Name: Solanki Mehul
Enrollment:202300819010047
(Hadoop Assigement)

Program 1 :

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

public class FirstProgram {
        public static class Map extends Mapper<LongWritable,Text,Text,IntWritable>{
                public void map(LongWritable key,Text value,Context context) throws
IOException,InterruptedException{
                        String line = value.toString();
                        StringTokenizer token = new StringTokenizer(line);
                        while(token.hasMoreElements()) {
                                value.set(token.nextToken());
                                context.write(value, new IntWritable(1));
                        }
                }
        }

        public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>{
                public void reduce(Text key,Iterable<IntWritable> value,Context context) throws
IOException,InterruptedException{
                        int sum = 0;
                        for(IntWritable i : value) {
                                sum += i.get();
                        }
                        context.write(key, new IntWritable(sum));
                }
        }

        public static void main(String args[]) throws Exception{
```

```
		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf,"FirstProgram");
		job.setJarByClass(FirstProgram.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);

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

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

		job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);

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

		outputPath.getFileSystem(conf).delete(outputPath, true);
		System.exit(job.waitForCompletion(true)?0:1);
	}
}
```

Text File :

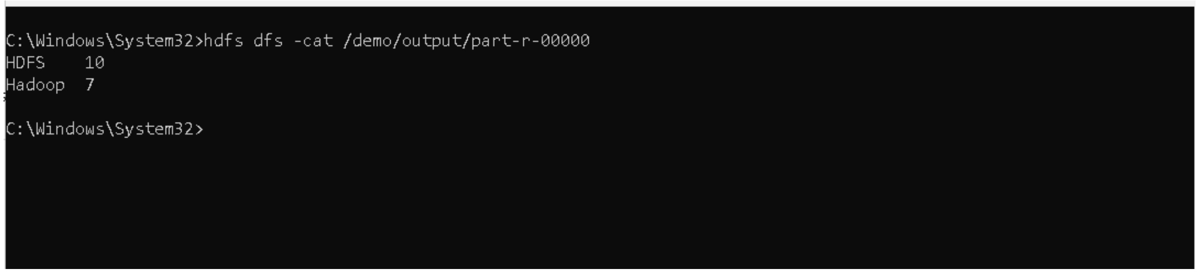
```
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
```

Commands :

Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :



```
C:\Windows\System32>hdfs dfs -cat /demo/output/part-r-00000
HDFS    10
Hadoop  7

C:\Windows\System32>
```

Program 2 :

```
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.LongWritable;
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 SecondProgram {
	public static class Map extends Mapper<LongWritable,Text,Text,IntWritable>{
		public void map(LongWritable key,Text value,Context context) throws
IOException,InterruptedException{
			String[] cols = value.toString().split(",");
			String year = cols[0];
			int temperature = Integer.parseInt(cols[1]);
			context.write(new Text(year),new IntWritable(temperature));
		}
	}

	public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>{
		public void reduce(Text key,Iterable<IntWritable> values,Context context) throws
IOException,InterruptedException{
			int minTemp = Integer.MAX_VALUE;
			for(IntWritable value : values) {
				minTemp = Math.min(minTemp, value.get());
			}
			context.write(key, new IntWritable(minTemp));
		}
	}

	public static void main(String args[]) throws Exception {
		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf,"SecondProgram");
		job.setJarByClass(SecondProgram.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.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);
	}
}
```

Text File :

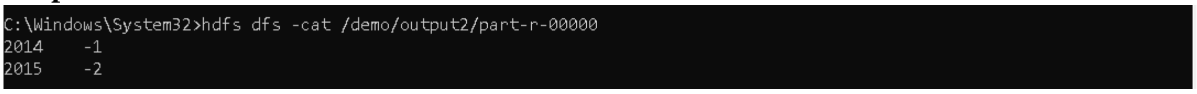
```
2014 1
2014 3
2014 -1
2014 5
2014 6
2014 8
2014 9
2014 10
2015 1
2015 -2
2015 5
2015 3
2015 4
```

Commands :

Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :



```
C:\Windows\System32>hdfs dfs -cat /demo/output2/part-r-00000
2014    -1
2015    -2
```

Program 3 :

```
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.LongWritable;
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 ThirdProgram {
        public static class Map extends Mapper<LongWritable,Text,Text,IntWritable>{
                public void map(LongWritable key,Text value,Context context) throws
IOException,InterruptedException{
                        String line = value.toString();
                        StringTokenizer token = new StringTokenizer(line);
                        while(token.hasMoreElements()) {
                                value.set(token.nextToken());
                                context.write(value, new IntWritable(1));
                        }
                }
        }

        public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>{
                private int outerSum = 0;

                public void reduce(Text key,Iterable<IntWritable> values,Context context) throws
IOException,InterruptedException{
                        int sum = 0;
                        for(IntWritable value : values) {
                                sum += value.get();
                                outerSum += value.get();
                        }
                        context.write(key, new IntWritable(sum));
                }

                public void cleanup(Context context) throws IOException,InterruptedException{
                        int avg = outerSum / 2;
                        context.write(new Text("Average"), new IntWritable(avg));
                }
        }

        public static void main(String args[]) throws Exception {
                Configuration conf = new Configuration();
                Job job = Job.getInstance(conf,"ThirdProgram");

                job.setJarByClass(ThirdProgram.class);
                job.setMapperClass(Map.class);
                job.setReducerClass(Reduce.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);
          }
}
```

Text File :

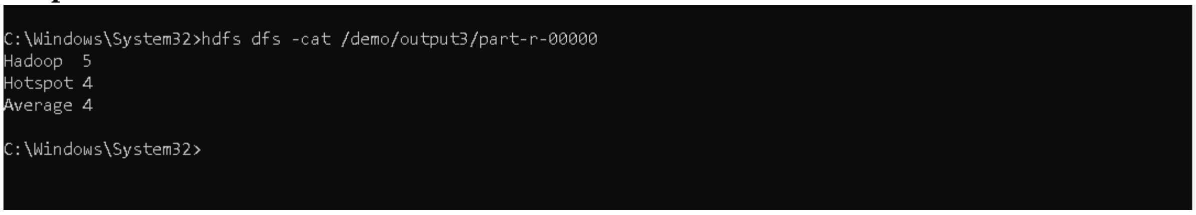
```
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
Hotspot
Hotspot
Hotspot
Hotspot
```

Commands :

Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :



```
C:\Windows\System32>hdfs dfs -cat /demo/output3/part-r-00000
Hadoop  5
Hotspot 4
Average 4

C:\Windows\System32>
```

Program 4 :

```
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.LongWritable;
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 FourthProgram {
	public static class Map extends Mapper<LongWritable,Text,Text,IntWritable>{
		public void map(LongWritable key,Text value,Context context) throws
IOException,InterruptedException{
			String line = value.toString();
			StringTokenizer token = new StringTokenizer(line);
			while(token.hasMoreElements()) {
				value.set(token.nextToken());
				if(value.getLength() >= 4) {
					context.write(value, new IntWritable(1));
				}
			}
		}
	}

	public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>{
		private int cnt = 0;
		public void reduce(Text key,Iterable<IntWritable> values,Context context) throws
IOException,InterruptedException{
			for(IntWritable value : values) {
				cnt += value.get();
			}
		}

		public void cleanup(Context context) throws IOException,InterruptedException{
			context.write(new Text("no of Count : "), new IntWritable(cnt));
		}
	}

	public static void main(String args[]) throws Exception {
		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf,"FourthProgram");

		job.setJarByClass(FourthProgram.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.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);
        }
}
```

Text File :

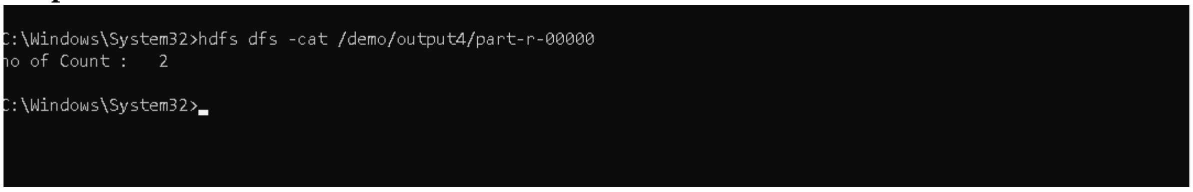
Java
Python
C
C++

Commands :

Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :



```
:\Windows\System32>hdfs dfs -cat /demo/output4/part-r-00000
o of Count :   2

:\Windows\System32>
```

Program 5 :

```
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.LongWritable;
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 FifthProgram {
	public static class Map extends Mapper<LongWritable,Text,Text,IntWritable>{
		public void map(LongWritable key,Text value,Context context) throws
IOException,InterruptedException{
			String[] cols = value.toString().split(",");
			String gender = cols[2];
			context.write(new Text(gender), new IntWritable(1));
		}
	}

	public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>{
		private int totalFemale = 0;
		public void reduce(Text key,Iterable<IntWritable> values,Context context) throws
IOException,InterruptedException{
			int sum = 0;
			for(IntWritable value : values) {
				sum += value.get();
			}
			if(key.equals(new Text("Female"))) {
				totalFemale = sum;
			}
		}

		public void cleanup(Context context) throws IOException,InterruptedException{
			context.write(new Text("Total female voters : "), new
IntWritable(totalFemale));
		}
	}

	public static void main(String args[]) throws Exception {
		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf,"FifthProgram");

		job.setJarByClass(FifthProgram.class);

		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.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);
	}
}
```

Text File :

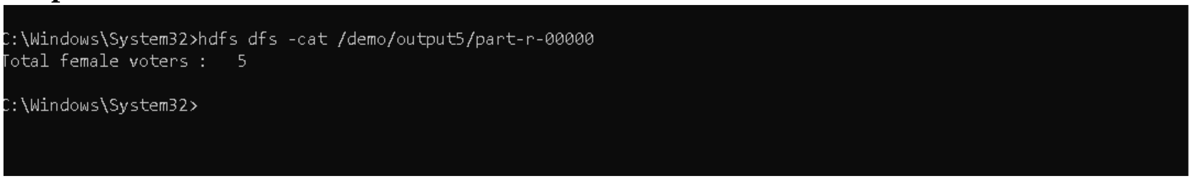
1,Divya,Male,20
2,Sumit,Male,20
3,Preksha,Female,20
4,Nikita,Female,20
5,Jishan,Male,20
6,Jhuveriya,Female,20
7,Nisarg,Male,20
8,Meet,Male,20
9,Kirsha,Female,20
10,Karina,Female,20

Commands :

Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :



```
:\Windows\System32>hdfs dfs -cat /demo/output5/part-r-00000
otal female voters :   5

:\Windows\System32>
```

Program 6 :

```
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.LongWritable;
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 SixthProgram {
	public static class Map extends Mapper<LongWritable,Text,Text,IntWritable>{
		public void map(LongWritable key,Text value,Context context) throws
IOException,InterruptedException {
			String cols[] = value.toString().split(",");
			String reviewID = cols[0];
			context.write(new Text(reviewID), new IntWritable(1));
		}
	}

	public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>{
		private int total_unique_reviews = 0;
		public void reduce(Text key,Iterable<IntWritable> values,Context context) throws
IOException,InterruptedException{
			int sum = 0;
			for(IntWritable value : values) {
				sum += value.get();
			}
			total_unique_reviews++;
			context.write(key, new IntWritable(sum));
		}

		public void cleanup(Context context) throws IOException,InterruptedException{
			context.write(new Text("Total unique reviews : "), new
IntWritable(total_unique_reviews));
		}
	}

	public static void main(String args[]) throws Exception {
		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf,"SixthProgram");

		job.setJarByClass(SixthProgram.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.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);
	}
}
```

CSV File :

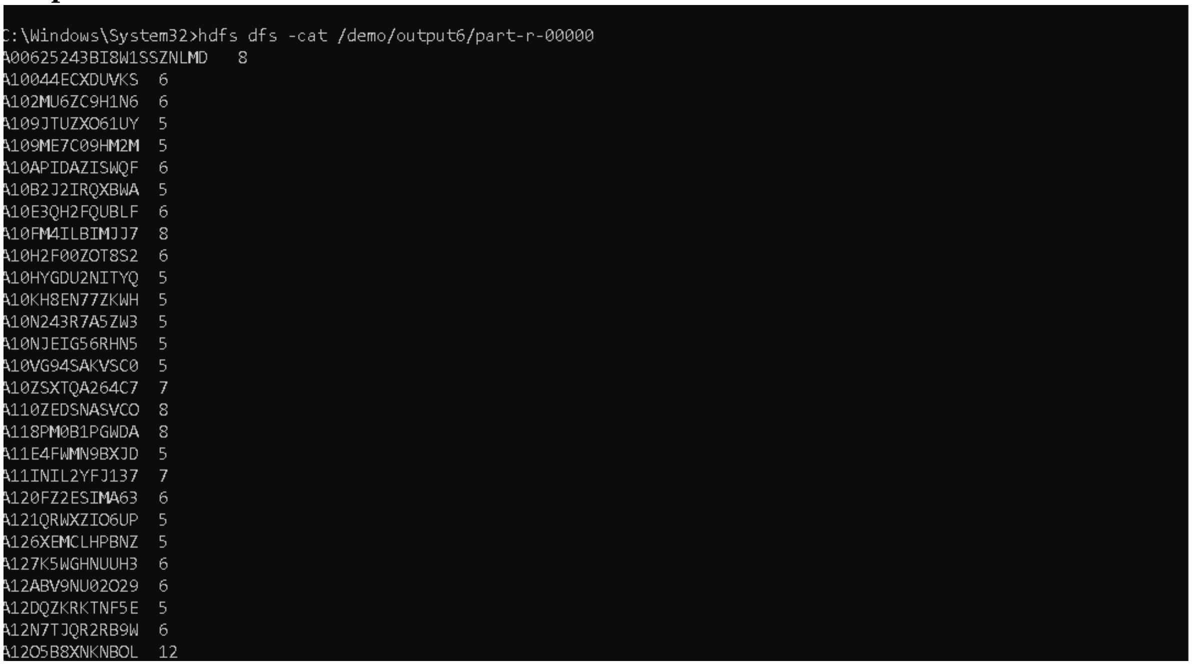
Note : file is large in size as well as rows wise.

Commands :

Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :



```
C:\Windows\System32>hdfs dfs -cat /demo/output6/part-r-00000
A00625243BI8W1SSZNLMD   8
A10044ECXDUVKS  6
A102MU6ZC9H1N6  6
A109JTUZXO61UY  5
A109ME7C09HM2M  5
A10APIDAZISWQF  6
A10B2J2IRQXBWA  5
A10E3QH2FQUBLF  6
A10FM4ILBIMJJ7  8
A10H2F00ZOT8S2  6
A10HYGDU2NITYQ  5
A10KH8EN77ZKWH  5
A10N243R7A5ZW3  5
A10NJEIG56RHN5  5
A10VG94SAKVSC0  5
A10ZSXTQA264C7  7
A110ZEDSNASVCO  8
A118PM0B1PGWDA  8
A11E4FWMN9BXJD  5
A11INIL2YFJ137  7
A120FZ2ESIMA63  6
A121QRWXZIO6UP  5
A126XEMCLHPBNZ  5
A127K5WGHNUUH3  6
A12ABV9NU02O29  6
A12DQZKRKTNF5E  5
A12N7TJQR2RB9W  6
A12O5B8XNKNBOL  12
```

Program 7 :

```
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.LongWritable;
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 SeventhProgram {
	public static class Map extends Mapper<LongWritable,Text,Text,IntWritable>{
		public void map(LongWritable key,Text value,Context context) throws
IOException,InterruptedException{
			String col[] = value.toString().split(",");
			String title = col[1].toString();
			String genres = col[2].toString();
			if(genres.contains("Comedy")) {
				context.write(new Text(title+" : "+genres),new IntWritable(1));
			}
			if(genres.contains("Documentary") && title.contains("1995")) {
				context.write(new Text("Documentry"),new IntWritable(1));
			}
			if(title.contains("Gold")) {
				context.write(new Text(title),new IntWritable(1));
			}
			if(genres.contains("Drama") && genres.contains("Romance")) {
				context.write(new Text(title + " : "+genres),new IntWritable(1));
			}
			if(genres.isEmpty()) {
				context.write(new Text("Missing"), new IntWritable(1));
			}
		}
	}

	public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>{
		private int count = 0;
		private int missing = 0;
		public void reduce(Text key,Iterable<IntWritable> values,Context context) throws
IOException,InterruptedException{
			int sum = 0;
			for(IntWritable value : values) {
				sum += value.get();
			}
			context.write(key, new IntWritable(sum));
			if(key.toString().contains("Documentary")) {
				count++;
			}
			if(key.toString().contains("Documentry")) {
				missing++;
			}
		}

		public void cleanup(Context context) throws IOException,InterruptedException{
			context.write(new Text("Total documentry movie in 1995 : "), new
IntWritable(count));
```

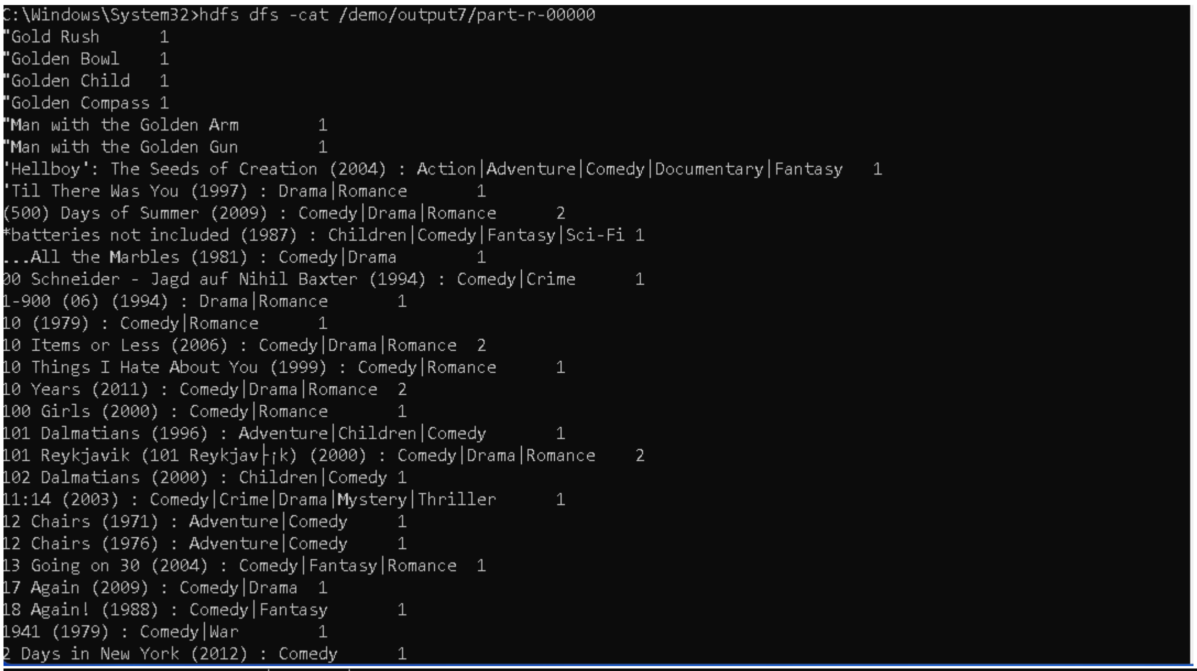
```
                        context.write(new Text("Total missing genres : "), new IntWritable(missing));
                }
        }

        public static void main(String args[]) throws Exception {
                Configuration conf = new Configuration();
                Job job = Job.getInstance(conf,"SeventhProgram");

                job.setJarByClass(SeventhProgram.class);
                job.setMapperClass(Map.class);
                job.setReducerClass(Reduce.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);
        }
}
```

CSV File :
Note : file is large in size as well as rows wise.

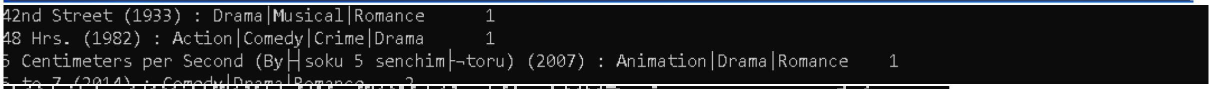
Commands :
Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

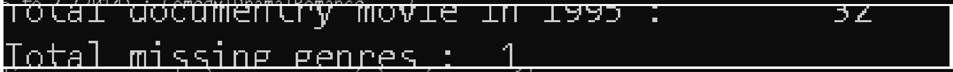
Output :



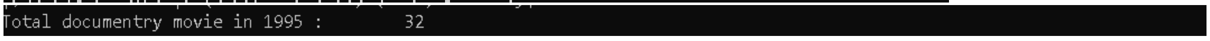
```
C:\Windows\System32>hdfs dfs -cat /demo/output7/part-r-00000
"Gold Rush      1
"Golden Bowl    1
"Golden Child   1
"Golden Compass 1
"Man with the Golden Arm        1
"Man with the Golden Gun        1
'Hellboy': The Seeds of Creation (2004) : Action|Adventure|Comedy|Documentary|Fantasy   1
'Til There Was You (1997) : Drama|Romance       1
(500) Days of Summer (2009) : Comedy|Drama|Romance      2
*batteries not included (1987) : Children|Comedy|Fantasy|Sci-Fi 1
...All the Marbles (1981) : Comedy|Drama        1
00 Schneider - Jagd auf Nihil Baxter (1994) : Comedy|Crime      1
1-900 (06) (1994) : Drama|Romance       1
10 (1979) : Comedy|Romance      1
10 Items or Less (2006) : Comedy|Drama|Romance  2
10 Things I Hate About You (1999) : Comedy|Romance      1
10 Years (2011) : Comedy|Drama|Romance  2
100 Girls (2000) : Comedy|Romance       1
101 Dalmatians (1996) : Adventure|Children|Comedy       1
101 Reykjavik (101 Reykjav├¡k) (2000) : Comedy|Drama|Romance    2
102 Dalmatians (2000) : Children|Comedy 1
11:14 (2003) : Comedy|Crime|Drama|Mystery|Thriller      1
12 Chairs (1971) : Adventure|Comedy     1
12 Chairs (1976) : Adventure|Comedy     1
13 Going on 30 (2004) : Comedy|Fantasy|Romance  1
17 Again (2009) : Comedy|Drama  1
18 Again! (1988) : Comedy|Fantasy       1
1941 (1979) : Comedy|War        1
2 Days in New York (2012) : Comedy      1
```



```
42nd Street (1933) : Drama|Musical|Romance      1
48 Hrs. (1982) : Action|Comedy|Crime|Drama      1
5 Centimeters per Second (By├soku 5 senchim├¬toru) (2007) : Animation|Drama|Romance    1
5 to 7 (2014) : Comedy|Drama|Romance    2
```



```
Total documentry movie in 1995 :        32
Total missing genres :  1
```



```
Total documentry movie in 1995 :        32
```