Week 4 Assignment

Guruprasad Velikadu Krishnamoorthy

College of Science and Technology, Bellevue University

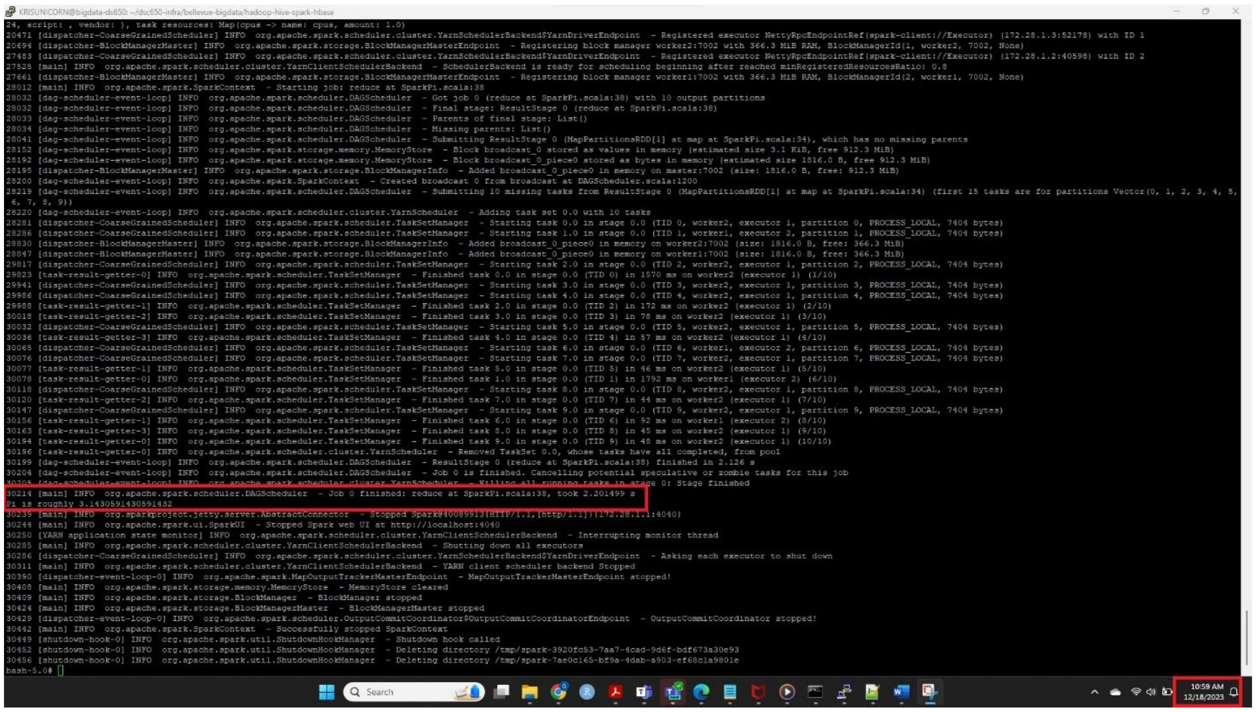
DSC650-T301: Big Data

Professor. Nasheb Ismaily

December 18, 2023

1. SparkPi Program:

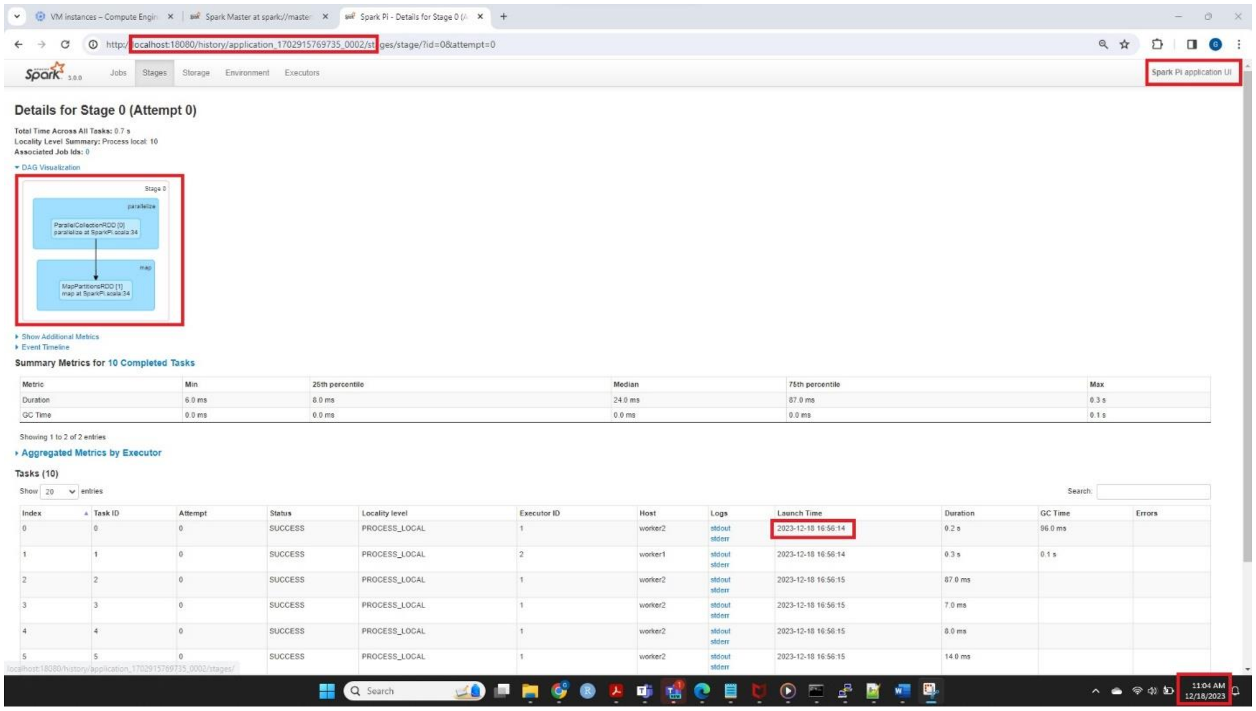
1.1 Screenshot of SparkPi program:



```
KRISUNICORN@bigdata-ds650: ~/dsc650-infra/bellevue-bigdata/hadoop-hive-spark-hbase
24, script: , vendor: ), task resources: Map(cpus -> name: cpus, amount: 1.0)
20471 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.cluster.YarnSchedulerBackend$YarnDriverEndpoint  - Registered executor NettyRpcEndpointRef(spark-client://Executor) (172.28.1.3:52178) with ID 1
20694 [dispatcher-BlockManagerMaster] INFO  org.apache.spark.storage.BlockManagerMasterEndpoint  - Registering block manager worker2:7002 with 366.3 MiB RAM, BlockManagerId(1, worker2, 7002, None)
27483 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.cluster.YarnSchedulerBackend$YarnDriverEndpoint  - Registered executor NettyRpcEndpointRef(spark-client://Executor) (172.28.1.2:40598) with ID 2
27525 [main] INFO  org.apache.spark.scheduler.cluster.YarnClientSchedulerBackend  - SchedulerBackend is ready for scheduling beginning after reached minRegisteredResourcesRatio: 0.8
27661 [dispatcher-BlockManagerMaster] INFO  org.apache.spark.storage.BlockManagerMasterEndpoint  - Registering block manager worker1:7002 with 366.3 MiB RAM, BlockManagerId(2, worker1, 7002, None)
28012 [main] INFO  org.apache.spark.SparkContext  - Starting job: reduce at SparkPi.scala:38
28032 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - Got job 0 (reduce at SparkPi.scala:38) with 10 output partitions
28032 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - Final stage: ResultStage 0 (reduce at SparkPi.scala:38)
28033 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - Parents of final stage: List()
28034 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - Missing parents: List()
28041 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - Submitting ResultStage 0 (MapPartitionsRDD[1] at map at SparkPi.scala:34), which has no missing parents
28152 [dag-scheduler-event-loop] INFO  org.apache.spark.storage.memory.MemoryStore  - Block broadcast_0 stored as values in memory (estimated size 3.1 KiB, free 912.3 MiB)
28192 [dag-scheduler-event-loop] INFO  org.apache.spark.storage.memory.MemoryStore  - Block broadcast_0_piece0 stored as bytes in memory (estimated size 1816.0 B, free 912.3 MiB)
28195 [dispatcher-BlockManagerMaster] INFO  org.apache.spark.storage.BlockManagerInfo  - Added broadcast_0_piece0 in memory on master:7002 (size: 1816.0 B, free: 912.3 MiB)
28200 [dag-scheduler-event-loop] INFO  org.apache.spark.SparkContext  - Created broadcast 0 from broadcast at DAGScheduler.scala:1200
28219 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - Submitting 10 missing tasks from ResultStage 0 (MapPartitionsRDD[1] at map at SparkPi.scala:34) (first 15 tasks are for partitions Vector(0, 1, 2, 3, 4, 5,
 6, 7, 8, 9))
28220 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.cluster.YarnScheduler  - Adding task set 0.0 with 10 tasks
28281 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 0.0 in stage 0.0 (TID 0, worker2, executor 1, partition 0, PROCESS_LOCAL, 7404 bytes)
28286 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 1.0 in stage 0.0 (TID 1, worker1, executor 2, partition 1, PROCESS_LOCAL, 7404 bytes)
28830 [dispatcher-BlockManagerMaster] INFO  org.apache.spark.storage.BlockManagerInfo  - Added broadcast_0_piece0 in memory on worker2:7002 (size: 1816.0 B, free: 366.3 MiB)
28847 [dispatcher-BlockManagerMaster] INFO  org.apache.spark.storage.BlockManagerInfo  - Added broadcast_0_piece0 in memory on worker1:7002 (size: 1816.0 B, free: 366.3 MiB)
29817 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 2.0 in stage 0.0 (TID 2, worker2, executor 1, partition 2, PROCESS_LOCAL, 7404 bytes)
29823 [task-result-getter-0] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 0.0 in stage 0.0 (TID 0) in 1570 ms on worker2 (executor 1) (1/10)
29941 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 3.0 in stage 0.0 (TID 3, worker2, executor 1, partition 3, PROCESS_LOCAL, 7404 bytes)
29986 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 4.0 in stage 0.0 (TID 4, worker2, executor 1, partition 4, PROCESS_LOCAL, 7404 bytes)
29988 [task-result-getter-1] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 2.0 in stage 0.0 (TID 2) in 172 ms on worker2 (executor 1) (2/10)
30018 [task-result-getter-2] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 3.0 in stage 0.0 (TID 3) in 78 ms on worker2 (executor 1) (3/10)
30032 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 5.0 in stage 0.0 (TID 5, worker2, executor 1, partition 5, PROCESS_LOCAL, 7404 bytes)
30036 [task-result-getter-3] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 4.0 in stage 0.0 (TID 4) in 57 ms on worker2 (executor 1) (4/10)
30065 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 6.0 in stage 0.0 (TID 6, worker1, executor 2, partition 6, PROCESS_LOCAL, 7404 bytes)
30076 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 7.0 in stage 0.0 (TID 7, worker2, executor 1, partition 7, PROCESS_LOCAL, 7404 bytes)
30077 [task-result-getter-1] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 5.0 in stage 0.0 (TID 5) in 46 ms on worker2 (executor 1) (5/10)
30078 [task-result-getter-0] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 1.0 in stage 0.0 (TID 1) in 1792 ms on worker1 (executor 2) (6/10)
30118 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 8.0 in stage 0.0 (TID 8, worker2, executor 1, partition 8, PROCESS_LOCAL, 7404 bytes)
30120 [task-result-getter-2] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 7.0 in stage 0.0 (TID 7) in 44 ms on worker2 (executor 1) (7/10)
30147 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.TaskSetManager  - Starting task 9.0 in stage 0.0 (TID 9, worker2, executor 1, partition 9, PROCESS_LOCAL, 7404 bytes)
30156 [task-result-getter-1] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 6.0 in stage 0.0 (TID 6) in 92 ms on worker1 (executor 2) (8/10)
30163 [task-result-getter-3] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 8.0 in stage 0.0 (TID 8) in 45 ms on worker2 (executor 1) (9/10)
30194 [task-result-getter-0] INFO  org.apache.spark.scheduler.TaskSetManager  - Finished task 9.0 in stage 0.0 (TID 9) in 48 ms on worker2 (executor 1) (10/10)
30196 [task-result-getter-0] INFO  org.apache.spark.scheduler.cluster.YarnScheduler  - Removed TaskSet 0.0, whose tasks have all completed, from pool
30199 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - ResultStage 0 (reduce at SparkPi.scala:38) finished in 2.126 s
30204 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.DAGScheduler  - Job 0 is finished. Cancelling potential speculative or zombie tasks for this job
30205 [dag-scheduler-event-loop] INFO  org.apache.spark.scheduler.cluster.YarnScheduler  - Killing all running tasks in stage 0: Stage finished
30214 [main] INFO  org.apache.spark.scheduler.DAGScheduler  - Job 0 finished: reduce at SparkPi.scala:38, took 2.201499 s
Pi is roughly 3.1430591430591432
30239 [main] INFO  org.sparkproject.jetty.server.AbstractConnector  - Stopped Spark@40089913{HTTP/1.1,[http/1.1]}{172.28.1.1:4040}
30244 [main] INFO  org.apache.spark.ui.SparkUI  - Stopped Spark web UI at http://localhost:4040
30250 [YARN application state monitor] INFO  org.apache.spark.scheduler.cluster.YarnClientSchedulerBackend  - Interrupting monitor thread
30285 [main] INFO  org.apache.spark.scheduler.cluster.YarnClientSchedulerBackend  - Shutting down all executors
30286 [dispatcher-CoarseGrainedScheduler] INFO  org.apache.spark.scheduler.cluster.YarnSchedulerBackend$YarnDriverEndpoint  - Asking each executor to shut down
30311 [main] INFO  org.apache.spark.scheduler.cluster.YarnClientSchedulerBackend  - YARN client scheduler backend Stopped
30390 [dispatcher-event-loop-0] INFO  org.apache.spark.MapOutputTrackerMasterEndpoint  - MapOutputTrackerMasterEndpoint stopped!
30408 [main] INFO  org.apache.spark.storage.memory.MemoryStore  - MemoryStore cleared
30409 [main] INFO  org.apache.spark.storage.BlockManager  - BlockManager stopped
30424 [main] INFO  org.apache.spark.storage.BlockManagerMaster  - BlockManagerMaster stopped
30429 [dispatcher-event-loop-0] INFO  org.apache.spark.scheduler.OutputCommitCoordinator$OutputCommitCoordinatorEndpoint  - OutputCommitCoordinator stopped!
30442 [main] INFO  org.apache.spark.SparkContext  - Successfully stopped SparkContext
30449 [shutdown-hook-0] INFO  org.apache.spark.util.ShutdownHookManager  - Shutdown hook called
30452 [shutdown-hook-0] INFO  org.apache.spark.util.ShutdownHookManager  - Deleting directory /tmp/spark-3920fc53-7aa7-4cad-9d6f-bdf673a30e93
30456 [shutdown-hook-0] INFO  org.apache.spark.util.ShutdownHookManager  - Deleting directory /tmp/spark-7ae0c165-bf9a-4dab-a903-ef68c1a9801e
bash-5.0#
```

10:59 AM
12/18/2023

1.2 Output from Spark History Server:



http://localhost:18080/history/application_1702915769735_0002/stages/stage/?id=0&attempt=0

Spark Pi application UI

Details for Stage 0 (Attempt 0)

Total Time Across All Tasks: 0.7 s
Locality Level Summary: Process local: 10
Associated Job Ids: 0

Summary Metrics for 10 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	6.0 ms	8.0 ms	24.0 ms	87.0 ms	0.3 s
GC Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.1 s

Showing 1 to 2 of 2 entries

Aggregated Metrics by Executor

Tasks (10)

Show 20 entries — Search:

Index	Task ID	Attempt	Status	Locality level	Executor ID	Host	Logs	Launch Time	Duration	GC Time	Errors
0	0	0	SUCCESS	PROCESS_LOCAL	1	worker2	stdout stderr	2023-12-18 16:56:14	0.2 s	96.0 ms	
1	1	0	SUCCESS	PROCESS_LOCAL	2	worker1	stdout stderr	2023-12-18 16:56:14	0.3 s	0.1 s	
2	2	0	SUCCESS	PROCESS_LOCAL	1	worker2	stdout stderr	2023-12-18 16:56:15	87.0 ms		
3	3	0	SUCCESS	PROCESS_LOCAL	1	worker2	stdout stderr	2023-12-18 16:56:15	7.0 ms		
4	4	0	SUCCESS	PROCESS_LOCAL	1	worker2	stdout stderr	2023-12-18 16:56:15	8.0 ms		
5	5	0	SUCCESS	PROCESS_LOCAL	1	worker2	stdout stderr	2023-12-18 16:56:15	14.0 ms		

localhost:18080/history/application_1702915769735_0002/stages/

11:04 AM
12/18/2023

1.3 Significance of the SparkPi program:

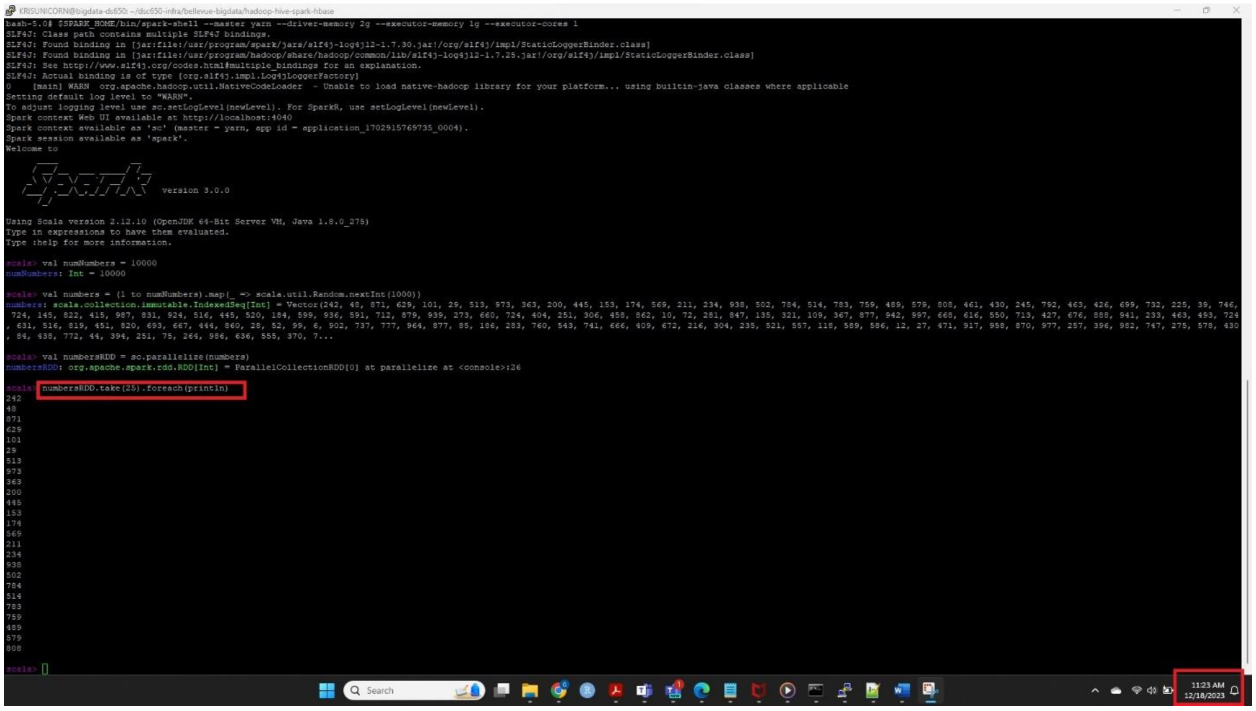
In the Monte Carlo method for estimating pi, random points are generated within a square, and the number of points falling inside a quarter-circle inside that square is counted. The ratio of the points inside the circle to the total points is then used to estimate the pi, based on the formula of the area of the circle and square.

In SparkPi Program, x and y coordinates of the "n" random points are calculated, and the number of points where x*2+y*2 < 1 is calculated. Spark parallelize creates a RDD and distributes the calculation between the executors and returns the result. The Pi is then calculated by using the formula 4 * count/ n.

Note, in our example the value of n passed is 10 and the program is executed in "client" mode with 2 GB of driver memory, 1 GB of executor memory, and 1 core. Yarn is specified as the Resource manager.

```
spark-submit --class org.apache.spark.examples.SparkPi \
    --master yarn \
    --deploy-mode client \
    --driver-memory 2g \
    --executor-memory 1g \
    --executor-cores 1 \
    $SPARK_HOME/examples/jars/spark-examples*.jar \
    10
```

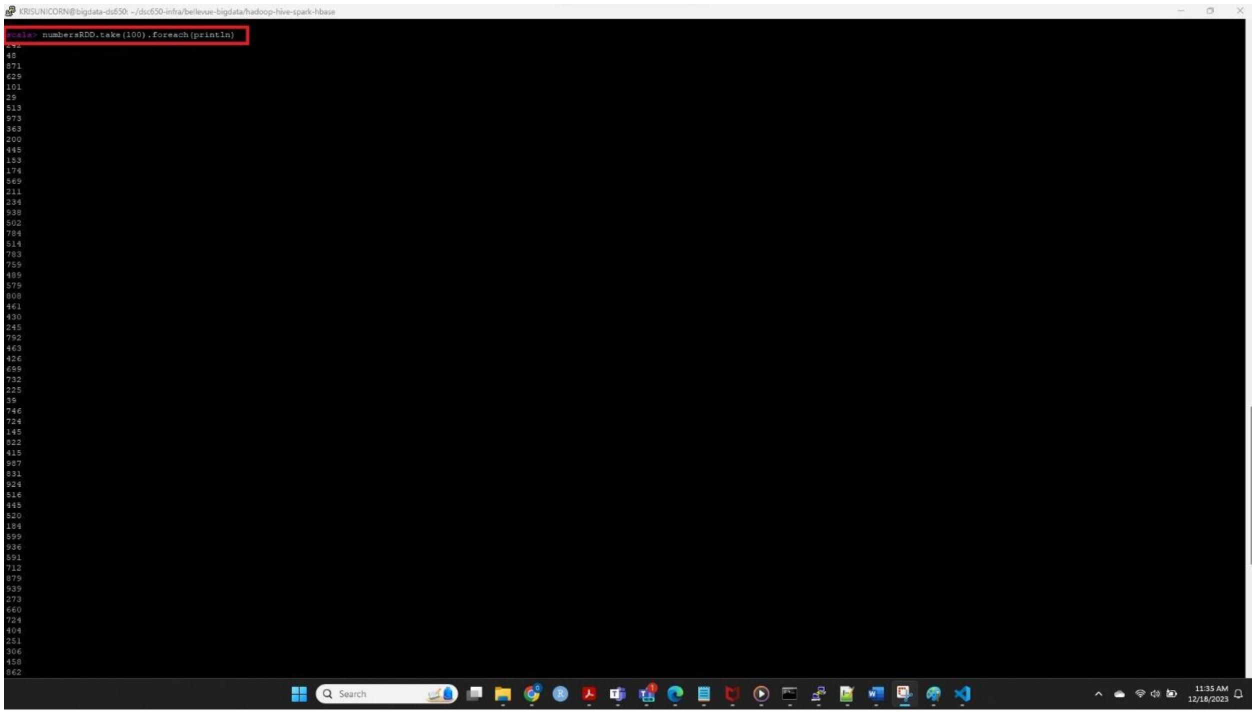
2. Screenshot of the first 100 generated random numbers.



KRISUNICORN@bigdata-ds650: ~/dsc650-infra/bellevue-bigdata/hadoop-hive-spark-hbase

```
bash-5.0# $SPARK_HOME/bin/spark-shell --master yarn --driver-memory 2g --executor-memory 1g --executor-cores 1
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/program/spark/jars/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/program/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
0    [main] WARN  org.apache.hadoop.util.NativeCodeLoader  - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://localhost:4040
Spark context available as 'sc' (master = yarn, app id = application_1702915769735_0004).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.0.0
      /_/

Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 1.8.0_275)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val numNumbers = 10000
numNumbers: Int = 10000

scala> val numbers = (1 to numNumbers).map(_ => scala.util.Random.nextInt(1000))
numbers: scala.collection.immutable.IndexedSeq[Int] = Vector(242, 48, 871, 629, 101, 29, 513, 973, 363, 200, 445, 153, 174, 569, 211, 234, 938, 502, 784, 514, 783, 759, 489, 579, 808, 461, 430, 245, 792, 463, 426, 699, 732, 225, 39, 746,
 724, 145, 822, 415, 987, 831, 924, 516, 445, 520, 184, 599, 936, 591, 712, 879, 939, 273, 660, 724, 404, 251, 306, 458, 862, 10, 72, 281, 847, 135, 321, 109, 367, 877, 942, 997, 668, 616, 550, 713, 427, 676, 888, 941, 233, 463, 493, 724
, 631, 516, 819, 451, 820, 693, 667, 444, 860, 28, 52, 99, 6, 902, 737, 777, 964, 877, 85, 186, 283, 760, 543, 741, 666, 409, 672, 216, 304, 235, 521, 557, 118, 589, 586, 12, 27, 471, 917, 958, 870, 977, 257, 396, 982, 747, 275, 578, 430
, 84, 438, 772, 44, 394, 251, 75, 264, 986, 636, 555, 370, 7...

scala> val numbersRDD = sc.parallelize(numbers)
numbersRDD: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at <console>:26

scala> numbersRDD.take(25).foreach(println)
242
48
871
629
101
29
513
973
363
200
445
153
174
569
211
234
938
502
784
514
783
759
489
579
808

scala> 
```

11:23 AM
12/18/2023



```
KRISUNICORN@bigdata-ds650: ~/dsc650-infra/bellevue-bigdata/hadoop-hive-spark-hbase
scala> numbersRDD.take(100).foreach(println)
242
48
871
629
101
29
513
973
363
200
445
153
174
569
211
234
938
502
784
514
783
759
489
579
808
461
430
245
792
463
426
699
732
225
39
746
724
145
822
415
987
831
924
516
445
520
184
599
936
591
712
879
939
273
660
724
404
251
306
458
862
```

3. **Screenshots of Custom Transformation and their significance:**

Due to space constraints while taking the screenshot of the output, only 10 rows are displayed for each transformation. 7 transformations were used, and their significance is listed below:

3.1. toUpperCase:

val transformedSentences = sentencesRDD.map(sentence => sentence.**toUpperCase**)

Significance:

This command will convert all the characters to Upper case and return the Uppercase converted values of the RDD. Similarly, **toLowerCase** can also be used.

3.2. startsWith with Map method:

val transformedSentences2 = sentencesRDD.**map**(sentence => sentence.startsWith("appl"))

Significance:

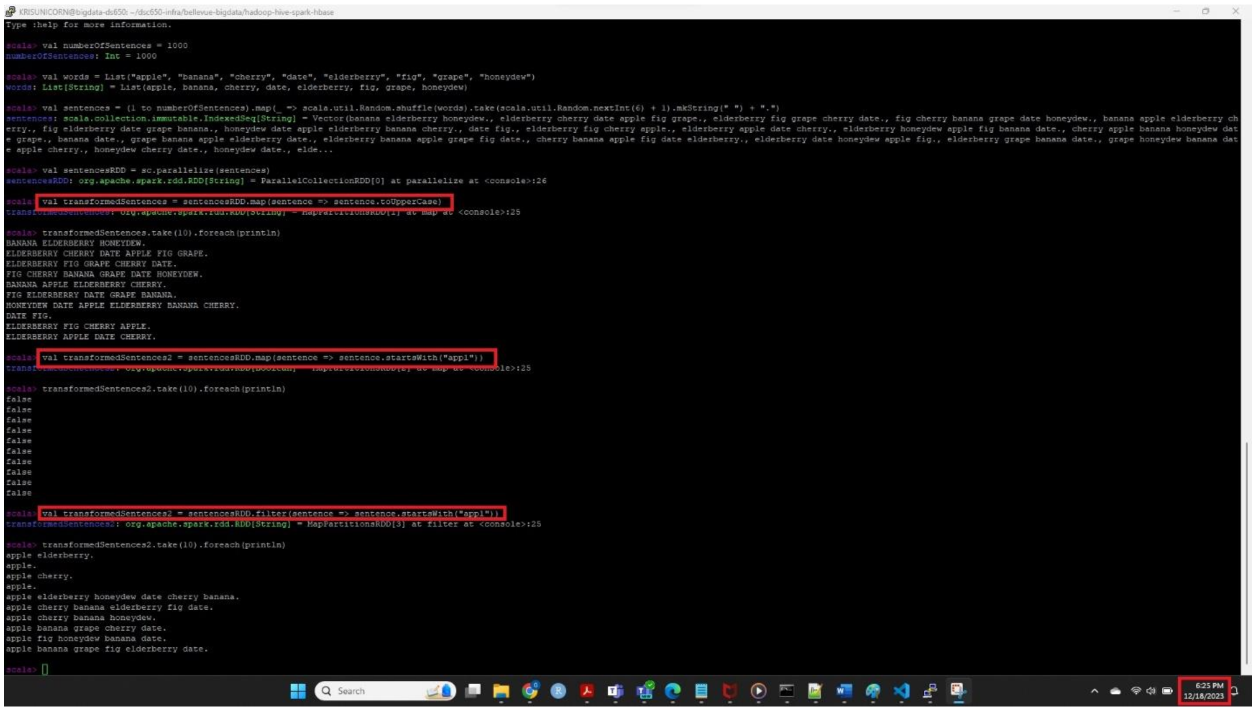
This command returns a Boolean value if the Array starts with the specified value. In the example, all arrays that start with “appl” returns True and everything else returns False.

3.3. startsWith used with filter method:

val transformedSentences2 = sentencesRDD.**filter**(sentence => sentence.**startsWith**("appl"))

Significance:

This is similar to the command above; the difference is it applies the filters and returns the array elements that start with the specified value.



```
KRISUNICORN@bigdata-ds650: ~/dsc650-infra/bellevue-bigdata/hadoop-hive-spark-hbase
Type :help for more information.

scala> val numberOfSentences = 1000
numberOfSentences: Int = 1000

scala> val words = List("apple", "banana", "cherry", "date", "elderberry", "fig", "grape", "honeydew")
words: List[String] = List(apple, banana, cherry, date, elderberry, fig, grape, honeydew)

scala> val sentences = (1 to numberOfSentences).map(_ => scala.util.Random.shuffle(words).take(scala.util.Random.nextInt(6) + 1).mkString(" ") + ".")
sentences: scala.collection.immutable.IndexedSeq[String] = Vector(banana elderberry honeydew., elderberry cherry date apple fig grape., elderberry fig grape cherry date., fig cherry banana grape date honeydew., banana apple elderberry ch
erry., fig elderberry date grape banana., honeydew date apple elderberry banana cherry., date fig., elderberry fig cherry apple., elderberry apple date cherry., elderberry honeydew apple fig banana date., cherry apple banana honeydew dat
e grape., banana date., grape banana apple elderberry date., elderberry banana apple grape fig date., cherry banana apple fig date elderberry., elderberry date honeydew apple fig., elderberry grape banana date., grape honeydew banana dat
e apple cherry., honeydew cherry date., honeydew date., elde...

scala> val sentencesRDD = sc.parallelize(sentences)
sentencesRDD: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[0] at parallelize at <console>:26

scala> val transformedSentences = sentencesRDD.map(sentence => sentence.toUpperCase)
transformedSentences: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[1] at map at <console>:25

scala> transformedSentences.take(10).foreach(println)
BANANA ELDERBERRY HONEYDEW.
ELDERBERRY CHERRY DATE APPLE FIG GRAPE.
ELDERBERRY FIG GRAPE CHERRY DATE.
FIG CHERRY BANANA GRAPE DATE HONEYDEW.
BANANA APPLE ELDERBERRY CHERRY.
FIG ELDERBERRY DATE GRAPE BANANA.
HONEYDEW DATE APPLE ELDERBERRY BANANA CHERRY.
DATE FIG.
ELDERBERRY FIG CHERRY APPLE.
ELDERBERRY APPLE DATE CHERRY.

scala> val transformedSentences2 = sentencesRDD.map(sentence => sentence.startsWith("appl"))
transformedSentences2: org.apache.spark.rdd.RDD[Boolean] = MapPartitionsRDD[2] at map at <console>:25

scala> transformedSentences2.take(10).foreach(println)
false
false
false
false
false
false
false
false
false
false

scala> val transformedSentences2 = sentencesRDD.filter(sentence => sentence.startsWith("appl"))
transformedSentences2: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[3] at filter at <console>:25

scala> transformedSentences2.take(10).foreach(println)
apple elderberry.
apple.
apple cherry.
apple.
apple elderberry honeydew date cherry banana.
apple cherry banana elderberry fig date.
apple cherry banana honeydew.
apple banana grape cherry date.
apple fig honeydew banana date.
apple banana grape fig elderberry date.

scala> 
```

3.4. Contains:

val transformedSentences3 = sentencesRDD.filter(sentence => sentence.contains("erry"))

Significance:

This command returns the list of elements that contains the string **"erry"**

3.5. substring:

val transformedSentences4 = sentencesRDD.map(sentence => sentence.substring(0,5))

Significance:

This transformation returns the substring of the element from position 0 to 4.

3.6. Replace:

val transformedSentences5 = sentencesRDD.map(sentence => sentence.replace(" ","-"))

Significance:

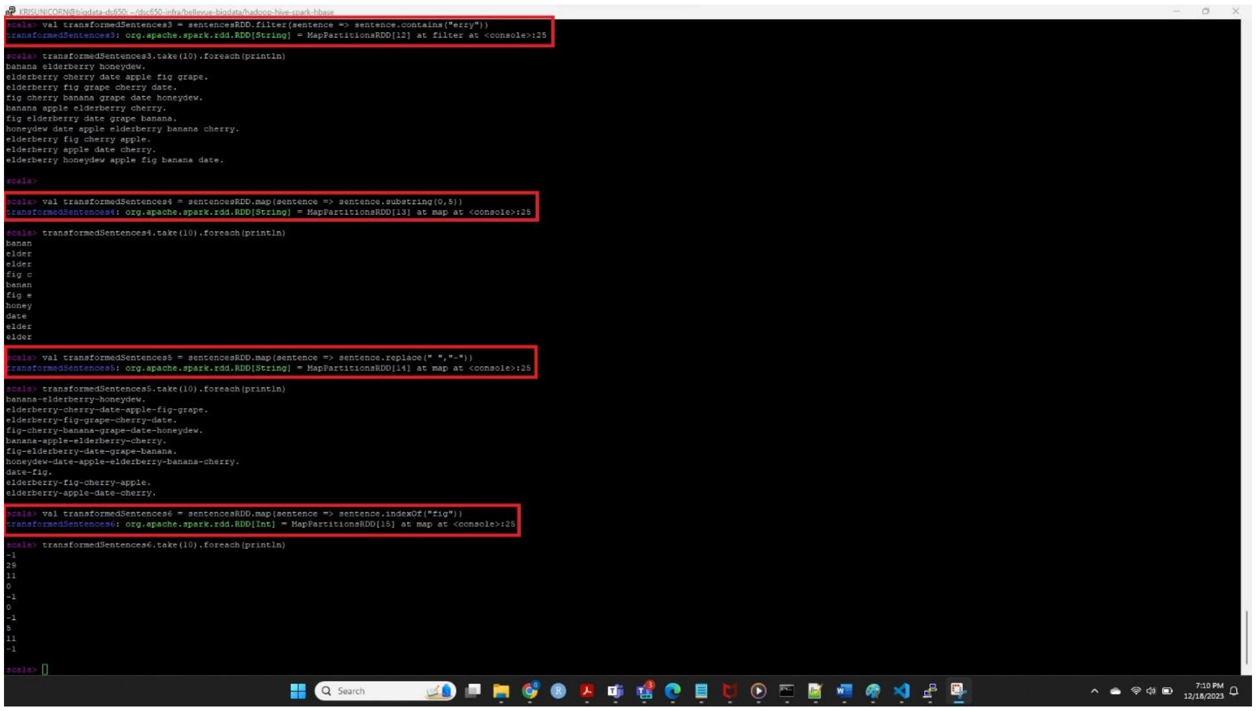
This transformation replaces the spaces with "-" in the Array of elements.

3.7. indexOf:

val transformedSentences6 = sentencesRDD.map(sentence => sentence.indexOf("fig"))

Significance:

This transformation returns the position of the string "fig" in the array. If the string does not exist, it returns -1.



```
KRISUNICORN@bigdata-ds650: ~/dsc650-infra/bellevue-bigdata/hadoop-hive-spark-hbase
scala> val transformedSentences3 = sentencesRDD.filter(sentence => sentence.contains("erry"))
transformedSentences3: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[12] at filter at <console>:25

scala> transformedSentences3.take(10).foreach(println)
banana elderberry honeydew.
elderberry cherry date apple fig grape.
elderberry fig grape cherry date.
fig cherry banana grape date honeydew.
banana apple elderberry cherry.
fig elderberry date grape banana.
honeydew date apple elderberry banana cherry.
elderberry fig cherry apple.
elderberry apple date cherry.
elderberry honeydew apple fig banana date.

scala>

scala> val transformedSentences4 = sentencesRDD.map(sentence => sentence.substring(0,5))
transformedSentences4: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[13] at map at <console>:25

scala> transformedSentences4.take(10).foreach(println)
banan
elder
elder
fig c
banan
fig e
honey
date
elder
elder

scala> val transformedSentences5 = sentencesRDD.map(sentence => sentence.replace(" ","-"))
transformedSentences5: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[14] at map at <console>:25

scala> transformedSentences5.take(10).foreach(println)
banana-elderberry-honeydew.
elderberry-cherry-date-apple-fig-grape.
elderberry-fig-grape-cherry-date.
fig-cherry-banana-grape-date-honeydew.
banana-apple-elderberry-cherry.
fig-elderberry-date-grape-banana.
honeydew-date-apple-elderberry-banana-cherry.
date-fig.
elderberry-fig-cherry-apple.
elderberry-apple-date-cherry.

scala> val transformedSentences6 = sentencesRDD.map(sentence => sentence.indexOf("fig"))
transformedSentences6: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[15] at map at <console>:25

scala> transformedSentences6.take(10).foreach(println)
-1
29
11
0
-1
0
-1
5
11
-1

scala> 
```