TOP GEAR ASSIGNMENTS FOR APACHE SPARK ASSIGNMENTS – L1

Assignments – 1 (Creation of RDD and operations on RDDs )

Note : Identify your spark context and answer the following. Wherever applicable create a new RDD.

Consider the following array.

{10,21,90,34,40,98,21,44,59,21,90,34,29,19, 21,90,34,29,49,78 } . Create a new RDD for each of the following assignments.

1. For the above array,

a) Create a RDD for the above array.

b) Display the array

c) Display the first element of the array

2. Consider the above array

a) Display the sorted output (ascending and descending) through an RDD.

b) Display the distinct elements of the array using an RDD

c) Display distinct elements without using a new RDD.

3. Consider the above array

a) Display maximum and minimum of given array using RDD.

b) Display top 5 list elements using RDD

c) Combine above array with a new array { 30,35,45,60,75,85} and display output.

d) Provide the sum of the array elements using reduce with distinct values.

e) Provide the sum of the array elements using reduce.

Solution –

def main(args: Array[String]): Unit = {

    val arr: Array[Int] = Array(10,21,90,34,40,98,21,44,59,21,90,34,29,19,21,90,34,29,49,78)

    val conf = new SparkConf().setMaster("local[\*]").setAppName("Topgear Training")

    val sc = new SparkContext(conf)

    //1-a)

    val rdd = sc.parallelize(arr)

    solution1(sc, rdd)

    solution2(sc,rdd)

    solution3(sc,rdd)

  }

  private def solution1(sc: SparkContext, rdd: RDD[Int]): Unit = {

    //b) Print the RDD

    rdd.foreach(println)

    //c) Finding the First element

    val firstElement = rdd.first()

    println(firstElement)

  }

  private def solution2(sc: SparkContext, rdd: RDD[Int]): Unit = {

    //2-a) Descending order Rdd

    val sortedRddDesc = rdd.sortBy(x => x, ascending = false)

    sortedRddDesc.collect().foreach(println)

    //2-a) Ascending order RDD

    val sortedRddAsc = rdd.sortBy(x => x)

    sortedRddAsc.collect().foreach(println)

    //2-b) Distinct RDD

    val distinctRdd = rdd.distinct()

    distinctRdd.collect().foreach(println)

    //2-c) Created a set to store the distinct values

    var distinctValues = Set[Int]()

    rdd.collect().foreach( el =>

      distinctValues += el

    )

    distinctValues.foreach(value => println(value))

  }

  private def solution3(sc: SparkContext, rdd: RDD[Int]): Unit = {

    //3-a) Maximum value

    val max = rdd.max()

    println(max)

    //3-a) Minimum Value

    val min = rdd.min()

    println(min)

    //3-b) Create a list of top 5 element and print those

    val listEl = rdd.collect().take(5).toList

    for(l <- listEl) {

      println(l)

    }

    //3-c) Combine a new Array newArr = Array(30,35,45,60,75,85)

    val newArr = Array(30,35,45,60,75,85)

    val newRDD = sc.parallelize(newArr)

    val afterJoiningRDD = rdd.union(newRDD)

    afterJoiningRDD.collect().foreach(println)

    //3-d) Sum with Distinct of the new RDD after union in 3-c

    val sumDistinct = afterJoiningRDD.distinct().reduce(\_ + \_)

    println(sumDistinct)

    //3-e) Sum of all the elements of the new RDD after union in 3-c

    val sumAll = afterJoiningRDD.reduce(\_ + \_)

    println(sumAll)

  }