Lab 2

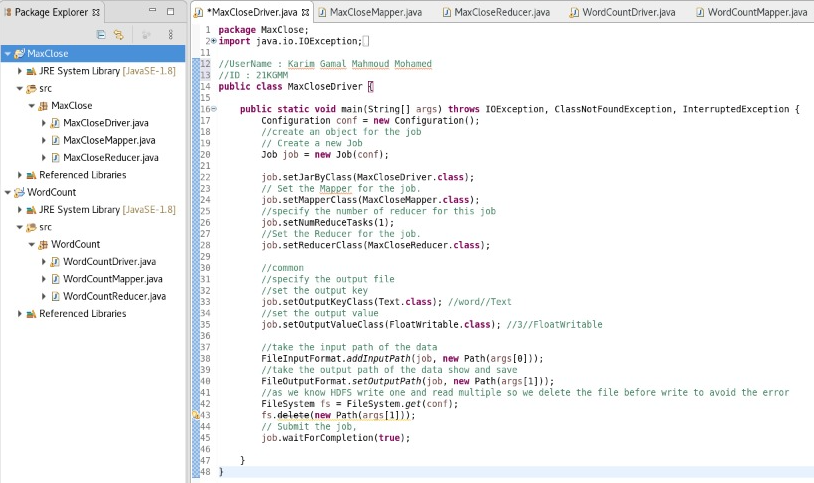
Name: Karim Gamal Mahmoud Mohamed

ID: 21KGMM

1. The jar file

The Jar file is uploaded with the Word file.

2-a) java source code for Driver :



package MaxClose;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.FileSystem;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.FloatWritable;

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.output.FileOutputFormat;

//UserName : Karim Gamal Mahmoud Mohamed

//ID : 21KGMM

public class MaxCloseDriver {

public static void main(String[] args) throws IOException, ClassNotFoundException, InterruptedException {

Configuration conf = new Configuration();

//create an object for the job

// Create a new Job

Job job = new Job(conf);

job.setJarByClass(MaxCloseDriver.class);

// Set the Mapper for the job.

job.setMapperClass(MaxCloseMapper.class);

//specify the number of reducer for this job

job.setNumReduceTasks(1);

//Set the Reducer for the job.

job.setReducerClass(MaxCloseReducer.class);

//common

//specify the output file

//set the output key

job.setOutputKeyClass(Text.class); //word//Text

//set the output value

job.setOutputValueClass(FloatWritable.class); //3//FloatWritable

//take the input path of the data

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

//take the output path of the data show and save

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

//as we know HDFS write one and read multiple so we delete the file before write to avoid the error

FileSystem fs = FileSystem.get(conf);

fs.delete(new Path(args[1]));

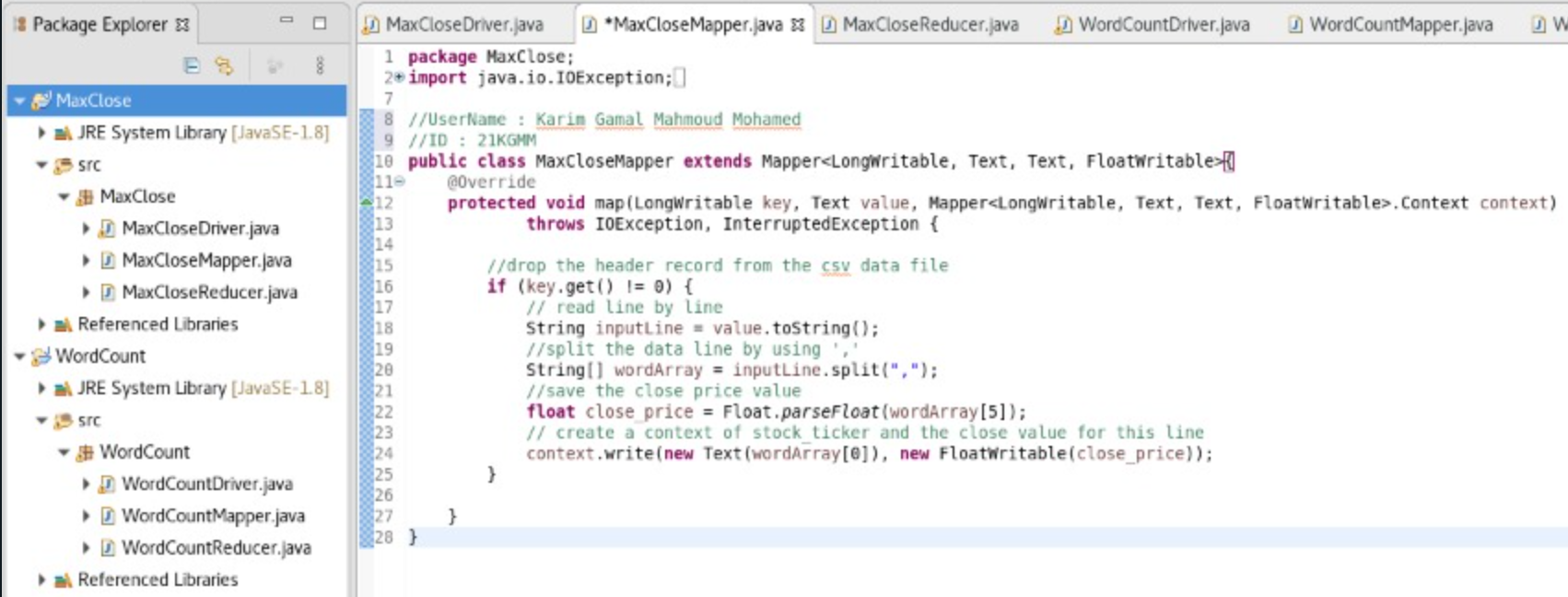
// Submit the job,

job.waitForCompletion(true);

}

}

2-b) java source code for Mapper :



package MaxClose;

import java.io.IOException;

import org.apache.hadoop.io.FloatWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

//UserName : Karim Gamal Mahmoud Mohamed

//ID : 21KGMM

public class MaxCloseMapper extends Mapper<LongWritable, Text, Text, FloatWritable>{

@Override

protected void map(LongWritable key, Text value, Mapper<LongWritable, Text, Text, FloatWritable>.Context context)

throws IOException, InterruptedException {

//drop the headr record from the csv data file

if (key.get() != 0) {

// read line by line

String inputLine = value.toString();

//split the data line by using ','

String[] wordArray = inputLine.split(",");

//save the close price value

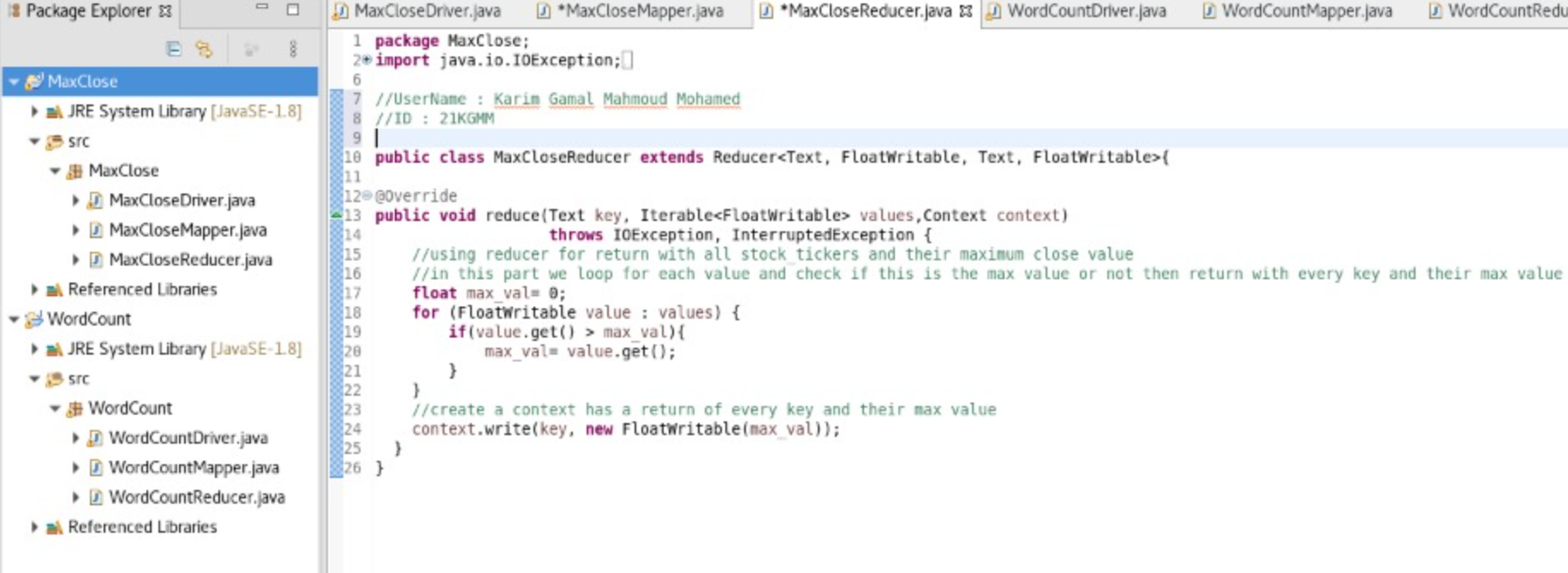
float close\_price = Float.parseFloat(wordArray[5]);

// create a context of stock\_ticker and the close value for this line

context.write(new Text(wordArray[0]), new FloatWritable(close\_price));

}}}

2-c) java source code for Reducer:



package MaxClose;

import java.io.IOException;

import org.apache.hadoop.io.FloatWritable;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.io.Text;

//UserName : Karim Gamal Mahmoud Mohamed

//ID : 21KGMM

public class MaxCloseReducer extends Reducer<Text, FloatWritable, Text, FloatWritable>{

@Override

public void reduce(Text key, Iterable<FloatWritable> values,Context context)

throws IOException, InterruptedException {

//using reducer for return with all stock\_tickers and their maximum close value

//in this part we loop for each value and check if this is the max value or not then return with every key and their max value

float max\_val= 0;

for (FloatWritable value : values) {

if(value.get() > max\_val){

max\_val= value.get();

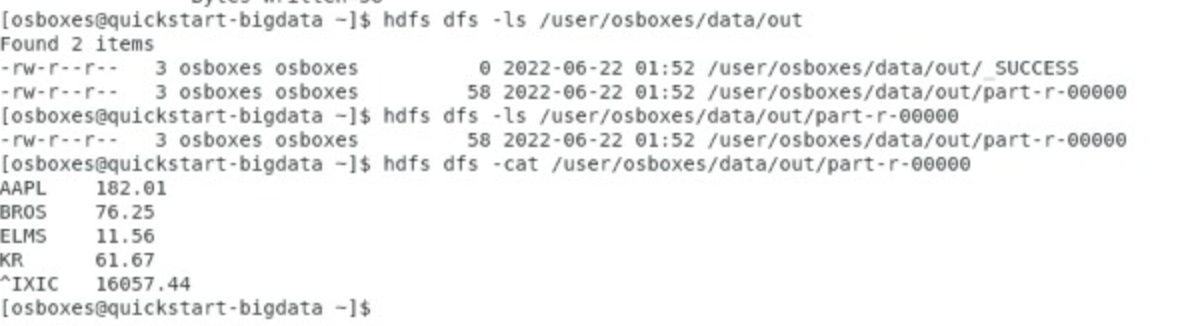
}}

//create a context has a return of every key and their max value

context.write(key, new FloatWritable(max\_val));

}}

3. The stocks.txt data file



-Stocks.txt data file :

AAPL 182.01

BROS 76.25

ELMS 11.56

KR 61.67

^IXIC 16057.44

This is the final results after the Mapreduce program is completed :

