počet rôznych slov dlhších ako 2 (Pozn. riadky treba rozdeliť na slová)

```
rdd.flatMap(line -> Arrays.asList(line.split(" ")).iterator()).distinct() .filter(word -> word.length() > 2) .count();
```
---

symetrickej diferencie množín reťazcov t.j. celkového počtu rôznych reťazcov, ktoré sa nachadzajú práve v jednej z kolekcií (ale nie v oboch)

```
rd1.union(rd2).distinct().subtract(rd1.intersection(rd2)).count();
```
---

symetrickej diferencie množín reťazcov t.j. celkového počtu reťazcov, ktoré sa nachádzajú len v jednej z kolekcií ale nie v oboch. Reťazce líšiace sa len VEĽKOSŤOU PÍSMEN považujte za totožné.

```
rd1 = rd1.map(s -> s.toUpperCase());
rd2 = rd2.map(s -> s.toUpperCase());
return rd1.union(rd2).distinct().subtract(rd1.intersection(rd2)).count();
```
---

boolovská hodnota hovoriaca či sú v kolekcii duplicitné reťazce

```
rdd.distinct().count() != rdd.count();
```
---

mapa (java.util.map) početností jednotlivých slov v kolekcii (t.j. kľúč je slovo a hodnota je počet výskytov). Slová lí šiace sa len veľkosťou písmen považujte za totožné.

```
rdd.map(s -> s.toLowerCase()).countByValue();
```
---

zoznam (java.util.List) obsahujúci všetky rôzne slová, ktoré sa nachádzajú v prvom súbore ale nenachádzajú v druho m. (Pozn. riadky treba rozdeliť na slová)

```
rdd1.flatMap(s -> Arrays.asList(s.split(" ")).iterator()).subtract(rdd2.flatMap(s -> Arrays.asList(s.split(" ")).iterator()
)).collect();
```
---

mapa (java.util.map) početností jednotlivých slov v súbore (t.j. kľúč je slovo a hodnota je počet výskytov). (Pozn. ria dky treba rozdeliť na slová)

```
rdd.flatMap(s -> Arrays.asList(s.split(" ")).iterator()).countByValue();
```
---

zoznam (java.util.List) obsahujúci všetky rôzne slová, ktoré sa nachádzajú v prvom súbore ale nenachádzajú v druho m. (Pozn. riadky treba rozdeliť na slová)

```
rdd1.flatMap( s -> Arrays.asList(s.split(" ")).iterator()).subtract(rdd2.flatMap( s -> Arrays.asList(s.split(" ")).iterator(
))).collect()
```
---

symetrickej diferencie množín reťazcov t.j. Celkového počtu reťazcov, ktoré sa nachádzajú len v jednej z kolekcií (al e nie v oboch). Reťazce líšiace sa len prázdnymi znakmi na začiatku a konci reťazca považujte za totožné.

```
rdd1 = rdd1.map(s -> s.trim());
rdd2 = rdd2.map(s -> s.trim());
```

```
rdd1.union(rdd2).distinct().subtract(rdd1.intersection(rdd2)).count();
```
---

boolovská hodnota hovoriaca či sú v kolekcii duplicity (t.j. ak sa v kolekcii vyskytuje reťazec viac krát výraz hodnot u true inak false). Reťazce líšiace sa len veľkosťou písmen považujte pri tom za rovnaké.

```
rdd.map(s -> s.toLowerCase()).distinct().count() != rdd.count();
```
---

mapa (java.util.map) udávajuca počet rôznych slov v každom súbore (t.j. kľúč je meno súboru a hodnota počet)

```
rdd.flatMapValues(x->Arrays.asList(x.split("\\s")).iterator()).distinct().countByKey();
```
---

počet rôznych slov v súbore, pričom slová líšiace sa len veľkosťou písmen považujte za totožné. (Pozn. riadky treba rozdeliť na slová)

```
rdd.flatMap(s -> Arrays.asList(s.split(" ")).iterator()).map(s -> s.toLowerCase()).distinct().count();
```

JavaPairRDD<String, List<String>> pdd je kolekcia dvojíc, kde prvá zložka je meno študenta druhá názvov predmet u, ktorý má zapísaný. S využitím operacií RDD-api napíšte výraz, ktorý pre každého študenta vypíše na štandardny výstup riadok obsahujúci meno študenta a reťazec zložený z názvov jeho predmetov oddelených čiarkou. (napr. Fero ASOS,VSA,RZZ)

```
pdd.collectAsMap().forEach((k,v) -> System.out.println(k + " " + String.join(",", v)));
```

JavaPairRDD<String, String> pdd je kolekcia dvojíc, kde prvá zložka je meno študenta druhá názvov predmetu, ktor ý má zapísaný. S využitím operacií RDD-api napíšte výraz, ktorý pre každého študenta vypíše na štandardny výstup riadok obsahujúci meno študenta a reťazec zložený z názvov jeho predmetov oddelených čiarkou. (napr. Fero ASOS ,VSA,RZZ)

```
pdd.flatMapValues(x->Arrays.asList(x.split(" ")).iterator()).groupByKey().collectAsMap().forEach((k, v) -> System .out.println(k + " " + String.join(",", v)));
```
---

počet výskytov slova ASOS v súbore, pričom nezáleží na veľkosti písmen (Pozn. riadky treba rozdeliť na slová)

```
rdd.flatMap(s -> Arrays.asList(s.split(" ")).iterator()).map(s -> s.toLowerCase()).filter(s -> s.equals("asos")).count()
```
---

dĺžku najdhšieho slova v súbore (Pozn. riadky treba rozdeliť na slová)

```
rdd.flatMap(s -> Arrays.asList(s.split(" ")).iterator()).map(String::length).reduce(Math::max)
```