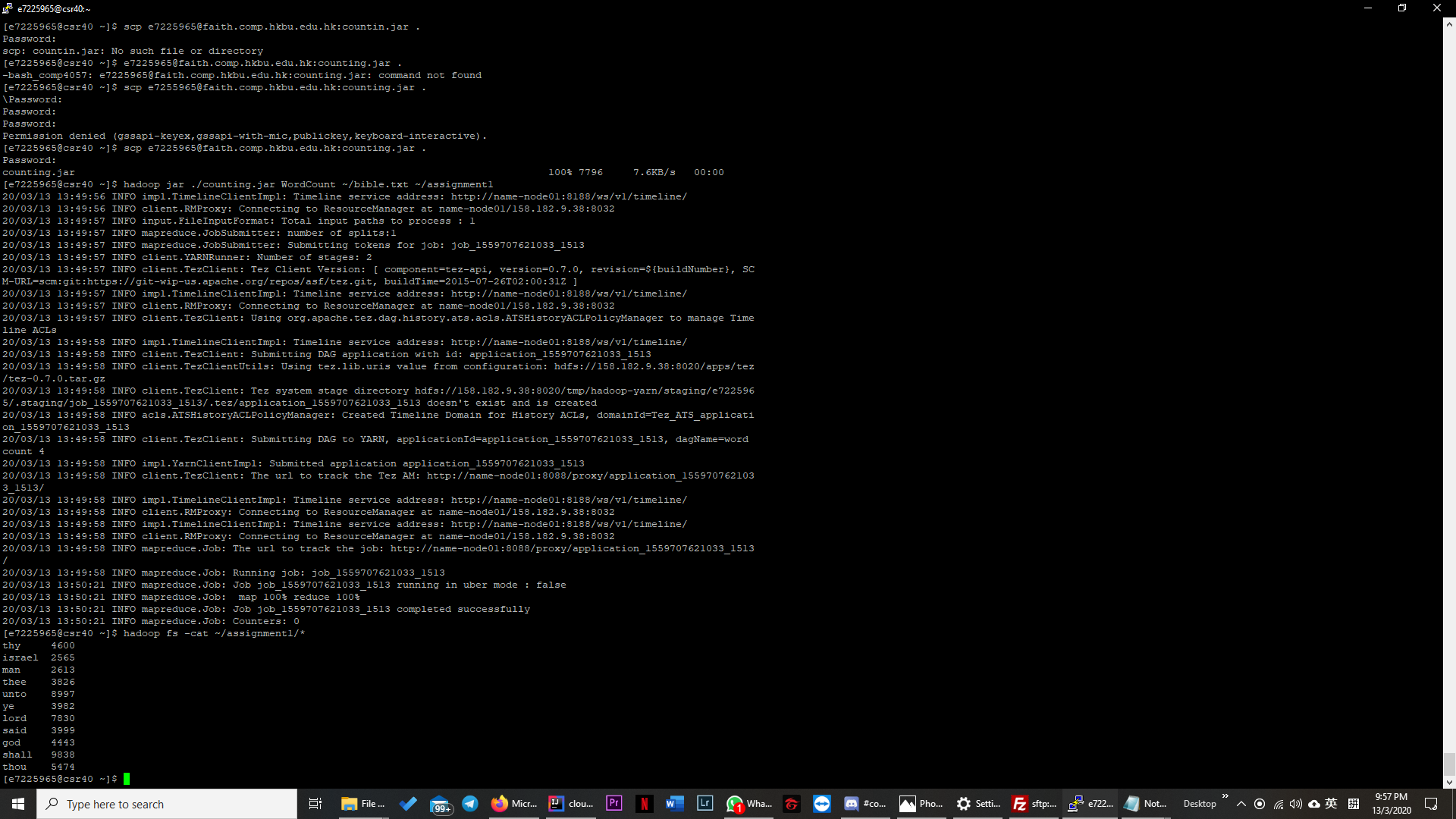
Name: Koo Tsz Kit

SID: 17225965



output

Input

public void map(Object key, Text value, Context context  
 ) throws IOException, InterruptedException {  
 StringTokenizer itr = new StringTokenizer(value.toString());  
 while (itr.hasMoreTokens()) {  
 String tmp = itr.nextToken();  
 tmp = tmp.replaceAll("[0-9]","");  
 tmp = tmp.replaceAll("\\W", "");  
 tmp = tmp.toLowerCase();  
 if(!Stopwords.*isOneOfThem*(tmp)&&!tmp.equals("")) {  
 word.set(tmp);  
 context.write(word, *one*);  
 }  
 }  
 }  
}

First, in the map function, before adding the index for the words, we need to filer out all the numbers and punctuations and change the word to lowercase. Therefore **tmp = tmp.replaceAll("[0-9]","");** is filtering all the numbers and **tmp = tmp.replaceAll("\\W", "");** is filtering all the punctuation. Moreover, **tmp = tmp.toLowerCase();** is changing all the word to lowercase.

Then, we need to filer out all the stop word so !Stopwords.isOneOfThem(tmp) is filtering out all of the stopword. Moreover, although all the punctuation, number has become ””, the words.set still will put a index for “”, for !tmp.,equals(“”) is filtering out all of those “”.

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();  
 }  
 if(sum>2500) {  
 result.set(sum);  
 context.write(key, result);  
 }  
 }  
}

The reducer function is the function reducing all the data from every groups, and output the total count result. But we only need the words that occurs more than 2500 times. Therefore before we use result.set(sum) and context.write, we add a if(sum>2500) to filter all the words that occurs less than 2500 times.