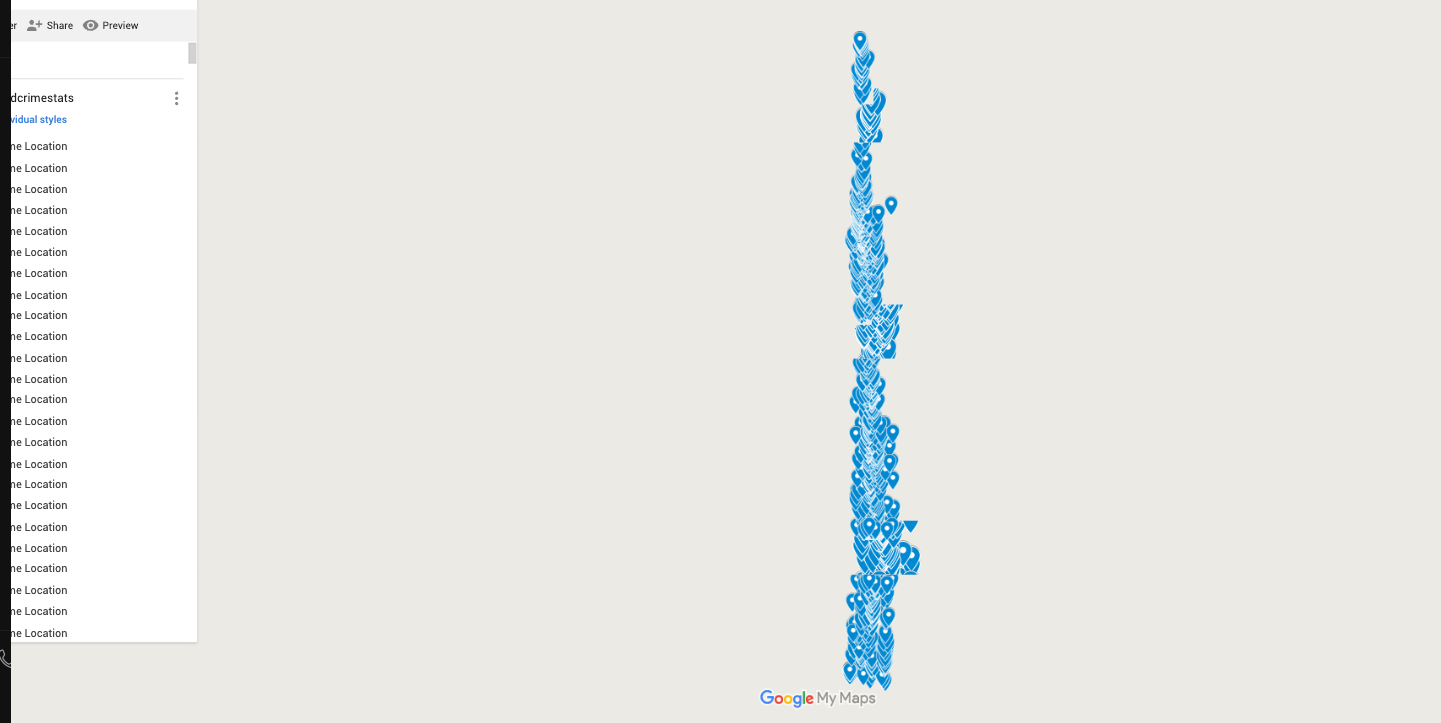
[zhengzen@andrew.cmu.edu](mailto:zhengzen@andrew.cmu.edu)

Zheng Zeng

Project 5

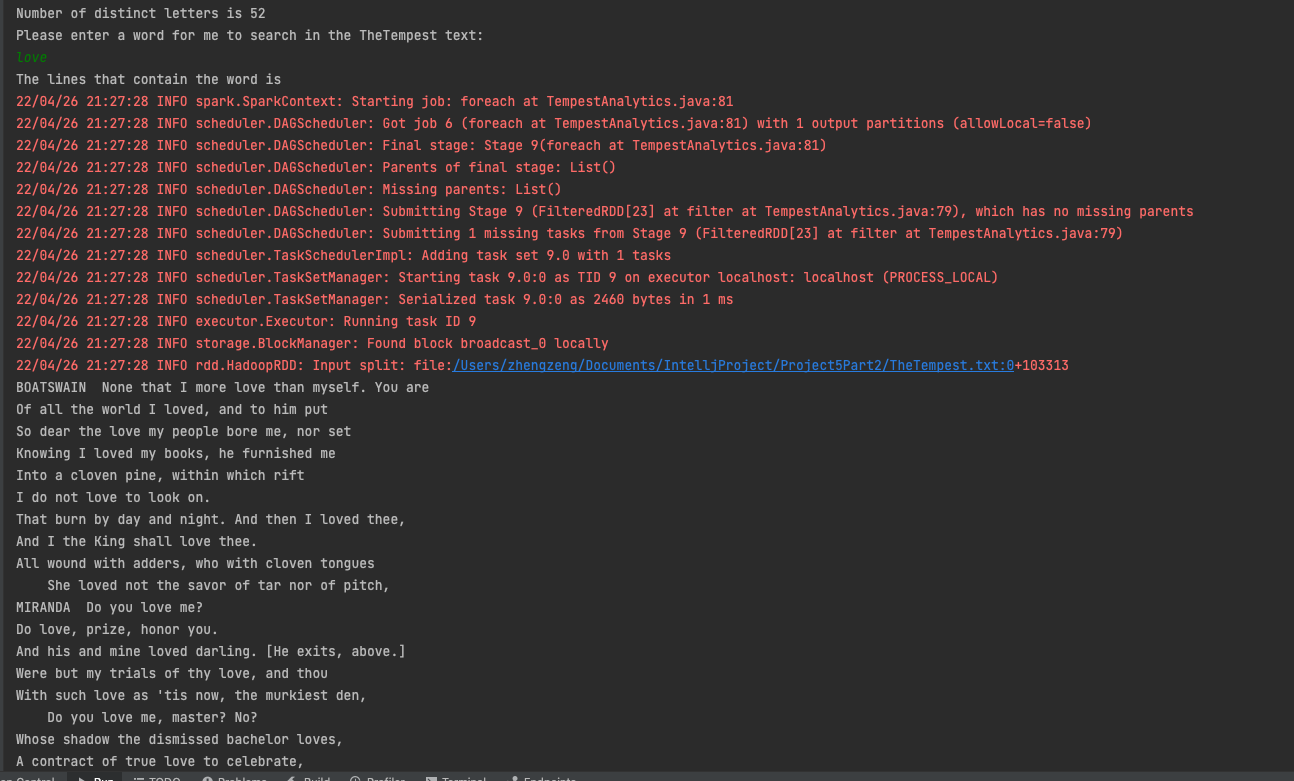
1. **Look at the screen shot of Google Earth from Part 1, Task 7.**

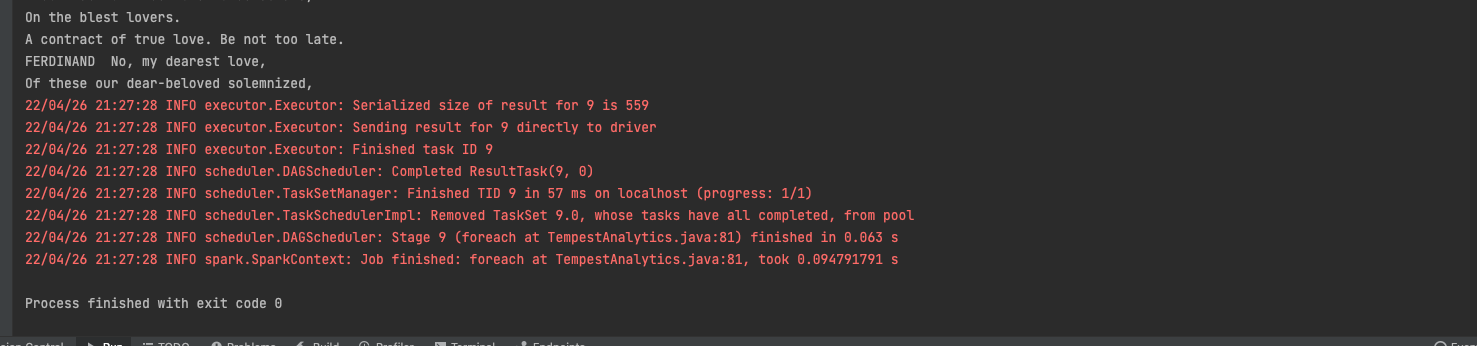


1. **Spark program from Part 2, Task 6. This also includes codes for Task0 to Task5.**

*/\*\*  
 \* Email: zhengzen@andrew.cmu.edu Author: Zheng Zeng This program is for DSIM Project 5. Hits Run on  
 \* the main method then the codes will run.  
 \*/*package cmu.edu.zhengzen;  
  
import java.util.Arrays;  
import org.apache.commons.lang.StringUtils;  
import org.apache.spark.SparkConf;  
import org.apache.spark.api.java.JavaRDD;  
import org.apache.spark.api.java.JavaSparkContext;  
  
import java.util.Scanner;  
  
public class TempestAnalytics {  
  
 public static void main(String[] args) {  
 // set file path  
 String filepath = "TheTempest.txt";  
 // start a spark session  
 SparkConf sparkConf = new SparkConf().setMaster("local").setAppName("Project 5");  
  
 JavaSparkContext sparkContext = new JavaSparkContext(sparkConf);  
 // read in the data  
 JavaRDD<String> input = sparkContext.textFile(filepath);  
  
 */\*\* Task 0 : count number of lines. \*/* long countline = input.count();  
 System.*out*.println("Number of lines in this article is " + (int) countline);  
  
 */\*\* Task 1: count number of words. \*/* // get rid of empty line  
 JavaRDD<String> not\_empty\_lines = input.filter(line -> !StringUtils.*isEmpty*(line));  
 // split by using symbols that are not characters  
 JavaRDD<String> wordlists =  
 not\_empty\_lines.flatMap(content -> Arrays.*asList*(content.split("[^a-zA-Z]+")));  
 long wordcount = wordlists.count();  
 System.*out*.println("Number of words in this document is " + (int) wordcount);  
  
 */\*\* Task 2 : display the number of distinct words in The Tempest. \*/* JavaRDD<String> dis\_words = wordlists.distinct();  
 long dis\_wc = dis\_words.count();  
 System.*out*.println("Number of distinct words in this document is " + (int) dis\_wc);  
 */\*\* Task 3: find the number of symbols \*/* JavaRDD<String> symbollists = not\_empty\_lines.flatMap(line -> Arrays.*asList*(line.split("")));  
 long symbolcount = symbollists.count();  
 System.*out*.println("Number of symbols in this document is " + (int) symbolcount);  
 */\*\* Task 4: count the number of distinct symbols \*/* JavaRDD<String> dis\_symbols = symbollists.distinct();  
 long dis\_sym\_count = dis\_symbols.count();  
 System.*out*.println("Number of distinct symbols in this document is " + (int) dis\_sym\_count);  
  
 */\*\* Task 5: find the number of distinct letters \*/* JavaRDD<String> distinct\_letter\_list =  
 not\_empty\_lines  
 .flatMap(content -> Arrays.*asList*(content.split("[^a-zA-Z]+")))  
 .flatMap(word -> Arrays.*asList*(word.split("")))  
 .distinct();  
 long letter\_count = distinct\_letter\_list.count();  
 System.*out*.println("Number of distinct letters is " + (int) letter\_count);  
  
 */\*\* Task 6: Interaction, finding lines that contains a certain word. Case sensitive \*/* Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Please enter a word for me to search in the TheTempest text: ");  
 String word = sc.next();  
 JavaRDD<String> lines\_including\_word =  
 not\_empty\_lines  
 .flatMap(  
 line -> {  
 // if the line contains the word  
 if (line.contains(word)) {  
 // return the line  
 return Arrays.*asList*(line);  
 } else {  
 // if the line does not contain the word  
 return Arrays.*asList*(""); // returns an empty string  
 }  
 })  
 .filter(line -> !line.equals("")); // filter out empty string  
 System.*out*.println("The lines that contain the word is ");  
 lines\_including\_word.foreach(line -> System.*out*.println(line));  
 }  
}

1. **Look at the output screenshot of Part 2, Task 6. This is the Part 2 execution.**





1. **Look at your Heinz cluster user ID (studentxxx). This is labelled and in the pdf.**

Student246

1. **Look at your Heinz cluster password. This is also labelled and in the pdf.**

4pril123!