Score: 3.500 (=100.0%)
Id: 42140
Veza između dostavljača hrane i naručitelja modelirana je klasom `Connection`:

```
class Connection {

    private String address;
    private float distance;

    public Connection(String address, float distance) {
        this.address=address;
        this.distance=distance;
    }

    public String getAddress() {
        return address;
    }

    public float getDistance() {
        return distance;
    }

}
```

Dostavljači hrane svakodnevno dostavljaju hranu na što je moguće više adresa. Potrebno je pronaći dostavljača koji je obavio najviše dostava ili skupio najviše kilometara (suma ostvarenih udaljenosti u svim dostavama).

Podaci o dostavama spremaju se u Mapu čiji je ključ ime dostavljača, a vrijednost kolekcija obavljenih dostava. Ulazni podaci mogu izgledati npr ovako:

```
"Mario" -> [("Mlinarska 23", 4.9), ("Dolčić 6a", 7.2), ("Maksimirska", 7.7)],

"Ivica" -> [("Filipovićeva 10", 7.4),("Jordanovac 8", 8.1)],

"Josip" -> [("Pantovčak 105", 4),("Zelengaj 48", 3.5),("Gornje Prekrižja 55", 2.3),("Mlinovi 15", 2.5)]
```

Vaš zadatak je napisati metode za dohvat i obradu ovih podataka, tako što ćete nadopuniti sljedeću klasu:

```
class DeliverySystem {

      public static Map<String, Integer> numOfDeliveriesPerDeliveryMan(Map<String,Collection<Connection>> data) {
            // Metoda vraća broj dostava za svakog dostavljača

      }

      public static Map<String, Float> distancePerDeliveryMan(Map<String,Collection<Connection>> data) {
            // Metoda vraća ukupnu udaljenost koju je svaki dostavljač prošao
      }

}
```

Radi jednostavnosti, pretpostavite da su sva imena dostavljača različita, tj. ne može se dogoditi da postoje dva dostavljača s istim imenom. **Elemenate u obje mape poredajte abecedno po imenu dostavljača**.

Za gornji primjer, ove dvije metode trebaju vratiti sljedeće:

```
metoda numOfDeliveriesPerDeliveryMan -> ["Ivica" -> 2, "Josip" -> 4, "Mario" -> 3]

metoda distancPerDeliveryMan -> ["Ivica" -> 15.5, "Josip" -> 12.3, "Mario" -> 19.8]
```

OPASKE:

- Prilikom predaje, predati kompletnu klasu DeliverySystem.
- Klasu `DeliverySystem` napisati bez modifikatora vidljivosti

Student's answer:

```
1  import java.util.Collection;
2  import java.util.Map;
3  import java.util.TreeMap;
4
5  class DeliverySystem {
6
7      public static Map<String, Integer> numOfDeliveriesPerDeliveryMan(Map<String, C
8          // Metoda vraća broj dostava za svakog dostavljača
9          TreeMap<String, Integer> deliveries = new TreeMap<>();
10         for (String man : data.keySet()) {
11             int count = data.get(man).size();
12             deliveries.put(man, count);
13         }
14         return deliveries;
15     }
16
17     public static Map<String, Float> distancePerDeliveryMan(Map<String,Collection<
18         // Metoda vraća ukupnu udaljenost koju je svaki dostavljač prošao
19         TreeMap<String, Float> distances = new TreeMap<>();
20         for (String man : data.keySet()) {
21             float distance = 0;
22             for (Connection obj : data.get(man)) {
23                 distance += obj.getDistance();
24             }
25             distances.put(man, distance);
26         }
27         return distances;
28     }
29
30 }
31
32
33
34
35
36
37
38
39
40
41
42
```

Correct answer:

```
1 This test does not have "show solutions" option enabled.
```

Hint: Correct. Well done!

Exam results obtained during submission evaluation:

#	stdin	stdout	expected	stderr	percentage	hint	mode	correct?
1	numOfDeliveriesPerDeliveryMan	{Ivica=2, Josip=4, Mario=3}	{Ivica=2, Josip=4, Mario=3}		50.00	Correct. Well done!	check elements order : false, case sensitive : false, ignore whitespace : true	true
2	distancePerDeliveryMan	{Ivica=15.5, Josip=12.3, Mario=19.8}	{Ivica=15.5, Josip=12.3, Mario=19.8}		50.00	Correct. Well done!	check elements order : false, case sensitive : true, ignore whitespace : true	true

Student's result | Correct result

Rerun student's code