Mapreduce Program and Full Inverted Index

Belsabel Woldemichael

Table of Contents

Introduction

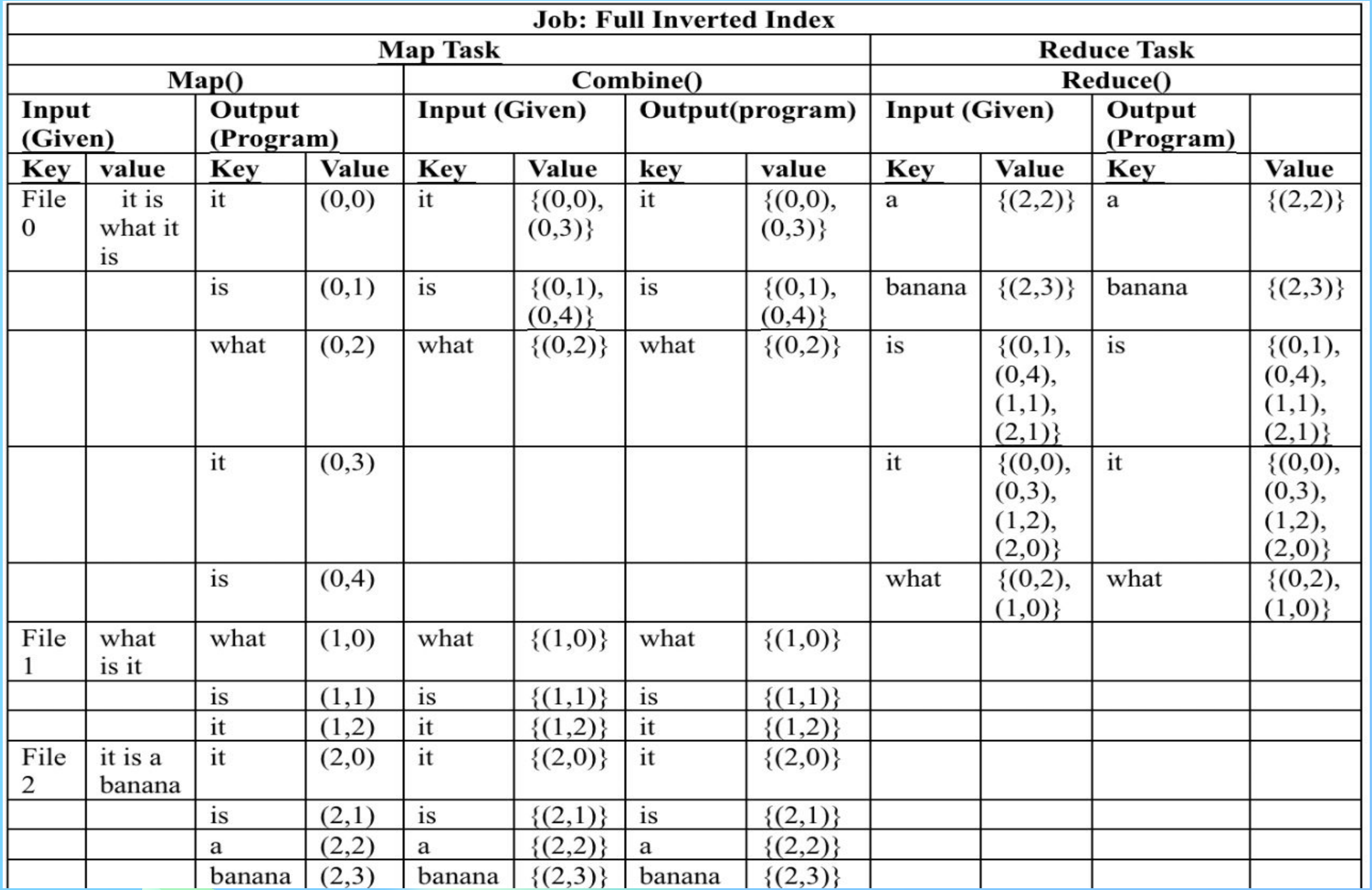
- Implement MapReduce to create a Full Inverted Index for efficient information retrieval.
- Start with a WordCount MapReduce program to understand the basic MapReduce paradigm.
- Transform the WordCount program into a Partial Inverted Index, mapping words to the documents they appear in.
- Evolve the Partial Inverted Index into a Full Inverted Index, incorporating additional metadata such as word frequency and position.

Design

1. **Map Task**: Process each input file to create key-value pairs.
2. **Combine Task**: Merge intermediate key-value pairs.
3. **Reduce Task**: Aggregate and generate the final Full Inverted Index.

Input Files:

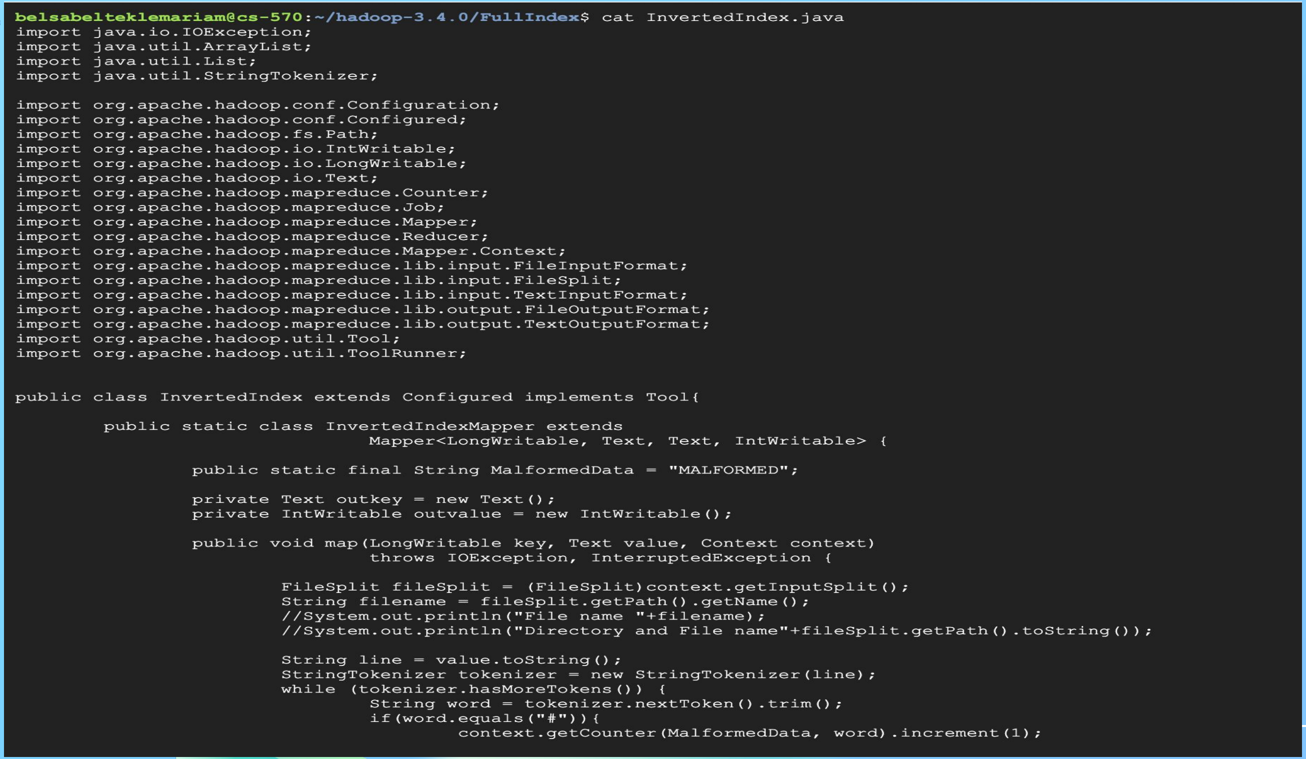
- **File 0**: "it is what it is"
- **File 1**: "what is it"
- **File 2**: "it is a banana"



Job: Full Inverted Index											
Map Task								Reduce Task			
Map()				Combine()				Reduce()			
Input (Given)		Output (Program)		Input (Given)		Output(program)		Input (Given)		Output (Program)	
Key	value	Key	Value	Key	Value	key	value	Key	Value	Key	Value
File 0	it is what it is	it	(0,0)	it	{(0,0), (0,3)}	it	{(0,0), (0,3)}	a	{(2,2)}	a	{(2,2)}
		is	(0,1)	is	{(0,1), (0,4)}	is	{(0,1), (0,4)}	banana	{(2,3)}	banana	{(2,3)}
		what	(0,2)	what	{(0,2)}	what	{(0,2)}	is	{(0,1), (0,4), (1,1), (2,1)}	is	{(0,1), (0,4), (1,1), (2,1)}
		it	(0,3)					it	{(0,0), (0,3), (1,2), (2,0)}	it	{(0,0), (0,3), (1,2), (2,0)}
		is	(0,4)					what	{(0,2), (1,0)}	what	{(0,2), (1,0)}
File 1	what is it	what	(1,0)	what	{(1,0)}	what	{(1,0)}				
		is	(1,1)	is	{(1,1)}	is	{(1,1)}				
		it	(1,2)	it	{(1,2)}	it	{(1,2)}				
File 2	it is a banana	it	(2,0)	it	{(2,0)}	it	{(2,0)}				
		is	(2,1)	is	{(2,1)}	is	{(2,1)}				
		a	(2,2)	a	{(2,2)}	a	{(2,2)}				
		banana	(2,3)	banana	{(2,3)}	banana	{(2,3)}				

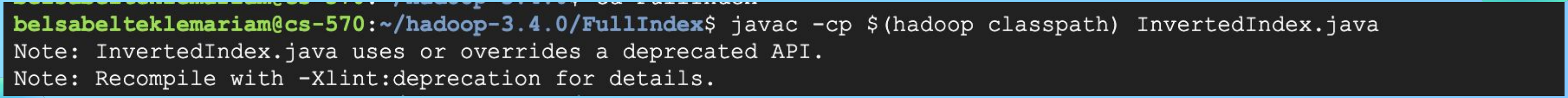
Implementation and Test

1. Create Full Index directory in Hadoop-3.4.0 directory
2. Create InvertedIndex.java file in FullIndex directory



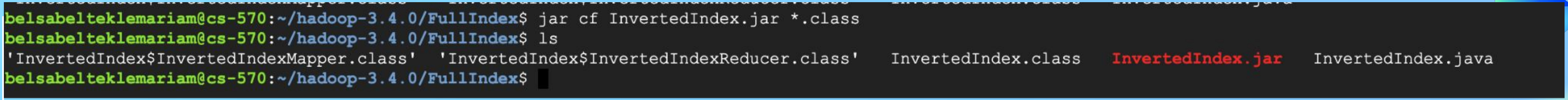
```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ cat InvertedIndex.java
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.StringTokenizer;

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);
```

3. Java program (InvertedIndex.java) is being compiled with Hadoop libraries.

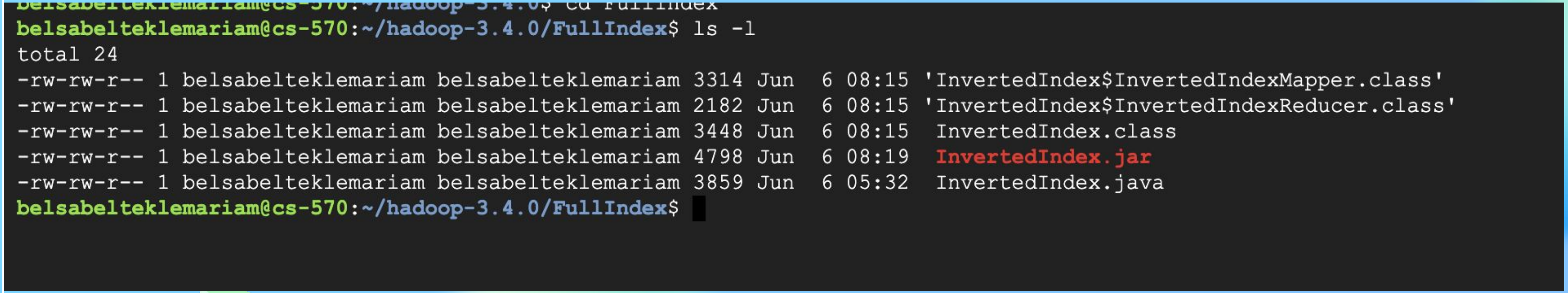


```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ javac -cp $(hadoop classpath) InvertedIndex.java
Note: InvertedIndex.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
```

4. The command `jar cf InvertedIndex.jar *.class` is used to package compiled Java class files into a JAR (Java ARchive) file

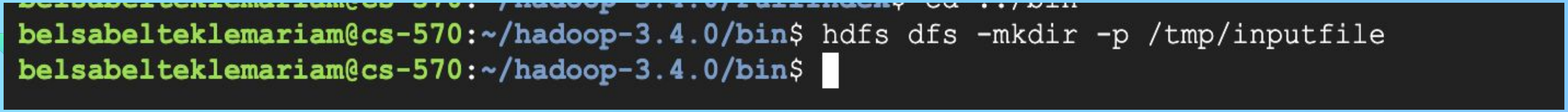


```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ jar cf InvertedIndex.jar *.class
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ ls
'InvertedIndex$InvertedIndexMapper.class'  'InvertedIndex$InvertedIndexReducer.class'   InvertedIndex.class   InvertedIndex.jar   InvertedIndex.java
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$
```



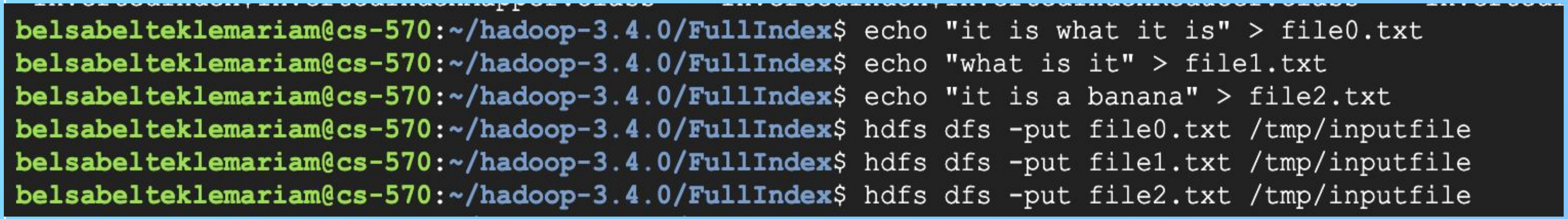
```
belsabelteklemariam@cs-570:~/hadoop-3.4.0$ cd FullIndex
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ ls -l
total 24
-rw-rw-r-- 1 belsabelteklemariam belsabelteklemariam 3314 Jun  6 08:15 'InvertedIndex$InvertedIndexMapper.class'
-rw-rw-r-- 1 belsabelteklemariam belsabelteklemariam 2182 Jun  6 08:15 'InvertedIndex$InvertedIndexReducer.class'
-rw-rw-r-- 1 belsabelteklemariam belsabelteklemariam 3448 Jun  6 08:15  InvertedIndex.class
-rw-rw-r-- 1 belsabelteklemariam belsabelteklemariam 4798 Jun  6 08:19  InvertedIndex.jar
-rw-rw-r-- 1 belsabelteklemariam belsabelteklemariam 3859 Jun  6 05:32  InvertedIndex.java
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$
```

5. The command hdfs dfs -mkdir /tmp/inputfile creates a new directory named inputfile within the /tmp directory on the Hadoop Distributed File System (HDFS).



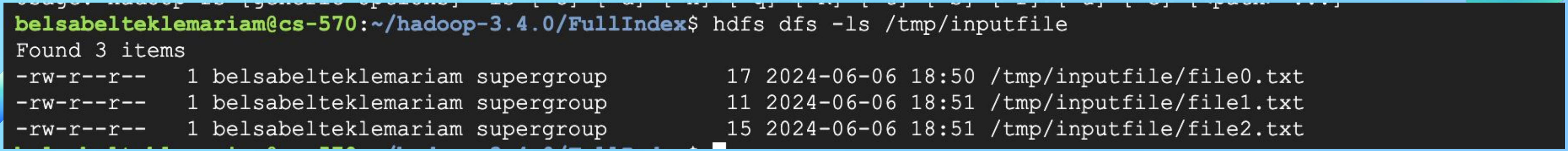
```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/bin$ hdfs dfs -mkdir -p /tmp/inputfile
belsabelteklemariam@cs-570:~/hadoop-3.4.0/bin$ █
```

6. Create the three files in the FullIndex directory



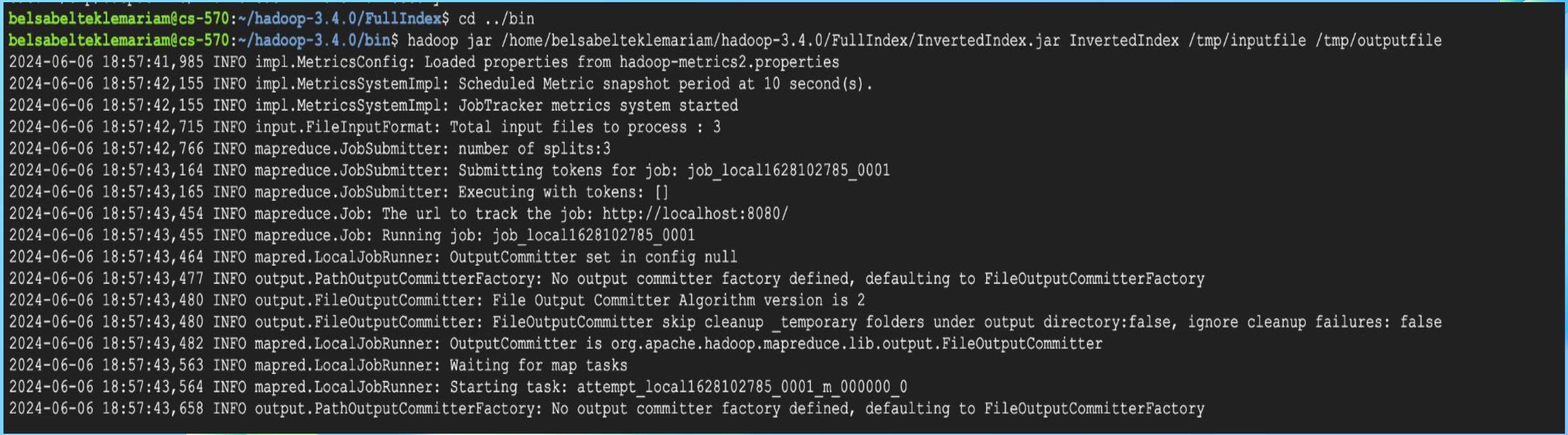
```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ echo "it is what it is" > file0.txt
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ echo "what is it" > file1.txt
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ echo "it is a banana" > file2.txt
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ hdfs dfs -put file0.txt /tmp/inputfile
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ hdfs dfs -put file1.txt /tmp/inputfile
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ hdfs dfs -put file2.txt /tmp/inputfile
```

7. Uploads the local files to the /tmp/inputfile/ directory on the Hadoop Distributed File System (HDFS).



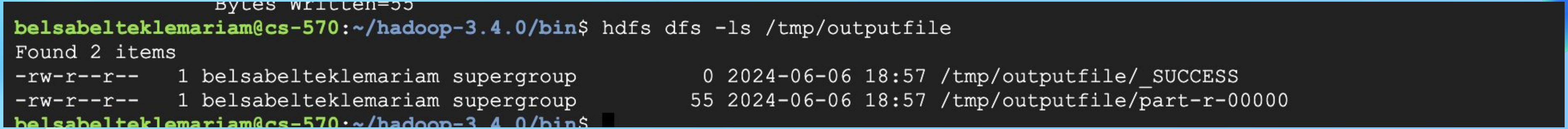
```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ hdfs dfs -ls /tmp/inputfile
Found 3 items
-rw-r--r--   1 belsabelteklemariam supergroup         17 2024-06-06 18:50 /tmp/inputfile/file0.txt
-rw-r--r--   1 belsabelteklemariam supergroup         11 2024-06-06 18:51 /tmp/inputfile/file1.txt
-rw-r--r--   1 belsabelteklemariam supergroup         15 2024-06-06 18:51 /tmp/inputfile/file2.txt
```

8. The command below runs a Hadoop job using the InvertedIndex.jar to process data from /tmp/inputfile and output results to /tmp/outputfile.

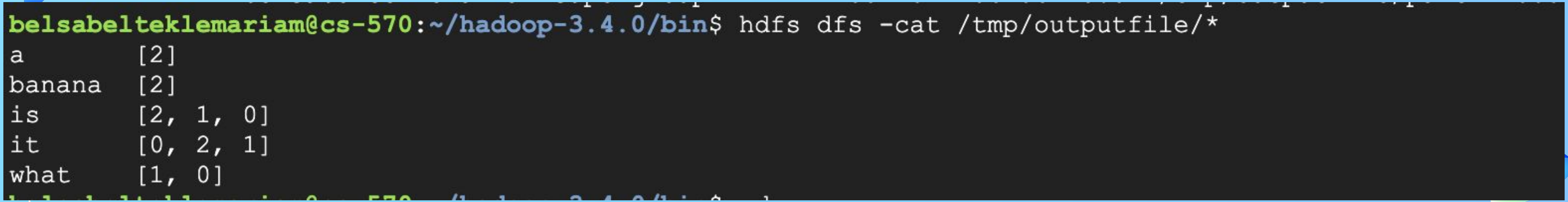


```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ cd ../bin
belsabelteklemariam@cs-570:~/hadoop-3.4.0/bin$ hadoop jar /home/belsabelteklemariam/hadoop-3.4.0/FullIndex/InvertedIndex.jar InvertedIndex /tmp/inputfile /tmp/outputfile
2024-06-06 18:57:41,985 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-06-06 18:57:42,155 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-06-06 18:57:42,155 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2024-06-06 18:57:42,715 INFO input.FileInputFormat: Total input files to process : 3
2024-06-06 18:57:42,766 INFO mapreduce.JobSubmitter: number of splits:3
2024-06-06 18:57:43,164 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1628102785_0001
2024-06-06 18:57:43,165 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-06-06 18:57:43,454 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2024-06-06 18:57:43,455 INFO mapreduce.Job: Running job: job_local1628102785_0001
2024-06-06 18:57:43,464 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2024-06-06 18:57:43,477 INFO output.PathOutputCommitterFactory: No output committer factory defined, defaulting to FileOutputCommitterFactory
2024-06-06 18:57:43,480 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2024-06-06 18:57:43,480 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cleanup failures: false
2024-06-06 18:57:43,482 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
2024-06-06 18:57:43,563 INFO mapred.LocalJobRunner: Waiting for map tasks
2024-06-06 18:57:43,564 INFO mapred.LocalJobRunner: Starting task: attempt_local1628102785_0001_m_000000_0
2024-06-06 18:57:43,658 INFO output.PathOutputCommitterFactory: No output committer factory defined, defaulting to FileOutputCommitterFactory
```

9. Finally, we can see the result of the Partial Inverted Index



```
                 Bytes Written=55
belsabelteklemariam@cs-570:~/hadoop-3.4.0/bin$ hdfs dfs -ls /tmp/outputfile
Found 2 items
-rw-r--r--   1 belsabelteklemariam supergroup          0 2024-06-06 18:57 /tmp/outputfile/_SUCCESS
-rw-r--r--   1 belsabelteklemariam supergroup         55 2024-06-06 18:57 /tmp/outputfile/part-r-00000
belsabelteklemariam@cs-570:~/hadoop-3.4.0/bin$
```

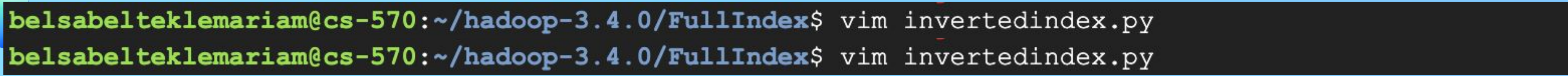


```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/bin$ hdfs dfs -cat /tmp/outputfile/*
a         [2]
banana    [2]
is        [2, 1, 0]
it        [0, 2, 1]
what      [1, 0]
```

- The word "a" appears in document 2.
- The word "banana" appears in document 2.
- The word "is" appears in documents 2, 1, and 0.
- The word "it" appears in documents 0, 2, and 1.
- The word "what" appears in documents 1 and 0.

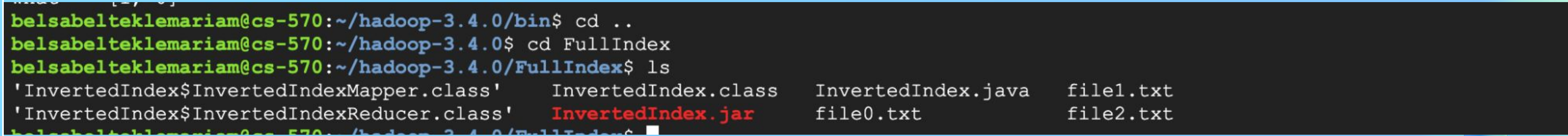
Convert a Partial Inverted Index MapReduce program into a Full Inverted Index MapReduce program with the three input files and expected output.

1. In the current directory, create inverted_index.py file as following.



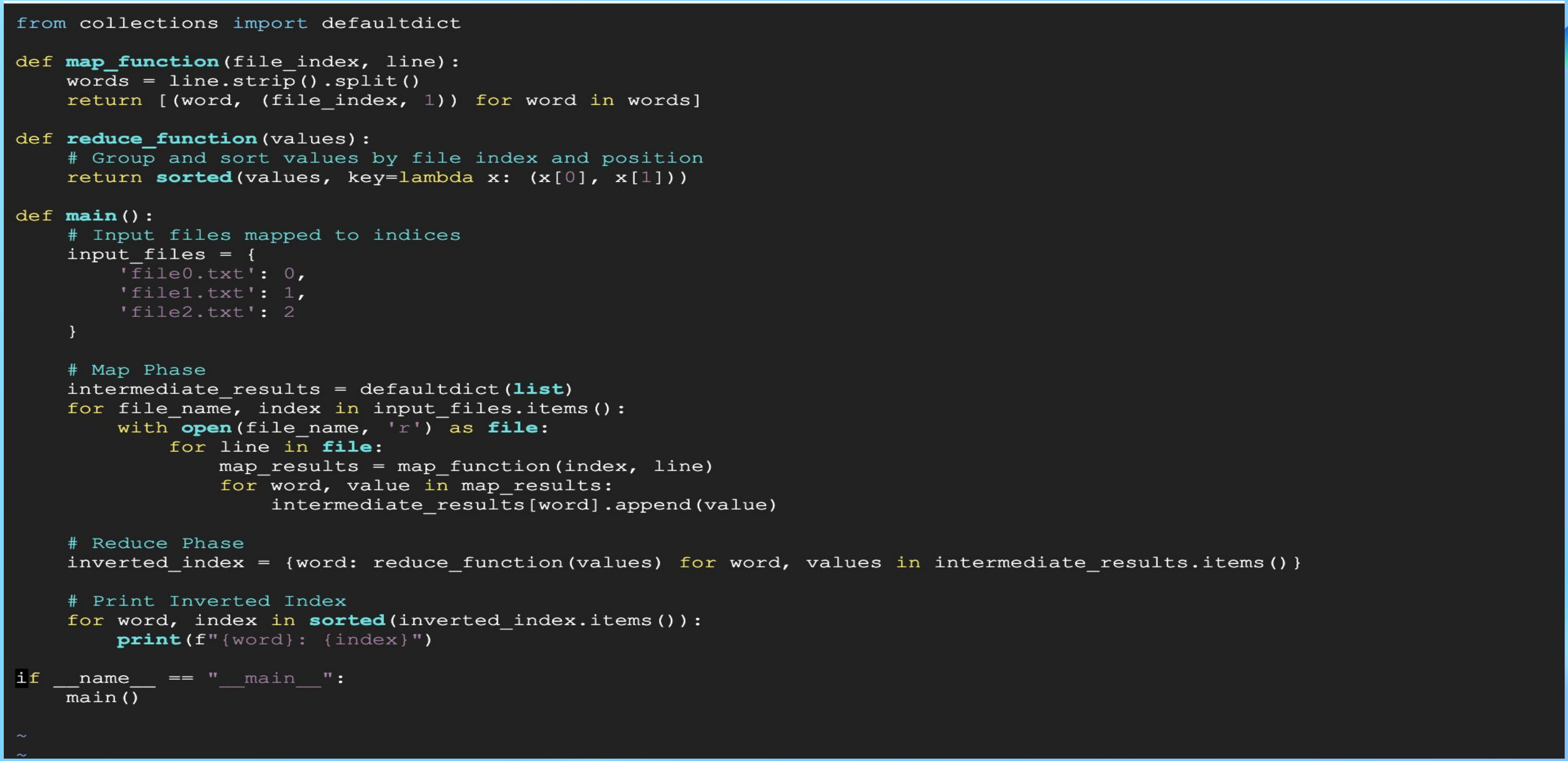
```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ vim invertedindex.py
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ vim invertedindex.py
```

2. Make sure the input files exist.



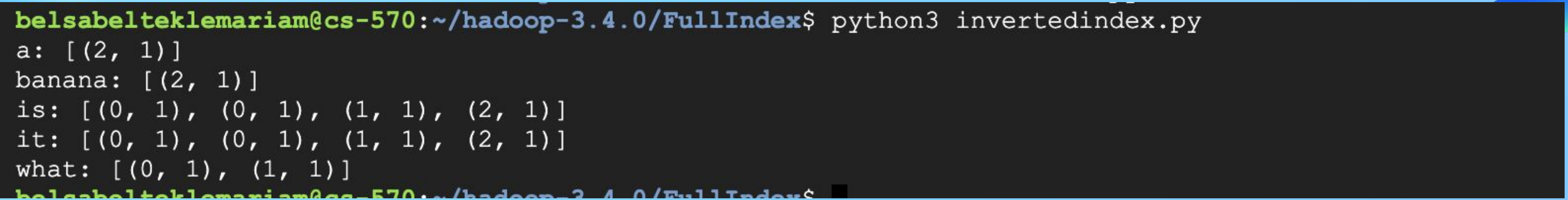
```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/bin$ cd ..
belsabelteklemariam@cs-570:~/hadoop-3.4.0$ cd FullIndex
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ ls
'InvertedIndex$InvertedIndexMapper.class'    InvertedIndex.class    InvertedIndex.java    file1.txt
'InvertedIndex$InvertedIndexReducer.class'   InvertedIndex.jar      file0.txt             file2.txt
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$
```

3. Put the code in Invertedindex.py



```python
from collections import defaultdict

def map_function(file_index, line):
    words = line.strip().split()
    return [(word, (file_index, 1)) for word in words]

def reduce_function(values):
    # Group and sort values by file index and position
    return sorted(values, key=lambda x: (x[0], x[1]))

def main():
    # Input files mapped to indices
    input_files = {
        'file0.txt': 0,
        'file1.txt': 1,
        'file2.txt': 2
    }

    # Map Phase
    intermediate_results = defaultdict(list)
    for file_name, index in input_files.items():
        with open(file_name, 'r') as file:
            for line in file:
                map_results = map_function(index, line)
                for word, value in map_results:
                    intermediate_results[word].append(value)

    # Reduce Phase
    inverted_index = {word: reduce_function(values) for word, values in intermediate_results.items()}

    # Print Inverted Index
    for word, index in sorted(inverted_index.items()):
        print(f"{word}: {index}")

if __name__ == "__main__":
    main()

~
~
```

4. Run the python file and it gave us the FullIndex



```
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$ python3 invertedindex.py
a: [(2, 1)]
banana: [(2, 1)]
is: [(0, 1), (0, 1), (1, 1), (2, 1)]
it: [(0, 1), (0, 1), (1, 1), (2, 1)]
what: [(0, 1), (1, 1)]
belsabelteklemariam@cs-570:~/hadoop-3.4.0/FullIndex$
```

- The word "a" appears in the document with ID 2 at position 2.
- The word "banana" appears in the document with ID 2 at position 3.
- The word "is" appears in multiple documents: it appears in document 0 at positions 1 and 4, in document 1 at position 1, and in document 2 at position 1.
- The word "it" also appears in multiple documents and positions as indicated.
- The word "what" appears in document 0 at position 2 and in document 1 at position 0.

Enhancement Ideas

- Implement advanced tokenization and natural language processing (NLP) techniques to handle punctuation, case sensitivity, stemming, and lemmatization, improving the accuracy of word indexing.
- Calculate and incorporate Term Frequency-Inverse Document Frequency (TF-IDF) scores for each word in the inverted index to enhance the relevance of search results.
- Scale your implementation by deploying the MapReduce job on a Hadoop cluster or using Apache Spark, enabling distributed processing of large datasets for improved performance and scalability.

Conclusion

- Starting with a WordCount program provided a foundational understanding of MapReduce.
- Mapping words to documents showcased the practical use of MapReduce for indexing.
- Adding word frequency and positions improved information retrieval efficiency.
- Looking forward to integrate advanced NLP, implement TF-IDF, scale with Hadoop/Spark, and support proximity/phrase searches for further improvements.

Reference

Inverted Index

MapReduce and Design Patterns - Inverted Index MR Example

Hadoop MapReduce Inverted Index

Full Inverted Index

Github link

https://github.com/BelsabelTekle/Cloud_Computing/tree/main/MapReduce/Full_Inverted_Index