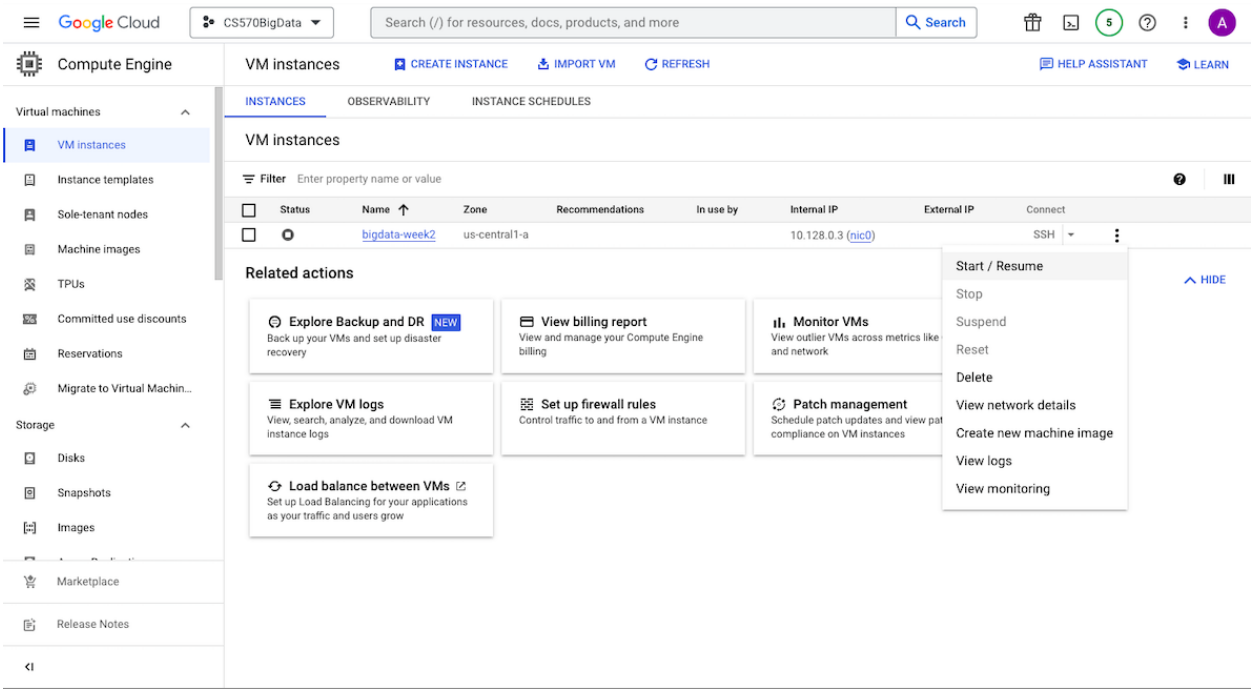
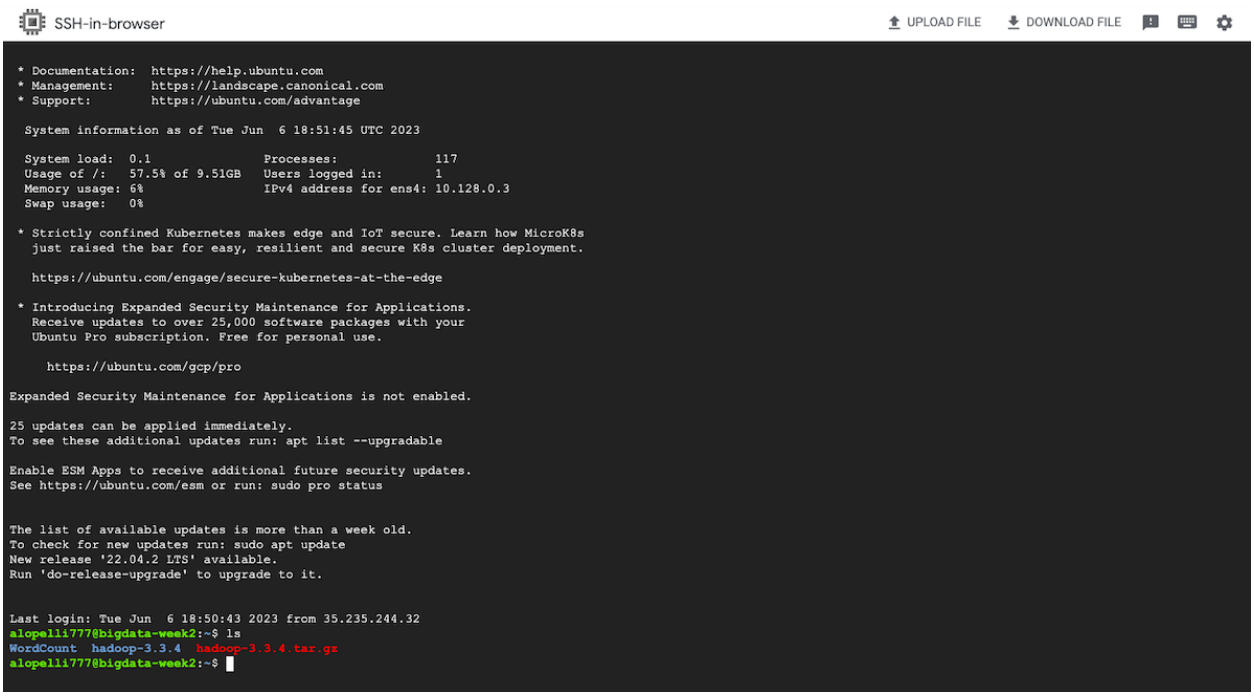
GCP Environment



Hadoop Environment



For Inverted index:

Code:

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Counter;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Mapper.Context;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileSplit;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;


public class InvertedIndex extends Configured implements Tool{

    public static class InvertedIndexMapper extends
            Mapper<LongWritable, Text, Text, IntWritable> {

        public static final String MalformedData = "MALFORMED";

        private Text outkey = new Text();
        private IntWritable outvalue = new IntWritable();

        public void map(LongWritable key, Text value, Context context)
                throws IOException, InterruptedException {
```

```
                FileSplit fileSplit = (FileSplit)context.getInputSplit();
                String filename = fileSplit.getPath().getName();
                //System.out.println("File name "+filename);
                //System.out.println("Directory and File name"+fileSplit.getPath().toString());

                String line = value.toString();
                StringTokenizer tokenizer = new StringTokenizer(line);
                while (tokenizer.hasMoreTokens()) {
                    String word = tokenizer.nextToken().trim();
                    if(word.equals("#")){
                        context.getCounter(MalformedData, word).increment(1);
                    }
                    else{
                        outkey.set(word);
                        outvalue = new IntWritable(
                                Integer.parseInt(filename.substring(4, filename.length()-4)));
                        System.out.println(outkey+"  "+outvalue);
                        context.write(outkey, outvalue);
                    }
                }
            }
        }

        public static class InvertedIndexReducer extends
        Reducer<Text, IntWritable, Text, Text> {
private Text outputkey = new Text();
private List<Integer> outputvalue = new ArrayList<Integer>();

public void reduce(Text key, Iterable<IntWritable> values,
            Context context) throws IOException, InterruptedException {
        outputkey = key;
        outputvalue = new ArrayList<Integer>();;
        for (IntWritable val : values) {
            if(!outputvalue.contains(val.get())){
                outputvalue.add(val.get());
            }
        }
        context.write(outputkey, new Text(outputvalue.toString()));
}
}

        public static void main(String[] args) throws Exception {
            int exitCode = ToolRunner.run(new Configuration(), new InvertedIndex(), args);
            System.exit(exitCode);
```

```
    }

    @Override
    public int run(String[] args) throws Exception {
        if (args.length != 2){
            System.out.printf("Usage: %s [generic options] <input> <output>\n",
                    getClass().getSimpleName());
            return -1;
        }
        Job job = new Job(getConf(), "InvertedIndex");
        job.setJarByClass(InvertedIndex.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setMapperClass(InvertedIndexMapper.class);
        job.setReducerClass(InvertedIndexReducer.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        boolean success = job.waitForCompletion(true);
        if(success){
            for(Counter counter: job.getCounters().getGroup(
                    InvertedIndexMapper.MalformedData)){

System.out.println(counter.getDisplayName()+"\t"+counter.getValue());
                    }
        }
        return success ? 0 : 1;
    }

}
```

Commnads:

Create the java file:

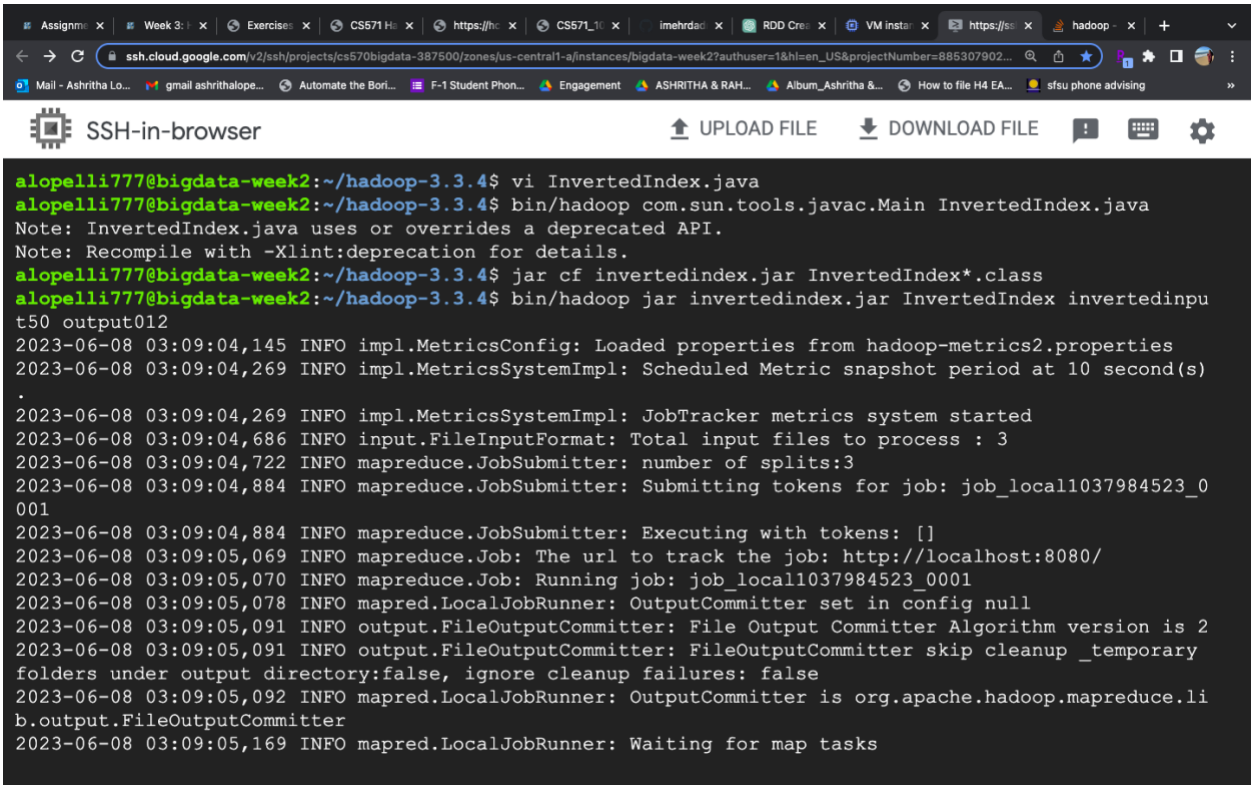
```
$ vi InvertedIndex.java
$ bin/hadoop com.sun.tools.javac.Main InvertedIndex.java
$ jar cf invertedindex.jar InvertedIndex *.class
$ hadoop dfs -copyFromLocal .  invertedindex50
```

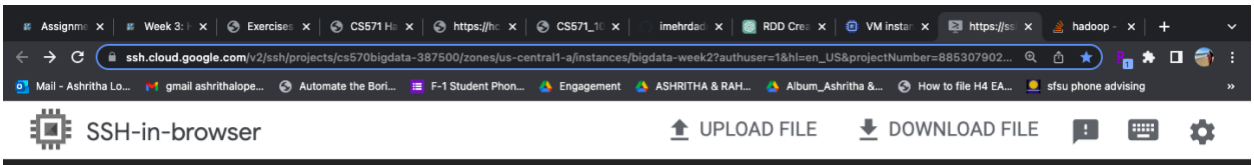
Copy input/file0, input/file1 and input/file2 of this project and place them inside the input folder of the Hadoop distrubution folder. While you are still there, run the following command to submit the job, get the input files from input folder, generate the inverted index and store its output in the output folder

```
$ bin/hadoop jar invertedindex.jar fullindex invertedindex50 output012
```

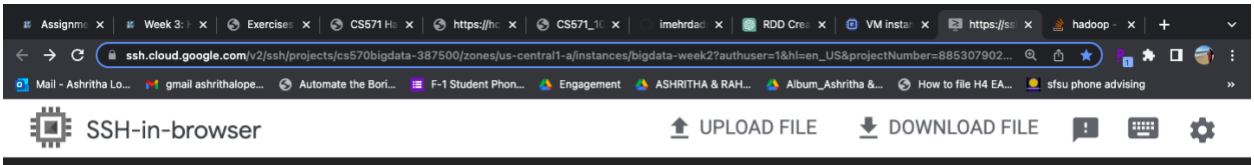
And finally to see the output, run the below command:

```
$ bin/hadoop dfs -cat output012/part-r-00000
```






```
alopelli777@bigdata-week2:~/hadoop-3.3.4$ bin/hadoop jar invertedindex.jar InvertedIndex invertedinpu
t50 output012
2023-06-08 03:09:04,145 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2023-06-08 03:09:04,269 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s)
.
2023-06-08 03:09:04,269 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2023-06-08 03:09:04,686 INFO input.FileInputFormat: Total input files to process : 3
2023-06-08 03:09:04,722 INFO mapreduce.JobSubmitter: number of splits:3
2023-06-08 03:09:04,884 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1037984523_0
001
2023-06-08 03:09:04,884 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-06-08 03:09:05,069 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2023-06-08 03:09:05,070 INFO mapreduce.Job: Running job: job_local1037984523_0001
2023-06-08 03:09:05,078 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2023-06-08 03:09:05,091 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-06-08 03:09:05,091 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary
folders under output directory:false, ignore cleanup failures: false
2023-06-08 03:09:05,092 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.li
b.output.FileOutputCommitter
2023-06-08 03:09:05,169 INFO mapred.LocalJobRunner: Waiting for map tasks
2023-06-08 03:09:05,170 INFO mapred.LocalJobRunner: Starting task: attempt_local1037984523_0001_m_000
000_0
2023-06-08 03:09:05,202 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-06-08 03:09:05,202 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary
folders under output directory:false, ignore cleanup failures: false
```




```
                Failed Shuffles=0
                Merged Map outputs=3
                GC time elapsed (ms)=19
                Total committed heap usage (bytes)=1518338048
        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=43
        File Output Format Counters
                Bytes Written=55
alopelli777@bigdata-week2:~/hadoop-3.3.4$ bin/hadoop dfs -cat output012/part-r-00000
WARNING: Use of this script to execute dfs is deprecated.
WARNING: Attempting to execute replacement "hdfs dfs" instead.

a       [2]
banana  [2]
is      [2, 1, 0]
it      [0, 2, 1]
what    [1, 0]
alopelli777@bigdata-week2:~/hadoop-3.3.4$
```

For Full Inverted Index

Code:

```
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.mapreduce.lib.input.FileSplit;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class FullIndexMR {
   public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {
      private Text word = new Text();
      private Text docInfo = new Text();

      public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
         String line = value.toString().toLowerCase();
         String[] words = line.split("\\s+");

         FileSplit fileSplit = (FileSplit) context.getInputSplit();
         String fileName = fileSplit.getPath().getName();
         int docId = Integer.parseInt(fileName.substring(4, fileName.length() - 4));

         for (int i = 0; i < words.length; i++) {
            String word = words[i];
            this.word.set(word);
            this.docInfo.set("{" + docId + "," + i + "}"); // Store the document ID and position of the
word
            context.write(this.word, this.docInfo);
         }
      }
   }

   public static class ReduceClass extends Reducer<Text, Text, Text, Text> {
      private Text result = new Text();
```

```
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
        StringBuilder sb = new StringBuilder();
        List<String> docInfoList = new ArrayList<>();

        for (Text value : values) {
          docInfoList.add(value.toString());
        }

        Collections.sort(docInfoList);

        sb.append("[");
        for (int i = 0; i < docInfoList.size(); i++) {
          sb.append(docInfoList.get(i));
          if (i < docInfoList.size() - 1) {
            sb.append(", ");
          }
        }
        sb.append("]");

        result.set(sb.toString());
        context.write(key, result);
    }
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "FullIndexMR");
    job.setJarByClass(FullIndexMR.class);
    job.setMapperClass(MapClass.class);
    job.setReducerClass(ReduceClass.class);
    job.setMapOutputKeyClass(Text.class);
    job.setMapOutputValueClass(Text.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);
    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    TextOutputFormat.setOutputPath(job, new Path(args[1]));
    job.waitForCompletion(true);
  }
}
```

Commands:

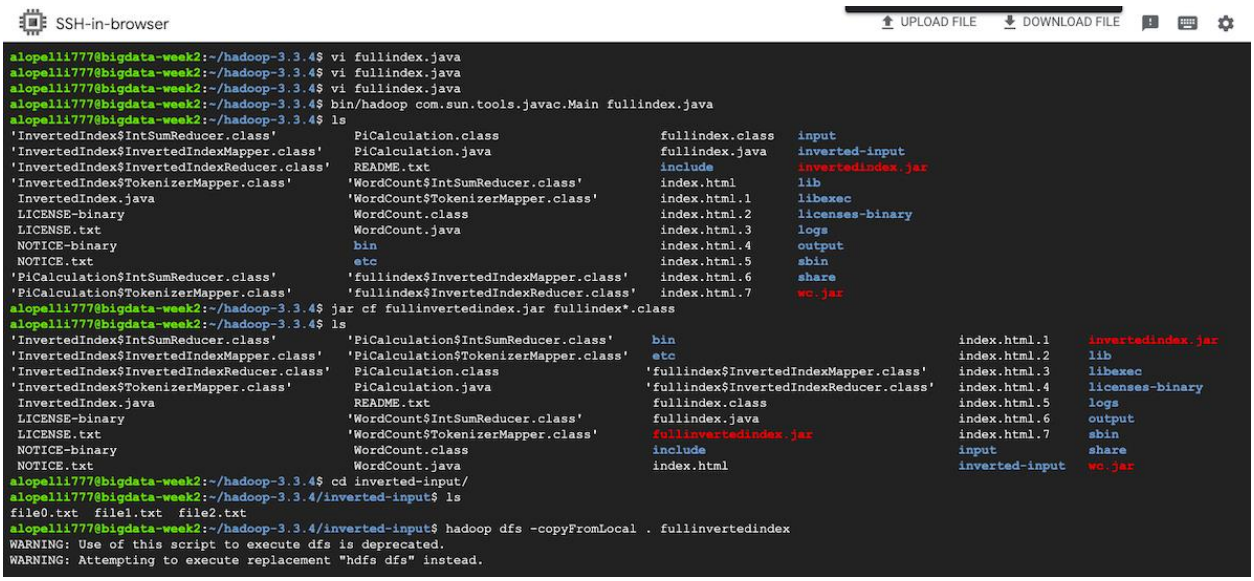
```
$ vi fullndex.java
$ bin/hadoop com.sun.tools.javac.Main fullindex.java
$ jar cf invertedindex.jar fullindex*.class
```

Copy input/file0, input/file1 and input/file2 of this project and place them inside the input folder of the Hadoop distrubution folder. While you are still there, run the following command to submit the job, get the input files from input folder, generate the inverted index and store its output in the output folder

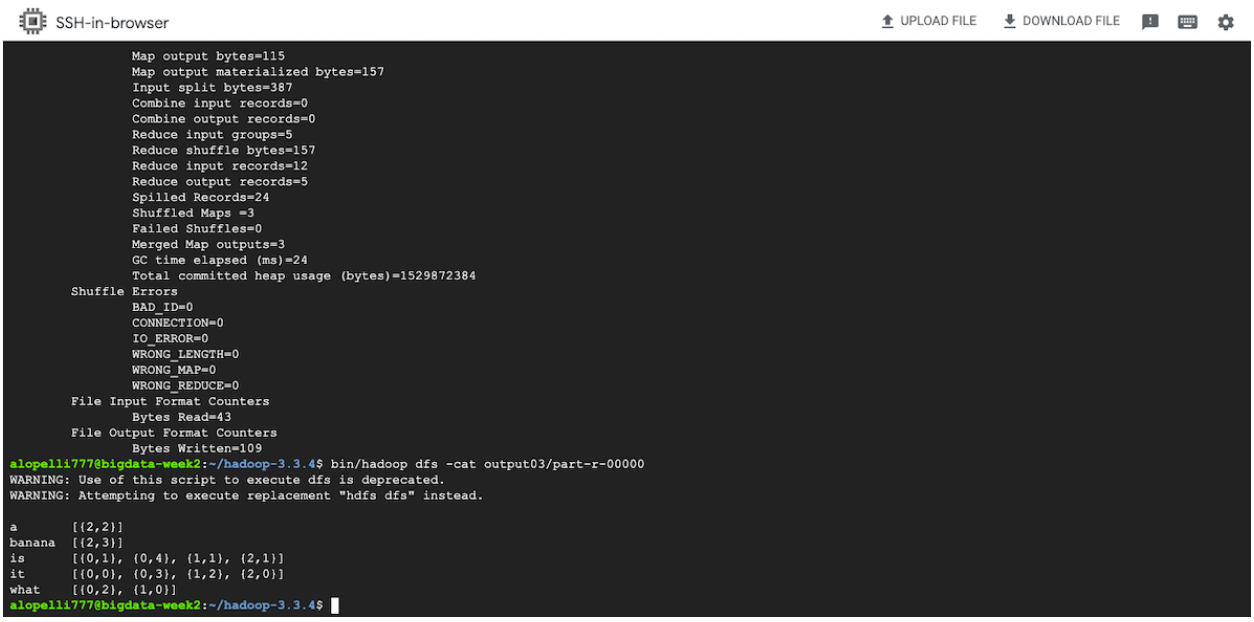
```
$ hadoop dfs -copyFromLocal .  invertedindex25
$ bin/hadoop jar fullinvertedindex.jar fullindex invertedindex25 output03
```

And finally to see the output, run the below command:

```
$ bin/hadoop dfs -cat output03/part-r-00000
```




```
alopelli777@bigdata-week2:~/hadoop-3.3.4/inverted-input$ cd ..
alopelli777@bigdata-week2:~/hadoop-3.3.4$ bin/hadoop jar fullinvertedindex.jar fullindex fullinvertedindex output20
2023-06-07 04:52:52,695 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2023-06-07 04:52:52,822 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2023-06-07 04:52:52,823 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2023-06-07 04:52:53,114 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your app
lication with ToolRunner to remedy this.
2023-06-07 04:52:53,319 INFO input.FileInputFormat: Total input files to process : 3
2023-06-07 04:52:53,349 INFO mapreduce.JobSubmitter: number of splits:3
2023-06-07 04:52:53,551 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1422010487_0001
2023-06-07 04:52:53,552 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-06-07 04:52:53,811 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2023-06-07 04:52:53,813 INFO mapreduce.Job: Running job: job_local1422010487_0001
2023-06-07 04:52:53,819 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2023-06-07 04:52:53,851 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-06-07 04:52:53,852 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failu
res: false
2023-06-07 04:52:53,853 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
2023-06-07 04:52:53,948 INFO mapred.LocalJobRunner: Waiting for map tasks
2023-06-07 04:52:53,949 INFO mapred.LocalJobRunner: Starting task: attempt_local1422010487_0001_m_000000_0
```


SSH-in-browser
UPLOAD FILE
DOWNLOAD FILE
Map output bytes=115
Map output materialized bytes=157
Input split bytes=387
Combine input records=0
Combine output records=0
Reduce input groups=5
Reduce shuffle bytes=157
Reduce input records=12
Reduce output records=5
Spilled Records=24
Shuffled Maps =3
Failed Shuffles=0
Merged Map outputs=3
GC time elapsed (ms)=24
Total committed heap usage (bytes)=1529872384
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=43
File Output Format Counters
Bytes Written=109
alopelli777@bigdata-week2:~/hadoop-3.3.4$ bin/hadoop dfs -cat output03/part-r-00000
WARNING: Use of this script to execute dfs is deprecated.
WARNING: Attempting to execute replacement "hdfs dfs" instead.
a [{2,2}]
banana [{2,3}]
is [{0,1}, {0,4}, {1,1}, {2,1}]
it [{0,0}, {0,3}, {1,2}, {2,0}]
what [{0,2}, {1,0}]
alopelli777@bigdata-week2:~/hadoop-3.3.4$