T.C. BAHÇEŞEHİR UNIVERSITY
FACULTY OF ENGINEERING AND NATURAL SCIENCES
COMPUTER ENGINEERING DEPARTMENT

CMP3005 Analysis of Algorithms
Term Project Report

Plagiarism Checker

Ercüment Burak Tokman (1315490)

Sertaç Bağcı (1802527)

İbrahim Barış Mumyakmaz (1804352)

Jan 16, 2020

1. INTRODUCTION

Plagiarism detection system that checks the similarity rate between selected document and a set of documents provided in text form.

2. DESIRED OUTPUTS

Developed program will print two distinct outputs for each document comparison process.

- Similarity rate between the main and each compared document.
- The most similar 5 sentences in each document.

3. OBJECTIVES

Primary objectives of the project ordered by their importance.

1. Running speed (efficiency) of the algorithm.
2. Similarity detection ability.
3. Readability of the code.

4. PROGRAMMING LANGUAGE

We preferred **Java** for this project due to its simplicity and ease of coding compared to C++.

5. LIBRARIES & METHODS

Libraries	java.io.*, java.nio.charset.StandartCharSets, java.nio.Files, java.nio.Path, java.nio.Paths, java.util.Arrays
Preprocessing	1. Separate documents content to sentences. 2. Convert sentences to lowercase. 3. Clean special characters including next line. 4. Separate sentence to words.
Rabin-Karp	String searching algorithm.
Hashing	Calculate the hash value of given text (string).
String Matching	If hashing matches, check each character of the text and the pattern if they actually match.
HashMap	Store sentences with their similarity rate along with their corresponding comparison sentences.

6. ANALYSIS OF THE STRING SEARCHING ALGORITHM

Our first design was a naïve method (brute force) solution with O(m*n) complexity. However, with the implementation of **Rabin-Karp** string searching algorithm, at the end we managed to achieve **O(n+m)** complexity with **501ms** average execution time. **Average execution time calculated for single main document and three comparison documents**.

Table 1. Comparison of string-matching algorithms

Algorithm	Avg. Time Complexity	Worst Case Complexity	Space Complexity	Average Speed (ms)
Brute-force	O(m*n)	O(m*n)	O(1)	2,437ms
Rabin-Karp	O(n+m)	O(m*n)	O(1)	501ms

m = average length of the pattern
n = length of the text that search made

7. HOW TO USE THE PROGRAM

We generated a **.jar** file since executing the Main class from terminal was bit problematic because of dependency to other classes.

Program can find and select main document and comparison folder by default if they are in the same directory. Move .**jar** file to a folder/directory where there are a **"main_doc.txt"** which will be checked and **"Documents"** folder containing text documents for plagiarism check.

```
Java -jar termproject.jar
```

Executing the jar file also can be done by giving **main document** and **documents folder** as arguments.

```
Java -jar termproject.jar -main="main_doc.txt" -compare="Documents/"
```

8. TOP LEVEL DESIGN OF THE PROGRAM

Each phase of the program without getting much into the detail.

1. Receive main file and documents folder as argument.
2. Read all files to Strings.

3. Take content of main file and check similarity with each document.
4. Print document similarity rate and most similar five sentences.

9. EXEMPLARY OUTPUTS

Plagiarised sentences colored with orange, red and green.

Main document:

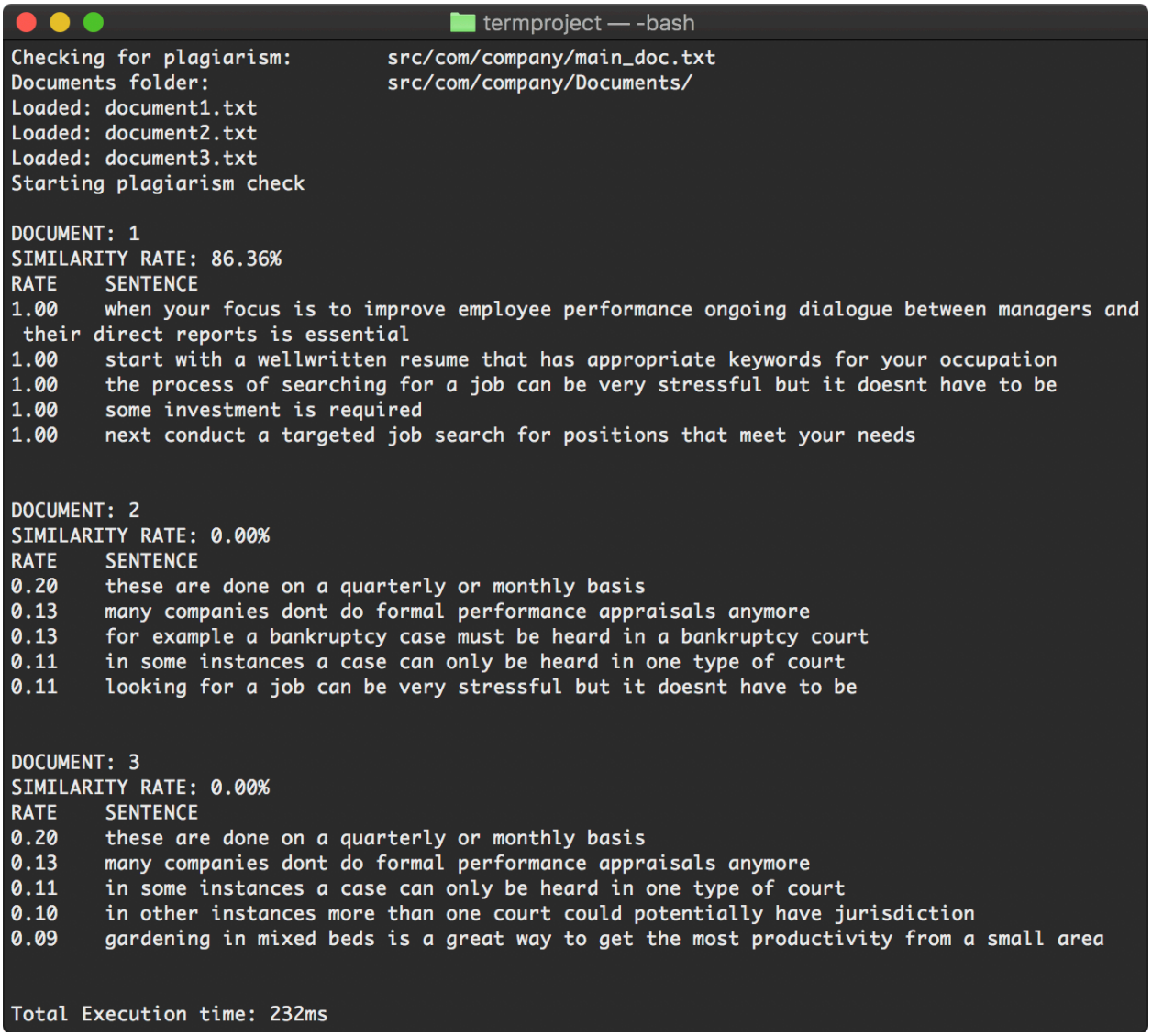
… For example, a federal criminal court and a state criminal court would each have jurisdiction over a crime that is a federal drug offense but that is also a state offense.
The process of searching for a job can be very stressful, but it doesn't have to be. Start with a well-written resume that has appropriate keywords for your occupation. Next, conduct a targeted job search for positions that meet your needs.
When your focus is to improve employee performance, ongoing dialogue between managers and their direct reports is essential. While performance management often involves conducting annual performance evaluations, it does involve more than just that. Many companies don't do formal performance appraisals anymore. Instead, they encourage one-on-one dialogues in which supervisors hold meetings with employees as a way to facilitate two-way communication and one-on one dialogue. These are done on a quarterly or monthly basis.
If you don't have a lot of space for a garden, raised beds can be a great option. Gardening in mixed beds is a great way to get the most productivity from a small area. **Some investment is required.** …

Document to be compared:

…For instance, bankruptcy cases can be ruled on only in bankruptcy court. In other situations, it is possible for more than one court to have jurisdiction. For instance, both a state and federal criminal court could have authority over a criminal case that is also considered an offense under federal and state drug laws.
The process of searching for a job can be very stressful, but it doesn't have to be. Start with a well-written resume that has appropriate keywords for your occupation. Next, conduct a targeted job search for positions that meet your needs.
Gardening in mixed beds is a great way to get the most productivity from a small space. **Some investment is required, to purchase materials for the beds themselves, as well as soil and compost.** The investment will likely pay-off in terms of increased productivity.
Performance management involves more than just conducting annual performance evaluations. In fact, many companies have done away with formal performance appraisals altogether. Instead, they opt for one-on-one dialogues between managers and employees on a quarterly or monthly basis.
When your focus is to improve employee performance, it's essential to encourage ongoing dialogue between managers and their direct reports. Some companies encourage

supervisors to hold one-on-one meetings with employees as a way to facilitate two-way communication.

9.1 OUTPUT OF THE PROGRAM:



10. DESIGN OF THE ALGORITHM

ALGORITHM	
STEP	OPERATION
0	Separate both documents to sentences and convert all letters to lowercase.
1	Pick sentence from **main** document.
2	Separate sentence to **words**.
3	Pick sentence from **comparison** document.
4	Remove **whitespaces** in **comparison** sentence
	APPLY RABIN-KARP
5	Pick word from **main** sentence as **pattern** and hash
6	Pick first letters from **comparison sentence** with pattern size and hash
7	If both **HASHES MATCH** then check each letter of the pattern and the text **if they really match**
8	If **PATTERN MATCH** then **INCREASE** the patternMatchCounter (then **Go to Step 11**)
9	If **patternMatchCounter** is equal or above 5 then mark as plagiarised
10	If **NO HASH** match then shift 1 letter to right
11	Pick the **next word** from **main** sentence **(Repeat from Step 5)**
12	**RESET** patternMatchCounter
13	Pick next sentence from **comparison** document **(Repeat from Step 1)**
14	Pick next sentence from **main** document **(Repeat from Step 1)**

10.1 EXAMPLE SCENARIO

MAIN DOCUMENT	
ID	SENTENCE
1	**Lorem ipsum** dolor **sit** amet, consectetur adipiscing **elit**.
2	**Donec** porttitor in nisi nec porttitor.
3	Praesent sagittis orci **ante**, ut **vestibulum** neque pretium viverra.
4	Ut imperdiet nunc at pharetra efficitur.
5	**Nullam non elit molestie ligula** euismod vestibulum nec et mauris.
6	Aliquam tristique quam id euismod sollicitudin.

COMPARISON DOCUMENT		
ID	SENTENCE	
1	Sed porttitor lectus vitae velit semper tincidunt.	
2	**Lorem ipsum Nullam non elit molestie ligula** adipiscing **elit**.	PLAGIARISED
3	**Donec vestibulum ante ipsum** primis in faucibus orci luctus et ultrices posuere cubilia curae.	
4	Etiam in vehicula mi, vel ullamcorper purus.	
5	Etiam **sit** amet malesuada lorem, **sit** amet dapibus turpis.	
6	Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos.	
7	Nulla interdum sapien urna.	

ID	Main Sentence									Main Document	
5	PATTERN	Nullam	non	elit	molestie	ligula	euismod	vestibulum	nec	et	mauris
	HASH	7	12	58	36	41	26	65	16	3	54

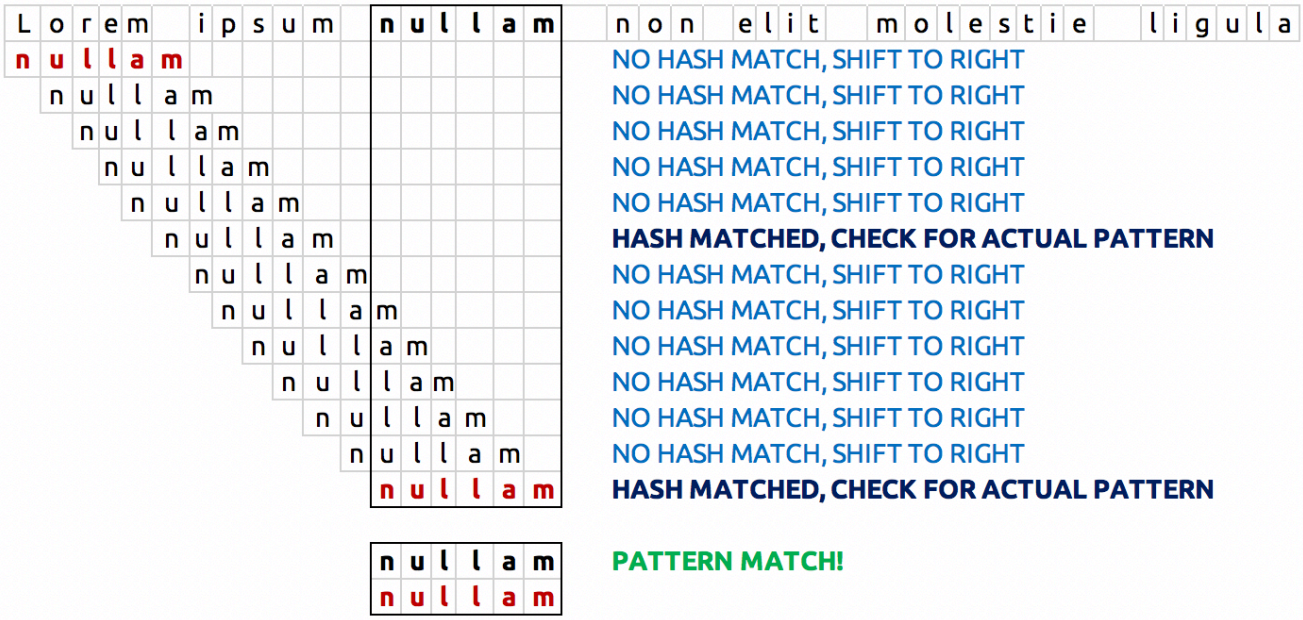
ID	Comparison Sentence									Comparison Document	
2	TEXT	Lorem ipsum nullam non elit molestie ligula adipiscing elit.									
		Lorem	ipsum	nullam	non	elit	molestie	ligula	adipiscing	elit	
	HASH	9	5	7	12	58	36	41	17	23	

Main Sentence

Nullam non elit molestie ligula euismod vestibulum nec et mauris.

Comparison Sentence

Lorem ipsum nullam non elit molestie ligula adipiscing elit.



11. CALCULATING SENTENCE SIMILARITY RATE

```
sentenceSimilarityRate = wordsMatched / MainSentenceTotalWordCount
```

sentenceSimilarityRate variable is the similarity percentage of the sentence from main document.

12. CALCULATING DOCUMENT SIMILARITY RATE

```
similarityRate = 100*PlagiarisedSentenceCount / totalSentenceCount
```

SimilarityRate variable is the similarity percentage of the main document and compared document.

13. SORTING HASHMAP

We used a map data structure to store the **sentences** of the main document with their **percentage** of plagiarism:

```
Map<String, Float> hm = new HashMap<String, Float>();
```

We put the data in the following way:

```
hm.put(mainSentence, (float) wordMatchCounter / mainWords.length);
```

Then we used a LinkedHashMap to store our map in a reverse sorted way (higher to lower):

```
LinkedHashMap<String, Float> reverseSortedMap = new LinkedHashMap<>();
    hm.entrySet()
        .stream()
        .sorted(Map.Entry.comparingByValue(Comparator.reverseOrder()))
        .forEachOrdered(x -> reverseSortedMap.put(x.getKey(), x.getValue()));
```

As a last step we took the first five values in the table with a for loop:

```
for (Map.Entry me : reverseSortedMap.entrySet()) {
        System.out.printf("%.2f\t%s\n", me.getValue(), me.getKey());
        if (i == 4)
           break;
        i++;
    }
```

14. OUTPUT TEMPLATE

```
DOCUMENT: {docNo}
SIMILARITY RATE: {similarityRate}%
{sentenceSimRate} 1st most similar sentence
{sentenceSimRate} 2nd most similar sentence
{sentenceSimRate} 3rd most similar sentence
{sentenceSimRate} 4th most similar sentence
{sentenceSimRate} 5th most similar sentence
```

15. REFERENCES

Remove null items from Array
https://stackoverflow.com/questions/31583523/best-way-to-remove-null-values-from-string-array

Remove whitespaces from a stream
https://stackoverflow.com/questions/56127505/how-to-remove-digits-and-whitespaces-from-a-stream

Sort HashMap in descending order
https://howtodoinjava.com/java/sort/java-sort-map-by-values/

Design logic & implementation of Rabin-Karp
https://www.youtube.com/watch?v=qQ8vS2btsxI
https://www.geeksforgeeks.org/rabin-karp-algorithm-for-pattern-searching/
https://www.tutorialcup.com/interview/string/rabin-karp-algorithm.htm