Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 2**	
Title of Lab Assignment: To study functionality of Mapreduce and implement following programs using Mapreduce. **1. Write a program in Map Reduce for WordCount operation.** **2. Write a program in Map Reduce for Matrix Multiplication**		
DOP: 30-08-2024	**DOS: 02-09-2024**	
CO Mapped: **CO1**	**PO Mapped:** **PO1, PO2, PO3, PO4, PO5, PO7, PSO1, PSO2**	**Signature:**

Practical No. 2

Aim: To study functionality of Mapreduce and implement following programs using Mapreduce.

1. **Write a program in Map Reduce for WordCount operation.**
2. **Write a program in Map Reduce for Matrix Multiplication.**

Theory:

What is MapReduce?

MapReduce is a programming paradigm that enables massive scalability across hundreds or thousands of servers in a Hadoop cluster. As the processing component, MapReduce is the heart of Apache Hadoop. The term "MapReduce" refers to two separate and distinct tasks that Hadoop programs perform.

The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples.

As the sequence of the name MapReduce implies, the reduce job is always performed after the map job. MapReduce programming offers several benefits to help you gain valuable insights from your big data:

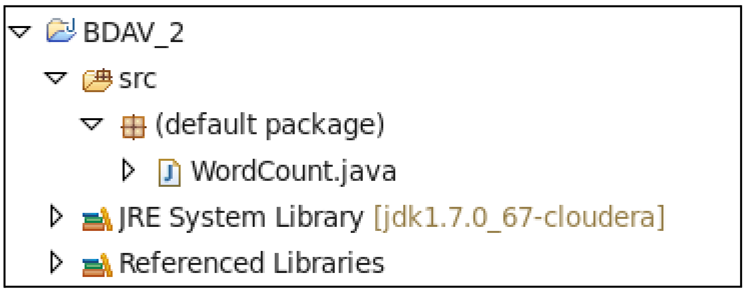
- Scalability: Businesses can process petabytes of data stored in the Hadoop Distributed File System (HDFS).
- Flexibility: Hadoop enables easier access to multiple sources of data and multiple types of data.
- Speed: With parallel processing and minimal data movement, Hadoop offers fast processing of massive amounts of data.
- Simple: Developers can write code in a choice of languages, including Java, C++ and Python.

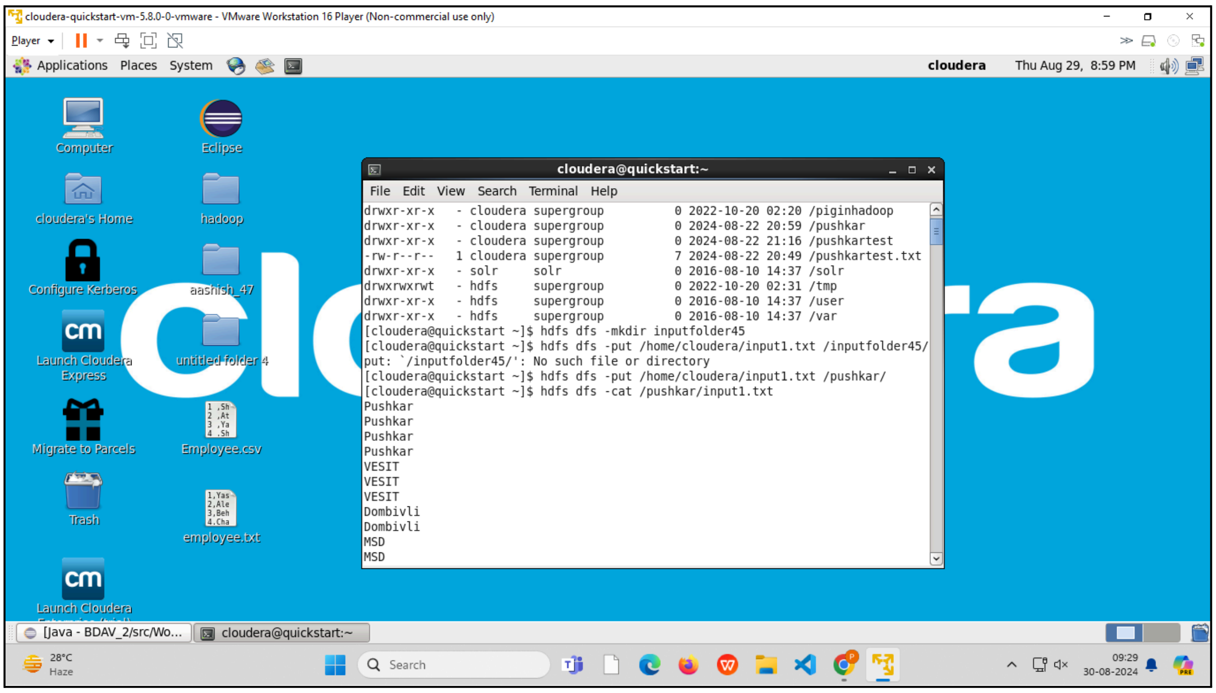
Program 1: MapReduce for Average Word Count

WordCount.java

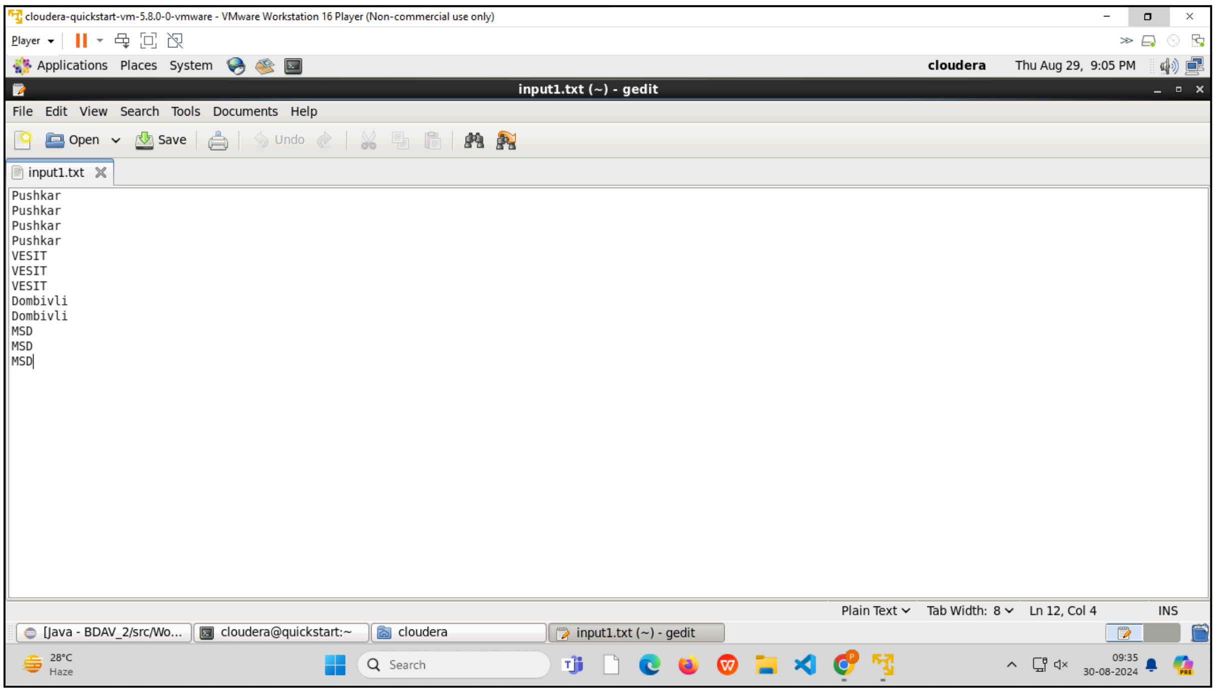
```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
public class WordCount {
        public static class TokenizerMapper extends
                        Mapper<Object, Text, Text, IntWritable> {
                private final static IntWritable one = new IntWritable(1);
                private Text word = new Text();
                public void map(Object key, Text value, Context context)
                                throws IOException, InterruptedException {
                        StringTokenizer itr = new StringTokenizer(value.toString());
                        while (itr.hasMoreTokens()) {
                                word.set(itr.nextToken());
                                context.write(word, one);
                        }
                }
        }

        public static class IntSumReducer 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);
                }
        }
        public static void main(String[] args) throws Exception {
                Configuration conf = new Configuration();
                Job job = Job.getInstance(conf, "word count");
                job.setJarByClass(WordCount.class);
                job.setMapperClass(TokenizerMapper.class);
                job.setCombinerClass(IntSumReducer.class);
                job.setReducerClass(IntSumReducer.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);
        }
}
```

Output:

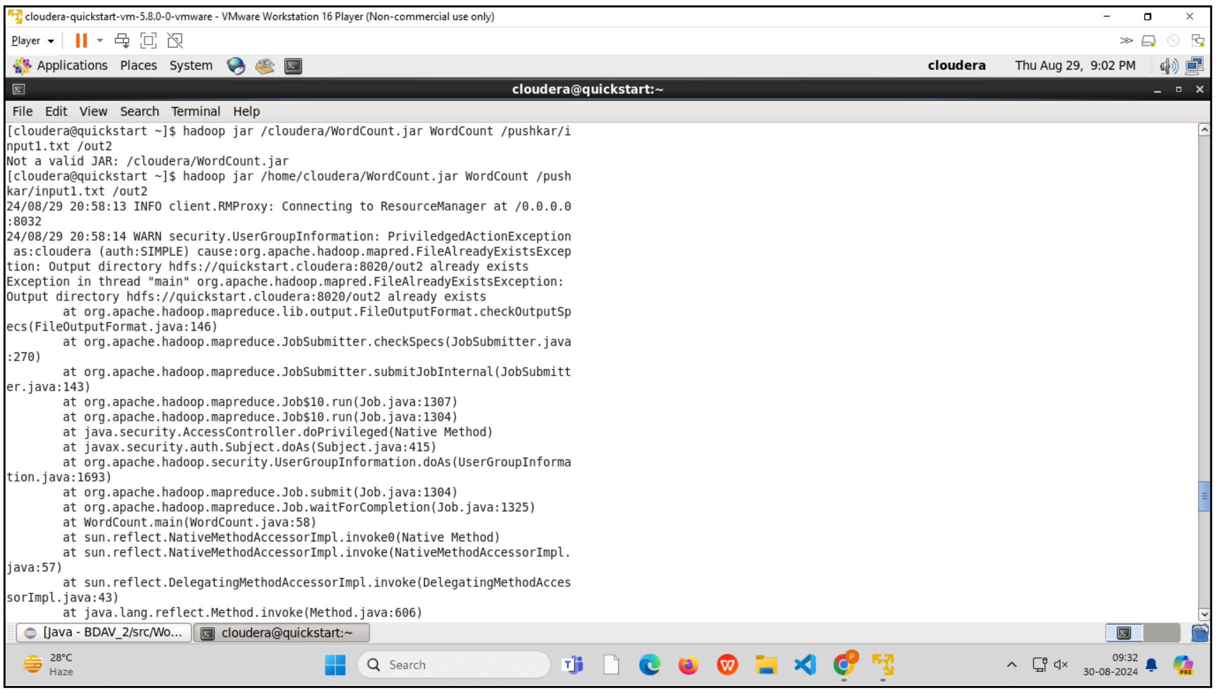



cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation 16 Player (Non-commercial use only)
Applications Places System
cloudera Thu Aug 29, 8:59 PM
cloudera@quickstart:~
File Edit View Search Terminal Help

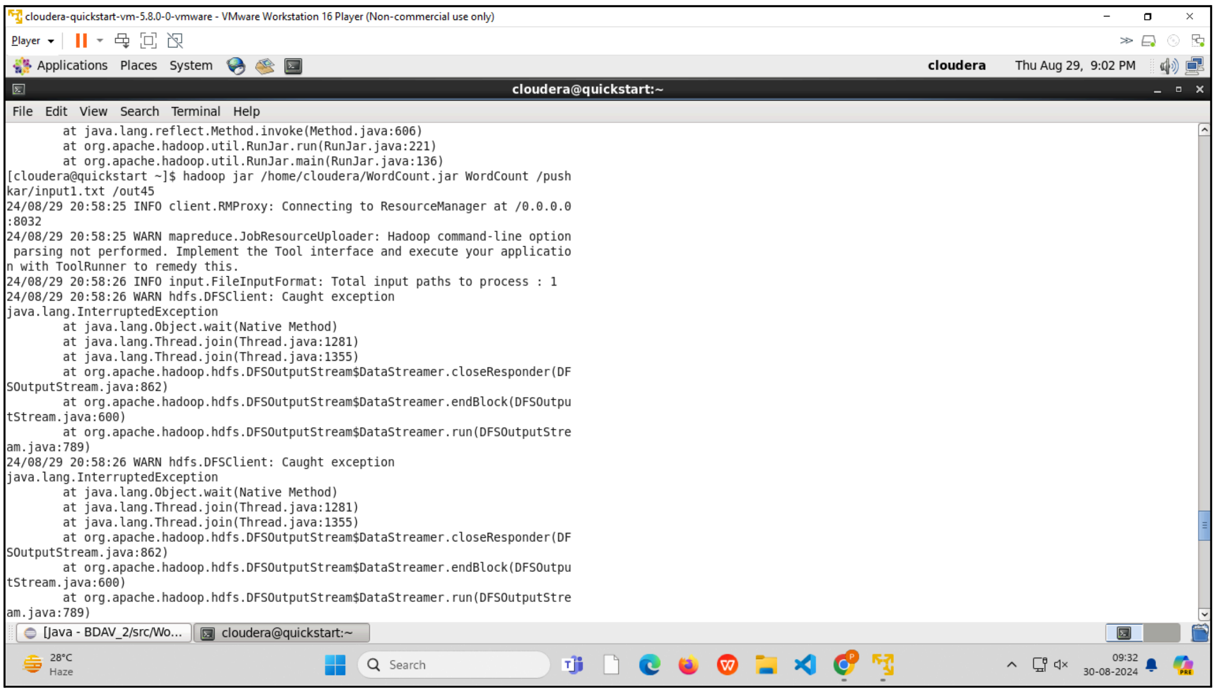
```
drwxr-xr-x   - cloudera supergroup          0 2022-10-20 02:20 /piginhadoop
drwxr-xr-x   - cloudera supergroup          0 2024-08-22 20:59 /pushkar
drwxr-xr-x   - cloudera supergroup          0 2024-08-22 21:16 /pushkartest
-rw-r--r--   1 cloudera supergroup          7 2024-08-22 20:49 /pushkartest.txt
drwxr-xr-x   - solr     solr                0 2016-08-10 14:37 /solr
drwxrwxrwt   - hdfs     supergroup          0 2022-10-20 02:31 /tmp
drwxr-xr-x   - hdfs     supergroup          0 2016-08-10 14:37 /user
drwxr-xr-x   - hdfs     supergroup          0 2016-08-10 14:37 /var
[cloudera@quickstart ~]$ hdfs dfs -mkdir inputfolder45
[cloudera@quickstart ~]$ hdfs dfs -put /home/cloudera/input1.txt /inputfolder45/
put: `/inputfolder45/': No such file or directory
[cloudera@quickstart ~]$ hdfs dfs -put /home/cloudera/input1.txt /pushkar/
[cloudera@quickstart ~]$ hdfs dfs -cat /pushkar/input1.txt
Pushkar
Pushkar
Pushkar
Pushkar
VESIT
VESIT
VESIT
Dombivli
Dombivli
MSD
MSD
```


cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation 16 Player (Non-commercial use only)
input1.txt (~) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
input1.txt
Plain Text Tab Width: 8 Ln 12, Col 4 INS
Thu Aug 29, 9:05 PM

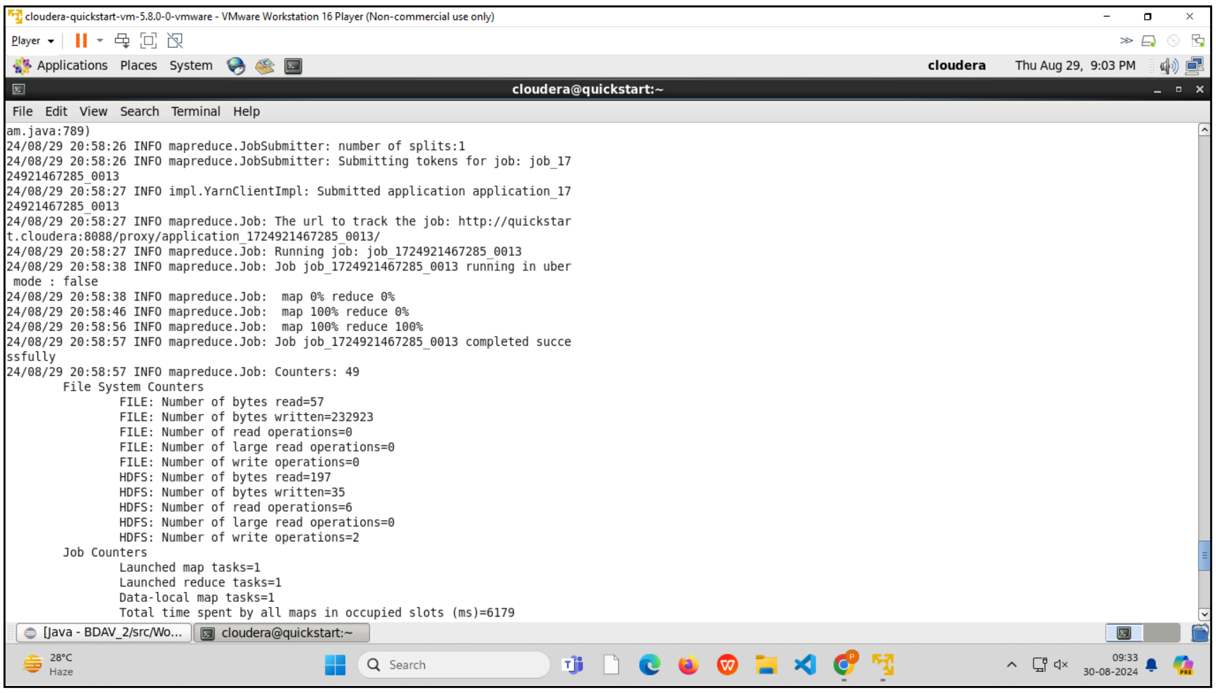
```
Pushkar
Pushkar
Pushkar
Pushkar
VESIT
VESIT
VESIT
Dombivli
Dombivli
MSD
MSD
MSD
```


cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation 16 Player (Non-commercial use only)
cloudera@quickstart:~
File Edit View Search Terminal Help
Thu Aug 29, 9:02 PM

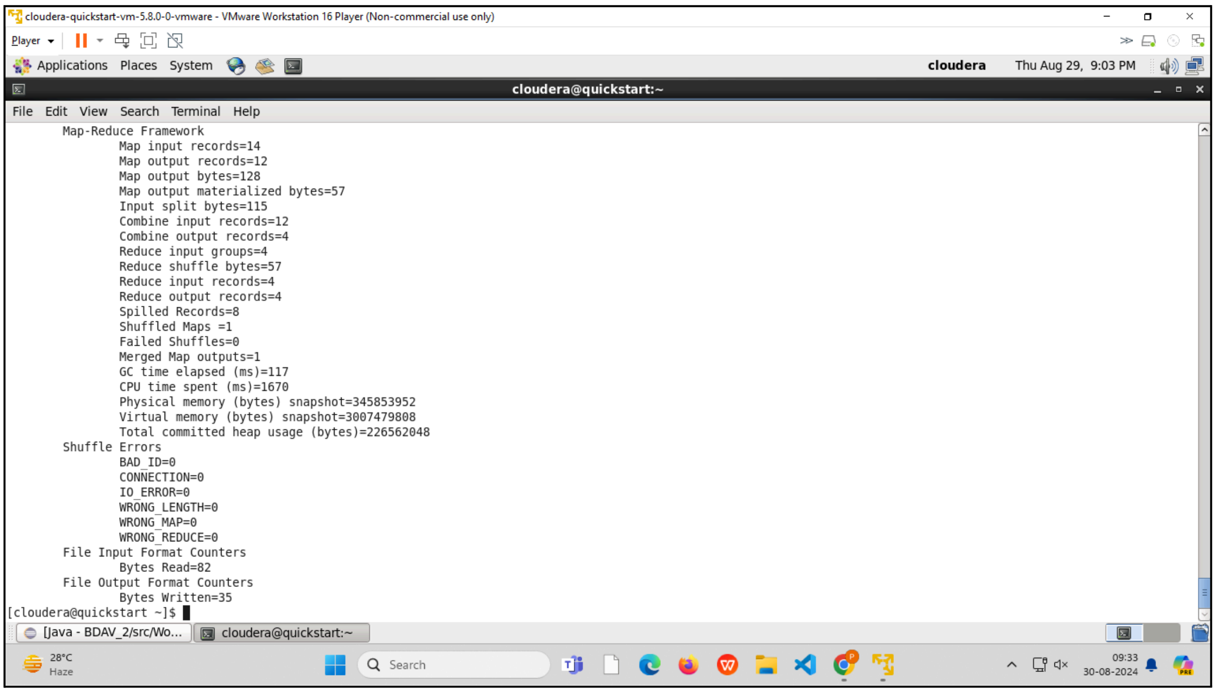
```
[cloudera@quickstart ~]$ hadoop jar /cloudera/WordCount.jar WordCount /pushkar/i
nput1.txt /out2
Not a valid JAR: /cloudera/WordCount.jar
[cloudera@quickstart ~]$ hadoop jar /home/cloudera/WordCount.jar WordCount /push
kar/input1.txt /out2
24/08/29 20:58:13 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0
:8032
24/08/29 20:58:14 WARN security.UserGroupInformation: PriviledgedActionException
 as:cloudera (auth:SIMPLE) cause:org.apache.hadoop.mapred.FileAlreadyExistsExcep
tion: Output directory hdfs://quickstart.cloudera:8020/out2 already exists
Exception in thread "main" org.apache.hadoop.mapred.FileAlreadyExistsException:
Output directory hdfs://quickstart.cloudera:8020/out2 already exists
        at org.apache.hadoop.mapreduce.lib.output.FileOutputFormat.checkOutputSp
ecs(FileOutputFormat.java:146)
        at org.apache.hadoop.mapreduce.JobSubmitter.checkSpecs(JobSubmitter.java
:270)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitt
er.java:143)
        at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1307)
        at org.apache.hadoop.mapreduce.Job$10.run(Job.java:1304)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:415)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInforma
tion.java:1693)
        at org.apache.hadoop.mapreduce.Job.submit(Job.java:1304)
        at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1325)
        at WordCount.main(WordCount.java:58)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.
java:57)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcces
sorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:606)
```


cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation 16 Player (Non-commercial use only)
Player
Applications Places System
cloudera Thu Aug 29, 9:02 PM
cloudera@quickstart:~
File Edit View Search Terminal Help
[Java - BDAV_2/src/Wo...
cloudera@quickstart:~
28°C Haze
Search
09:32 30-08-2024

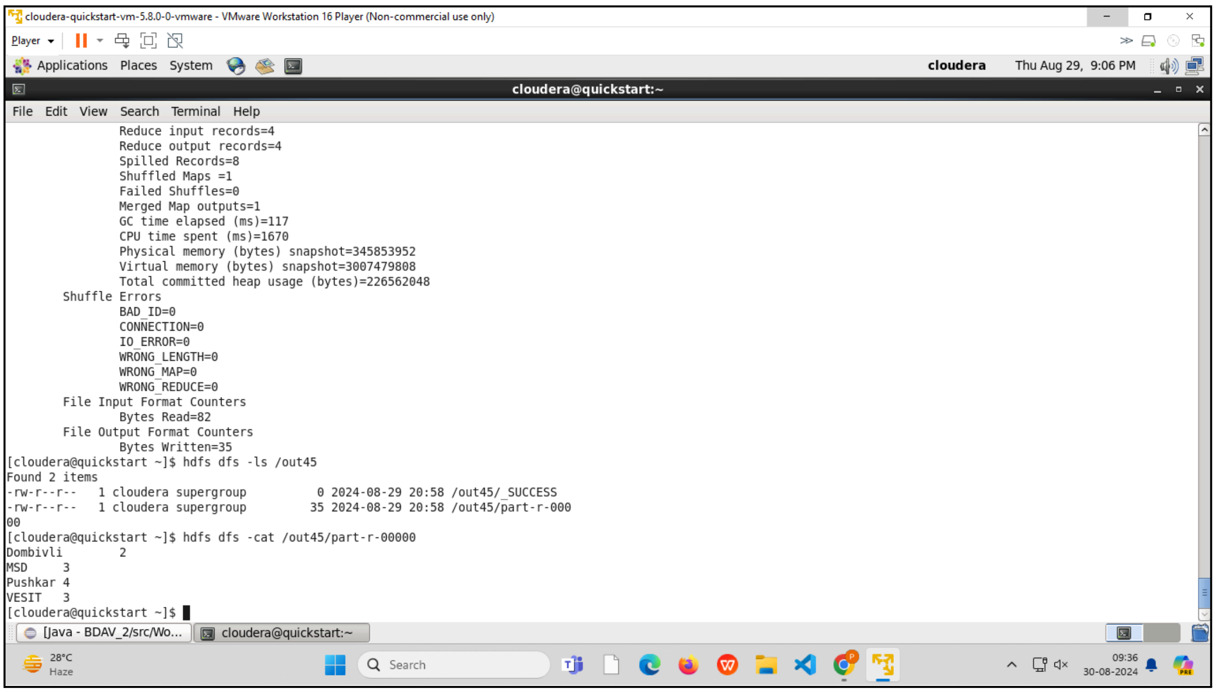
```
        at java.lang.reflect.Method.invoke(Method.java:606)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:221)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:136)
[cloudera@quickstart ~]$ hadoop jar /home/cloudera/WordCount.jar WordCount /push
kar/input1.txt /out45
24/08/29 20:58:25 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0
:8032
24/08/29 20:58:25 WARN mapreduce.JobResourceUploader: Hadoop command-line option
 parsing not performed. Implement the Tool interface and execute your applicatio
n with ToolRunner to remedy this.
24/08/29 20:58:26 INFO input.FileInputFormat: Total input paths to process : 1
24/08/29 20:58:26 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
        at java.lang.Object.wait(Native Method)
        at java.lang.Thread.join(Thread.java:1281)
        at java.lang.Thread.join(Thread.java:1355)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DF
SOutputStream.java:862)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutpu
tStream.java:600)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStre
am.java:789)
24/08/29 20:58:26 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
        at java.lang.Object.wait(Native Method)
        at java.lang.Thread.join(Thread.java:1281)
        at java.lang.Thread.join(Thread.java:1355)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DF
SOutputStream.java:862)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutpu
tStream.java:600)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStre
am.java:789)
```


cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation 16 Player (Non-commercial use only)
Player
Applications Places System
cloudera Thu Aug 29, 9:03 PM
cloudera@quickstart:~
File Edit View Search Terminal Help
[Java - BDAV_2/src/Wo...
cloudera@quickstart:~
28°C Haze
Search
09:33 30-08-2024

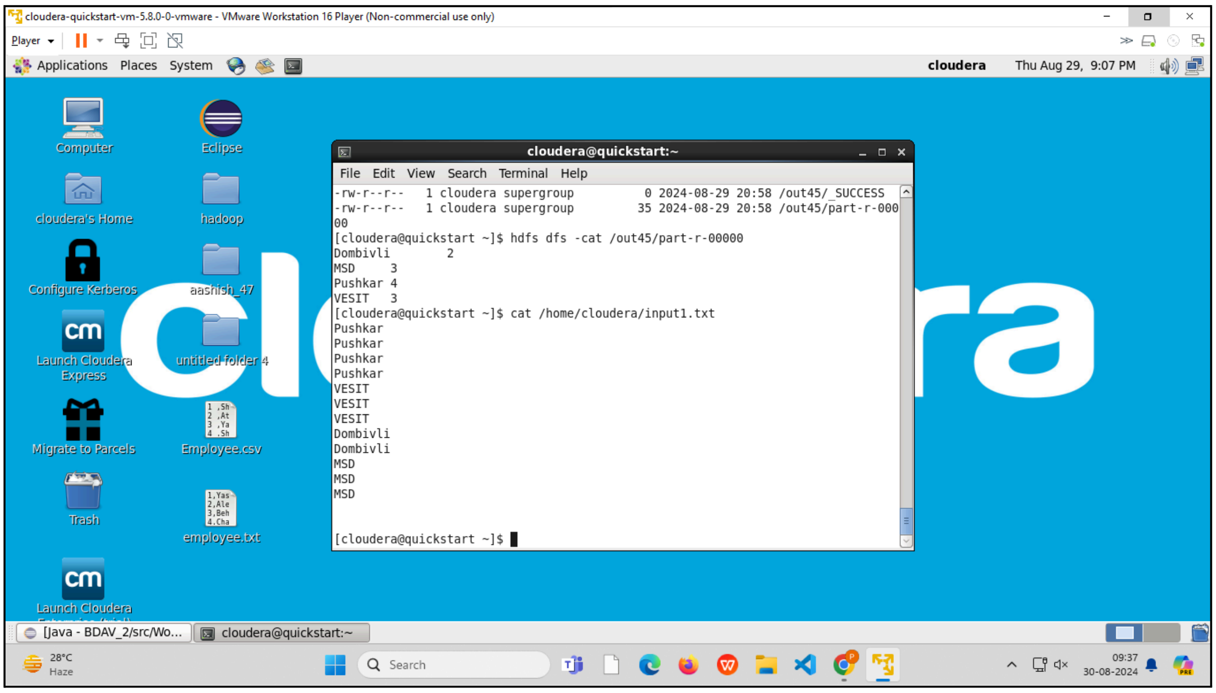
```
am.java:789)
24/08/29 20:58:26 INFO mapreduce.JobSubmitter: number of splits:1
24/08/29 20:58:26 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_17
24921467285_0013
24/08/29 20:58:27 INFO impl.YarnClientImpl: Submitted application application_17
24921467285_0013
24/08/29 20:58:27 INFO mapreduce.Job: The url to track the job: http://quickstar
t.cloudera:8088/proxy/application_1724921467285_0013/
24/08/29 20:58:27 INFO mapreduce.Job: Running job: job_1724921467285_0013
24/08/29 20:58:38 INFO mapreduce.Job: Job job_1724921467285_0013 running in uber
 mode : false
24/08/29 20:58:38 INFO mapreduce.Job:  map 0% reduce 0%
24/08/29 20:58:46 INFO mapreduce.Job:  map 100% reduce 0%
24/08/29 20:58:56 INFO mapreduce.Job:  map 100% reduce 100%
24/08/29 20:58:57 INFO mapreduce.Job: Job job_1724921467285_0013 completed succe
ssfully
24/08/29 20:58:57 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=57
                FILE: Number of bytes written=232923
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=197
                HDFS: Number of bytes written=35
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=6179
```



```
        Map-Reduce Framework
                Map input records=14
                Map output records=12
                Map output bytes=128
                Map output materialized bytes=57
                Input split bytes=115
                Combine input records=12
                Combine output records=4
                Reduce input groups=4
                Reduce shuffle bytes=57
                Reduce input records=4
                Reduce output records=4
                Spilled Records=8
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=117
                CPU time spent (ms)=1670
                Physical memory (bytes) snapshot=345853952
                Virtual memory (bytes) snapshot=3007479808
                Total committed heap usage (bytes)=226562048
        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=82
        File Output Format Counters
                Bytes Written=35
[cloudera@quickstart ~]$
```



```
                Reduce input records=4
                Reduce output records=4
                Spilled Records=8
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=117
                CPU time spent (ms)=1670
                Physical memory (bytes) snapshot=345853952
                Virtual memory (bytes) snapshot=3007479808
                Total committed heap usage (bytes)=226562048
        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=82
        File Output Format Counters
                Bytes Written=35
[cloudera@quickstart ~]$ hdfs dfs -ls /out45
Found 2 items
-rw-r--r--   1 cloudera supergroup          0 2024-08-29 20:58 /out45/_SUCCESS
-rw-r--r--   1 cloudera supergroup         35 2024-08-29 20:58 /out45/part-r-000
00
[cloudera@quickstart ~]$ hdfs dfs -cat /out45/part-r-00000
Dombivli        2
MSD     3
Pushkar 4
VESIT   3
[cloudera@quickstart ~]$
```



```
-rw-r--r--   1 cloudera supergroup          0 2024-08-29 20:58 /out45/_SUCCESS
-rw-r--r--   1 cloudera supergroup         35 2024-08-29 20:58 /out45/part-r-000
00
[cloudera@quickstart ~]$ hdfs dfs -cat /out45/part-r-00000
Dombivli        2
MSD     3
Pushkar 4
VESIT   3
[cloudera@quickstart ~]$ cat /home/cloudera/input1.txt
Pushkar
Pushkar
Pushkar
Pushkar
VESIT
VESIT
VESIT
Dombivli
Dombivli
MSD
MSD
MSD


[cloudera@quickstart ~]$ 
```

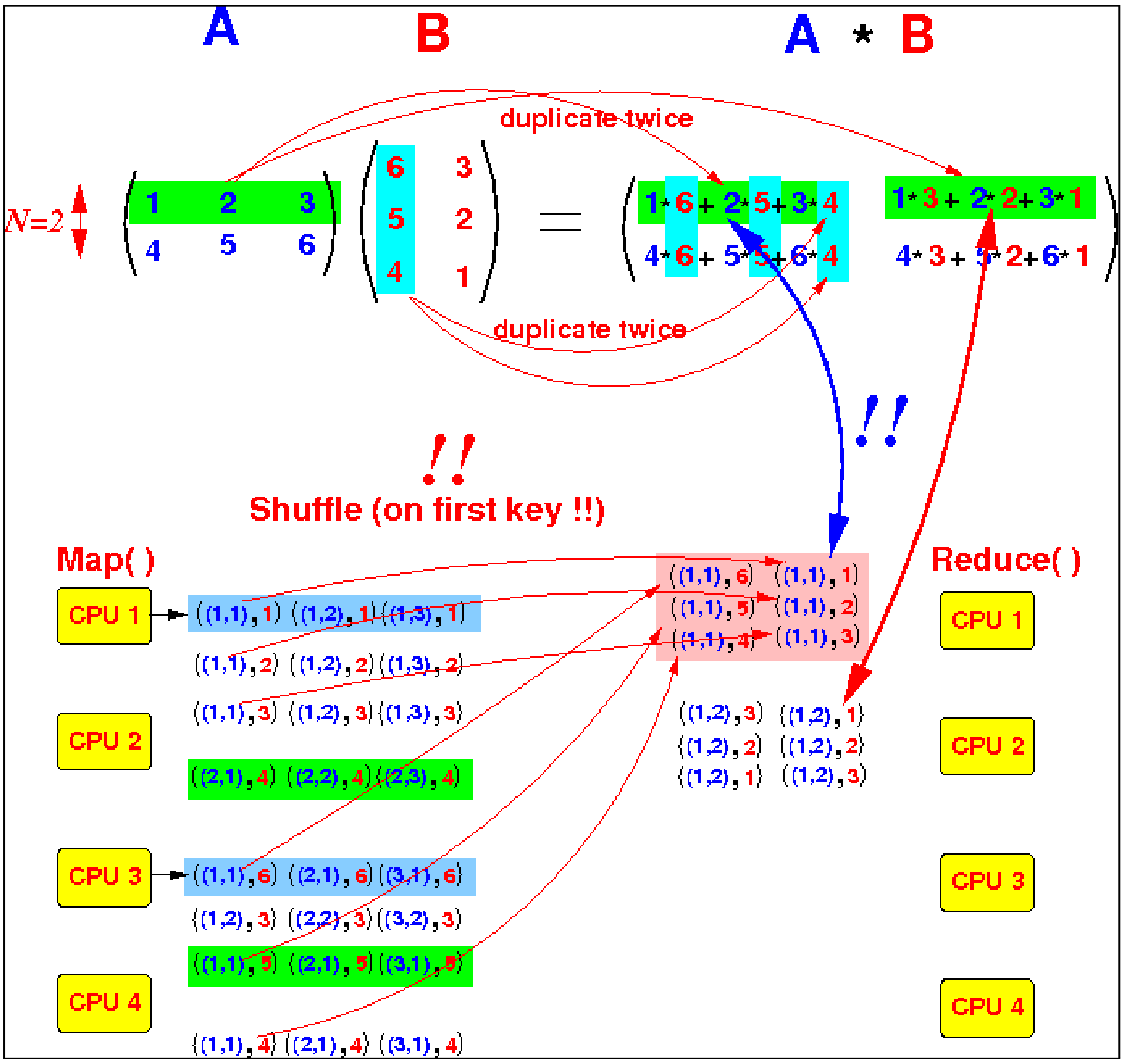
Program 2. To implement Matrix-Multiplication using MapReduce

Theory:

To perform matrix multiplication, 2 sets of MapReduce are required,

1. MapReduce to multiply:
    a. Mapper: Form pairs of Row from matrix A and Column from Matrix B
    b. Reducer: Multiply the results of the above Mapper
2. MapReduce to Calculate:
    a. Mapper: Identity, returns the same Key value pair
    b. Reducer: Calculates the sum of all the same keys and returns the key with the sum.

The diagram below shows an example of Matrix Multiplication using MapReduce.



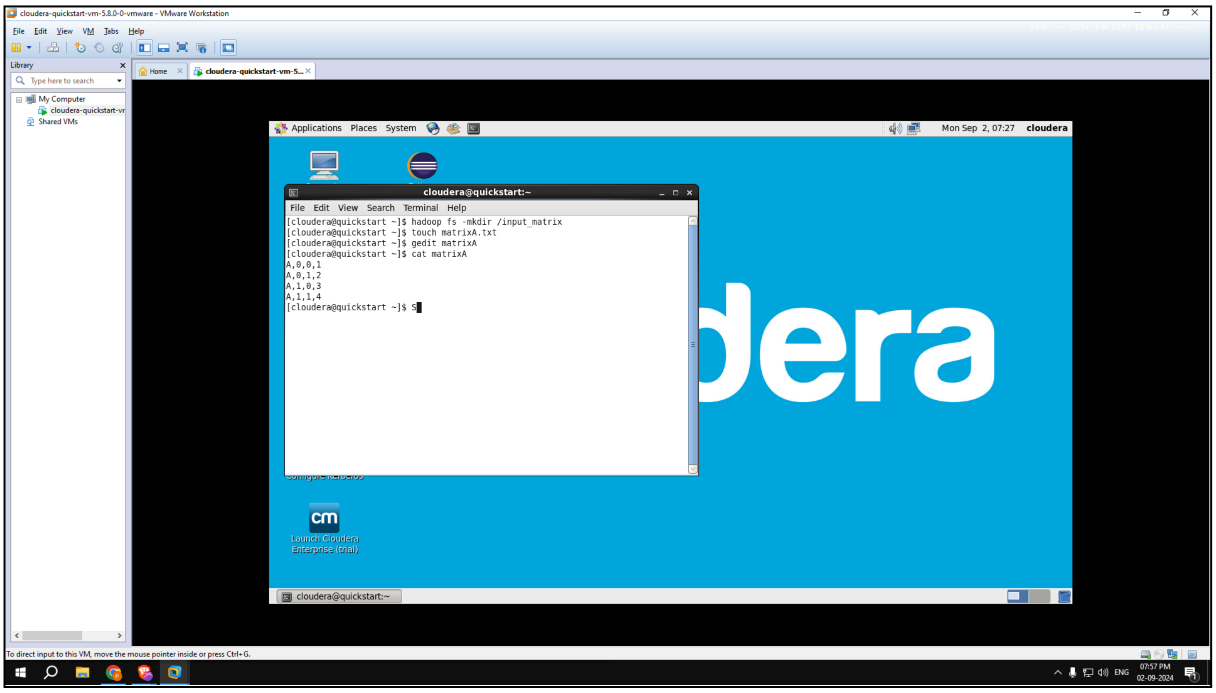
Code:

```
package expt4;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import java.util.ArrayList;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.Writable;
import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;
import org.apache.hadoop.util.ReflectionUtils;

class Element implements Writable {
        int tag;
        int index;
        double value;

        Element() {
                tag = 0;
                index = 0;
                value = 0.0;
        }

        Element(int tag, int index, double value) {
                this.tag = tag;
                this.index = index;
                this.value = value;
```

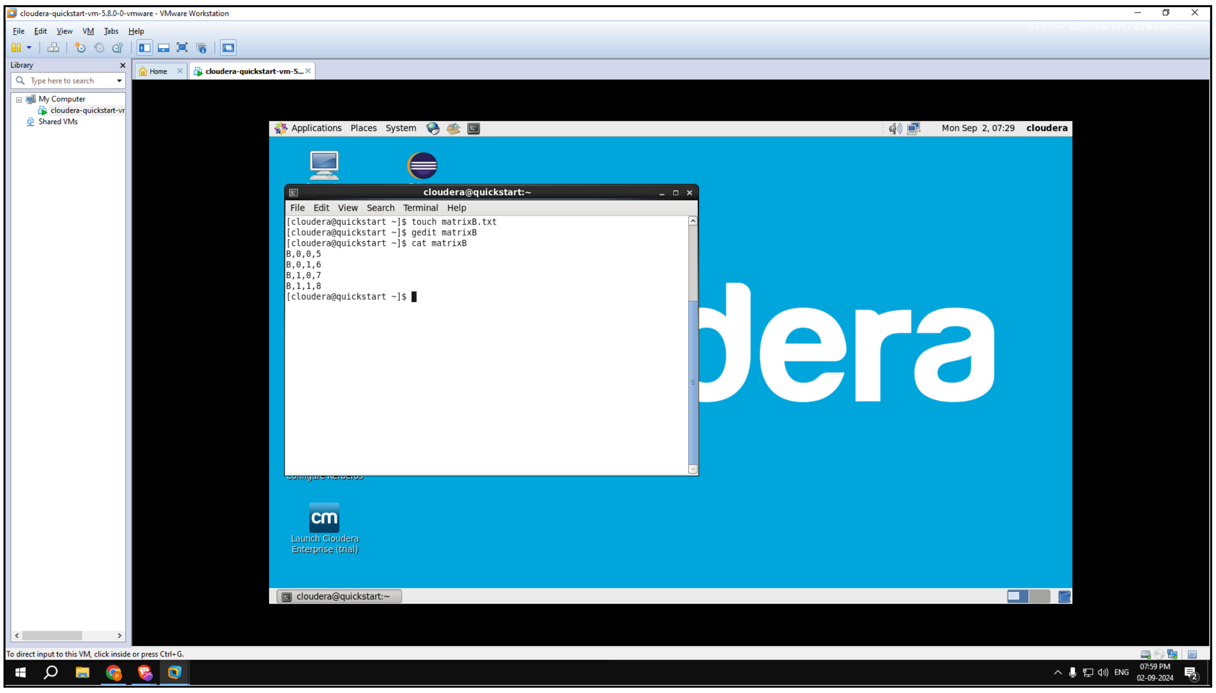
```
	}

	@Override
	public void readFields(DataInput input) throws IOException {
		tag = input.readInt();
		index = input.readInt();
		value = input.readDouble();
	}

	@Override
	public void write(DataOutput output) throws IOException {
		output.writeInt(tag);
		output.writeInt(index);
		output.writeDouble(value);
	}
}

class Pair implements WritableComparable<Pair> {

	int i;
	int j;

	Pair() {
		i = 0;
		j = 0;
	}

	Pair(int i, int j) {
		this.i = i;
		this.j = j;
	}

	@Override
	public void readFields(DataInput input) throws IOException {
		i = input.readInt();
		j = input.readInt();
	}

	@Override
	public void write(DataOutput output) throws IOException {
		output.writeInt(i);
		output.writeInt(j);
	}

	@Override
	public int compareTo(Pair compare) {

		if (i > compare.i) {
			return 1;
		} else if ( i < compare.i) {
			return -1;
		} else {
			if(j > compare.j) {
				return 1;
			} else if (j < compare.j) {
				return -1;
			}
		}
		return 0;
	}

	public String toString() {
		return i + " " + j + " ";
	}

}

public class MatrixMulitply {

	public static class MatriceMapperM extends
Mapper<Object,Text,IntWritable,Element> {
		@Override
		public void map(Object key, Text value, Context context)
				throws IOException, InterruptedException {
			String readLine = value.toString();
			String[] stringTokens = readLine.split(",");

			int index = Integer.parseInt(stringTokens[0]);
			double elementValue = Double.parseDouble(stringTokens[2]);

			Element e = new Element(0, index, elementValue);

			IntWritable keyValue = new
IntWritable(Integer.parseInt(stringTokens[1]));
			context.write(keyValue, e);
		}
	}

	public static class MatriceMapperN extends
Mapper<Object,Text,IntWritable,Element> {

		@Override
		public void map(Object key, Text value, Context context)
				throws IOException, InterruptedException {
			String readLine = value.toString();
			String[] stringTokens = readLine.split(",");

			int index = Integer.parseInt(stringTokens[1]);
			double elementValue = Double.parseDouble(stringTokens[2]);

			Element e = new Element(1,index, elementValue);

			IntWritable keyValue = new
IntWritable(Integer.parseInt(stringTokens[0]));
			context.write(keyValue, e);
		}
	}
	public static class ReducerMxN extends Reducer<IntWritable,Element, Pair,
DoubleWritable> {

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

		ArrayList<Element> M = new ArrayList<Element>();
		ArrayList<Element> N = new ArrayList<Element>();

		Configuration conf = context.getConfiguration();

		for(Element element : values) {

			Element tempElement = ReflectionUtils.newInstance(Element.class,
conf);
			ReflectionUtils.copy(conf, element, tempElement);

			if (tempElement.tag == 0) {
				M.add(tempElement);
			} else if(tempElement.tag == 1) {
				N.add(tempElement);
			}
		}

		for(int i=0;i<M.size();i++) {
			for(int j=0;j<N.size();j++) {

				Pair p = new Pair(M.get(i).index,N.get(j).index);
				double multiplyOutput = M.get(i).value * N.get(j).value;

				context.write(p, new DoubleWritable(multiplyOutput));
				}
			}
		}
	}

	public static class MapMxN extends Mapper<Object, Text, Pair, DoubleWritable> {
		@Override
		public void map(Object key, Text value, Context context)
				throws IOException, InterruptedException {

			String readLine = value.toString();
			String[] pairValue = readLine.split(" ");

			Pair p = new
Pair(Integer.parseInt(pairValue[0]),Integer.parseInt(pairValue[1]));
			DoubleWritable val = new
DoubleWritable(Double.parseDouble(pairValue[2]));

			context.write(p, val);
		}
	}

	public static class ReduceMxN extends Reducer<Pair, DoubleWritable, Pair,
DoubleWritable> {
		@Override
		public void reduce(Pair key, Iterable<DoubleWritable> values, Context
context)
		throws IOException, InterruptedException {

			double sum = 0.0;
			for(DoubleWritable value : values) {
				sum += value.get();
			}

			context.write(key, new DoubleWritable(sum));
		}
	}

	public static void main(String[] args) throws Exception {
		Job job = Job.getInstance();
		job.setJobName("MapIntermediate");
		job.setJarByClass(MatrixMulitply.class);

		MultipleInputs.addInputPath(job, new Path(args[0]), TextInputFormat.class,
MatriceMapperM.class);
		MultipleInputs.addInputPath(job, new Path(args[1]), TextInputFormat.class,
MatriceMapperN.class);
		job.setReducerClass(ReducerMxN.class);

		job.setMapOutputKeyClass(IntWritable.class);
		job.setMapOutputValueClass(Element.class);

		job.setOutputKeyClass(Pair.class);
		job.setOutputValueClass(DoubleWritable.class);

		job.setOutputFormatClass(TextOutputFormat.class);

		FileOutputFormat.setOutputPath(job, new Path(args[2]));

		job.waitForCompletion(true);

		Job job2 = Job.getInstance();
		job2.setJobName("MapFinalOutput");
		job2.setJarByClass(MatrixMulitply.class);

		job2.setMapperClass(MapMxN.class);
		job2.setReducerClass(ReduceMxN.class);

		job2.setMapOutputKeyClass(Pair.class);
		job2.setMapOutputValueClass(DoubleWritable.class);

		job2.setOutputKeyClass(Pair.class);
		job2.setOutputValueClass(DoubleWritable.class);
		job2.setInputFormatClass(TextInputFormat.class);
		job2.setOutputFormatClass(TextOutputFormat.class);
```

```
                FileInputFormat.setInputPaths(job2, new Path(args[2]));
                FileOutputFormat.setOutputPath(job2, new Path(args[3]));

                job2.waitForCompletion(true);
        }
}
```

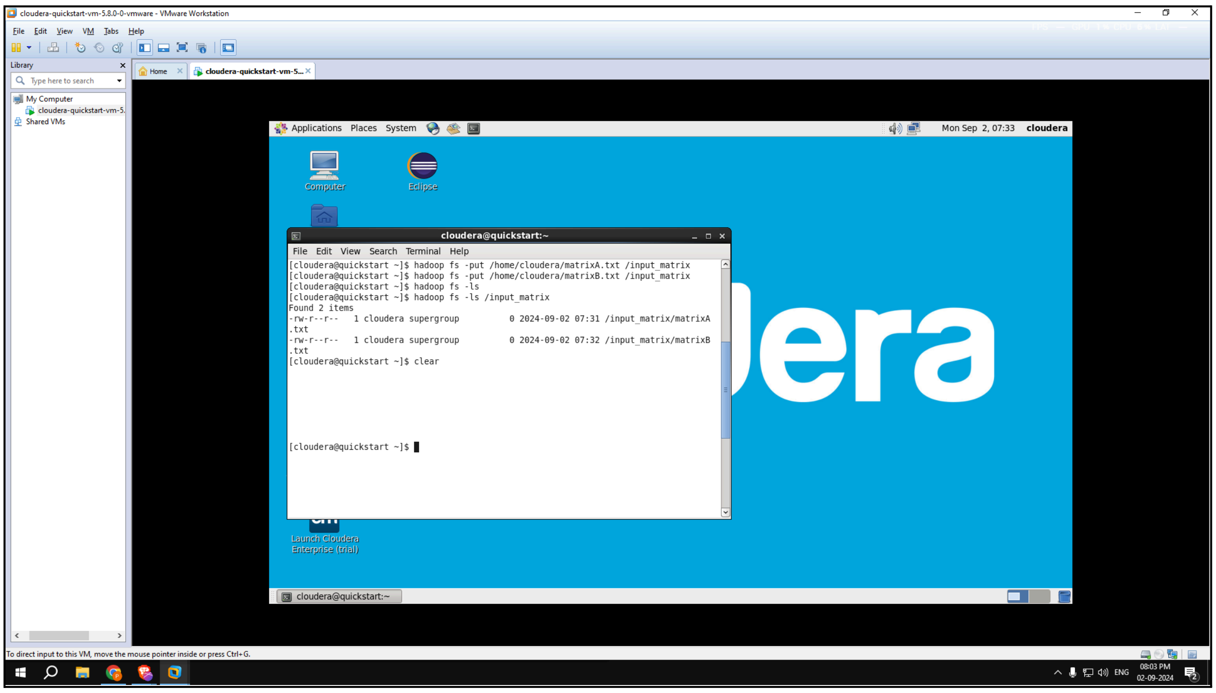
Output:


cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation
File Edit View VM Tabs Help
Library
My Computer
cloudera-quickstart-vm
Shared VMs
Home
cloudera-quickstart-vm-5...
Applications Places System
Mon Sep 2, 07:27 cloudera
cloudera@quickstart:~
File Edit View Search Terminal Help
Launch Cloudera Enterprise (trial)
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
07:57 PM
02-09-2024

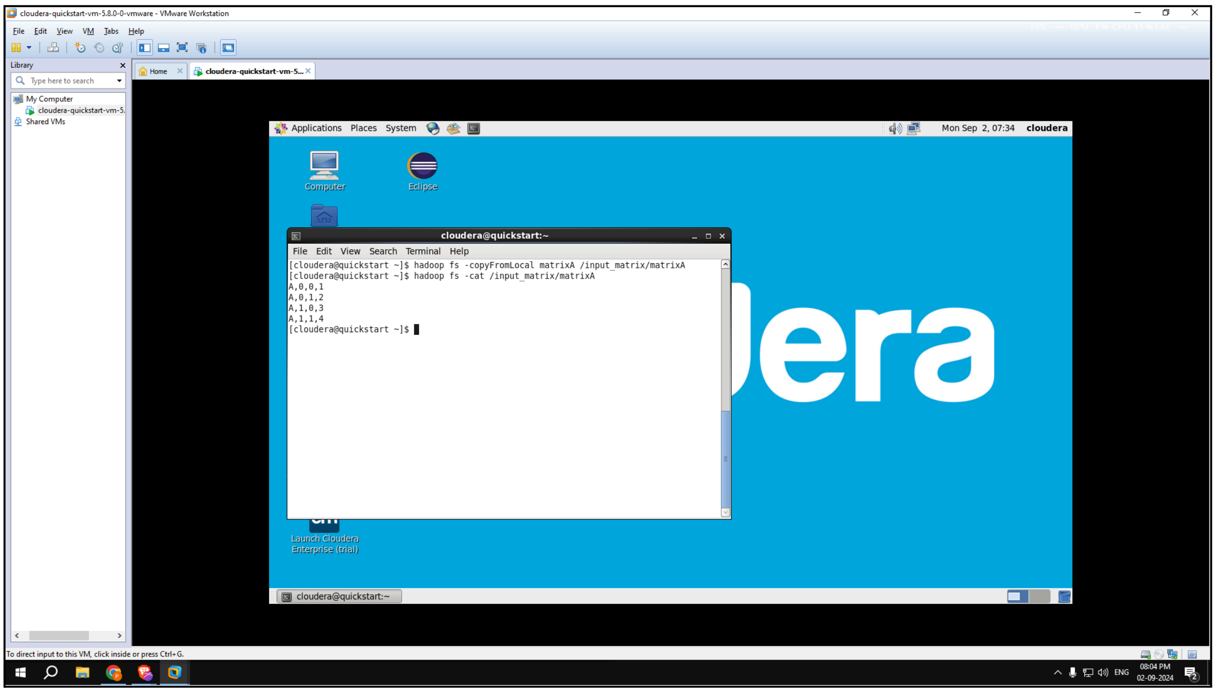
```
[cloudera@quickstart ~]$ hadoop fs -mkdir /input_matrix
[cloudera@quickstart ~]$ touch matrixA.txt
[cloudera@quickstart ~]$ gedit matrixA
[cloudera@quickstart ~]$ cat matrixA
A,0,0,1
A,0,1,2
A,1,0,3
A,1,1,4
[cloudera@quickstart ~]$ S
```



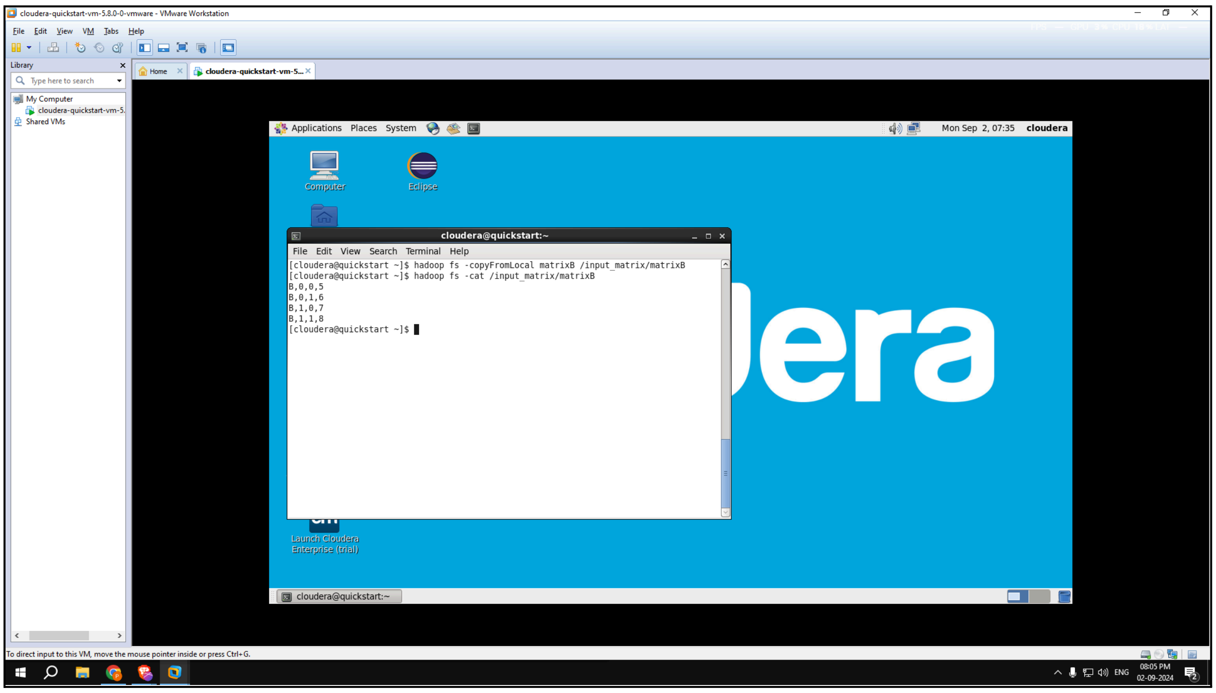
```
[cloudera@quickstart ~]$ touch matrixB.txt
[cloudera@quickstart ~]$ gedit matrixB
[cloudera@quickstart ~]$ cat matrixB
B,0,0,5
B,0,1,6
B,1,0,7
B,1,1,8
[cloudera@quickstart ~]$ 
```



```
[cloudera@quickstart ~]$ hadoop fs -put /home/cloudera/matrixA.txt /input_matrix
[cloudera@quickstart ~]$ hadoop fs -put /home/cloudera/matrixB.txt /input_matrix
[cloudera@quickstart ~]$ hadoop fs -ls
[cloudera@quickstart ~]$ hadoop fs -ls /input_matrix
Found 2 items
-rw-r--r--   1 cloudera supergroup          0 2024-09-02 07:31 /input_matrix/matrixA
.txt
-rw-r--r--   1 cloudera supergroup          0 2024-09-02 07:32 /input_matrix/matrixB
.txt
[cloudera@quickstart ~]$ clear




[cloudera@quickstart ~]$ 
```

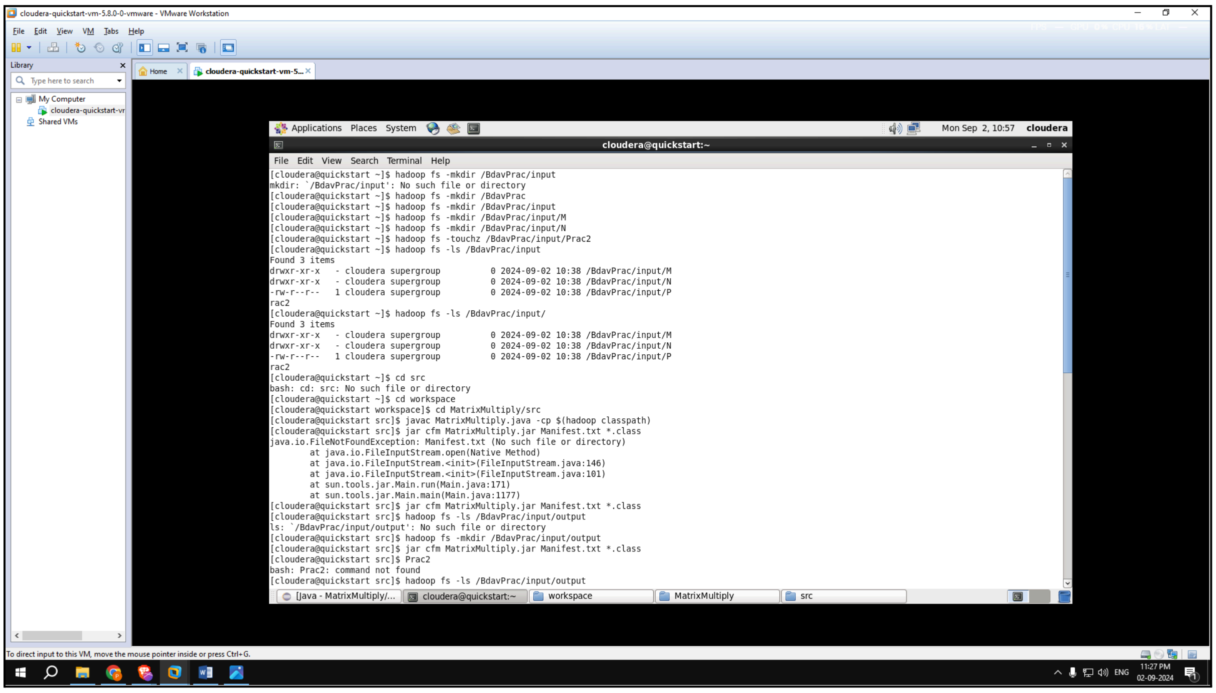

cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation
Applications Places System
Mon Sep 2, 07:34 cloudera
cloudera@quickstart:~
File Edit View Search Terminal Help
08:04 PM
02-09-2024

```
[cloudera@quickstart ~]$ hadoop fs -copyFromLocal matrixA /input_matrix/matrixA
[cloudera@quickstart ~]$ hadoop fs -cat /input_matrix/matrixA
A,0,0,1
A,0,1,2
A,1,0,3
A,1,1,4
[cloudera@quickstart ~]$
```


cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation
Applications Places System
Mon Sep 2, 07:35 cloudera
cloudera@quickstart:~
File Edit View Search Terminal Help
08:05 PM
02-09-2024

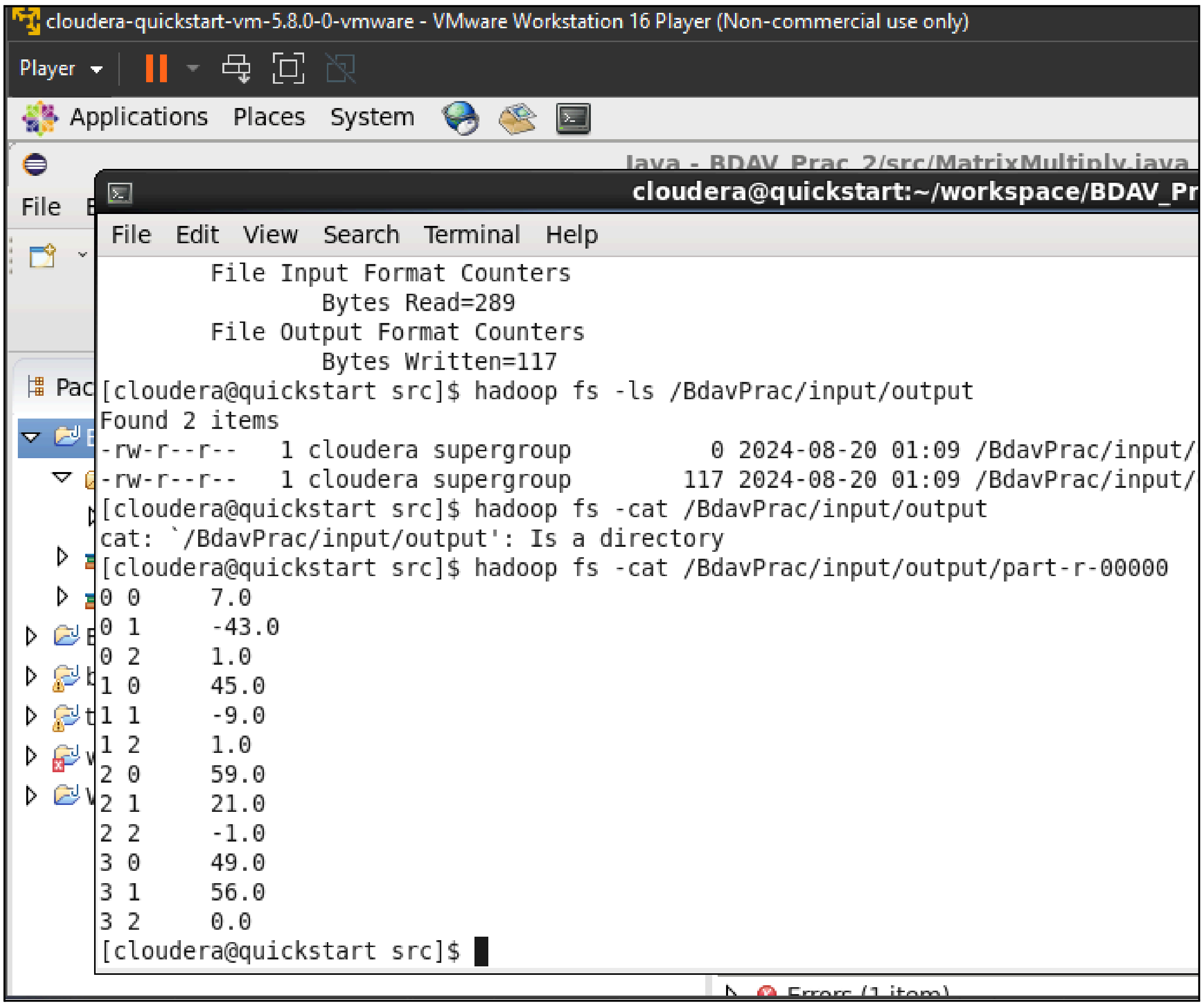
```
[cloudera@quickstart ~]$ hadoop fs -copyFromLocal matrixB /input_matrix/matrixB
[cloudera@quickstart ~]$ hadoop fs -cat /input_matrix/matrixB
B,0,0,5
B,0,1,6
B,1,0,7
B,1,1,8
[cloudera@quickstart ~]$
```

```
hadoop fs –mkdir /input_matrix
[cloudera@quickstart ~]$ hadoop fs -put /home/cloudera/matrixA.txt /input_matrix
[cloudera@quickstart ~]$ hadoop fs -put /home/cloudera/matrixB.txt /input_matrix
[cloudera@quickstart ~]$ hadoop fs -copyFromLocal matrixA /input_matrix/matrixA
[cloudera@quickstart ~]$ hadoop fs -cat /input_matrix/matrixA
[cloudera@quickstart ~]$ hadoop fs -copyFromLocal matrixB /input_matrix/matrixB
[cloudera@quickstart ~]$ hadoop fs -cat /input_matrix/matrixB
[cloudera@quickstart ~]$ hadoop jar MatrixMulti.jar OneStepMatrixMultiplication
/input_matrix/* /output
[cloudera@quickstart ~]$ hadoop fs -cat /output/*
```



```
[cloudera@quickstart ~]$ hadoop fs -mkdir /BdavPrac/input
mkdir: `/BdavPrac/input': No such file or directory
[cloudera@quickstart ~]$ hadoop fs -mkdir /BdavPrac
[cloudera@quickstart ~]$ hadoop fs -mkdir /BdavPrac/input
[cloudera@quickstart ~]$ hadoop fs -mkdir /BdavPrac/input/M
[cloudera@quickstart ~]$ hadoop fs -mkdir /BdavPrac/input/N
[cloudera@quickstart ~]$ hadoop fs -touchz /BdavPrac/input/Prac2
[cloudera@quickstart ~]$ hadoop fs -ls /BdavPrac/input
Found 3 items
drwxr-xr-x   - cloudera supergroup          0 2024-09-02 10:38 /BdavPrac/input/M
drwxr-xr-x   - cloudera supergroup          0 2024-09-02 10:38 /BdavPrac/input/N
-rw-r--r--   1 cloudera supergroup          0 2024-09-02 10:38 /BdavPrac/input/P
rac2
[cloudera@quickstart ~]$ hadoop fs -ls /BdavPrac/input/
Found 3 items
drwxr-xr-x   - cloudera supergroup          0 2024-09-02 10:38 /BdavPrac/input/M
drwxr-xr-x   - cloudera supergroup          0 2024-09-02 10:38 /BdavPrac/input/N
-rw-r--r--   1 cloudera supergroup          0 2024-09-02 10:38 /BdavPrac/input/P
rac2
[cloudera@quickstart ~]$ cd src
bash: cd: src: No such file or directory
[cloudera@quickstart ~]$ cd workspace
[cloudera@quickstart workspace]$ cd MatrixMultiply/src
[cloudera@quickstart src]$ javac MatrixMultiply.java -cp $(hadoop classpath)
[cloudera@quickstart src]$ jar cfm MatrixMultiply.jar Manifest.txt *.class
java.io.FileNotFoundException: Manifest.txt (No such file or directory)
        at java.io.FileInputStream.open(Native Method)
        at java.io.FileInputStream.<init>(FileInputStream.java:146)
        at java.io.FileInputStream.<init>(FileInputStream.java:101)
        at sun.tools.jar.Main.run(Main.java:171)
        at sun.tools.jar.Main.main(Main.java:1177)
[cloudera@quickstart src]$ jar cfm MatrixMultiply.jar Manifest.txt *.class
[cloudera@quickstart src]$ hadoop fs -ls /BdavPrac/input/output
ls: `/BdavPrac/input/output': No such file or directory
[cloudera@quickstart src]$ hadoop fs -mkdir /BdavPrac/input/output
[cloudera@quickstart src]$ jar cfm MatrixMultiply.jar Manifest.txt *.class
[cloudera@quickstart src]$ Prac2
bash: Prac2: command not found
[cloudera@quickstart src]$ hadoop fs -ls /BdavPrac/input/output
```

cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation
File Edit View VM Tabs Help
Library
My Computer
Shared VMs
Home
Applications Places System
Mon Sep 2, 10:57 cloudera
cloudera@quickstart:~
File Edit View Search Terminal Help
[Java - MatrixMultiply/...
workspace
MatrixMultiply
src
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
11:27 PM
02-09-2024


cloudera-quickstart-vm-5.8.0-0-vmware - VMware Workstation 16 Player (Non-commercial use only)
Player
Applications Places System
Java - BDAV_Prac_2/src/MatrixMultiply.java
cloudera@quickstart:~/workspace/BDAV_Pr
File Edit View Search Terminal Help
Errors (1 item)

```
        File Input Format Counters
                Bytes Read=289
        File Output Format Counters
                Bytes Written=117
[cloudera@quickstart src]$ hadoop fs -ls /BdavPrac/input/output
Found 2 items
-rw-r--r--   1 cloudera supergroup          0 2024-08-20 01:09 /BdavPrac/input/
-rw-r--r--   1 cloudera supergroup        117 2024-08-20 01:09 /BdavPrac/input/
[cloudera@quickstart src]$ hadoop fs -cat /BdavPrac/input/output
cat: `/BdavPrac/input/output': Is a directory
[cloudera@quickstart src]$ hadoop fs -cat /BdavPrac/input/output/part-r-00000
0 0     7.0
0 1     -43.0
0 2     1.0
1 0     45.0
1 1     -9.0
1 2     1.0
2 0     59.0
2 1     21.0
2 2     -1.0
3 0     49.0
3 1     56.0
3 2     0.0
[cloudera@quickstart src]$ 
```

Conclusion: Successfully implemented Word Count and Matrix Multiplication in hadoop using MapReduce.