Quiz 10 **Name:_____**

1. Implement Q1.java such that it will work as follows when executed:

```
C:\IS442> java Q1 1.1 2.2 3.3
Sum = 6.6

C:\IS442> java Q1 1.1
Sum = 1.1

C:\IS442> java Q1 2 3
Sum = 5.0

C:\IS442> java Q1
no inputs!
```

Answer:

2. [**5 marks]** Implement the following function `buyfruits`. This function takes in 2 parameters:
   a. `inventory` (type: `Map`): Represents the stock available in the shop's inventory. Each entry is a key-value pair. The key is the product's name, and the value is the quantity available.
   b. `order` (type: `Map`): Represents items in a customer's order. Each entry is a key-value pair is in the form `{name:quantity}`, i.e., the key is the product's name and the value is the quantity that the customer wishes to purchase.

   For each entry in the order, this function will check the inventory to see if the item is available, and of sufficient quantity. If all items are available and are of sufficient quantities, proceed to process the order (i.e. reduce the quantity from the inventory) and return `true`. Otherwise, return `false`.

```
public class BuyFruitsTest {
    public static void main(String[] args) {
     {
        /*
         This is equivalent to
           Map<String, String> inventory =
                   new HashMap<String, Integer>();
           inventory.put("apple", 3);
           inventory.put("orange", 4);
           inventory.put("pear", 5);
        */
       Map<String, Integer> inventory = new TreeMap<String, Integer>(
               Map.of("apple", 3, "orange", 4, "pear", 5));

       Map<String, Integer> order = new TreeMap<String, Integer>(
               Map.of("apple", 4)); // not enough apple

       boolean result = buyFruits(inventory, order);
       System.out.println("Test 1");
       System.out.println("Expected:false {apple=3, orange=4, pear=5}");
       System.out.println("Actual  :" + result + " " + inventory);
       System.out.println();

     }

     {
       Map<String, Integer> inventory = new TreeMap<String, Integer>(
               Map.of("apple", 3, "orange", 4, "pear", 5));

       Map<String, Integer> order = new TreeMap<String, Integer>(
               Map.of("apple", 2, "orange", 2));

       boolean result = buyFruits(inventory, order);
       System.out.println("Test 2");
       System.out.println("Expected:true {apple=1, orange=2, pear=5}");
       System.out.println("Actual  :" + result + " " + inventory);
       System.out.println();
     }
```

```
    {
      Map<String, Integer> inventory = new TreeMap<String, Integer>(
              Map.of("apple", 3, "orange", 4, "pear", 5));

      Map<String, Integer> order = new TreeMap<String, Integer>(
              Map.of("apple", 5, "orange", 2)); // not enough apple

      boolean result = buyFruits(inventory, order);
      System.out.println("Test 3");
      System.out.println("Expected:false {apple=3, orange=4, pear=5}");
      System.out.println("Actual  :" + result + " " + inventory);
      System.out.println();
    }

    {
      Map<String, Integer> inventory = new TreeMap<String, Integer>(
              Map.of("apple", 3, "orange", 4, "pear", 5));

      Map<String, Integer> order = new TreeMap<String, Integer>(
              Map.of("guava", 5)); // guava is not available

      boolean result = buyFruits(inventory, order);
      System.out.println("Test 4");
      System.out.println("Expected:false {apple=3, orange=4, pear=5}");
      System.out.println("Actual  :" + result + " " + inventory);
      System.out.println();
    }
   }
}
```

It will generate the following output:

```
Test 1
Expected:false {apple=3, orange=4, pear=5}
Actual  :false {apple=3, orange=4, pear=5}

Test 2
Expected:true {apple=1, orange=2, pear=5}
Actual  :true {apple=1, orange=2, pear=5}

Test 3
Expected:false {apple=3, orange=4, pear=5}
Actual  :false {apple=3, orange=4, pear=5}

Test 4
Expected:false {apple=3, orange=4, pear=5}
Actual  :false {apple=3, orange=4, pear=5}
```

Answer