Free Questions for CCA175 by vceexamstest

Shared by Swanson on 15-04-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

Problem Scenario 63 : You have been given below code snippet.

val a = sc.parallelize(List("dog", "tiger", "lion", "cat", "panther", "eagle"), 2)

val b = a.map(x => (x.length, x))

operation1

Write a correct code snippet for operationl which will produce desired output, shown below. Array[(lnt, String}] = Array((4,lion), (3,dogcat), (7,panther), (5,tigereagle))

Options:

A- Solution :
b.reduceByKey(_ + _).collect
reduceByKey JPair] : This function provides the well-known reduce functionality in Spark. Please note that any function f you provide, should be commutative in order to generate reproducible results.

B- Solution :
b.reduceByKey(_ + _).collect
reduceByKey JPair] : This function provides the well-known reduce functionality in Spark.

Answer:

A

Question 2

Question Type: MultipleChoice

Problem Scenario 62 : You have been given below code snippet.

val a = sc.parallelize(List("dogM, "tiger", "lion", "cat", "panther", "eagle"), 2)

val b = a.map(x => (x.length, x))

operation1

Write a correct code snippet for operationl which will produce desired output, shown below. Array[(lnt, String)] = Array((3,xdogx), (5,xtigerx), (4,xlionx), (3,xcatx), (7,xpantherx), (5,xeaglex))

Options:

A- Solution :

b.mapValuesf'x' + _ + 'x').collect

mapValues [Pair] : Takes the values of a RDD that consists of two-component tuples, and applies the provided function to transform

each value. Tlien,.it.forms newtwo-componendtuples using the key and the transformed value and stores them in a new RDD.

B- Solution :

b.mapValuesf'x' + _ + 'x').collect

mapValues [Pair] : Takes the values of a RDD that consists of two-component tuples, and applies the provided function to transform each value.

Answer:

A

Question 3

Question Type: **MultipleChoice**

Problem Scenario 61 : You have been given below code snippet.

val a = sc.parallelize(List("dog", "salmon", "salmon", "rat", "elephant"), 3)

val b = a.keyBy(_.length)

val c = sc.parallelize(List("dog","cat","gnu","salmon","rabbit","turkey","wolf","bear","bee"), 3)

val d = c.keyBy(_.length) operationl

Write a correct code snippet for operationl which will produce desired output, shown below.

Array[(Int, (String, Option[String]}}] = Array((6,(salmon,Some(salmon))), (6,(salmon,Some(rabbit))),

(6,(salmon,Some(turkey))), (6,(salmon,Some(salmon))), (6,(salmon,Some(rabbit))), (6,(salmon,Some(turkey))), (3,(dog,Some(dog))), (3,(dog,Some(cat))), (3,(dog,Some(dog))), (3,(dog,Some(bee))), (3,(rat,Some(dogg)), (3,(rat,Some(cat)j), (3,(rat.Some(gnu))). (3,(rat,Some(bee))), (8,(elephant,None)))

Options:

A- Solution :

b.leftOuterJoin(d}.collect

leftOuterJoin [Pair]: Performs an left outer join using two key-value RDDs. Please note that the keys must be generally comparable to make this work keyBy : Constructs two-component tuples (key-value pairs) by applying a function on each data item. Trie result of the function becomes the key and the original data item becomes the value of the newly created tuples.

B- Solution :

b.leftOuterJoin(d}.collect

leftOuterJoin [Pair]: Performs an left outer join using two key-value RDDs. Please note that the keys must be generally comparable to make this work keyBy : Constructs two-component tuples (key-value pairs) by applying a function on each data item.

Answer:

A

Question 4

Question Type: MultipleChoice

Problem Scenario 60 : You have been given below code snippet.

val a = sc.parallelize(List("dog", "salmon", "salmon", "rat", "elephant"}, 3}

val b = a.keyBy(_.length)

val c = sc.parallelize(List("dog","cat","gnu","salmon","rabbit","turkey","woif","bear","bee"), 3)

val d = c.keyBy(_.length)

operation1

Write a correct code snippet for operationl which will produce desired output, shown below.

Array[(lnt, (String, String))] = Array((6,(salmon,salmon)), (6,(salmon,rabbit)), (6,(salmon,turkey)), (6,(salmon,salmon)), (6,(salmon,rabbit)),

(6,(salmon,turkey)), (3,(dog,dog)), (3,(dog,cat)), (3,(dog,gnu)), (3,(dog,bee)), (3,(rat,dog)), (3,(rat,cat)), (3,(rat,gnu)), (3,(rat,bee)))

Options:

A- solution:
b.join(d).collect
join [Pair]: Performs an inner join using two key-value RDDs. Please note that the keys must be generally comparable to make this work.
keyBy : Constructs two-component tuples (key-value pairs) by applying a function on each data item.

B- solution:

b.join(d).collect

join [Pair]: Performs an inner join using two key-value RDDs. Please note that the keys must be generally comparable to make this work.

keyBy : Constructs two-component tuples (key-value pairs) by applying a function on each data item. The result of the function becomes the data item becomes the key and the originalvalue of the newly created tuples.

Answer:

B

Question 5

Question Type: **MultipleChoice**

Problem Scenario 59 : You have been given below code snippet.

val x = sc.parallelize(1 to 20)

val y = sc.parallelize(10 to 30) operationl

z.collect

Write a correct code snippet for operationl which will produce desired output, shown below. Array[lnt] = Array(16,12, 20,13,17,14,18,10,19,15,11)

Options:

A- Solution :

val z = x.intersection(y)

intersection : Returns the elements in the two RDDs which are the same.

B- Solution :

val z = x.intersection(y)

intersection : Returns the elements in the two RDs which are the same.

Answer:

A

Question 6

Question Type: **MultipleChoice**

Problem Scenario 58 : You have been given below code snippet.

val a = sc.parallelize(List("dog", "tiger", "lion", "cat", "spider", "eagle"), 2) val b = a.keyBy(_.length)

operation1

Write a correct code snippet for operationl which will produce desired output, shown below.

Array[(Int, Seq[String])] = Array((4,ArrayBuffer(lion)), (6,ArrayBuffer(spider)), (3,ArrayBuffer(dog, cat)), (5,ArrayBuffer(tiger, eagle}}}

Options:

A- Solution :

b.groupByKey.collect

groupByKey [Pair]

Very similar to groupBy, but instead of supplying a function, the key-component of each pair will automatically be presented to the partitioner.

Listing Variants

def groupByKeyO: RDD[(K, lterable[V]}]

def groupByKey(numPartittons: Int): RDD[(K, lterable[R])]

def groupByKey(partitioner: Partitioner): RDD[(K, lterable[V])]

B- Solution :

b.groupByKey.collect

groupByKey [Pair]

Very similar to groupBy, but instead of supplying a function, the key-component of each pair will automatically be presented to the partitioner.

Listing Variants

def groupByKeyQ: RDD[(K, lterable[V]}]

def groupByKey(numPartittons: Int): RDD[(K, lterable[V])]

def groupByKey(partitioner: Partitioner): RDD[(K, lterable[V])]

Answer:

B

Question 7

Question Type: MultipleChoice

Problem Scenario 57 : You have been given below code snippet.

val a = sc.parallelize(1 to 9, 3) operationl

Write a correct code snippet for operationl which will produce desired output, shown below.

Array[(String, Seq[lnt])] = Array((even,ArrayBuffer(2, 4, G, 8)), (odd,ArrayBuffer(1, 3, 5, 7, 9)))

Options:

A- Solution :

a.groupBy(x => {if (x % 2 == 0) 'even' else 'odd' }).collect

B- Solution :

a.groupBy(x => {if (x % 3 == 8) 'even' else 'odd' }).collect

Answer:

A

Question 8

Question Type: MultipleChoice

Problem Scenario 56 : You have been given below code snippet.

val a = sc.parallelize(l to 100. 3)

operation1

Write a correct code snippet for operationl which will produce desired output, shown below.

Array [Array [I nt]] = Array(Array(1, 2, 3,4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33),

Array(34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66),

Array(67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100))

Options:

A- Solution : a.glom.collect
glom
Assembles an array that contains all elements of the partition and embeds it in an RDD.

B- Solution : a.glom.collect
glom
Assembles an array that contains all elements of the partition and embeds it in an RDD. Each returned array contains the contents of one panition

Answer:

B

Question 9

Question Type: **MultipleChoice**

Problem Scenario 55 : You have been given below code snippet.

val pairRDDI = sc.parallelize(List(("cat",2), ("cat", 5), ("book", 4),("cat", 12))) val pairRDD2 = sc.parallelize(List(("cat",2), ("cup", 5), ("mouse", 4),("cat", 12)))

operation1

Write a correct code snippet for operationl which will produce desired output, shown below.

Array[(String, (Option[lnt], Option[lnt]))] = Array((book,(Some(4},None)), (mouse,(None,Some(4))), (cup,(None,Some(5))), (cat,(Some(2),Some(2)),(cat,(Some(2),Some(12))), (cat,(Some(5),Some(2))), (cat,(Some(5),Some(12))), (cat,(Some(12),Some(2))), (cat,(Some(12),Some(12)))J

Options:

A- Solution : pairRDD1.fullOuterJoin(pairRDD2).collect
fullOuterJoin [Pair]
Performs the full outer join between two paired RDDs.
Listing Variants
def fullOuterJoin[W](other: RDD[(K, W)], numPartitions: Int): RDD[(K, (Option[V], OptionfW]))]
def fullOuterJoin[W](other: RDD[(K, W}]}: RDD[(K, (Option[V], OptionfW]))]
def fullOuterJoin[W](other: RDD[(K, W)], partitioner: Partitioner): RDD[(K, (Option[V], Option[W]))]

B- Solution : pairRDD1.fullOuterJoin(pairRDD2).collect
fullOuterJoin [Pair]
Performs the full outer join between two paired RDDs.
Listing Variants
def fullOuterJoin[W](other: RDD[(K, W)], partitioner: Partitioner): RDD[(K, (Option[V], Option[W]))]

Answer:

A

Question 10

Question Type: MultipleChoice

Problem Scenario 54 : You have been given below code snippet.

val a = sc.parallelize(List("dog", "tiger", "lion", "cat", "panther", "eagle"))

val b = a.map(x => (x.length, x))

operation1

Write a correct code snippet for operationl which will produce desired output, shown below.

Array[(lnt, String)] = Array((4,lion), (7,panther), (3,dogcat), (5,tigereagle))

Options:

A- Solution :
b.foidByKey('')(_ + J.collect
foldByKey [Pair]
Very similar to fold, but performs the folding separately for each key of the RDD. This function is only available if the RDD consists of two-component tuples
Listing Variants
def foldByKey(zeroValue: V)(func: (V, V) => V): RDD[(K, V}]

def foldByKey(zeroValue: V, numPartitions: lnt)(func: (V, V) => V): RDD[(K, V)]
def foldByKey(zeroValue: V, partitioner: Partitioner)(func: (V, V) => V): RDD[(K, V}]

B- Solution :
b.foidByKey('')(_ + J.collect
foldByKey [Pair]
Very similar to fold, but performs the folding separately for each key of the RDD. This function is only available if the RDD consists of two-component tuples
Listing Variants
def foldByKey(zeroValue: V)(func: (V, V) => V): RDD[(K, V}]
def foldByKey(zeroValue: V, numPartitions: lnt)(func: (V, V) => V): RDD[(K, V)]

Answer:

A

Question 11

Question Type: MultipleChoice

Problem Scenario 53 : You have been given below code snippet.

val a = sc.parallelize(1 to 10, 3)

operation1

b.collect

Output 1

Array[lnt] = Array(2, 4, 6, 8,10)

operation2

Output 2

Array[lnt] = Array(1,2, 4)

Write a correct code snippet for operation1 and operation2 which will produce desired output, shown above.

Options:

A- Solution :
valb = a.filter(_%2==0)
a.filter(_ <4).collect
filter
Evaluates a boolean function for each data item of the RDD and puts the items for which the function returned true into the resulting RDD.
When you provide a filter function, it must be able to handle all data items contained in the RDD. Scala provides so-called partial functions to deal with mixed data types (Tip: Partial functions to deal are very useful if you have some data which may be bad and you do not want to handle but for the good data (matching data) you want to apply some Kind of map function. The following article is good. It teaches you about partial functions in a very nice way and explains why case has to be used for partial functions:article)
Examples for mixed data without partial functions

val b = sc.parallelize(1 to 8)
b.filter(_ <4)xollect
res15: Arrayjlnt] = Array(1, 2, 3)
val a = sc.parallelize(List('cat'\ 'horse', 4.8, 2.5, 2, 'dog'))
a.filter(_<4).collect
error: value < is not a member of Any

B- Solution :
valb = a.filter(_%2==0)
a.filter(_ <4).collect
filter
Evaluates a boolean function for each data item of the RDD and puts the items for which the function returned true into the resulting RDD.
When you provide a filter function, it must be able to handle all data items contained in the RDD. Scala provides so-called partial functions to deal with mixed data types (Tip: Partial functions to deal are very useful if you have some data which may be bad and you do not want to handle but for the good data (matching data) you want to apply some Kind of map function. The following article is good. It teaches you about partial functions in a very nice way and explains why case has to be used for partial functions:article)
Examples for mixed data without partial functions
val b = sc.parallelize(1 to 8)
b.filter(_ <4)xollect
res15: Arrayjlnt] = Array(1, 2, 3)
val a = sc.parallelize(List('cat'\ 'horse', 4.0, 3.5, 2, 'dog'))
a.filter(_<4).collect
error: value < is not a member of Any

Answer:

B

Question 12

Question Type: MultipleChoice

Problem Scenario 52 : You have been given below code snippet.

val b = sc.parallelize(List(1,2,3,4,5,6,7,8,2,4,2,1,1,1,1,1))

Operation_xyz

Write a correct code snippet for Operation_xyz which will produce below output. scalaxollection.Map[lnt,Long] = Map(5 -> 1, 8 -> 1, 3 -> 1, 6 -> 1, 1 -> S, 2 -> 3, 4 -> 2, 7 -> 1)

Options:

A- Solution :

b.countByValue

countByValue

Returns a map that contains all unique values of the RDD and their respective occurrence counts. (Warning: This operation will finallyaggregate the information in a single reducer.)

Listing Variants
def countByValue(): Map[T, Long]

B- Solution :
b.countByValue
countByValue
Returns a map that contains all unique values of the RTT and their respective occurrence counts. (Warning: This operation will finallyaggregate the information in a single reducer.)
Listing Variants
def countByValue(): Map[T, Long]

Answer:

A