**Session 6b**

**12/05/2021**

**Problem Statement:**

**Program 3: Use the Hadoop framework to write a custom MapReduce program to perform word count operation on a custom data set.**

**Word Count Program:**

package my.wordcount.demo;

import java.io.IOException;

import java.util.\*;

import org.apache.hadoop.fs.Path;

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

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

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

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

public class WordCount {

//MAPPER CODE

public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {

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

private Text word = new Text();

public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {

String line = value.toString();

StringTokenizer tokenizer = new StringTokenizer(line);

while (tokenizer.hasMoreTokens()) {

word.set(tokenizer.nextToken());

output.collect(word, one); //{Mary, 1} //{had, 1} {little, 1} {lamb,1} {little, 1}

}

}

}

//REDUCER CODE

public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {

public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException { //{little: {1,1}}

int sum = 0;

while (values.hasNext()) {

sum += values.next().get(); // sum = 0 , sum <- sum + 1 < 1 , sum = 1+1 = > {key:little, sum=2}

}

output.collect(key, new IntWritable(sum));

}

}

//DRIVER CODE

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

JobConf conf = new JobConf(WordCount.class);

conf.setJobName("wordcount");

conf.setOutputKeyClass(Text.class);

conf.setOutputValueClass(IntWritable.class);

conf.setMapperClass(Map.class);

conf.setCombinerClass(Reduce.class);

conf.setReducerClass(Reduce.class);

conf.setInputFormat(TextInputFormat.class);

conf.setOutputFormat(TextOutputFormat.class); // hadoop jar jarname classpath inputfolder outputfolder

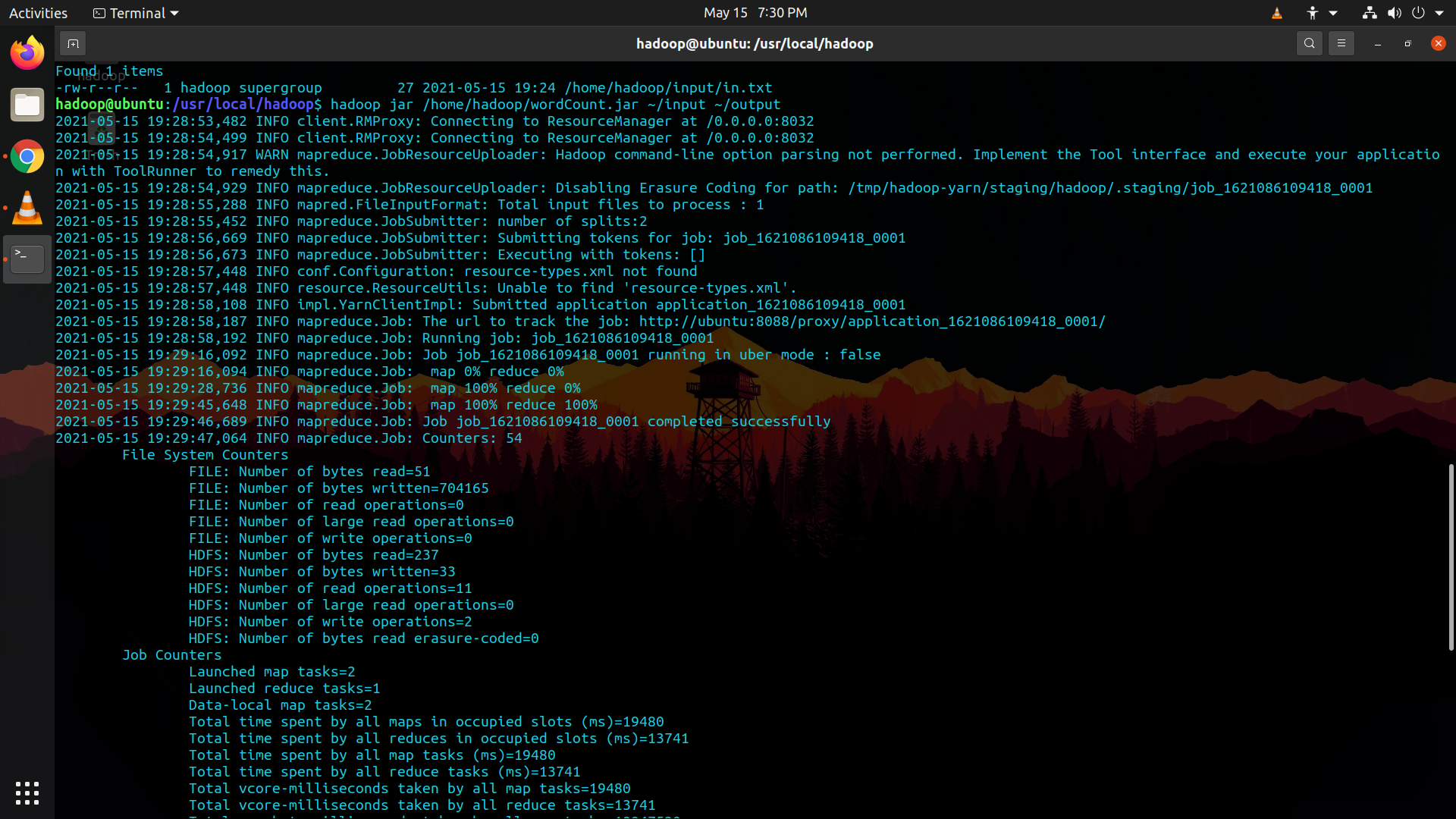
FileInputFormat.setInputPaths(conf, new Path(args[0]));

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

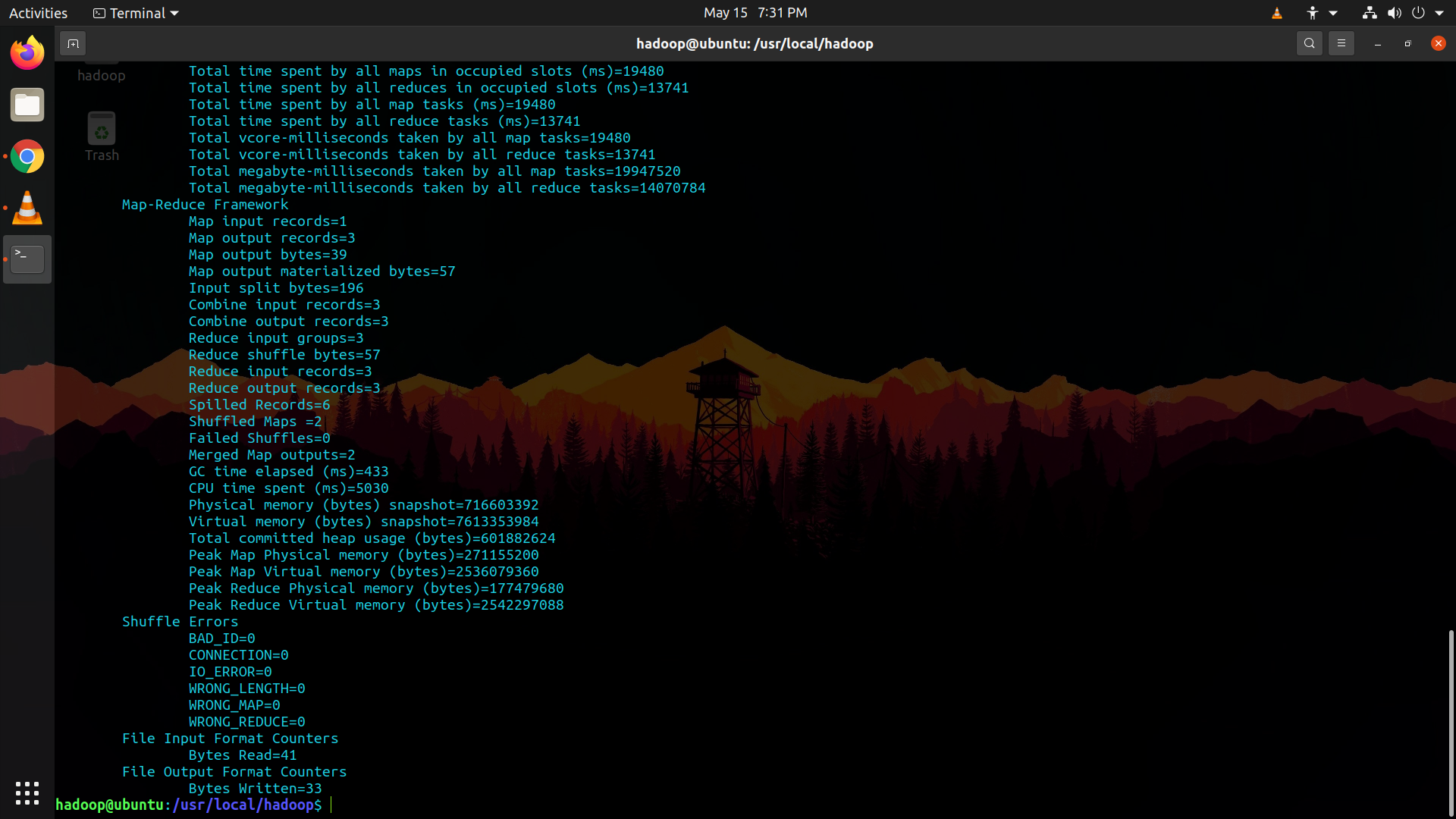
JobClient.runJob(conf);

}

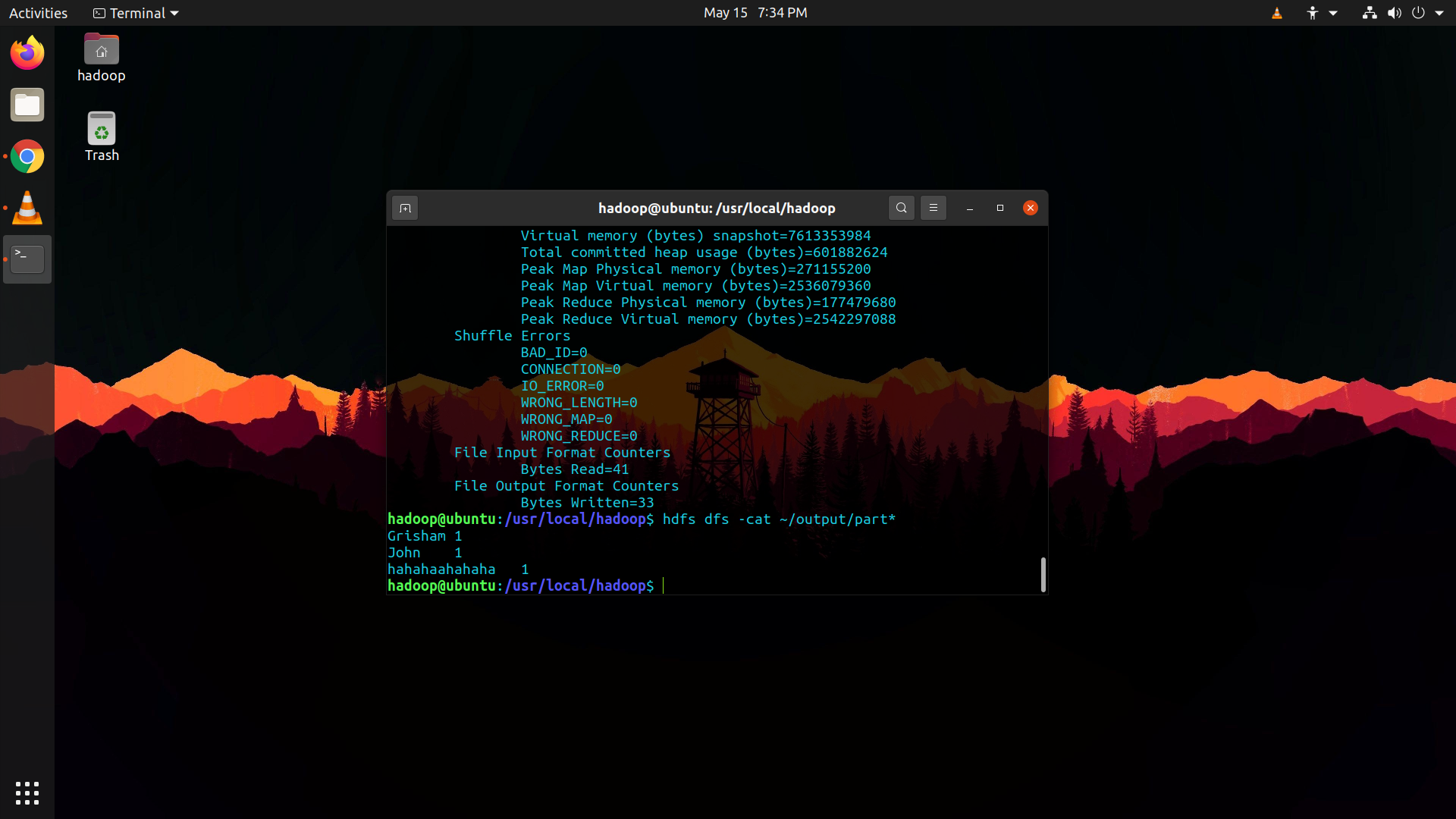
}

**Output:**

**$ hadoop jar /home/hadoop/wordCount.jar ~/input ~/output**

****

**$ hdfs dfs -cat ~/output/part\***

****

**Program 4: Use the Hadoop framework to write a MapReduce program to read a .csv file into a single node Hadoop cluster containing following fields**

**Sl. No.**

**CARD name**

**UserName**

**Amount withdrawn**

**Implement the following,**

**1. Count the Number of transactions done by each user**

**Write the following program.**

package studentMarks;

import java.io.IOException;

import java.util.\*;

import org.apache.hadoop.fs.Path;

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

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

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

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

public class TransCount {

//MAPPER CODE

public static class Map extends MapReduceBase implements Mapper <LongWritable, Text, Text, IntWritable> {

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

private Text word = new Text();

public void map(LongWritable key, Text value, OutputCollector <Text,IntWritable> output, Reporter reporter) throws IOException {

String myvalue = value.toString();

String[] transtokens = myvalue.split(",");

output.collect(new Text(transtokens[2]), one);

}

}

//REDUCER CODE

public static class Reduce extends MapReduceBase implements Reducer <Text, IntWritable, Text, IntWritable> {

public void reduce(Text key, Iterator <IntWritable> values, OutputCollector <Text, IntWritable> output, Reporter reporter) throws IOException {

//{little: {1,1}}

int transcount = 0;

while (values.hasNext()) {

transcount += values.next().get();

}

output.collect(key, new IntWritable(transcount));

}

}

//DRIVER CODE

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

JobConf conf = new JobConf(TransCount.class);

conf.setJobName("TransCount");

conf.setOutputKeyClass(Text.class);

conf.setOutputValueClass(IntWritable.class);

conf.setMapperClass(Map.class);

conf.setCombinerClass(Reduce.class);

conf.setReducerClass(Reduce.class);

conf.setInputFormat(TextInputFormat.class);

conf.setOutputFormat(TextOutputFormat.class);

FileInputFormat.setInputPaths(conf, new Path(args[0]));

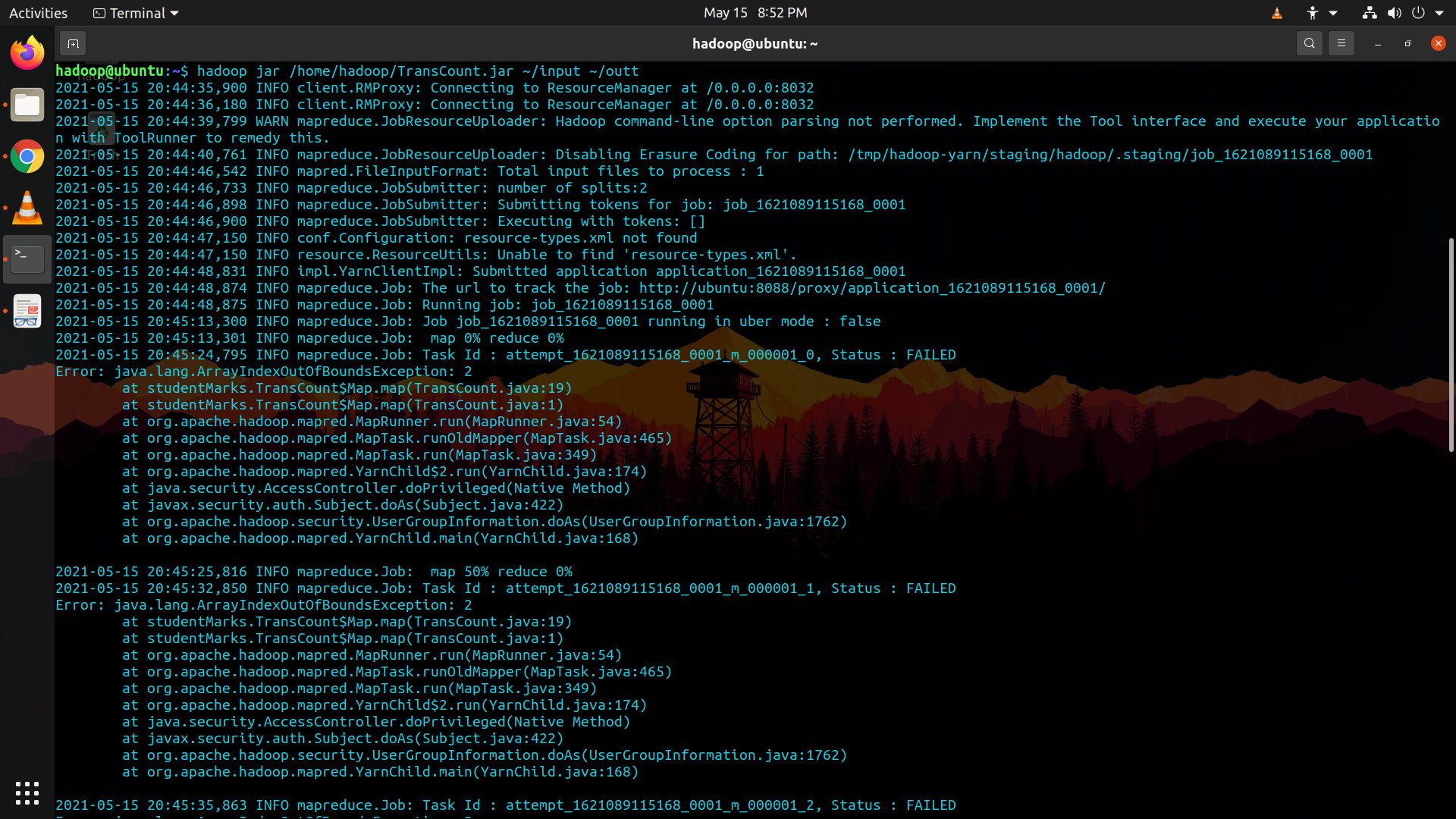
FileOutputFormat.setOutputPath(conf, new Path(args[1]));

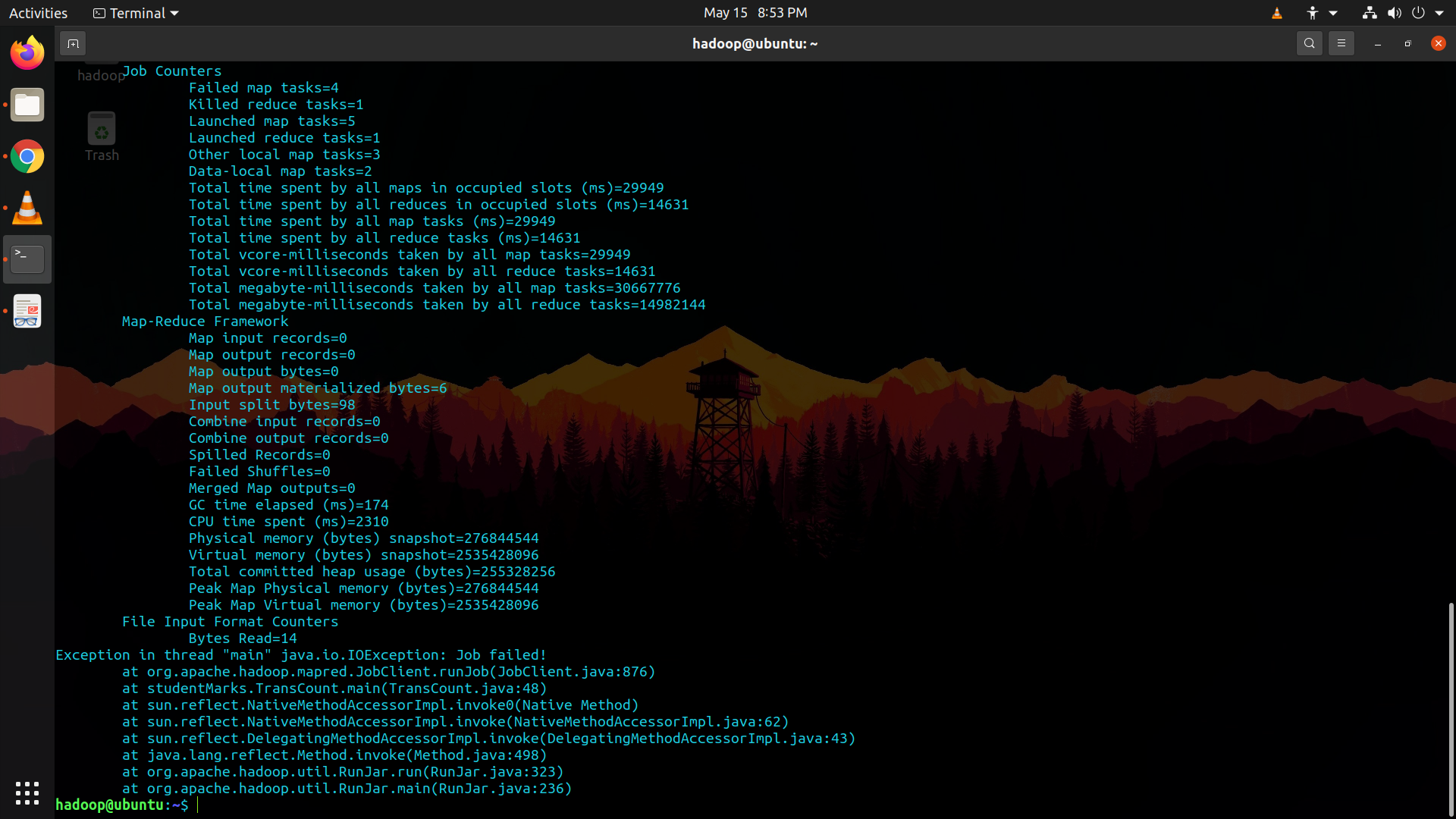
JobClient.runJob(conf);

}

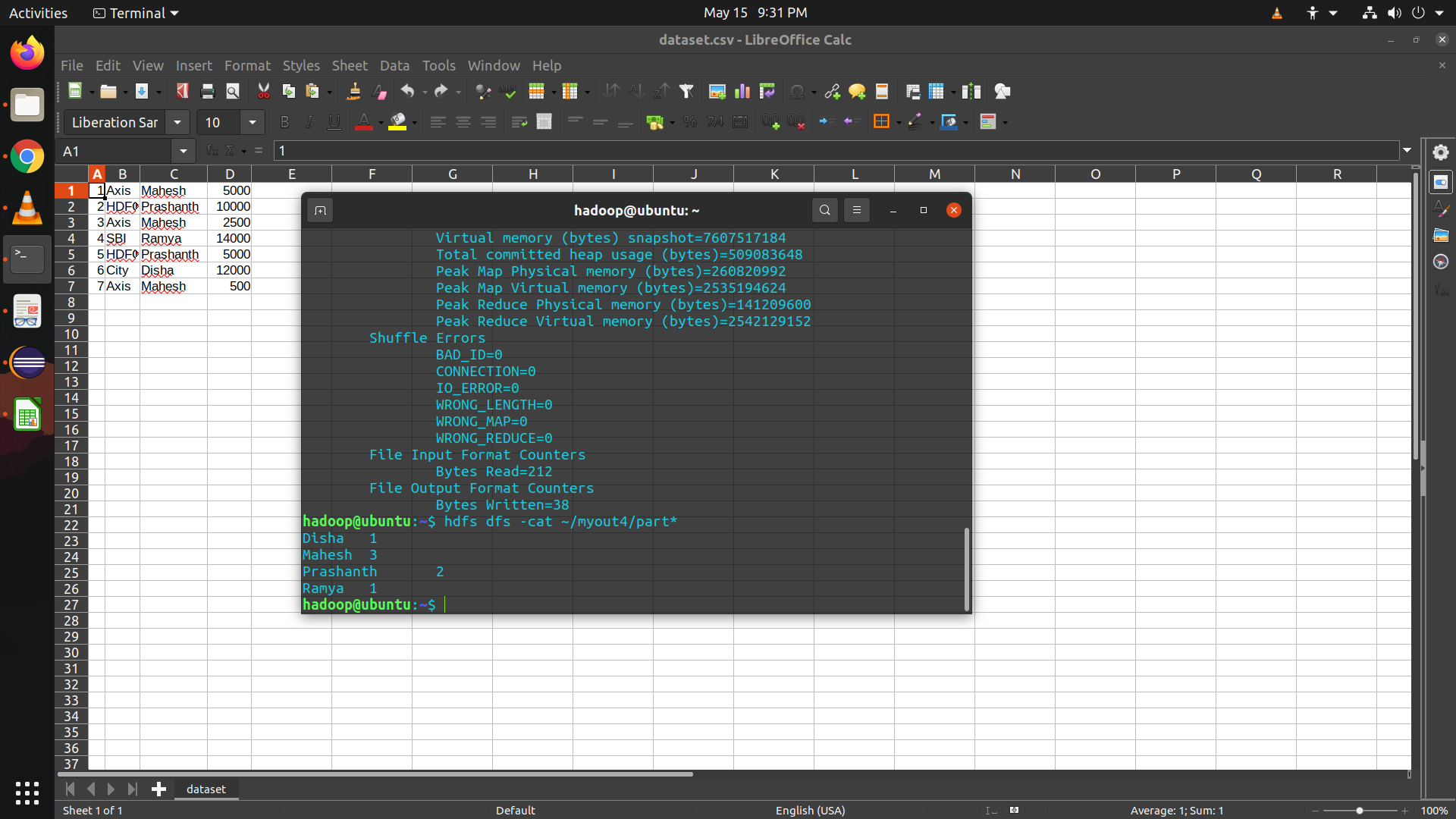
}

**Output: $ hadoop jar /home/hadoop/TransCount.jar ~/input ~/output**

****

****

**$ hdfs dfs -cat ~/myout4/part\***

****

**2. Find the total amount of money transacted by each user**

package studentMarks;

import java.io.IOException;

import java.util.\*;

import org.apache.hadoop.fs.Path;

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

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

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

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

public class TotalAmount {

//MAPPER CODE

public static class Map extends MapReduceBase implements Mapper <LongWritable, Text, Text, IntWritable> {

//private final static IntWritable one = new IntWritable(1);

private Text word = new Text();

public void map(LongWritable key, Text value, OutputCollector <Text, IntWritable> output, Reporter reporter) throws IOException {

String myvalue = value.toString();

String[] TransactionTokens = myvalue.split(",");

output.collect(new Text(TransactionTokens[2]), new IntWritable(Integer.parseInt(TransactionTokens[3])));

}

}

//REDUCER CODE

public static class Reduce extends MapReduceBase implements Reducer

<Text, IntWritable, Text, IntWritable> {

public void reduce(Text key, Iterator <IntWritable> values,

OutputCollector <Text, IntWritable> output, Reporter reporter) throws IOException {

//{little: {1,1}}

int transactionAmountCount = 0;

while (values.hasNext()) {

transactionAmountCount += values.next().get();

}

output.collect(key, new IntWritable(transactionAmountCount));

}

}

//DRIVER CODE

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

JobConf conf = new JobConf(TotalAmount.class);

conf.setJobName("Counting the total amount");

conf.setOutputKeyClass(Text.class);

conf.setOutputValueClass(IntWritable.class);

conf.setMapperClass(Map.class);

conf.setCombinerClass(Reduce.class);

conf.setReducerClass(Reduce.class);

conf.setInputFormat(TextInputFormat.class);

conf.setOutputFormat(TextOutputFormat.class);

FileInputFormat.setInputPaths(conf, new Path(args[0]));

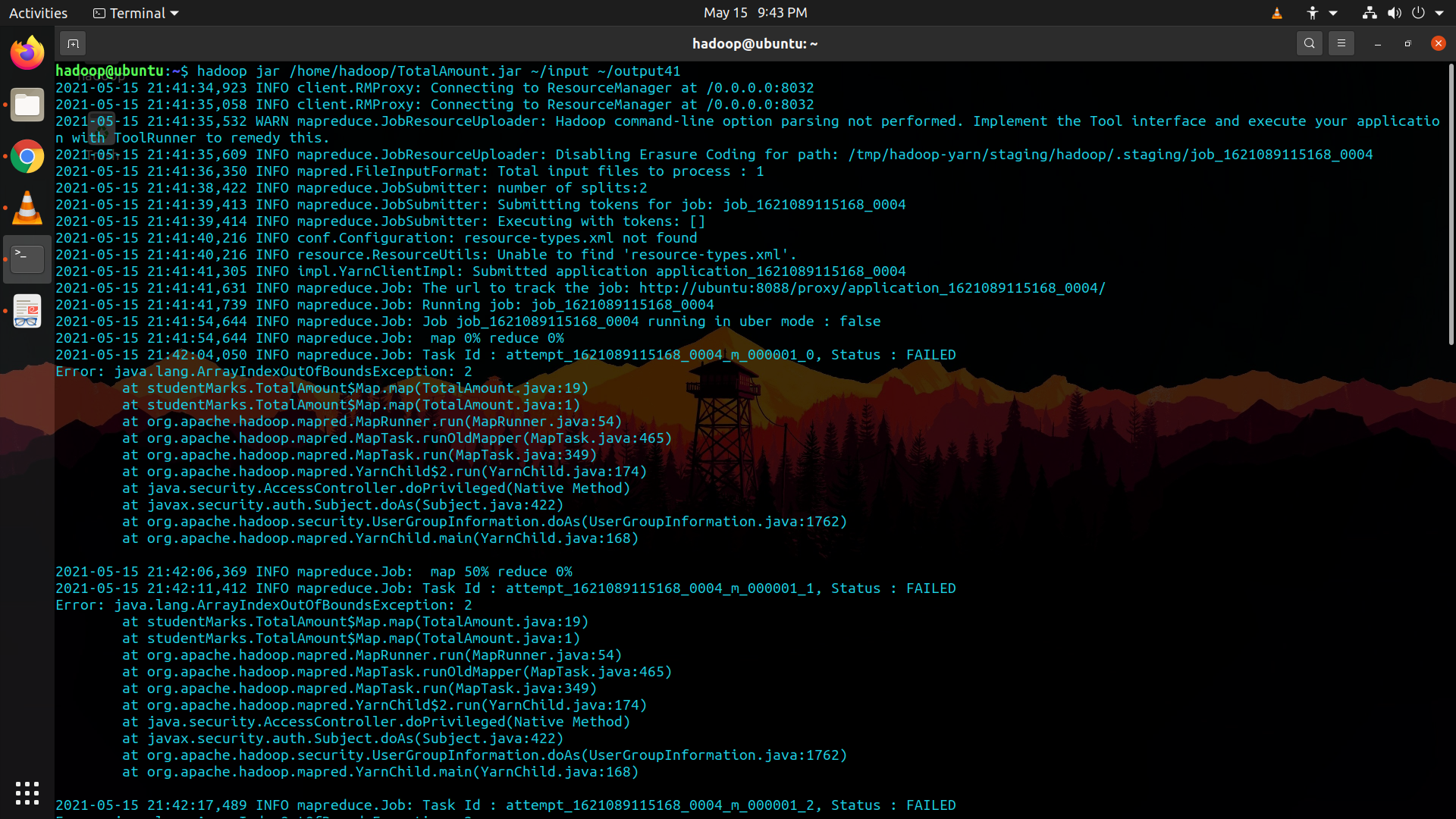
FileOutputFormat.setOutputPath(conf, new Path(args[1]));

JobClient.runJob(conf);

}

}

**Output: hadoop jar /home/hadoop/TotalAmount.jar ~/input ~/output41**

****

****