BIG DATA LAB-07

**>PROGRAM-03**

**Use the Hadoop Framework to write a custom MapReduce to perform wordcount operation on a custom dataset.**

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.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 WordCount {

public static class TokenizerMapper

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

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

private Text word = new Text();

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

) throws IOException, InterruptedException {

StringTokenizer itr = new StringTokenizer(value.toString());

DEPARTMENT OF ISE, NMIT, BANGALORE 29

BIG DATA LAB

while (itr.hasMoreTokens()) {

word.set(itr.nextToken());

context.write(word, one);

}

}

}

public static class IntSumReducer

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

private IntWritable result = new IntWritable();

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

Context context

) throws IOException, InterruptedException {

int sum = 0;

for (IntWritable val : values) {

sum += val.get();

}

result.set(sum);

context.write(key, result);

}

}

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "word count");

DEPARTMENT OF ISE, NMIT, BANGALORE 30

BIG DATA LAB

job.setJarByClass(WordCount.class);

job.setMapperClass(TokenizerMapper.class);

job.setCombinerClass(IntSumReducer.class);

job.setReducerClass(IntSumReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

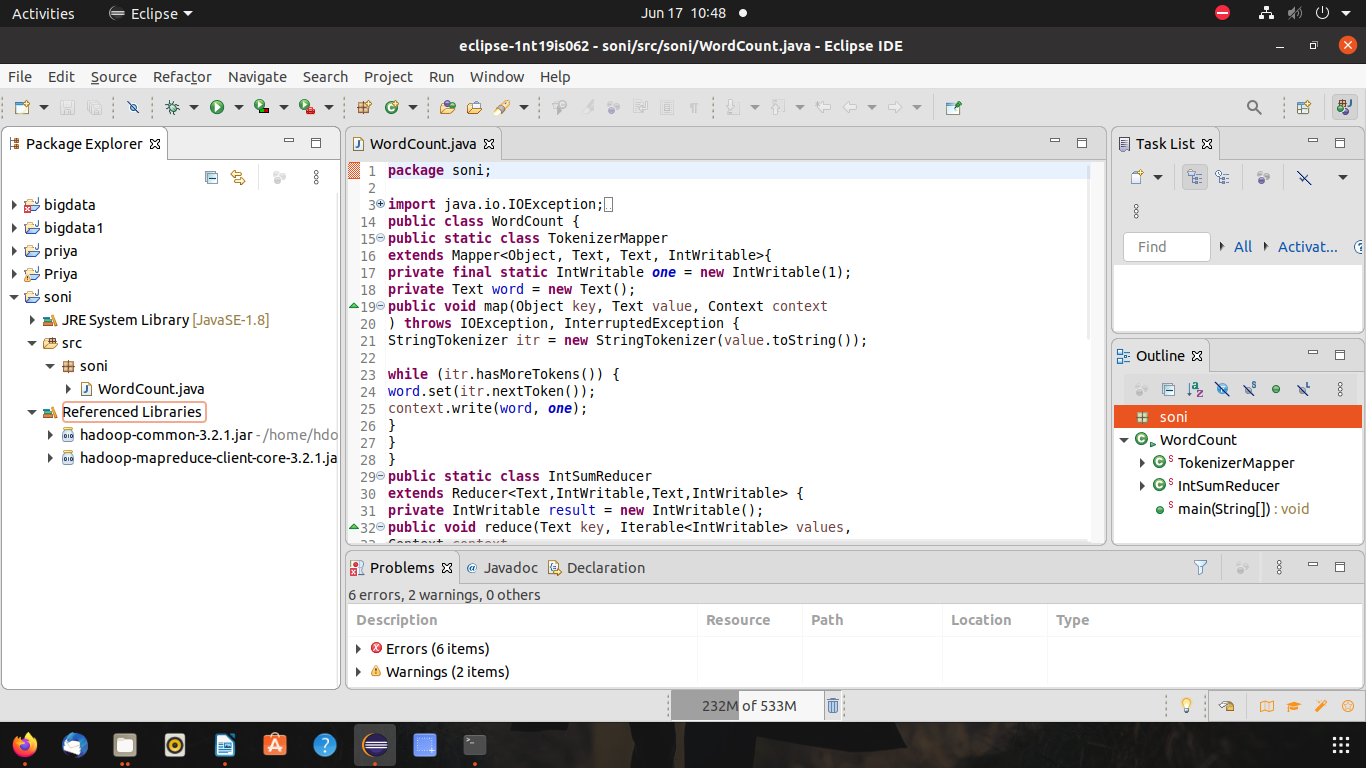
FileInputFormat.addInputPath(job, new Path(args[0]));

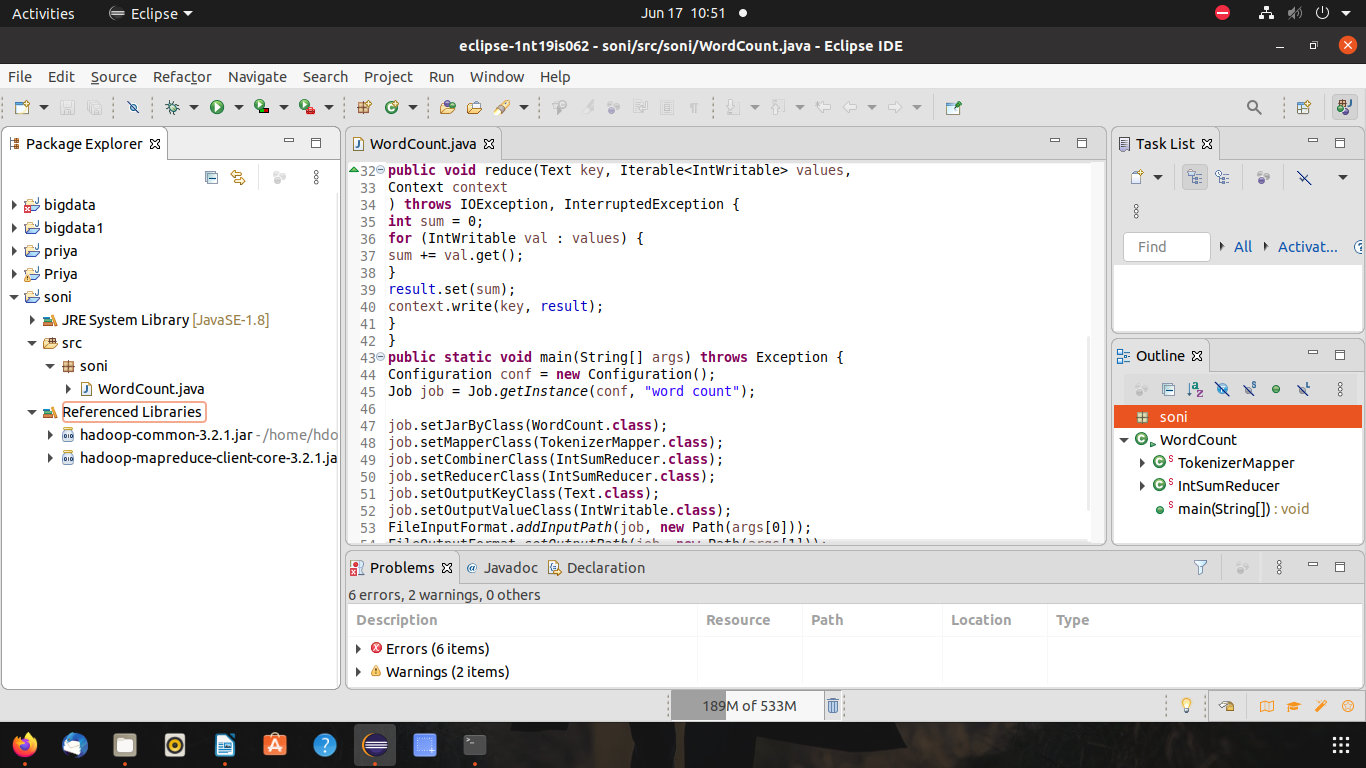
FileOutputFormat.setOutputPath(job, new Path(args[1]));

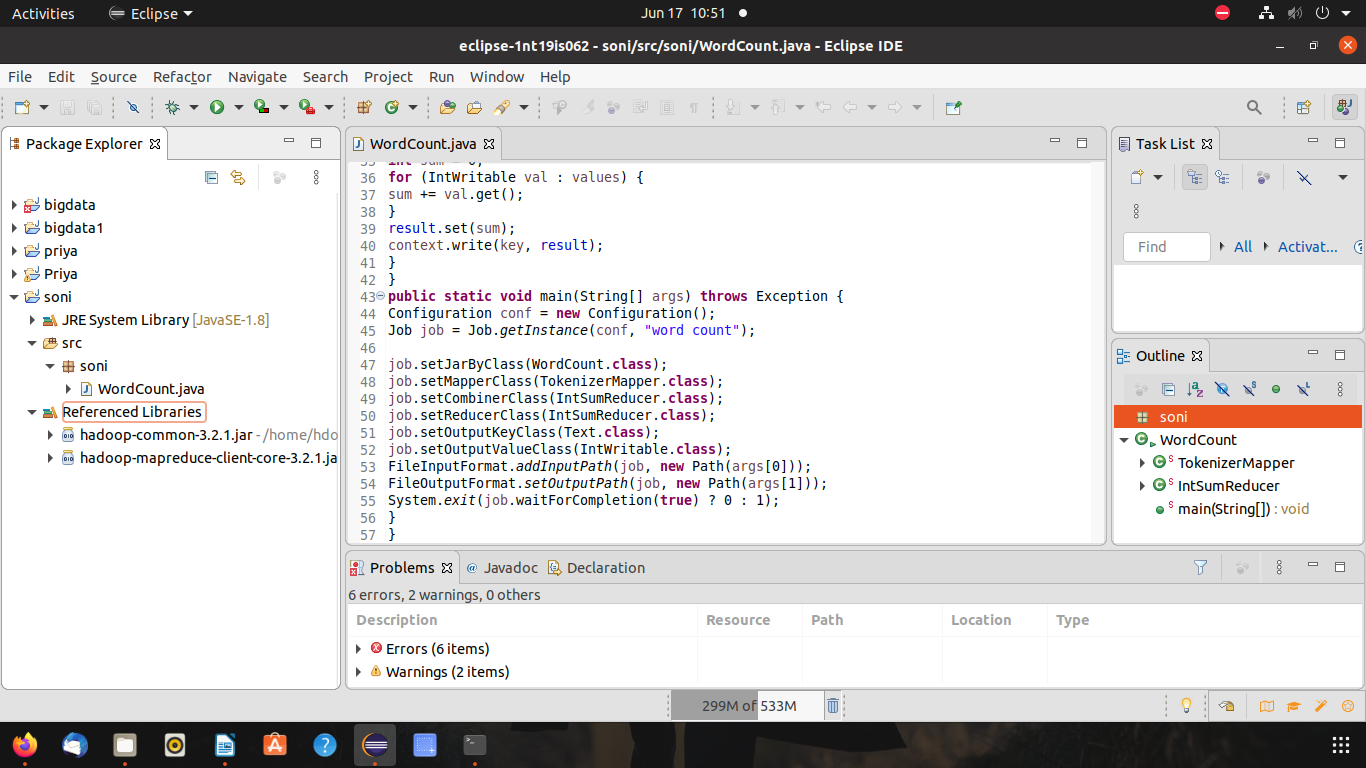
System.exit(job.waitForCompletion(true) ? 0 : 1);

}

}





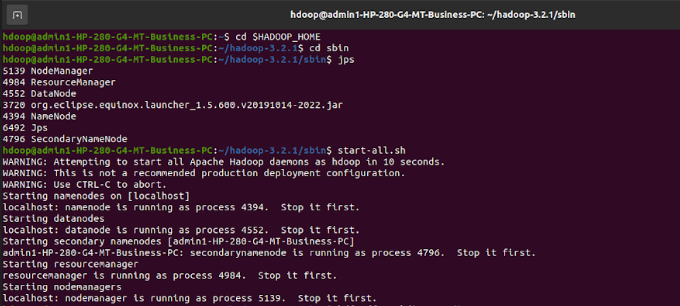


CREATED A JAR FILE NAMED SONI



COMMANDS ON TERMINAL

hdoop@admin1-HP-280-G4-MT-Business-PC:~$ cd $HADOOP\_HOME  
hdoop@admin1-HP-280-G4-MT-Business-PC:~/hadoop-3.2.1$ cd sbin  
hdoop@admin1-HP-280-G4-MT-Business-PC:~/hadoop-3.2.1/sbin$ jps



CREATING A INPUT DIRECTORY

hdfs dfs -mkdir -p ~/input



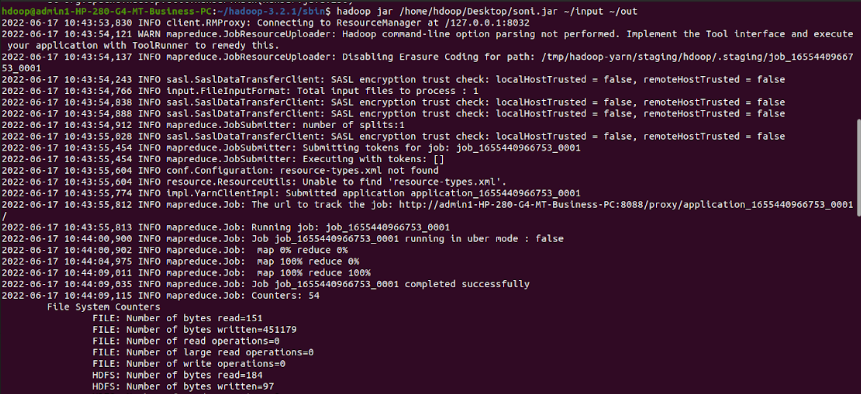
APPEND THE CONTENT OF THE FILE

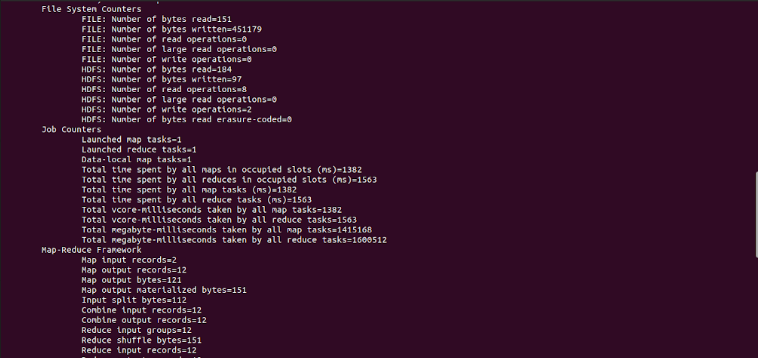
hdfs dfs -appendToFile - ~/input/test.txt

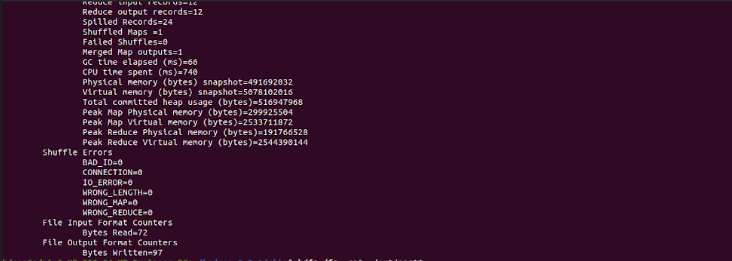


RUNNING THE JOB BY PASSING IN THE INPUT AND OUTPUT DIRECTORIES CREATED EARLIER

hadoop jar /home/hdoop/Desktop/son.jar ~/input ~/out







CAT THE FILE TO GET THE OUTPUT

hdfs dfs -cat ~/out/part\*

