IMAAD IMRAN HAJWANE

202101132 / 21

ASSIGNEMNT 03

MAP REDUCER ASSIGNEMNT

LY – A1

QUESTION: 01

Q.1

Consider the following table snippet:

Author	Paper Title	Citations
Claudio Gutierrez	Semantics and Complexity of SPARQL	320
Claudio Gutierrez	Survey of graph database models	315
Claudio Gutierrez	Foundations of semantic web databases	232
Claudio Gutierrez	The expressive power of SPARQL	157
Claudio Gutierrez	Minimal deductive systems for RDF	137
...	...	...
Jorge Perez	Semantics and Complexity of SPARQL	320
Jorge Perez	Minimal deductive systems for RDF	137
Jorge Perez	The recovery of a schema mapping	66
...	...	...
Renzo Angles	Survey of graph database models	315
Renzo Angles	The expressive power of SPARQL	157
Renzo Angles	Current graph database models	20
...	...	...

The table is a large tab-separated values (TSV) file contains millions of records about authors, their papers, and the citations of their papers. Multiple authors may write a single paper (as seen above). Paper titles and author names can be assumed to be unique.

From this table, you wish to compute a new table with pairs of co-authors and the sum of the number of citations of those papers they have co-authored together. Based on the partial data input above, the result would look like the following (avoiding duplicates by ensuring that Author 1 is alphabetically lower than Author 2):

Author 1	Author 2	Citations
Claudio Gutierrez	Jorge Perez	457
Claudio Gutierrez	Renzo Angles	472
...	...	...

You then wish to sort the results in descending order by total citations.

Given this input and desired output, design a series of MapReduce jobs to perform the required processing. In particular, detail the sequence of map/reduce phases of your algorithm: what are the map keys, what are the map values, what are the reduce keys, what are the reduce values, what does the map function do, what does the reduce function do. Also indicate if there is a possibility to use a combiner at each step. You can use natural language, diagrams, examples AND/OR pseudo-code to describe the algorithm, as you prefer (so long as it is readable).

SOLUTION:

Code for TSV file Generation.

```java
import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class TsvFileGenerator {

    public static void main(String[] args) {

        // Define the file path for the TSV file

        String filePath = "author.tsv";

        // Define the sample data

        String[][] data = {

                { "Author", "Paper Title", "Citations" },

                { "Claudio Gutierrez, Jorge Perez", "Semantics and Complexity of SPARQL", "320" },

                { "Claudio Gutierrez", "Survey of graph database models", "315" },

                { "Claudio Gutierrez", "Foundations of semantic web databases", "232" },

                { "Claudio Gutierrez, Jorge Perez", "Minimal deductive systems for RDF", "315" },

                { "Claudio Gutierrez, Renzo Angles", "The expressive power of SPARQL", "157" },

                { "Jorge Perez", "The recovery of a schema mapping", "232" },

                { "Renzo Angles, Claudio Gutierrez", "Survey of graph database models", "315" },

                { "Renzo Angles", "Current graph database models", "157" }

        };

        // Write the data to a TSV file

        try (BufferedWriter writer = new BufferedWriter(new FileWriter(filePath))) {

            for (String[] row : data) {

                writer.write(String.join("\t", row)); // Join the row elements with tab character

                writer.newLine(); // Add a new line after each row

            }

            System.out.println("TSV file generated successfully at: " + filePath);

        } catch (IOException e) {

            System.err.println("Error writing to TSV file: " + e.getMessage());

        }

    }

}
```

Output of TSV file:

	A	B	C
1	Author	Paper Title	Citations
2	Claudio Gutierrez, Jorge Perez	Semantics and Complexity of SPARQL	320
3	Claudio Gutierrez	Survey of graph database models	315
4	Claudio Gutierrez	Foundations of semantic web databases	232
5	Claudio Gutierrez, Jorge Perez	Minimal deductive systems for RDF	315
6	Claudio Gutierrez, Renzo Angles	The expressive power of SPARQL	157
7	Jorge Perez	The recovery of a schema mapping	232
8	Renzo Angles, Claudio Gutierrez	Survey of graph database models	315
9	Renzo Angles	Current graph database models	157
10			

Code for MapReduce:

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.io.WritableComparator;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Partitioner;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import java.io.IOException;

public class CoAuthorCitationAnalysis {

    // Mapper Class
    public static class CoAuthorMapper extends Mapper<Object, Text,
Text, IntWritable> {

        private IntWritable citationCount = new IntWritable();
        private Text coAuthorPair = new Text();

        // Map function processes each line of the input file
        public void map(Object key, Text value, Context context) throws
IOException, InterruptedException {

            String line = value.toString();
            String[] parts = line.split("\t"); // Split by tab
delimiter
```

```
            // Check for header or malformed lines
            if (parts.length < 3 ||
parts[2].trim().equals("Citations")) {
                return; // Skip the line if it's a header or malformed
            }

            try {
                // Parse authors and citation count
                String[] authors = parts[0].split(", "); // Split
authors by ", " delimiter
                int citations = Integer.parseInt(parts[2].trim()); //
Parse citations
                citationCount.set(citations);

                // Emit all pairs of co-authors
                for (int i = 0; i < authors.length; i++) {
                    for (int j = i + 1; j < authors.length; j++) {
                        String author1 = authors[i].trim();
                        String author2 = authors[j].trim();

                        // Ensure pairs are ordered alphabetically
                        if (author1.compareTo(author2) < 0) {
                            coAuthorPair.set(author1 + "," + author2);
                        } else {
                            coAuthorPair.set(author2 + "," + author1);
                        }

                        // Write the co-author pair and citation count
to context
                        context.write(coAuthorPair, citationCount);
                    }
                }
            } catch (NumberFormatException e) {
                // Handle number format exceptions gracefully
                System.err.println("Skipping line due to format error:
" + line);
            }
        }
    }

    // Partitioner Class
    public static class CoAuthorPartitioner extends Partitioner<Text,
IntWritable> {

        @Override
```

```
        public int getPartition(Text key, IntWritable value, int
numReduceTasks) {
            return (key.hashCode() & Integer.MAX_VALUE) %
numReduceTasks;
        }
    }

    // Comparator Class for Secondary Sorting
    public static class DescendingCitationComparator extends
WritableComparator {

        protected DescendingCitationComparator() {
            super(IntWritable.class, true);
        }

        @SuppressWarnings("rawtypes")
        @Override
        public int compare(WritableComparable a, WritableComparable b)
{
            IntWritable int1 = (IntWritable) a;
            IntWritable int2 = (IntWritable) b;
            return -1 * int1.compareTo(int2); // Multiply by -1 for
descending order
        }
    }

    // Reducer Class
    public static class CoAuthorReducer extends Reducer<Text,
IntWritable, Text, IntWritable> {

        private IntWritable result = new IntWritable();

        public void reduce(Text key, Iterable<IntWritable> values,
Context context)
                throws IOException, InterruptedException {

            int sum = 0;
            for (IntWritable val : values) {
                sum += val.get();
            }

            result.set(sum);
            context.write(key, result);
        }
    }
```

```java
    // Main Method
    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Co-Author Citation Analysis");

        job.setJarByClass(CoAuthorCitationAnalysis.class);
        job.setMapperClass(CoAuthorMapper.class);
        job.setPartitionerClass(CoAuthorPartitioner.class);
        job.setReducerClass(CoAuthorReducer.class);
        job.setSortComparatorClass(DescendingCitationComparator.class);
        job.setCombinerClass(CoAuthorReducer.class); // Using Reducer
as Combiner

        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(IntWritable.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

Execution Steps:

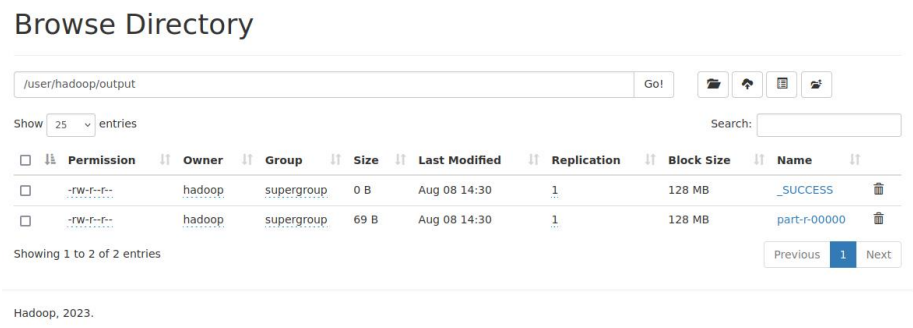
```
hadoop@imaad:~/Desktop/L3$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [imaad]
Starting resourcemanager
Starting nodemanagers
hadoop@imaad:~/Desktop/L3$ jps
4402 ResourceManager
4195 SecondaryNameNode
4885 Jps
4007 DataNode
3880 NameNode
4522 NodeManager
hadoop@imaad:~/Desktop/L3$   235  javac -classpath `hadoop classpath` -d . CoAuthorCitationAnalysis.java
235: command not found
hadoop@imaad:~/Desktop/L3$ javac -classpath `hadoop classpath` -d . CoAuthorCitationAnalysis.java
hadoop@imaad:~/Desktop/L3$ ls
 author.tsv                                  'CoAuthorCitationAnalysis$CoAuthorReducer.class'                CoAuthorCitationAnalysis.java
'CoAuthorCitationAnalysis$CoAuthorMapper.class'      'CoAuthorCitationAnalysis$DescendingCitationComparator.class'   execution.txt
'CoAuthorCitationAnalysis$CoAuthorPartitioner.class'   CoAuthorCitationAnalysis.class                                tsvgen.java
hadoop@imaad:~/Desktop/L3$ jar -cvf CoAuthorCitationAnalysis.jar -C . .
added manifest
adding: CoAuthorCitationAnalysis$CoAuthorPartitioner.class(in = 835) (out= 429)(deflated 48%)
adding: CoAuthorCitationAnalysis$DescendingCitationComparator.class(in = 661) (out= 363)(deflated 45%)
adding: tsvgen.java(in = 1620) (out= 658)(deflated 59%)
adding: CoAuthorCitationAnalysis$CoAuthorMapper.class(in = 2557) (out= 1195)(deflated 53%)
adding: CoAuthorCitationAnalysis.class(in = 1923) (out= 961)(deflated 50%)
adding: CoAuthorCitationAnalysis.java(in = 5017) (out= 1494)(deflated 70%)
adding: execution.txt(in = 86) (out= 66)(deflated 23%)
adding: CoAuthorCitationAnalysis$CoAuthorReducer.class(in = 1788) (out= 752)(deflated 57%)
adding: author.tsv(in = 513) (out= 248)(deflated 51%)
hadoop@imaad:~/Desktop/L3$ ls
 author.tsv                                  'CoAuthorCitationAnalysis$CoAuthorReducer.class'                CoAuthorCitationAnalysis.jar    tsvgen.java
'CoAuthorCitationAnalysis$CoAuthorMapper.class'      'CoAuthorCitationAnalysis$DescendingCitationComparator.class'   CoAuthorCitationAnalysis.java
'CoAuthorCitationAnalysis$CoAuthorPartitioner.class'   CoAuthorCitationAnalysis.class                                execution.txt
hadoop@imaad:~/Desktop/L3$ hadoop jar CoAuthorCitationAnalysis.jar CoAuthorCitationAnalysis /user/hadoop/input /user/hadoop/output
2024-08-08 14:27:43,773 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-08-08 14:27:43,960 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-08-08 14:27:43,960 INFO impl.MetricsSystemImpl: JobTracker metrics system started
Exception in thread "main" org.apache.hadoop.mapred.FileAlreadyExistsException: Output directory hdfs://localhost:9000/user/hadoop/output already exists
        at org.apache.hadoop.mapreduce.lib.output.FileOutputFormat.checkOutputSpecs(FileOutputFormat.java:164)
        at org.apache.hadoop.mapreduce.JobSubmitter.checkSpecs(JobSubmitter.java:277)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:143)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1678)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1675)
```

```
hadoop@imaad:~/Desktop/L3$ hadoop jar CoAuthorCitationAnalysis.jar CoAuthorCitationAnalysis /user/hadoop/input /user/hadoop/output
2024-08-08 14:27:43,773 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-08-08 14:27:43,960 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-08-08 14:27:43,960 INFO impl.MetricsSystemImpl: JobTracker metrics system started
Exception in thread "main" org.apache.hadoop.mapred.FileAlreadyExistsException: Output directory hdfs://localhost:9000/user/hadoop/output already exists
        at org.apache.hadoop.mapreduce.lib.output.FileOutputFormat.checkOutputSpecs(FileOutputFormat.java:164)
        at org.apache.hadoop.mapreduce.JobSubmitter.checkSpecs(JobSubmitter.java:277)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:143)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1678)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1675)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1899)
        at org.apache.hadoop.mapreduce.Job.submit(Job.java:1675)
        at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1696)
        at CoAuthorCitationAnalysis.main(CoAuthorCitationAnalysis.java:121)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:328)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:241)
hadoop@imaad:~/Desktop/L3$ hadoop jar CoAuthorCitationAnalysis.jar CoAuthorCitationAnalysis /user/hadoop/inputfile /user/hadoop/outputfile
2024-08-08 14:29:00,748 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-08-08 14:29:00,940 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-08-08 14:29:00,940 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2024-08-08 14:29:01,216 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2024-08-08 14:29:01,374 INFO mapreduce.JobSubmitter: Cleaning up the staging area file:/tmp/hadoop/mapred/staging/hadoop503172231/.staging/job_local503172231_0001
Exception in thread "main" org.apache.hadoop.mapreduce.lib.input.InvalidInputException: Input path does not exist: hdfs://localhost:9000/user/hadoop/inputfile
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(FileInputFormat.java:340)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.listStatus(FileInputFormat.java:279)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.getSplits(FileInputFormat.java:404)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeNewSplits(JobSubmitter.java:310)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeSplits(JobSubmitter.java:327)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:200)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1678)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1675)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1899)
        at org.apache.hadoop.mapreduce.Job.submit(Job.java:1675)
        at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1696)
        at CoAuthorCitationAnalysis.main(CoAuthorCitationAnalysis.java:121)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
```

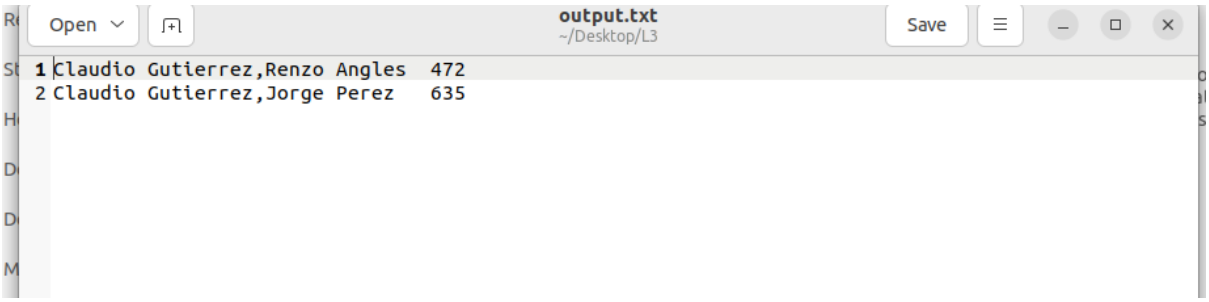
```
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:328)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:241)
Caused by: java.io.IOException: Input path does not exist: hdfs://localhost:9000/user/hadoop/inputfile
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(FileInputFormat.java:313)
        ... 19 more
hadoop@lnaad:~/Desktop/L3$ hadoop jar CoAuthorCitationAnalysis.jar CoAuthorCitationAnalysis /user/hadoop/input /user/hadoop/output
2024-08-08 14:29:59,072 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-08-08 14:29:59,210 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-08-08 14:29:59,210 INFO impl.MetricsSystemImpl: JobTracker metrics system started
Exception in thread "main" org.apache.hadoop.mapred.FileAlreadyExistsException: Output directory hdfs://localhost:9000/user/hadoop/output already exists
        at org.apache.hadoop.mapreduce.lib.output.FileOutputFormat.checkOutputSpecs(FileOutputFormat.java:164)
        at org.apache.hadoop.mapreduce.JobSubmitter.checkSpecs(JobSubmitter.java:277)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:143)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1678)
        at org.apache.hadoop.mapreduce.Job$11.run(Job.java:1675)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1899)
        at org.apache.hadoop.mapreduce.Job.submit(Job.java:1675)
        at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1696)
        at CoAuthorCitationAnalysis.main(CoAuthorCitationAnalysis.java:121)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:328)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:241)
hadoop@lnaad:~/Desktop/L3$ hadoop fs -rm -r /user/hadoop/output
Deleted /user/hadoop/output
hadoop@lnaad:~/Desktop/L3$ hadoop jar CoAuthorCitationAnalysis.jar CoAuthorCitationAnalysis /user/hadoop/input /user/hadoop/output
2024-08-08 14:30:38,597 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-08-08 14:30:38,761 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-08-08 14:30:38,766 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2024-08-08 14:30:39,003 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2024-08-08 14:30:39,337 INFO input.FileInputFormat: Total input files to process : 1
2024-08-08 14:30:39,424 INFO mapreduce.JobSubmitter: number of splits:1
2024-08-08 14:30:39,670 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local984520015_0001
2024-08-08 14:30:39,671 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-08-08 14:30:39,933 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2024-08-08 14:30:39,934 INFO mapreduce.Job: Running job: job_local984520015_0001
2024-08-08 14:30:39,938 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2024-08-08 14:30:39,991 INFO output.PathOutputCommitterFactory: No output committer factory defined, defaulting to FileOutputCommitterFactory
2024-08-08 14:30:39,993 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
```

```
                Map output records=4
                Map output bytes=138
                Map output materialized bytes=79
                Input split bytes=115
                Combine input records=4
                Combine output records=2
                Reduce input groups=2
                Reduce shuffle bytes=79
                Reduce input records=2
                Reduce output records=2
                Spilled Records=4
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=53
                Total committed heap usage (bytes)=296988032
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=513
        File Output Format Counters
                Bytes Written=69
hadoop@lnaad:~/Desktop/L3$ hdfs dfs -get /user/hadoop/output/part-r-00000 /home/hadoop/Desktop/L3/output.txt
hadoop@lnaad:~/Desktop/L3$ cat output.txt
Claudio Gutierrez,Renzo Angles  472
Claudio Gutierrez,Jorge Perez   635
hadoop@lnaad:~/Desktop/L3$
```

HDFS Output:



Output File:



Algorithm:

• **Imports**: The code imports the necessary Hadoop libraries, such as `Configuration`, `Job`, `Mapper`, `Reducer`, and others, to set up the MapReduce job.

• **Mapper Class (`CoAuthorMapper`)**:

- **Input**: Reads a line from the input file.
- **Process**: Splits the line by tab delimiter, checks for malformed lines or headers, and then parses authors and citation counts.
- **Output**: Emits pairs of co-authors with the citation count. It ensures that author pairs are ordered alphabetically.

• **Partitioner Class (`CoAuthorPartitioner`)**:

- This class assigns the partition for each key-value pair. The partition is determined by hashing the key and taking the modulo with the number of reduce tasks.

• **Comparator Class (`DescendingCitationComparator`)**:

- Provides a custom comparator to sort the citation counts in descending order for secondary sorting during the shuffle and sort phase of MapReduce.

• **Reducer Class (`CoAuthorReducer`)**:

- **Input**: Receives co-author pairs with their citation counts.
- **Process**: Sums up the citation counts for each co-author pair.
- **Output**: Writes the co-author pair and the total citation count.

• **Main Method**:

- Sets up the configuration and job properties.
- Defines the Mapper, Partitioner, Reducer, Sort Comparator, and Combiner classes.
- Specifies the output key and value classes.
- Takes input and output paths from command-line arguments and waits for the job to complete.