

### **Get practice to use methodreferences**

1. In the lecture, one of the examples of a method reference of type *object::instanceMethod* was *this::equals*. Since every lambda expression must be converted to a functional interface, find a functional interface in the `java.util.function` package that would be used for this lambda expression.

Hint #1: The implicit reference 'this' refers to the currently active object. So, to answer this question, create a class `MyClass` in which you have referenced *this::equals* with an appropriate type; add a method `myMethod(MyClass cl)` [testing method to check the equality] which uses this method expression to return true if `cl` is equal to 'this'.

Hint #2: Take a look at the api docs here:

<http://docs.oracle.com/javase/8/docs/api/java/util/function/package-summary.html>

2. An example of a method reference is:

`Math::random`

Its corresponding functional interface is `Supplier<Double>`. Do the following in separate java file:

- i. Put this method expression in a `main` method in a Java class and use it to print a random number to the console(using method reference)
- ii. Rewrite this method reference as a lambda expression (using lambda)
- iii. Create a Java class to print the random number using an inner class by implementing `Supplier` interface; call this inner class from a `main` method and use it to output a random number to the console. (using inner class)

Problem 3:

```
List<String> fruits = Arrays.asList("Apple", "Banana", "Orange", "Cherries", "blums");
```

- a. Print the given list using `forEach` with Lambdas
- b. Print the given list using method reference

Problem 4:

```
String[] names = {"Alexis", "Tim", "Kyleen", "KRISTY"};
```

- a. Use `Arrays.sort()` to sort the names by ignore case using Method reference.