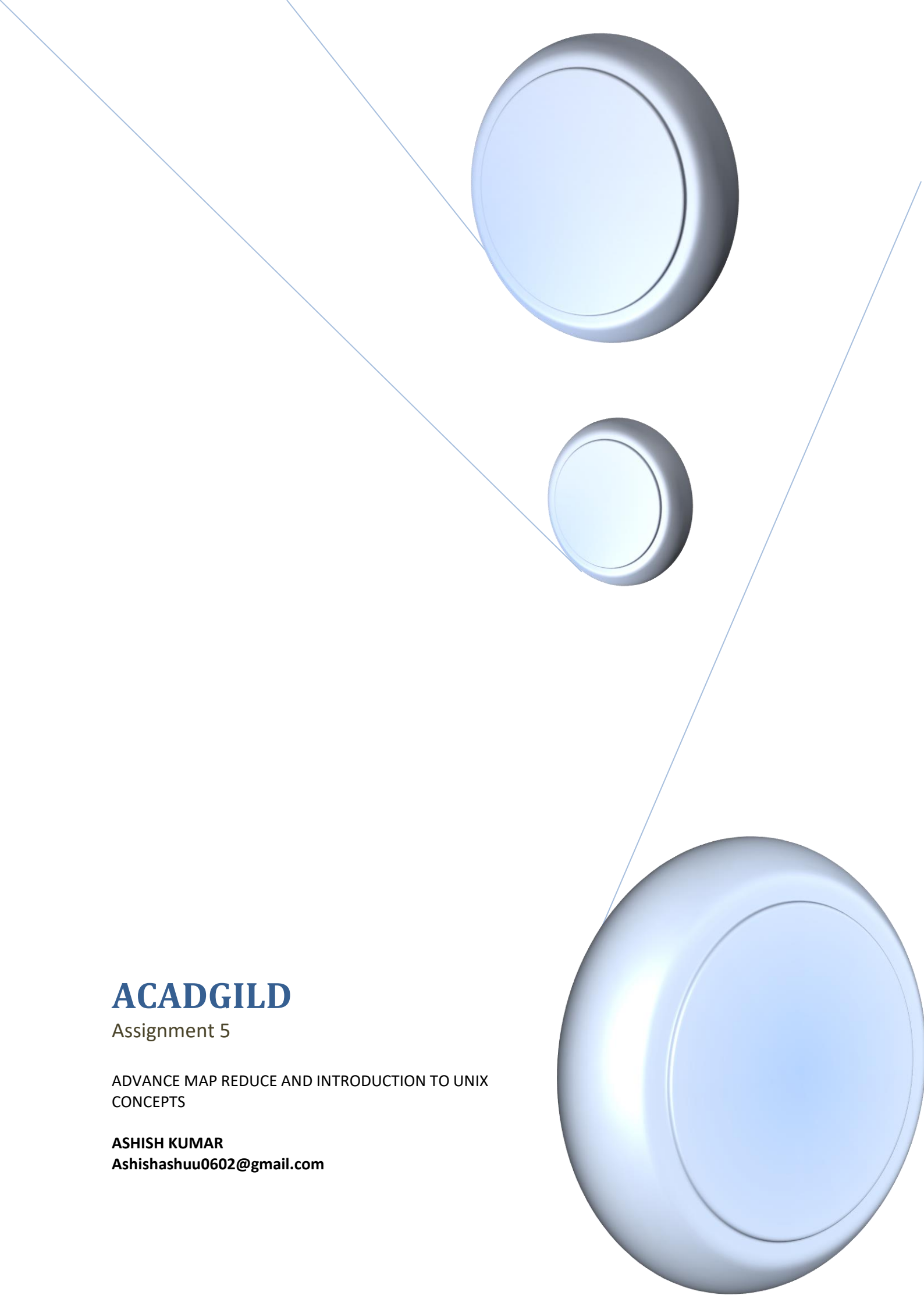


ACADGILD

Assignment 5

ADVANCE MAP REDUCE AND INTRODUCTION TO UNIX CONCEPTS

ASHISH KUMAR
Ashishashuu0602@gmail.com

Task 1: Find the Unique Number of Listeners.

Solution:

Mapper Code

```
/** Code to find number of Unique Listeners
package acadgild.assign5.task1;

import java.io.IOException;

/**
 * @author Ashish
 *
 */
public class NumUniqueListnMapper extends Mapper<LongWritable, Text,Text, IntWritable>{

        private static final IntWritable cou = new IntWritable(1);


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

                String[] userid = value.toString().split("[|]");
                Text uniquelistn = new Text(userid[0]);

                //Writing userid value as key
                context.write(uniquelistn, cou);
        }
}
```

Reducer Code

```
/** Reducer Code to generate Duplicate value for Listner
 *
 */
package acadgild.assign5.task1;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
//import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;


/**
 * @author Ashish
 * @param <Text>
 * @param <IntWritable>
 *
 */
//public class NumUniqueListnReducer extends Reducer<Text, IntWritable, Text, NullWritable>{
public class NumUniqueListnReducer extends Reducer<Text, IntWritable, Text, IntWritable>{
            private int numlistmusic;

    @Override
            protected void setup(Context context) {
            numlistmusic = 0;
        }


    public void reduce(Text uniquelistn, Iterable<IntWritable> counting, Context context) throws IOException,
    {
```

```
32          ++numlistmusic;
33         //  int sum=0;
34         //  for(IntWritable value:counting)
35         //  {
36         //      sum+=value.get();
37         //  if(sum == 1)
38                 //context.write(uniquelistn,new IntWritable(sum)); --  shwing number of times repeating
39         //      context.write(uniquelistn, NullWritable());  ------  working
40
41             }
42     protected void cleanup(Context context) throws IOException, InterruptedException {
43         context.write(new Text("Number of Unique Listners are - "), new IntWritable(numlistmusic));
44     }
45     }
46
47 //  private NullWritable NullWritable() {
48         // TODO Auto-generated method stub
49 //      return null;
50 //  }
51 //}
52
```

Driver Code

```
1  package acadgild.assign5.task1;
2
3  import java.io.IOException;
14
15
16 public class NumUniqueListnDriver {
17
18     public static void main(String[] args) throws ClassNotFoundException, IOException, InterruptedException {
19         // TODO Auto-generated method stub
20         if (args.length != 2) {
21               System.err.println("Usage: Provide Input file path and output path <input path> <output path>");
22               System.exit(-1);
23             }
24
25             //Job Related Configurations
26             Configuration conf = new Configuration();
27             @SuppressWarnings("deprecation")
28             Job job = new Job(conf, "Assignment 5 Task 1- Number of Unique Listners");
29             System.out.println("Assignment 5 Task 1 - Unique Listners in Number");
30             job.setJarByClass(NumUniqueListnDriver.class);
31
32
33             //Provide paths to pick the input file for the job
34             FileInputFormat.setInputPaths(job, new Path(args[0]));
35
36
37             //Provide paths to pick the output file for the job, and delete it if already present
38             Path outputPath = new Path(args[1]);
39             FileOutputFormat.setOutputPath(job, outputPath);
40             outputPath.getFileSystem(conf).delete(outputPath, true);
41
```

```
42
43            //To set the mapper and reducer of this job
44            job.setMapperClass(NumUniqueListnMapper.class);
45            job.setReducerClass(NumUniqueListnReducer.class);
46            // Specify the number of reducer to 1
47            job.setNumReduceTasks(1);
48
49
50            //set the input and output format class
51            job.setInputFormatClass(TextInputFormat.class);
52            job.setOutputFormatClass(TextOutputFormat.class);
53
54            //set up the output key and value classes
55            job.setMapOutputKeyClass(Text.class);
56            job.setMapOutputValueClass(IntWritable.class);
57
58            job.setOutputKeyClass(Text.class);
59            job.setOutputValueClass(IntWritable.class);
60
61            //execute the job
62            System.exit(job.waitForCompletion(true) ? 0 : 1);
63          }
64
65
66      }
67
```

Command to execute Jar –

```
[acadgild@localhost ~]$ hadoop jar /home/acadgild/Desktop/Assignment5/task1test/assign5task1.jar /hadoopdata/assignment5/musicdata.txt /hadoopdata/assignment5/
musictask1
18/07/29 00:12:07 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Assignment 5 Task 1 - Unique Listners in Number
18/07/29 00:12:08 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/07/29 00:12:09 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your applicati
on with ToolRunner to remedy this.
18/07/29 00:12:09 INFO input.FileInputFormat: Total input paths to process : 1
18/07/29 00:12:09 INFO mapreduce.JobSubmitter: number of splits:1
18/07/29 00:12:09 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1532799989690_0003
18/07/29 00:12:10 INFO impl.YarnClientImpl: Submitted application application_1532799989690_0003
18/07/29 00:12:10 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1532799989690_0003/
18/07/29 00:12:10 INFO mapreduce.Job: Running job: job_1532799989690_0003
18/07/29 00:12:19 INFO mapreduce.Job: Job job_1532799989690_0003 running in uber mode : false
```

Output –

The Number of Unique Listeners should be 3 as per input provided.

```
[acadgild@localhost ~]$ hadoop fs -cat /hadoopdata/assignment5/musictask1/part-r-00000
18/07/29 00:20:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform
Number of Unique Listners are -         3
[acadgild@localhost ~]$
```

Task 2: Number of times a song was heard fully.

Solution:

Mapper Code

```
13
14 /**
15  * @author Ashish
16  *
17  */
18 @SuppressWarnings("unused")
19 public class SongsHeardFullyMapper extends Mapper<LongWritable, Text,Text, IntWritable>{
20
21     private static final IntWritable cou = new IntWritable(1);
22
23
24     public void map(LongWritable key, Text value, Context context)
25                         throws IOException, InterruptedException {
26
27                 String[] user = value.toString().split("[|]");
28                 //Text songid = new Text(user[1]);
29
30                 String textconcat = new String ("The number of times a song with ID as ");
31                 String song = new String(user[1]);
32                 String concat = textconcat.concat(song).concat(new String(" heard fully is - "));
33
34                 Text songid = new Text(concat);
35
36                 if (user[4].equals("1"))
37
38                 context.write(songid, cou);    //Writing songid value as key
39
40 }
41 }
```

Reducer Code

```
11
12 /**
13  * @author Ashish
14  *
15  */
16 public class SongsHeardFullyReducer extends Reducer<Text, IntWritable, Text, IntWritable>{
17 //          private int songsfullheard;
18
19     @Override
20   //        protected void setup(Context context) {
21     //  songsfullheard = 0;
22   //    }
23
24
25     public void reduce(Text songid, Iterable<IntWritable> counting, Context context) throws IOException, Inter
26     {
27 //       ++songsfullheard;
28         int sum = 0;
29         for(IntWritable value: counting){
30             sum+= value.get();
31             }
32     //  context.write(new Text("Number of times a Song heared fully - "), new IntWritable(sum));
33         context.write(songid, new IntWritable(sum));
34             }
35 //  protected void cleanup(Context context) throws IOException, InterruptedException {
36  //       context.write(new Text("Number of times a Song heared fully - "), new IntWritable(songsfullheard));
37  //   }
38     }
39
40
```

Driver Code

```
31    public static void main(String[] args) throws ClassNotFoundException, IOException, InterruptedException {
32        // TODO Auto-generated method stub
33
34        if (args.length != 2) {
35              System.err.println("Usage: Provide Input file path and output path <input path> <output path>");
36              System.exit(-1);
37            }
38
39            //Job Related Configurations
40            Configuration conf = new Configuration();
41            @SuppressWarnings("deprecation")
42            Job job = new Job(conf, "Assignment 5 Task 2- Number of Times a Song Heard Fully");
43            System.out.println("Assignment 5 Task 2 - Number of Times a Song Heard Fully");
44            job.setJarByClass(SongsHeardFullyDriver.class);
45
46
47            //Provide paths to pick the input file for the job
48            FileInputFormat.setInputPaths(job, new Path(args[0]));
49
50
51            //Provide paths to pick the output file for the job, and delete it if already present
52            Path outputPath = new Path(args[1]);
53            FileOutputFormat.setOutputPath(job, outputPath);
54            outputPath.getFileSystem(conf).delete(outputPath, true);
55
56
57            //To set the mapper and reducer of this job
58            job.setMapperClass(SongsHeardFullyMapper.class);
59            job.setReducerClass(SongsHeardFullyReducer.class);
60            // Specify the number of reducer to 1
```

```
61            job.setNumReduceTasks(1);
62
63
64            //set the input and output format class
65            job.setInputFormatClass(TextInputFormat.class);
66            job.setOutputFormatClass(TextOutputFormat.class);
67
68            //set up the output key and value classes
69            job.setMapOutputKeyClass(Text.class);
70            job.setMapOutputValueClass(IntWritable.class);
71
72            job.setOutputKeyClass(Text.class);
73            job.setOutputValueClass(IntWritable.class);
74
75            //execute the job
76            System.exit(job.waitForCompletion(true) ? 0 : 1);
77          }
78
79    }
80
81
```

Command used to execute Jar file –

```
[acadgild@localhost ~]$ hadoop jar /home/acadgild/Desktop/Assignment5/Task2/assign5task2.jar  /hadoopdata/assignment5/musicdata.txt /hadoopdata/assignment5/mus
ictask2
18/07/29 02:53:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Assignment 5 Task 2 - Number of Times a Song Heard Fully
18/07/29 02:53:17 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/07/29 02:53:18 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your applicati
on with ToolRunner to remedy this.
18/07/29 02:53:18 INFO input.FileInputFormat: Total input paths to process : 1
18/07/29 02:53:18 INFO mapreduce.JobSubmitter: number of splits:1
18/07/29 02:53:18 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1532799989690_0013
18/07/29 02:53:19 INFO impl.YarnClientImpl: Submitted application application_1532799989690_0013
18/07/29 02:53:19 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1532799989690_0013/
18/07/29 02:53:19 INFO mapreduce.Job: Running job: job_1532799989690_0013
```

Output:

The output is 1 as per input text file provided.

[acadgild@localhost ~]$ hadoop fs -cat /hadoopdata/assignment5/musictask2/part-r-00000
18/07/29 02:54:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

The number of times a song with ID as 223 heard fully is - 1

```
[acadgild@localhost ~]$ hadoop fs -cat /hadoopdata/assignment5/musictask2/part-r-00000
18/07/29 02:54:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your plat
The number of times a song with ID as 223 heard fully is -      1
[acadgild@localhost ~]$
```

Task 3: What are the number of times a song was shared.

Solution:

Mapper Code

```
/**
 * @author Ashish
 *
 */
@SuppressWarnings("unused")
public class SongSharedFullyMapper extends Mapper<LongWritable, Text,Text, IntWritable>{
private static final IntWritable cou = new IntWritable(1);


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

                String[] user = value.toString().split("[|]");

                String textconcat = new String ("The number of times a song with ID ");
                String song = new String(user[1]);
                String concat = textconcat.concat(song).concat(new String(" Shared fully is - "));

                Text sharedsong = new Text(concat);

                if (user[2].equals("1"))

                context.write(sharedsong, cou);   //Writing songshared as key

}
}
```

Reducer Code

```java
/**
package com.acadgild.assign5task3;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;

/**
 * @author Ashish
 *
 */
@SuppressWarnings("unused")
public class SongsSharedFullyReducer extends Reducer<Text, IntWritable, Text, IntWritable>{
    public void reduce(Text sharedsong, Iterable<IntWritable> counting, Context context) throws IOException, I
    {

        int sum = 0;
        for(IntWritable value: counting){
            sum+= value.get();
            }
        context.write(sharedsong, new IntWritable(sum));
            }
}
```

Driver Code

```java
import java.io.IOException;

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.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;


/**
 * @author Ashish
 *
 */
public class SongsSharedFullyDriver {

    /**
     * @param args
     * @throws InterruptedException
     * @throws IOException
     * @throws ClassNotFoundException
     */
    public static void main(String[] args) throws ClassNotFoundException, IOException, InterruptedException {
        // TODO Auto-generated method stub

        if (args.length != 2) {
              System.err.println("Usage: Provide Input file path and output path <input path> <output path>");
              System.exit(-1);
```

```
37              }
38
39              //Job Related Configurations
40              Configuration conf = new Configuration();
41              @SuppressWarnings("deprecation")
42              Job job = new Job(conf, "Assignment 5 Task 3- Number of Times a Song Shared Fully");
43              System.out.println("Assignment 5 Task 3 - Number of Times a Song Shared Fully");
44              job.setJarByClass(SongsSharedFullyDriver.class);
45
46
47              //Provide paths to pick the input file for the job
48              FileInputFormat.setInputPaths(job, new Path(args[0]));
49
50
51              //Provide paths to pick the output file for the job, and delete it if already present
52              Path outputPath = new Path(args[1]);
53              FileOutputFormat.setOutputPath(job, outputPath);
54              outputPath.getFileSystem(conf).delete(outputPath, true);
55
56
57              //To set the mapper and reducer of this job
58              job.setMapperClass(SongSharedFullyMapper.class);
59              job.setReducerClass(SongsSharedFullyReducer.class);
60              // Specify the number of reducer to 1
61              job.setNumReduceTasks(1);
62
63
64              //set the input and output format class
65              job.setInputFormatClass(TextInputFormat.class);
66              job.setOutputFormatClass(TextOutputFormat.class);
67
```

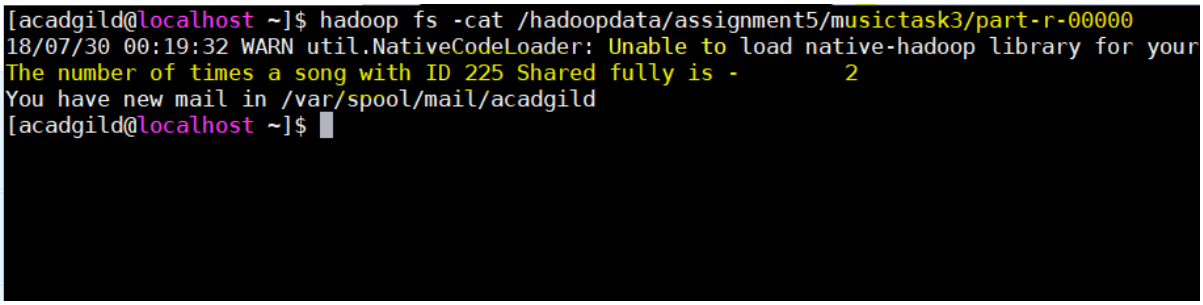
```
68              //set up the output key and value classes
69              job.setMapOutputKeyClass(Text.class);
70              job.setMapOutputValueClass(IntWritable.class);
71
72              job.setOutputKeyClass(Text.class);
73              job.setOutputValueClass(IntWritable.class);
74
75              //execute the job
76              System.exit(job.waitForCompletion(true) ? 0 : 1);
77          }
78
79      }
80
81
```

The Command to execute:

```
[acadgild@localhost ~]$ hadoop jar /home/acadgild/Desktop/Assignment5/Task3/assign5task3.jar  /hadoopdata/assignment5/musicdata.txt /hadoopdata/assignment5/mus
ictask3
18/07/30 00:09:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Assignment 5 Task 3 - Number of Times a Song Shared Fully
18/07/30 00:09:24 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/07/30 00:09:25 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your applicati
on with ToolRunner to remedy this.
18/07/30 00:09:25 INFO input.FileInputFormat: Total input paths to process : 1
18/07/30 00:09:26 INFO mapreduce.JobSubmitter: number of splits:1
18/07/30 00:09:26 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1532886532739_0003
18/07/30 00:09:26 INFO impl.YarnClientImpl: Submitted application application_1532886532739_0003
18/07/30 00:09:26 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1532886532739_0003/
18/07/30 00:09:26 INFO mapreduce.Job: Running job: job_1532886532739_0003
18/07/30 00:09:34 INFO mapreduce.Job: Job job_1532886532739_0003 running in uber mode : false
18/07/30 00:09:34 INFO mapreduce.Job:  map 0% reduce 0%
18/07/30 00:09:40 INFO mapreduce.Job:  map 100% reduce 0%
18/07/30 00:09:47 INFO mapreduce.Job:  map 100% reduce 100%
18/07/30 00:09:47 INFO mapreduce.Job: Job job_1532886532739_0003 completed successfully
18/07/30 00:09:47 INFO mapreduce.Job: Counters: 49
```

Output:

Song ID 225 is being shared 2 times. So the final output should be **2** with Song ID as **225.**


[acadgild@localhost ~]$ hadoop fs -cat /hadoopdata/assignment5/musictask3/part-r-00000
18/07/30 00:19:32 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
The number of times a song with ID 225 Shared fully is - 2
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$