```
pip install pyspark
```

```
Requirement already satisfied: pyspark in /usr/local/lib/python3.11/dist-packages (3.5.5)
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.11/dist-packages (from pyspark) (0.10
```

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("Word Count").getOrCreate()
```

```
sc = spark.sparkContext
```

```
sc.defaultParallelism
```

```
2
```

```
dataRDD = sc.textFile("/content/sample_data/data/")
```

```
dataRDD.count()
```

```
8
```

```
dataRDD.getNumPartitions()
```

```
3
```

```
dataRDD = dataRDD.repartition(2)
```

```
dataRDD.getNumPartitions()
```

```
2
```

```
for line in dataRDD.collect():
  print(line)
```

```
car river
river
bear bear
car river dear bear
river car car
bear bear river river car
car car river
river bear river
```

```
mapRDD = dataRDD.flatMap(lambda a : a.split(' '))
```

```
mapRDD.getNumPartitions()
```

```
2
```

```
for line in mapRDD.collect():
  print(line)
```

```
car
river
river
bear
bear
car
river
dear
bear
```

```
river
car
car
bear
bear
river
river
car
car
car
river
river
bear
river
```

```
mapRDD.count()
```

```
23
```

```
keybyword2 = mapRDD.map(lambda word : (word,1))
```

```
for line in keybyword2.collect():
  print(line)
```

```
('car', 1)
('river', 1)
('river', 1)
('bear', 1)
('bear', 1)
('car', 1)
('river', 1)
('dear', 1)
('bear', 1)
('river', 1)
('car', 1)
('car', 1)
('bear', 1)
('bear', 1)
('river', 1)
('river', 1)
('car', 1)
('car', 1)
('car', 1)
('river', 1)
('river', 1)
('bear', 1)
('river', 1)
```

```
counts = keybyword2.reduceByKey(lambda a,b : a+b)
```

```
for line in counts.collect():
  print(line)
```

```
('car', 7)
('river', 9)
('bear', 6)
('dear', 1)
```

```
counts.saveAsTextFile("/content/sample_data/pyspark1")
```

```
keySorted = counts.sortByKey(False)
```

```
for line in keySorted.collect():
  print(line)
```

```
('river', 9)
('dear', 1)
```

```
        ('car', 7)
        ('bear', 6)
```

```
keySorted.saveAsTextFile("/content/sample_data/pyspark2")
```

```
valueSorted = counts.sortBy(lambda a : -a[1])
```

```
for line in valueSorted.collect():
  print(line)
```

```
('river', 9)
('car', 7)
('bear', 6)
('dear', 1)
```

```
counts.cache()
```

```
PythonRDD[21] at collect at <ipython-input-24-e159b5767c36>:1
```