**import** java.io.{BufferedWriter, File, FileWriter}  
  
**import** sun.plugin.util.UIUtil  
  
*/\*\*  
 \* Created by Administrator on 2017/3/25 0025.  
 \*/***object** ProData {  
 **def** main(args: Array[String]): Unit = {  
 **val** file = **new** File(**"G:\\data.txt"**)  
 **if**(!file.exists()) file.createNewFile()  
 *//数据产生* **var** flag = **true  
 var** num = 0  
  
 **val** writer = **new** BufferedWriter(**new** FileWriter(file))  
 **while**(flag){  
 **val** buffer = **new** StringBuffer()  
 **for**(i <- 0 to 10){  
 buffer.append(**"Hello "**)  
 **for**(i <- 0 to 8){  
 buffer.append(**"Apache "**)  
 }  
 **for**(i<- 0 to 3){  
 buffer.append(**"Spark "**)  
 }  
 buffer.append(**"! "**)  
 }  
 buffer.append(**"\n"**)  
 writer.write(buffer.toString)  
 writer.flush()  
 num +=1  
 **if**(num == 50000000){  
 flag = **false** }  
 }  
 }  
}

**import** org.apache.log4j.{Level, Logger}  
**import** org.apache.spark.{Partitioner, SparkConf, SparkContext}  
**import** org.apache.spark.rdd.RDD  
  
**import** scala.util.Random  
  
*/\*\*  
 \* Created by Administrator on 2017/4/9 0009.  
 \*/***object** WordCount {  
 **def** main(args: Array[String]): Unit = {  
 Logger.*getLogger*(**"org.apache.spark"**).setLevel(Level.*WARN*)  
 Logger.*getLogger*(**"org.eclipse.jetty.server"**).setLevel(Level.*OFF*)  
 *//spark入口  
 //spark conf配置对象* **val** conf = **new** SparkConf().setAppName(**"WordCount"**).setMaster(**"local[2]"**)  
 **val** sc = **new** SparkContext(conf)  
  
 *//RDD* **val** file: RDD[String] = sc.textFile(**"file:\\G:\\data.txt"**)  
 **val** flatMap: RDD[String] = file.flatMap(line => line.split(**" "**))  
  
 *//自定义分区  
// val map: RDD[(String, Long)] = flatMap.map(word => (word,1l))  
// val byKey = map.reduceByKey(\_ + \_).sortByKey(false).groupByKey(new Partitioner {override def numPartitions: Int = 6  
//  
// override def getPartition(key: Any): Int = key.hashCode() % 6  
// })  
// byKey.foreach( wordCount => println(wordCount))  
  
 //添加随机前缀  
// var map: RDD[(String, Long)] = flatMap.map(word => (new Random().nextInt(5)+"\_"+word,1l))* **var** map = flatMap.map(word => **if**(word.contains(**"Apache"**)){ *//通过配置文件，或者是数组等对类似类别id,地区id等通常会造成数据倾斜的id做随机处理* (**new** Random().nextInt(5)+**"\_"**+word,1l)  
 }**else**{  
 (word,1l)  
 }  
 )  
 **var** byKey = map.reduceByKey(\_ + \_)  
 byKey.foreach( wordCount => *println*(wordCount))  
*// var rdd = byKey.map(word => (word.\_1.split("\_")(1),word.\_2))* **var** rdd = byKey.map(word => **if**(word.\_1.contains(**"\_"**)){  
 (word.\_1.split(**"\_"**)(1),word.\_2)  
 }**else**{  
 (word.\_1,word.\_2)  
 }  
 )  
  
 **var** key = rdd.reduceByKey(\_ + \_).sortByKey(**false**).groupByKey()  
 key.foreach( wordCount => *println*(wordCount))  
  
 }  
}