import org.apache.spark.SparkConf  
import org.apache.spark.SparkContext  
  
object TemperatureCount {  
 def main(args: Array[String]): Unit = {  
 // Create Spark configuration  
 val conf = new SparkConf().setAppName("TemperatureCount").setMaster("local")  
 val sc = new SparkContext(conf)  
 val input\_data = List(  
 "s1,2016-01-01,20.5",  
 "s2,2016-01-01,30.1",  
 "s1,2016-01-02,60.2",  
 "s2,2016-01-02,20.4",  
 "s1,2016-01-03,55.5",  
 "s2,2016-01-03,52.5"  
 )  
 val rdd = sc.parallelize(input\_data)  
 val sensorTemperaturesRDD = rdd.map(line => {  
 val parts = line.split(",")  
 (parts(0), parts(2).toFloat)  
 })  
 val filteredTemperaturesRDD = sensorTemperaturesRDD.filter(\_.\_2 > 50)  
 val sensorCountsRDD = filteredTemperaturesRDD.map(sensorTemp => (sensorTemp.\_1, 1))  
 val sensorTemperatureCounts = sensorCountsRDD.reduceByKey(\_ + \_)  
 val result = sensorTemperatureCounts.collect()  
 *println*("Count, Sensor Name")  
 result.foreach { case (sensor, count) =>  
 *println*(s"**$**count, **$**sensor")  
 }  
 sc.stop()  
 }  
}

Output:

Count, Sensor Name

1, s2

2, s1

import org.apache.spark.SparkConf  
import org.apache.spark.SparkContext  
  
object TemperatureCount {  
 def main(args: Array[String]): Unit = {  
 val conf = new SparkConf().setAppName("TemperatureCount").setMaster("local")  
 val sc = new SparkContext(conf)  
 val input\_data = List(  
 "s1,2016-01-01,20.5",  
 "s2,2016-01-01,30.1",  
 "s1,2016-01-02,60.2",  
 "s2,2016-01-02,20.4",  
 "s1,2016-01-03,55.5",  
 "s2,2016-01-03,52.5"  
 )  
 val rdd = sc.parallelize(input\_data)  
 val temperaturesRDD = rdd.map(line => {  
 val parts = line.split(",")  
 parts(2).toDouble  
 })  
 val maxTemperature = temperaturesRDD.max()  
 *println*(s"Highest temperature recorded: **$**maxTemperature")  
 sc.stop()  
 }  
}

Output:

Highest temperature recorded: 60.2