**Gebze Technical University**
**Computer Engineering**

**CSE 222 - 2018 Spring**

# HOMEWORK 6 REPORT

**BERKE SÜSLÜ**
**161044076**

Course Assistant: Ayşe Şerbetçi Turan

# 1 INTRODUCTION

## 1.1 Problem Definition

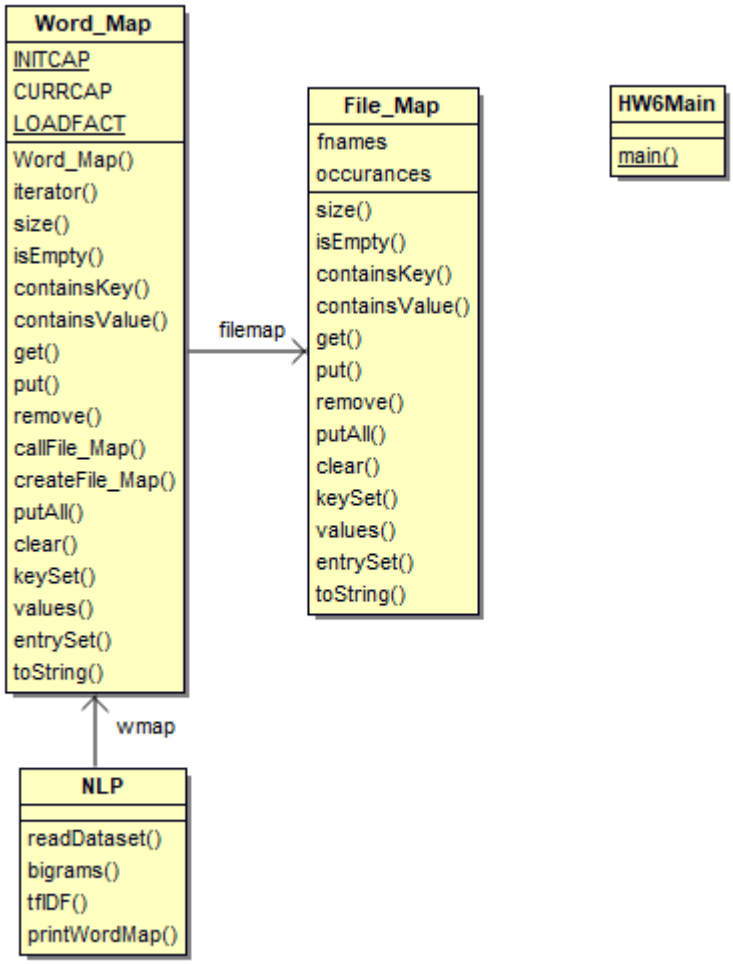
Reading texts with using hashMap data structures.

## 1.2 System Requirements

I used IntelliJ IDEA 2018.3.5 (Community Edition) with
Build #IC-183.5912.21, built on February 26, 2019
JRE: 1.8.0_152-release-1343-b28 amd64
JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o
Windows 10 10.0

# 2 METHOD

## 2.1 Class Diagrams




## 2.2 Problem Solution Approach

I implemented the given methods for File_Map. In Word_Map class, i created my iterator and wrapper function in order to use File_Map class. In NLP class, i implemented the given methods again.

## 2.3 Time Complexities

File_Map methods:

| | |
|---|---|
| Size() | O(1) |
| isEmpty() | O(1) |
| containsKey() | O(n) (Due to the arrayList.contains() implementation) |

| | |
|---|---|
| containsValue() | O(n) (Due to the linear for loop) |
| get() | Best case O(1) – Average O(n+1)/2 = O(n) – Worst case O(n) |
| put() | Best case O(n)+O(n) = O(n) (If statements)<br>Worst case O(n)+O(n) = O(n) (ArrayList add method)<br>In conclusion : O(n) |
| remove() | O(n) (ArrayList remove method) |
| putAll() | O(n) (ArrayList addAll method) |
| clear() | O(1) |
| keySet() | O(n) (Due to the linear for loop) |
| values() | O(1) |
| entrySet() | Not implemented |
| toString() | O(n) (Due to the linear for loop) |

Word_Map methods:

| | |
|---|---|
| Size() | O(1) |
| isEmpty() | O(1) |
| containsKey(Object key) | Best case O(1) – Worst case O(n) |
| containsValue(Object value) | Best case O(1) – Worst case O(n) |
| get(Object key) | Best case O(1) – Worst case O(n) (Linear probe) |
| Put(Object key, Object value) | Best case O(1) – Worst case O(n^2) (Nested linear for loops) |
| putAll(map m) | Same as put method (includes put method) |
| Clear() | O(1) |
| keySet() | Best case O(1) – Worst case O(n) (Due to the hashSet add method) |
| callFile_Map() | Best case O(1) – Worst case O(n) (Due to the while loop) |
| createFile_Map() | Same as callFile_Map (includes it) |
| values() | Best case O(1) – Worst case O(n) (Due to the for loop) |
| entrySet() | Not implemented |
| toString() | O(n) (Due to the linear list search) |
| Iterator functions(hasNext,next,remove) | O(1) |

NLP methods:

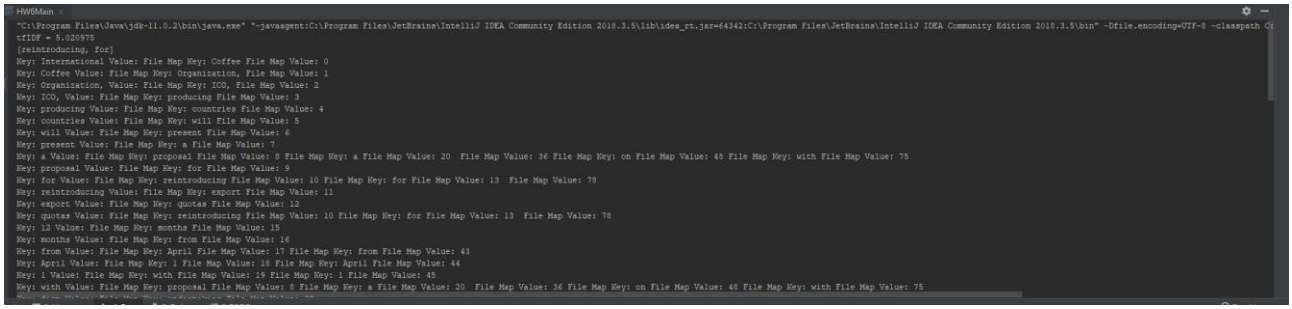
| | |
|---|---|
| readDataset() | Best case O(n) – Worst case O(n^2) (Due to the Word_Map.put() method) |
| bigrams() | O(n) (Due to the ArrayList putAll method) |
| tfIDF() | O(1) |
| printWordMap() | O(n) (Same as Word_Map.toString() method) |

# 3 RESULT

## 3.1 Test Cases

I used 0000026 file, index.txt and my custom driver function to test data structure.

## 3.2 Running Results




- Main titles -> 16pt , 2 line break
- Subtitles -> 14pt, 1.5 line break
- Paragraph -> 12pt, 1.5 line break