**MapReduce Program to find the Deanery wise, Department wise and Class wise Toppers of a University. (2 Methods)**

*Aayushi Shrivastava | 2139461*

**Program 1: Simple Mapper and Reducer**

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.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 Topper {

public static class MapIt

extends Mapper<Object, Text, Text, Text>{

@Override

public void map(Object key, Text value, Context context

) throws IOException, InterruptedException {

String data[]=value.toString().split(",");

String key1 = data[0];

String dean\_deets = String.format("%s-%s",data[0],data[5]);

String dept\_deets = String.format("%s-%s",data[0],data[5]);

String class\_deets = String.format("%s-%s",data[0],data[5]);

//Send only Deanery specific details to Deanery Keys

context.write(new Text(String.format("Deanery: %s Topper ----->",data[6])), new Text(dean\_deets));

//Send only Department specific details to Deanery Keys

context.write(new Text(String.format("Department: %s Topper ----->",data[2])), new Text(dept\_deets));

//Send only Class specific details to Deanery Keys

context.write(new Text(String.format("Class: %s Topper ----->",data[3])), new Text(class\_deets));

}

}

public static class ReduceIt

extends Reducer<Text,Text,Text,Text> {

public void reduce(Text key, Iterable <Text> values,

Context context

) throws IOException, InterruptedException {

int sum=0;

String max\_roll = "";

int max\_marks = 0;

String tied\_marks = "";

for (Text rec : values)

{

//Retrieving all the calculated values from the Mapper

String [] variables = rec.toString().split("-");

String roll = variables[0];

String mark = variables[1];

if (Integer.parseInt(mark) == max\_marks)

{

tied\_marks = tied\_marks + " and " + String.format("%s %s",roll,mark); //To handle tied scores

}

else if (Integer.parseInt(mark) > max\_marks)

{

max\_roll = roll;

max\_marks = Integer.parseInt(mark);

tied\_marks = String.format("%s %s",roll,mark); //To handle tied scored

}

}

String output\_str = String.format("%s",tied\_marks);

context.write(key, new Text(output\_str));

}

}

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "Practice");

job.setJarByClass(Topper.class);

job.setMapperClass(MapIt.class);

job.setReducerClass(ReduceIt.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

System.exit(job.waitForCompletion(true) ? 0 : 1);

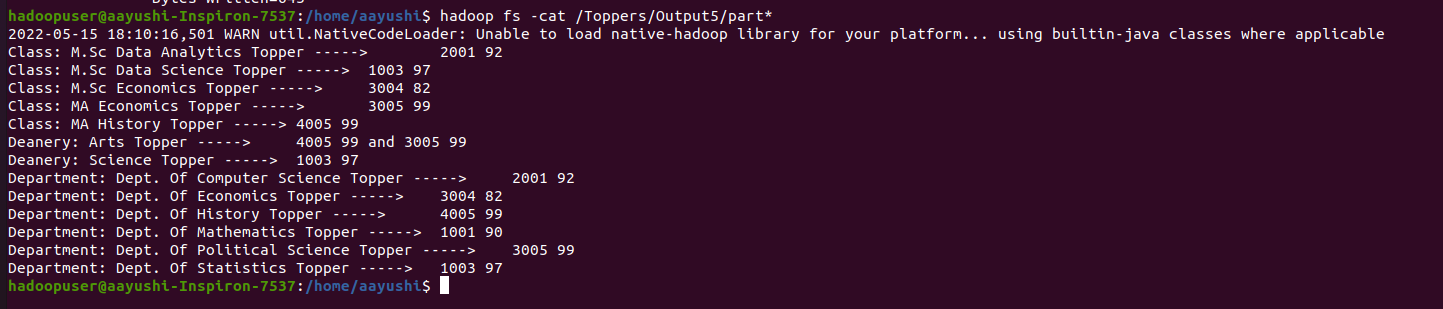
}

}

**Output 1:**

The output displays if the Topper is for the Given Class, Dept or Deanery followed with the Deanery/Dept/Class name.

The content after --------> signifies the Topper’s roll number and their respective marks.

****

**Program 2: Mapper and Reducer and Partitioner Class**

import java.io.\*;

import org.apache.hadoop.io.\*;

import org.apache.hadoop.mapreduce.\*;

import org.apache.hadoop.conf.\*;

import org.apache.hadoop.conf.\*;

import org.apache.hadoop.fs.\*;

import org.apache.hadoop.mapreduce.lib.input.\*;

import org.apache.hadoop.mapreduce.lib.output.\*;

import org.apache.hadoop.util.\*;

public class Topper {

public static class MapIt

extends Mapper<Object, Text, Text, Text>{

@Override

public void map(Object key, Text value, Context context

) throws IOException, InterruptedException {

String data[]=value.toString().split(",");

String key1 = data[0];

String dean\_deets = String.format("%s-%s",data[0],data[5]);

String dept\_deets = String.format("%s-%s",data[0],data[5]);

String class\_deets = String.format("%s-%s",data[0],data[5]);

//Send only Deanery specific details to Deanery Keys

context.write(new Text(String.format("Deanery: %s Topper ----->",data[6])), new Text(dean\_deets));

//Send only Department specific details to Deanery Keys

context.write(new Text(String.format("Department: %s Topper ----->",data[2])), new Text(dept\_deets));

//Send only Class specific details to Deanery Keys

context.write(new Text(String.format("Class: %s Topper ----->",data[3])), new Text(class\_deets));

}

}

public static class ReduceIt

extends Reducer<Text,Text,Text,Text> {

public void reduce(Text key, Iterable <Text> values,

Context context

) throws IOException, InterruptedException {

int sum=0;

String max\_roll = "";

int max\_marks = 0;

String tied\_marks = "";

for (Text rec : values)

{

//Retrieving all the calculated values from the Mapper

String [] variables = rec.toString().split("-");

String roll = variables[0];

String mark = variables[1];

if (Integer.parseInt(mark) == max\_marks)

{

tied\_marks = tied\_marks + " and " + String.format("%s %s",roll,mark); //To handle tied scores

}

else if (Integer.parseInt(mark) > max\_marks)

{

max\_roll = roll;

max\_marks = Integer.parseInt(mark);

tied\_marks = String.format("%s %s",roll,mark); //To handle tied scored

}

}

String output\_str = String.format("%s",tied\_marks);

context.write(key, new Text(output\_str));

}

}

//Partitioner class

public static class PartitionIt extends Partitioner < Object, Text >

{

@Override

public int getPartition(Object key, Text value, int numReduceTasks)

{

String[] str = key.toString().split(":");

String div = str[0];

if(numReduceTasks == 0)

{

return 0;

}

if(div.equals("Deanery"))

{

return 0;

}

else if(div.equals("Department"))

{

return 1;

}

else

{

return 2;

}

}

}

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "Practice");

job.setJarByClass(Topper.class);

job.setMapperClass(MapIt.class);

job.setPartitionerClass(PartitionIt.class);

job.setReducerClass(ReduceIt.class);

job.setNumReduceTasks(3);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

System.exit(job.waitForCompletion(true) ? 0 : 1);

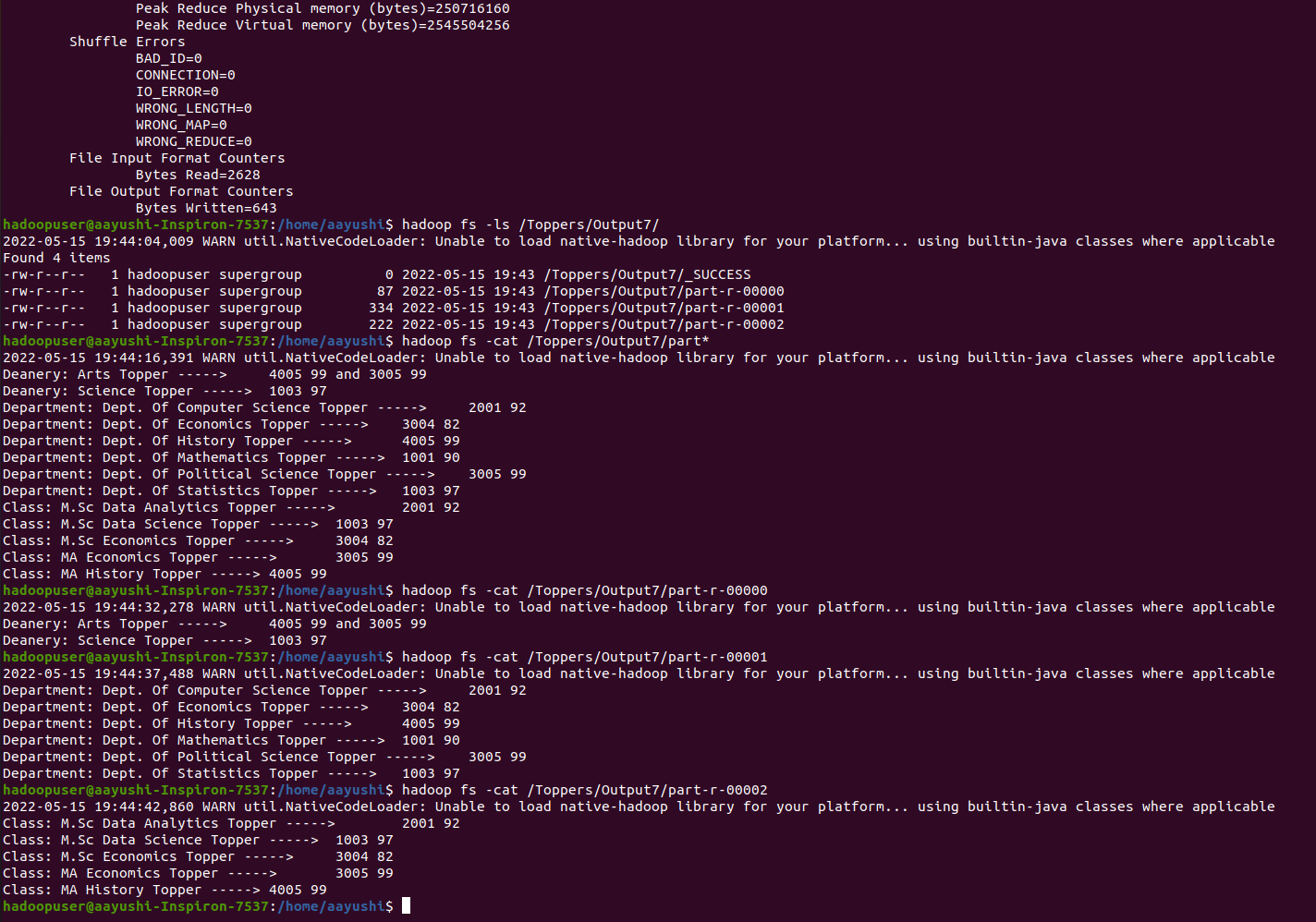
}

}

**Output 2:**

The output displays if the Topper is for the Given Class, Dept or Deanery followed with the Deanery/Dept/Class name in different Paritions of the Reducer. First reducer will return Deanery wise toppers. Second reducer will return Department wise toppers.

The content after --------> signifies the Topper’s roll number and their respective marks.



**Data:**

1001,Kelly,Dept. Of Statistics,M.Sc Data Science,Statistics,94,Science

1002,Josh,Dept. Of Statistics,M.Sc Data Science,Statistics,85,Science

1003,Hannah,Dept. Of Statistics,M.Sc Data Science,Statistics,97,Science

1004,Zack,Dept. Of Statistics,M.Sc Data Science,Statistics,84,Science

1005,Cody,Dept. Of Statistics,M.Sc Data Science,Statistics,74,Science

1001,Kelly,Dept. Of Mathematics,M.Sc Data Science,Mathematics,90,Science

1002,Josh,Dept. Of Mathematics,M.Sc Data Science,Mathematics,88,Science

1003,Hannah,Dept. Of Mathematics,M.Sc Data Science,Mathematics,71,Science

1004,Zack,Dept. Of Mathematics,M.Sc Data Science,Mathematics,82,Science

1005,Cody,Dept. Of Mathematics,M.Sc Data Science,Mathematics,77,Science

2001,Ben,Dept. Of Computer Science,M.Sc Data Analytics,Machine Learning,92,Science

2002,Jake,Dept. Of Computer Science,M.Sc Data Analytics,Machine Learning,86,Science

2003,Raven,Dept. Of Computer Science,M.Sc Data Analytics,Machine Learning,79,Science

2004,Gordon,Dept. Of Computer Science,M.Sc Data Analytics,Machine Learning,72,Science

2005,Ramsay,Dept. Of Computer Science,M.Sc Data Analytics,Machine Learning,67,Science

2001,Ben,Dept. Of Computer Science,M.Sc Data Analytics,Data Mining,67,Science

2002,Jake,Dept. Of Computer Science,M.Sc Data Analytics,Data Mining,79,Science

2003,Raven,Dept. Of Computer Science,M.Sc Data Analytics,Data Mining,70,Science

2004,Gordon,Dept. Of Computer Science,M.Sc Data Analytics,Data Mining,84,Science

2005,Ramsay,Dept. Of Computer Science,M.Sc Data Analytics,Data Mining,76,Science

3001,Nike,Dept. Of Economics,M.Sc Economics,Microeconomics,72,Arts

3002,Athena,Dept. Of Economics,M.Sc Economics,Microeconomics,68,Arts

3003,Kratos,Dept. Of Economics,M.Sc Economics,Microeconomics,73,Arts

3004,Ramirez,Dept. Of Economics,M.Sc Economics,Microeconomics,82,Arts

3005,Medusa,Dept. Of Economics,M.Sc Economics,Microeconomics,77,Arts

3001,Nike,Dept. Of Political Science,MA Economics,International Politics,80,Arts

3002,Athena,Dept. Of Political Science,MA Economics,International Politics,81,Arts

3003,Kratos,Dept. Of Political Science,MA Economics,International Politics,61,Arts

3004,Rami,Dept. Of Political Science,MA Economics,International Politics,62,Arts

3005,Medusa,Dept. Of Political Science,MA Economics,International Politics,99,Arts

4001,Adam,Dept. Of History,MA History,History of Politics,88,Arts

4002,Kevin,Dept. Of History,MA History,History of Politics,84,Arts

4003,Steve,Dept. Of History,MA History,History of Politics,65,Arts

4004,Pam,Dept. Of History,MA History,History of Politics,53,Arts

4005,Lola,Dept. Of History,MA History,History of Politics,99,Arts