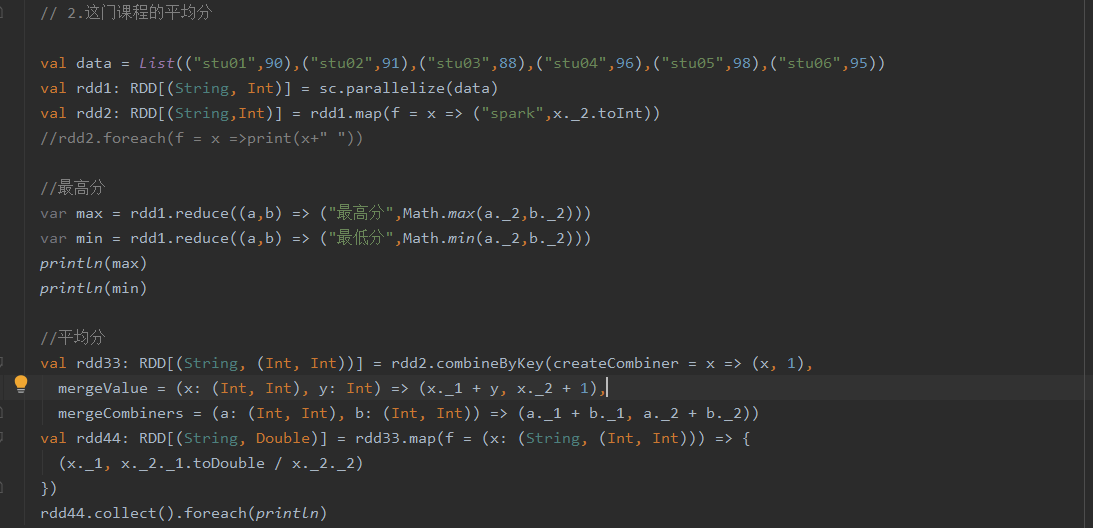
RDD编程求最高分





代码：

package scala

import org.apache.spark.rdd.RDD  
import org.apache.spark.{SparkConf, SparkContext}  
  
object homework\_1 {  
 def main(args: Array[String]): Unit = {  
 val conf: SparkConf = new SparkConf()  
 conf.setAppName("homework\_1")  
 conf.setMaster("local")  
 val sc: SparkContext = new SparkContext(conf)  
 // 1.最高分的学生和分数  
 // 2.这门课程的平均分  
  
 val data = *List*(("stu01",90),("stu02",91),("stu03",88),("stu04",96),("stu05",98),("stu06",95))  
 val rdd1: RDD[(String, Int)] = sc.parallelize(data)  
 val rdd2: RDD[(String,Int)] = rdd1.map(f = x => ("spark",x.\_2.toInt))  
 //rdd2.foreach(f = x =>print(x+" "))  
  
 //最高分  
 var max = rdd1.reduce((a,b) => ("最高分",Math.*max*(a.\_2,b.\_2)))  
 var min = rdd1.reduce((a,b) => ("最低分",Math.*min*(a.\_2,b.\_2)))  
 *println*(max)  
 *println*(min)  
  
 //平均分  
 val rdd33: RDD[(String, (Int, Int))] = rdd2.combineByKey(createCombiner = x => (x, 1),  
 mergeValue = (x: (Int, Int), y: Int) => (x.\_1 + y, x.\_2 + 1),  
 mergeCombiners = (a: (Int, Int), b: (Int, Int)) => (a.\_1 + b.\_1, a.\_2 + b.\_2))  
 val rdd44: RDD[(String, Double)] = rdd33.map(f = (x: (String, (Int, Int))) => {  
 (x.\_1, x.\_2.\_1.toDouble / x.\_2.\_2)  
 })  
 rdd44.collect().foreach(*println*)  
 }  
}