LAPORAN TUGAS KECIL I

IF2211 STRATEGI ALGORITMA

Gede Prasidha Bhawarnawa

13520004

K01

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

I. Algoritma *Brute Force* yang Digunakan

Dalam melakukan pencarian solusi atau jawaban dari *word search puzzle* seperti yang dilakukan oleh penulis, digunakan algoritma *brute force* dengan langkah-langkah sebagai berikut:

1. Menerima input berupa sebuah matriks huruf dan sebuah list kata kunci.
2. Memasukkan isi matriks huruf ke dalam sebuah *array of char* "word_matrix" dan diisi padding (diberikan 1 tambahan baris dan kolom di ujung atas, kanan, kiri, dan bawah). Tujuan diberikan padding adalah agar saat dilakukan pengecekan, tidak akan muncul *error* "*Index out of range*" karena bila dilakukan pengecekan sampai ke luar matriks huruf, scanner akan bertemu dengan padding / ujung matriks.
3. Memasukkan list kata kunci yang ingin dicari ke dalam sebuah *array of strings* "answer_matrix"
4. Untuk setiap kata kunci di dalam "answer_matrix", diambil huruf pertamanya dan dibandingkan dengan huruf pertama pada "word_matrix".
    a. Bila tidak ditemukan, maka search dilakukan ke elemen selanjutnya hingga semua huruf di "word_matrix" telah diperiksa.
    b. Bila ditemukan, maka akan diperiksa elemen-elemen disekitar huruf tersebut dengan huruf kedua, ketiga, keempat, sampai huruf ke-*n* kata kunci. Arah pemeriksaan disamakan dengan arah delapan mata angin (*north, northeast, east, southeast, south, southwest, west, northwest*).
        i. Bila sampai huruf ke-*n* semua huruf ditemukan pada suatu arah tertentu, maka posisi huruf pertama (*starting point*) pada "word_matrix" serta arah yang digunakan untuk menemukan kata kunci akan disimpan dalam sebuah *array of integer* "where_and_how". *Output*-kan ke *console* kata kunci yang ditemukan, posisi huruf pertama, serta arah pembacaan kata kunci di "word_matrix".
        ii. Bila sampai huruf ke-*n* tidak semua huruf ditemukan pada arah tertentu, maka akan direset pencarian ke posisi huruf

pertama namun akan mencoba arah baru. Kembali ke langkah 4.b.

iii. Bila semua arah telah dicoba tapi tidak ada yang memenuhi kondisi langkah 4.b, maka kembali ke langkah 4.a.

c. Bila semua huruf pada “word_matrix” telah diperiksa dan tidak ada yang memenuhi langkah 4.b.i, maka outputkan teks yang menunjukkan bahwa kata tidak dapat ditemukan.

5. Untuk setiap elemen di “answer_matrix” dan setiap baris di “where_and_how”:
    a. Buat duplikat matriks “word_matrix” yang semua elemennya berisikan “-“ bernama “result_matrix”
    b. Untuk setiap baris di “word_matrix”:
        i. Baca huruf pertama di “word_matrix” sesuai posisi yang terdapat di “where_and_how”.
        ii. Iterasi huruf kedua sampai akhir di “word_matrix” sesuai arah yang telah ditandai di “where_and_how”.
        iii. Untuk setiap huruf yang diiterasi di “word_matrix”, pindahkan huruf tersebut sesuai indeksnya ke “result_matrix”.
6. Tampilkan waktu yang dibutuhkan untuk mencari semua kata kunci dalam nano detik
7. Tampilkan jumlah perbandingan huruf yang dilakukan untuk mencari semua kata kunci

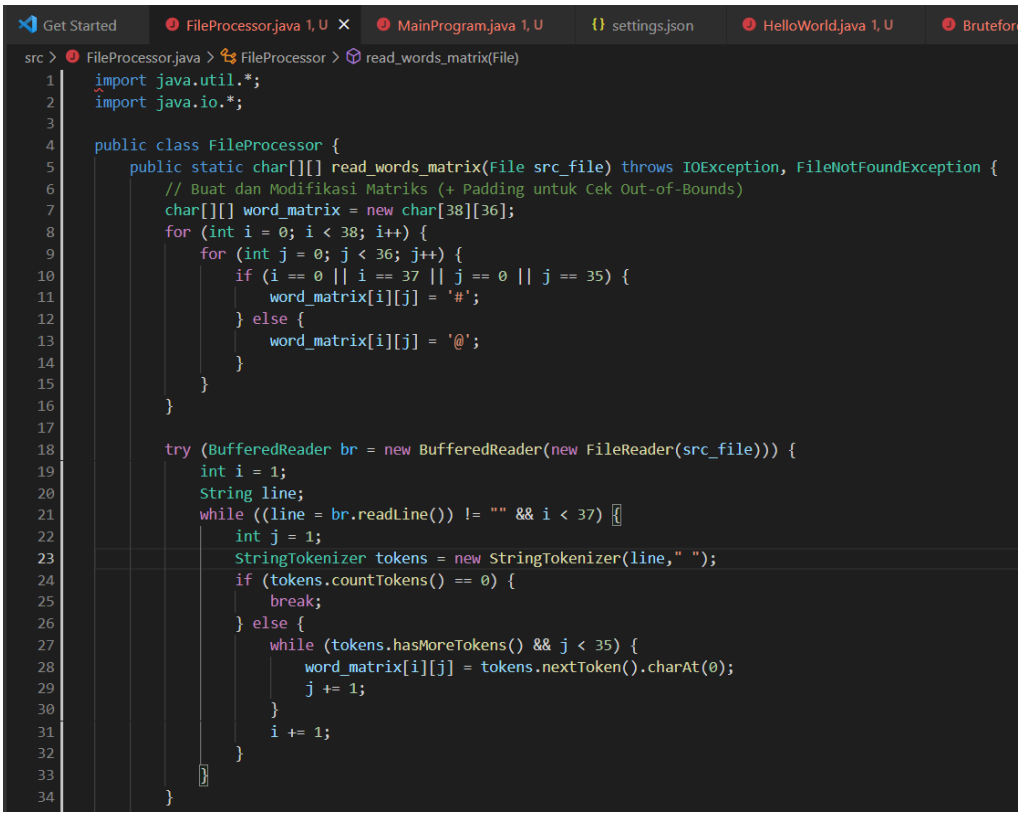
II. Source Program

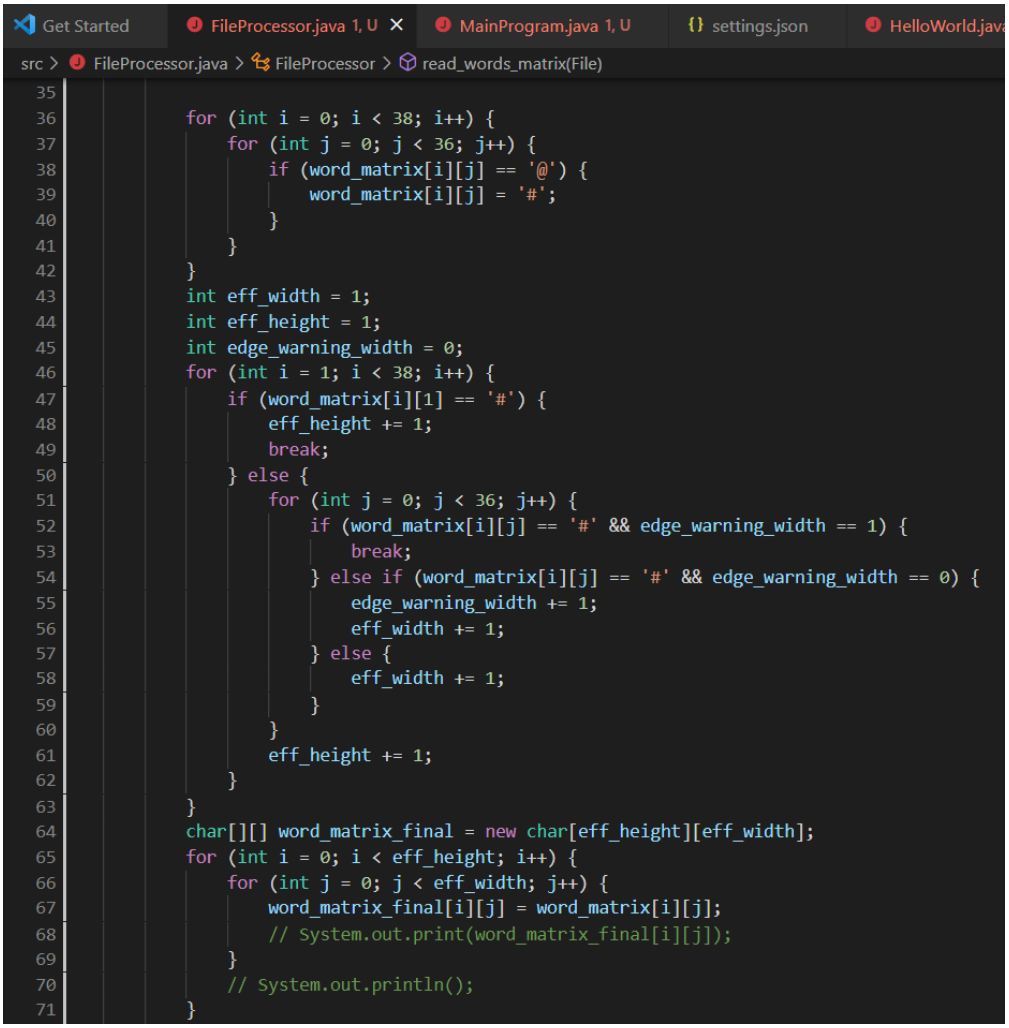
Program yang dibuat untuk melakukan *brute force word search puzzle* menggunakan bahasa Java. Program dapat dijalankan dengan membuka terminal/shell di folder “bin”. Lalu, gunakan kode “java MainProgram” untuk membuka program. Ketika diminta *prompt*, masukkan nama file test yang berada di dalam folder “testfile”. Bila file tidak ditemukan di dalam folder “testfile”, maka program akan meminta sampai mendapatkan file yang valid.

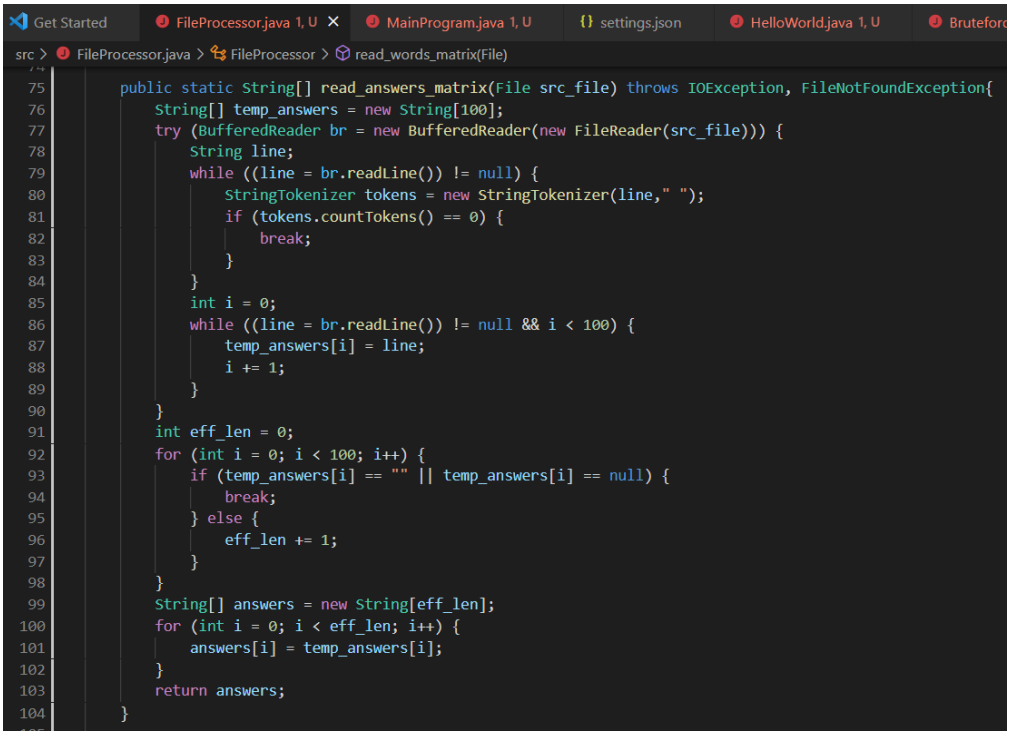
Program dibagi menjadi tiga bagian: “FileProcessing.java”, “BruteforceMethod.java”, “MainProgram.java”. “FileProcessing,java” adalah

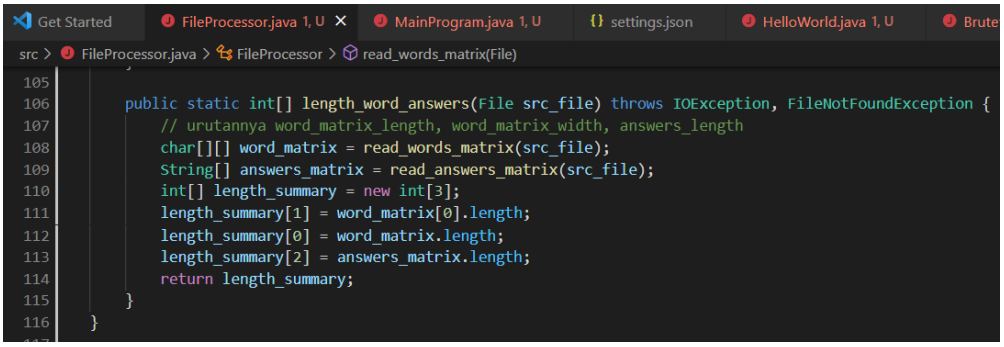
program untuk membaca isi txt file ke dalam array “word_matrix” dan “answer_matrix”. File ini juga akan membuat *array of integer* “len_summary” yang berisikan dimensi “word_matrix” dan “answer_matrix”.

Berikut adalah potretan layar (*screenshot*) untuk program “FileProcessing,java”:

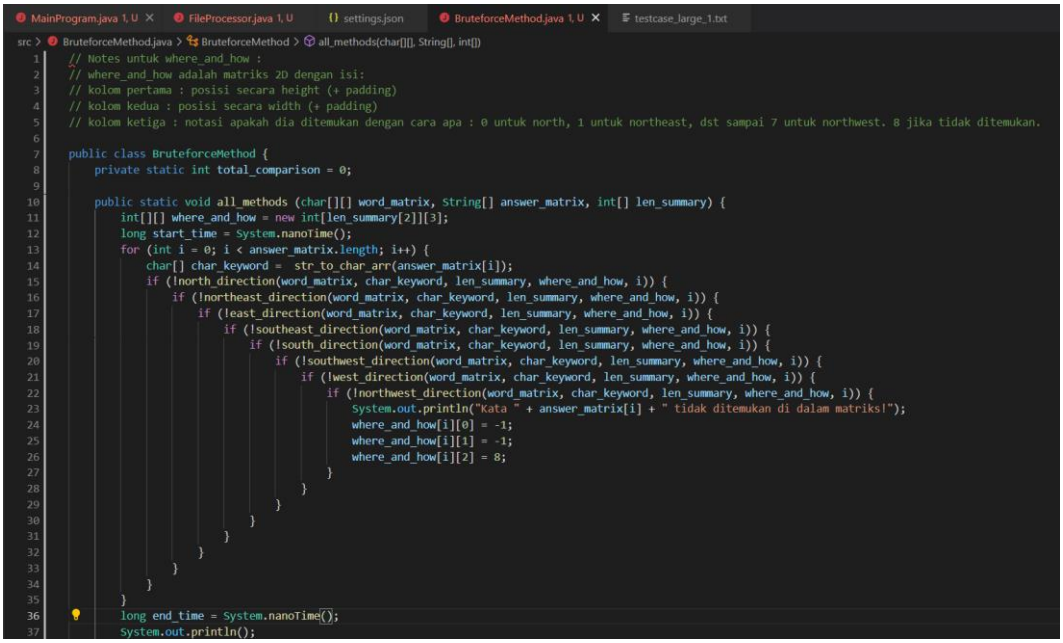


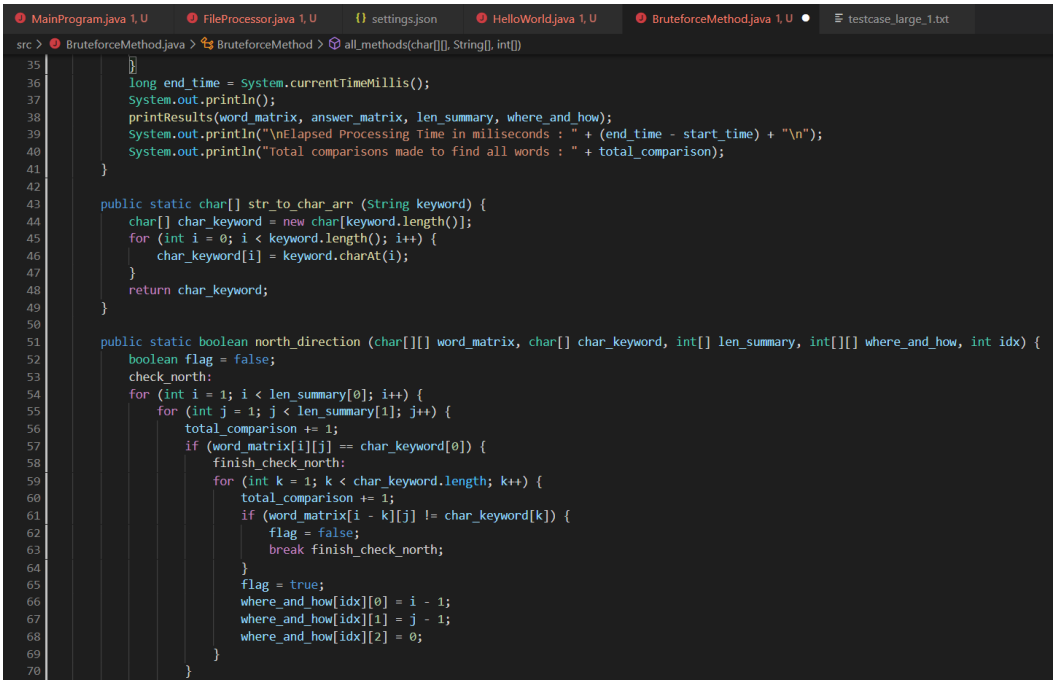

Get Started
FileProcessor.java 1, U
MainProgram.java 1, U
settings.json
HelloWorld.java
src > FileProcessor.java > FileProcessor > read_words_matrix(File)
for (int i = 0; i < 38; i++) {
for (int j = 0; j < 36; j++) {
if (word_matrix[i][j] == '@') {
word_matrix[i][j] = '#';
}
}
}
int eff_width = 1;
int eff_height = 1;
int edge_warning_width = 0;
for (int i = 1; i < 38; i++) {
if (word_matrix[i][1] == '#') {
eff_height += 1;
break;
} else {
for (int j = 0; j < 36; j++) {
if (word_matrix[i][j] == '#' && edge_warning_width == 1) {
break;
} else if (word_matrix[i][j] == '#' && edge_warning_width == 0) {
edge_warning_width += 1;
eff_width += 1;
} else {
eff_width += 1;
}
}
eff_height += 1;
}
}
char[][] word_matrix_final = new char[eff_height][eff_width];
for (int i = 0; i < eff_height; i++) {
for (int j = 0; j < eff_width; j++) {
word_matrix_final[i][j] = word_matrix[i][j];
// System.out.print(word_matrix_final[i][j]);
}
// System.out.println();
}


Get Started
FileProcessor.java 1, U
MainProgram.java 1, U
settings.json
HelloWorld.java 1, U
src > FileProcessor.java > FileProcessor > read_words_matrix(File)
public static String[] read_answers_matrix(File src_file) throws IOException, FileNotFoundException{
String[] temp_answers = new String[100];
try (BufferedReader br = new BufferedReader(new FileReader(src_file))) {
String line;
while ((line = br.readLine()) != null) {
StringTokenizer tokens = new StringTokenizer(line," ");
if (tokens.countTokens() == 0) {
break;
}
}
int i = 0;
while ((line = br.readLine()) != null && i < 100) {
temp_answers[i] = line;
i += 1;
}
}
int eff_len = 0;
for (int i = 0; i < 100; i++) {
if (temp_answers[i] == "" || temp_answers[i] == null) {
break;
} else {
eff_len += 1;
}
}
String[] answers = new String[eff_len];
for (int i = 0; i < eff_len; i++) {
answers[i] = temp_answers[i];
}
return answers;
}

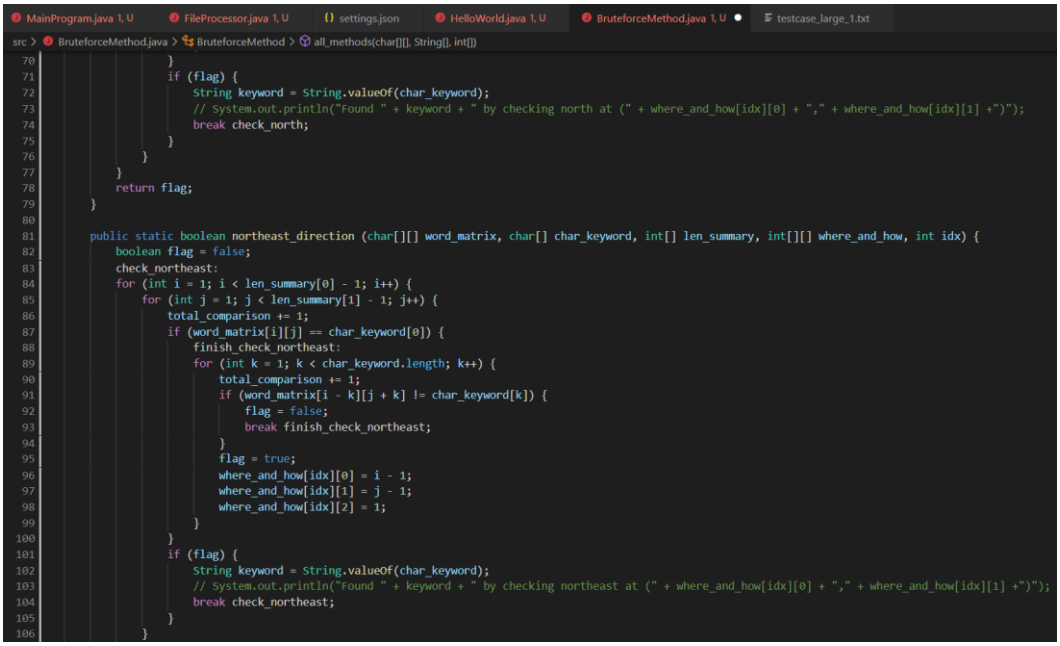


"BruteforceMethod.java" adalah program untuk mencari semua kata kunci pada array "word_matrix". Program ini membandingkan huruf pertama kata kunci dan mencari pasangannya di "word_matrix". Lalu, program akan mencari ke delapan arah mata angin dari setiap huruf yang berpasangan dengan huruf pertama kata kunci. Program lalu akan mencatat posisi huruf pertama dan arah bila sampai huruf terakhir kata kunci ditemukan. Bila tidak ditemukan, maka akan di-output tidak ditemukan kata tersebut. Bila ditemukan, maka akan di-output lokasi kata tersebut pada "word_matrix".

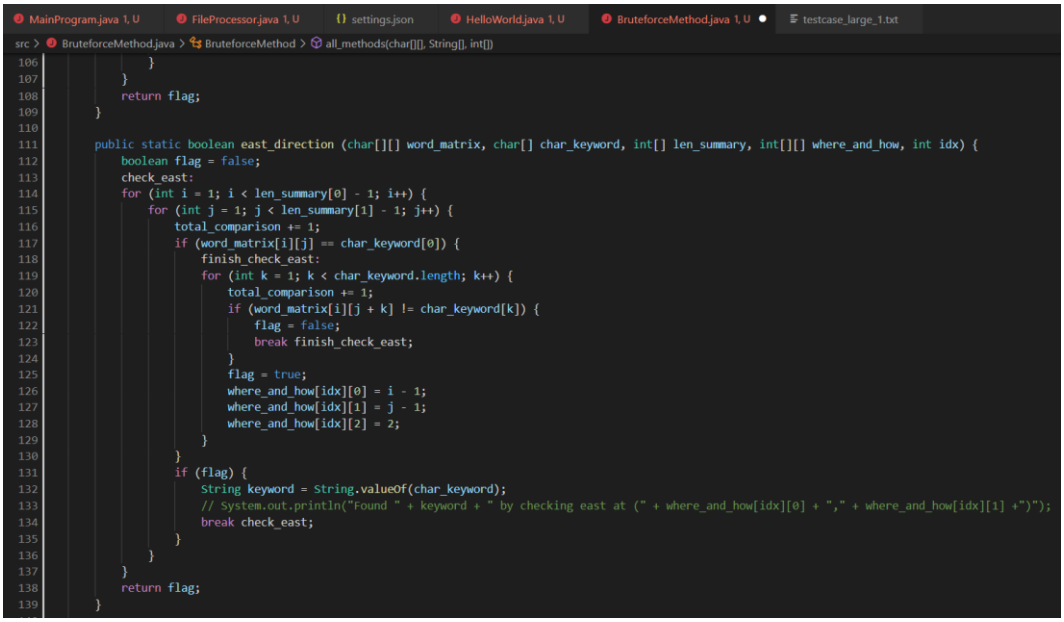




```
MainProgram.java 1, U | FileProcessor.java 1, U | settings.json | HelloWorld.java 1, U | BruteforceMethod.java 1, U | testcase_large_1.txt
src > BruteforceMethod.java > BruteforceMethod > all_methods(char[][], String[], int[])
35              }
36              long end_time = System.currentTimeMillis();
37              System.out.println();
38              printResults(word_matrix, answer_matrix, len_summary, where_and_how);
39              System.out.println("\nElapsed Processing Time in miliseconds : " + (end_time - start_time) + "\n");
40              System.out.println("Total comparisons made to find all words : " + total_comparison);
41          }
42
43          public static char[] str_to_char_arr (String keyword) {
44              char[] char_keyword = new char[keyword.length()];
45              for (int i = 0; i < keyword.length(); i++) {
46                  char_keyword[i] = keyword.charAt(i);
47              }
48              return char_keyword;
49          }
50
51          public static boolean north_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
52              boolean flag = false;
53              check_north:
54              for (int i = 1; i < len_summary[0]; i++) {
55                  for (int j = 1; j < len_summary[1]; j++) {
56                      total_comparison += 1;
57                      if (word_matrix[i][j] == char_keyword[0]) {
58                          finish_check_north:
59                          for (int k = 1; k < char_keyword.length; k++) {
60                              total_comparison += 1;
61                              if (word_matrix[i - k][j] != char_keyword[k]) {
62                                  flag = false;
63                                  break finish_check_north;
64                              }
65                              flag = true;
66                              where_and_how[idx][0] = i - 1;
67                              where_and_how[idx][1] = j - 1;
68                              where_and_how[idx][2] = 0;
69                          }
70                      }
```



```
MainProgram.java 1, U | FileProcessor.java 1, U | settings.json | HelloWorld.java 1, U | BruteforceMethod.java 1, U | testcase_large_1.txt
src > BruteforceMethod.java > BruteforceMethod > all_methods(char[][], String[], int[])
70                  }
71                  if (flag) {
72                      String keyword = String.valueOf(char_keyword);
73                      // System.out.println("Found " + keyword + " by checking north at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
74                      break check_north;
75                  }
76              }
77          }
78          return flag;
79      }
80
81      public static boolean northeast_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
82          boolean flag = false;
83          check_northeast:
84          for (int i = 1; i < len_summary[0] - 1; i++) {
85              for (int j = 1; j < len_summary[1] - 1; j++) {
86                  total_comparison += 1;
87                  if (word_matrix[i][j] == char_keyword[0]) {
88                      finish_check_northeast:
89                      for (int k = 1; k < char_keyword.length; k++) {
90                          total_comparison += 1;
91                          if (word_matrix[i - k][j + k] != char_keyword[k]) {
92                              flag = false;
93                              break finish_check_northeast;
94                          }
95                          flag = true;
96                          where_and_how[idx][0] = i - 1;
97                          where_and_how[idx][1] = j - 1;
98                          where_and_how[idx][2] = 1;
99                      }
100                 }
101                 if (flag) {
102                     String keyword = String.valueOf(char_keyword);
103                     // System.out.println("Found " + keyword + " by checking northeast at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
104                     break check_northeast;
105                 }
106             }
```




```java
                }
            }
            return flag;
        }

        public static boolean east_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
            boolean flag = false;
            check_east:
            for (int i = 1; i < len_summary[0] - 1; i++) {
                for (int j = 1; j < len_summary[1] - 1; j++) {
                    total_comparison += 1;
                    if (word_matrix[i][j] == char_keyword[0]) {
                        finish_check_east:
                        for (int k = 1; k < char_keyword.length; k++) {
                            total_comparison += 1;
                            if (word_matrix[i][j + k] != char_keyword[k]) {
                                flag = false;
                                break finish_check_east;
                            }
                            flag = true;
                            where_and_how[idx][0] = i - 1;
                            where_and_how[idx][1] = j - 1;
                            where_and_how[idx][2] = 2;
                        }
                    }
                    if (flag) {
                        String keyword = String.valueOf(char_keyword);
                        // System.out.println("Found " + keyword + " by checking east at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
                        break check_east;
                    }
                }
            }
            return flag;
        }
```

```java
        public static boolean southeast_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
            boolean flag = false;
            check_southeast:
            for (int i = 1; i < len_summary[0] - 1; i++) {
                for (int j = 1; j < len_summary[1] - 1; j++) {
                    total_comparison += 1;
                    if (word_matrix[i][j] == char_keyword[0]) {
                        finish_check_southeast:
                        for (int k = 1; k < char_keyword.length; k++) {
                            total_comparison += 1;
                            if (word_matrix[i + k][j + k] != char_keyword[k]) {
                                flag = false;
                                break finish_check_southeast;
                            }
                            flag = true;
                            where_and_how[idx][0] = i - 1;
                            where_and_how[idx][1] = j - 1;
                            where_and_how[idx][2] = 3;
                        }
                    }
                    if (flag) {
                        String keyword = String.valueOf(char_keyword);
                        // System.out.println("Found " + keyword + " by checking southeast at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
                        break check_southeast;
                    }
                }
            }
            return flag;
        }
```

```java
        public static boolean south_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
            boolean flag = false;
            check_south:
            for (int i = 1; i < len_summary[0] - 1; i++) {
                for (int j = 1; j < len_summary[1] - 1; j++) {
                    total_comparison += 1;
                    if (word_matrix[i][j] == char_keyword[0]) {
                        finish_check_south:
                        for (int k = 1; k < char_keyword.length; k++) {
                            total_comparison += 1;
                            if (word_matrix[i + k][j] != char_keyword[k]) {
                                flag = false;
                                break finish_check_south;
                            }
                            flag = true;
                            where_and_how[idx][0] = i - 1;
                            where_and_how[idx][1] = j - 1;
                            where_and_how[idx][2] = 4;
                        }
                    }
                    if (flag) {
                        String keyword = String.valueOf(char_keyword);
                        // System.out.println("Found " + keyword + " by checking south at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
                        break check_south;
                    }
                }
            }
            return flag;
        }
```

```java
public static boolean southwest_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
    boolean flag = false;
    check_southwest:
    for (int i = 1; i < len_summary[0] - 1; i++) {
        for (int j = 1; j < len_summary[1] - 1; j++) {
            total_comparison += 1;
            if (word_matrix[i][j] == char_keyword[0]) {
                finish_check_southwest:
                for (int k = 1; k < char_keyword.length; k++) {
                    total_comparison += 1;
                    if (word_matrix[i + k][j - k] != char_keyword[k]) {
                        flag = false;
                        break finish_check_southwest;
                    }
                    flag = true;
                    where_and_how[idx][0] = i - 1;
                    where_and_how[idx][1] = j - 1;
                    where_and_how[idx][2] = 5;
                }
            }
            if (flag) {
                String keyword = String.valueOf(char_keyword);
                // System.out.println("Found " + keyword + " by checking southwest at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
                break check_southwest;
            }
        }
    }
    return flag;
}
```

```java
public static boolean west_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
    boolean flag = false;
    check_west:
    for (int i = 1; i < len_summary[0] - 1; i++) {
        for (int j = 1; j < len_summary[1] - 1; j++) {
            total_comparison += 1;
            if (word_matrix[i][j] == char_keyword[0]) {
                finish_check_west:
                for (int k = 1; k < char_keyword.length; k++) {
                    total_comparison += 1;
                    if (word_matrix[i][j - k] != char_keyword[k]) {
                        flag = false;
                        break finish_check_west;
                    }
                    flag = true;
                    where_and_how[idx][0] = i - 1;
                    where_and_how[idx][1] = j - 1;
                    where_and_how[idx][2] = 6;
                }
            }
            if (flag) {
                String keyword = String.valueOf(char_keyword);
                // System.out.println("Found " + keyword + " by checking west at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
                break check_west;
            }
        }
    }
    return flag;
}
```

```java
public static boolean northwest_direction (char[][] word_matrix, char[] char_keyword, int[] len_summary, int[][] where_and_how, int idx) {
    boolean flag = false;
    check_northwest:
    for (int i = 1; i < len_summary[0] - 1; i++) {
        for (int j = 1; j < len_summary[1] - 1; j++) {
            total_comparison += 1;
            if (word_matrix[i][j] == char_keyword[0]) {
                finish_check_northwest:
                for (int k = 1; k < char_keyword.length; k++) {
                    total_comparison += 1;
                    if (word_matrix[i - k][j - k] != char_keyword[k]) {
                        flag = false;
                        break finish_check_northwest;
                    }
                    flag = true;
                    where_and_how[idx][0] = i - 1;
                    where_and_how[idx][1] = j - 1;
                    where_and_how[idx][2] = 7;
                }
            }
            if (flag) {
                String keyword = String.valueOf(char_keyword);
                // System.out.println("Found " + keyword + " by checking northwest at (" + where_and_how[idx][0] + "," + where_and_how[idx][1] +")");
                break check_northwest;
            }
        }
    }
    return flag;
}
```

```java
    public static void printResults (char[][] word_matrix, String[] answer_matrix, int[] len_summary, int[][] where_and_how) {
        // Menghilangkan padding dari word_matriks untuk ditampilkan
        // System.out.println(word_matrix.length);
        // System.out.println(word_matrix[0].length);
        // for (int i = 0; i < where_and_how.length; i++) {
        //     System.out.println(where_and_how[i][0] + " " + where_and_how[i][1] + " " + where_and_how[i][2]);
        // }
        int starting_direction, starting_height, starting_width;
        char[] char_keyword;
        char[][] new_word_matrix = new char[len_summary[0] - 2][len_summary[1] - 2];
        for (int i = 0; i < len_summary[0] - 2; i++) {
            for (int j = 0; j < len_summary[1] - 2; j++) {
                new_word_matrix[i][j] = word_matrix[i + 1][j + 1];
            }
        }
        for (int i = 0; i < answer_matrix.length; i++) {
            char[][] results_matrix = new char[len_summary[0] - 2][len_summary[1] - 2];
            for (int j = 0; j < results_matrix.length; j++) {
                for (int k = 0; k < results_matrix[0].length; k++) {
                    results_matrix[j][k] = '-';
                }
            }
            starting_height = where_and_how[i][0];
            starting_width = where_and_how[i][1];
            starting_direction = where_and_how[i][2];
            char_keyword = str_to_char_arr(answer_matrix[i]);
```

```java
            char_keyword = str_to_char_arr(answer_matrix[i]);
            switch (starting_direction) {
                case 0:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_height -= 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah north di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
                case 1:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_height -= 1;
                            starting_width += 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah northeast di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
```

```
                case 2:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_width += 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah east di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
                case 3:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_height += 1;
                            starting_width += 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah southeast di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
```

```
                case 4:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_height += 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah south di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
                case 5:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_height += 1;
                            starting_width -= 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah southwest di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
```

```
                case 6:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_width -= 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah west di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
                case 7:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_height -= 1;
                            starting_width -= 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah northwest di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
```

```
                case 7:
                    for (int k = 0; k < char_keyword.length; k++) {
                        if (word_matrix[starting_height + 1][starting_width + 1] == char_keyword[k]) {
                            results_matrix[starting_height][starting_width] = word_matrix[starting_height + 1][starting_width + 1];
                            starting_height -= 1;
                            starting_width -= 1;
                        }
                    }
                    System.out.println("Kata " + answer_matrix[i] + " ditemukan arah northwest di :");
                    System.out.println();
                    for (int l = 0; l < results_matrix.length; l++) {
                        for (int m = 0; m < results_matrix[0].length; m++) {
                            System.out.print(results_matrix[l][m]);
                        }
                        System.out.println();
                    }
                    System.out.println();
                    break;
                case 8:
                    System.out.println("Kata " + answer_matrix[i] + " tidak ditemukan\n");
            }
        }
    }
}
```

“MainProgram.java” adalah program utama dan yang menjadi penerima aliran I/O utama dari pengguna program. Program ini akan memberikan prompt untuk meminta nama file testcase yang berada pada folder “testcase”. Bila file ditemukan, maka program akan berlanjut ke langkah selanjutnya. Bila tidak ditemukan, maka prompt akan terus diberikan hingga file ditemukan. Setelah itu, program akan memanggil fungsi-fungsi yang berada pada “FileProcessor.java” dan “BruteforceMethod.java” dan menampilkan outputnya.

```java
import java.util.Scanner;
import java.io.*;

public class MainProgram {
    Run | Debug
    public static void main(String[] args) throws IOException {
        boolean file_nonexistant = true;
        char[][] word_matrix;
        String[] answer_matrix;
        int[] len_summary;
        try (Scanner readFile = new Scanner(System.in)) {
            while (file_nonexistant) {
                System.out.print("Insert your path to the textfile here (please watch the formatting) : ");
                String filename = readFile.next();
                File puzzlefile = new File("../testfile/" + filename);
                if (puzzlefile.exists()) {
                    // System.out.println("File exists\n");
                    file_nonexistant = false;
                    word_matrix = FileProcessor.read_words_matrix(puzzlefile);
                    answer_matrix = FileProcessor.read_answers_matrix(puzzlefile);
                    len_summary = FileProcessor.length_word_answers(puzzlefile);
                    BruteforceMethod.all_methods(word_matrix, answer_matrix, len_summary);
                } else {
                    System.out.println("File does not exists, try again!");
                }
            }
        }


    }
}
```

III. Hasil Eksekusi Program

Untuk menguji coba program, program akan diuji menggunakan sembilan file extension .txt yang berisikan tiga ukuran puzzle berbeda. Format nama file yang digunakan adalah <ukuran puzzle>_<nomor file>.txt.

Uji coba pertama dilakukan dengan menggunakan file “small_1.txt” dengan dimensi matriks hurufnya 16x14 dan jumlah kata kunci yang harus dicari sebanyak 14 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat small_1.txt
M P Q W V Q S O J E T O X G E P
O X T A I L S P J M C T P Q R U
D O W X P M Y E K C O J S B R Z
I O H E L C W W T E I C F B Q V
O L K A H M N D I T I V Y L W S
H Y C D B U N P Y U P R I S E B
W E R I C N F H X O R C T M J S
B M U N A T J R N P K A S I C Q
W M T M H P J U O S T J Y H N Q
E J D T P M Y P A P I W Y B Q K
T I A G L E D W W O P P I S W A
D S Q F A T B E Q R E I I N J R
Y L E M O H U U N T P U X D Q K
G D W B T E I U Q Y L O H O F I

ALPHA
BECALM
GAIT
HOMELY
HYPE
JOCKEY
KNIT
OXTAIL
QUIET
SELF
SPORTY
TEMPT
TWIN
UPRISE
```

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : small_1.txt

Kata ALPHA ditemukan arah north di :

---------------
---------------
---------------
---------------
---------------
---------------
---------------
---A-----------
---H-----------
---P-----------
---L-----------
---A-----------
---------------
---------------

Kata BECALM ditemukan arah northeast di :

---------------
---------------
-----M---------
---L-----------
--A------------
-C-------------
E--------------
---------------
---------------
---------------
---------------
---------------
---------------

Kata GAIT ditemukan arah west di :

---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
TIAG-----------
---------------
```

```
Kata HOMELY ditemukan arah west di :

---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
YLEMOH----------
---------------
```

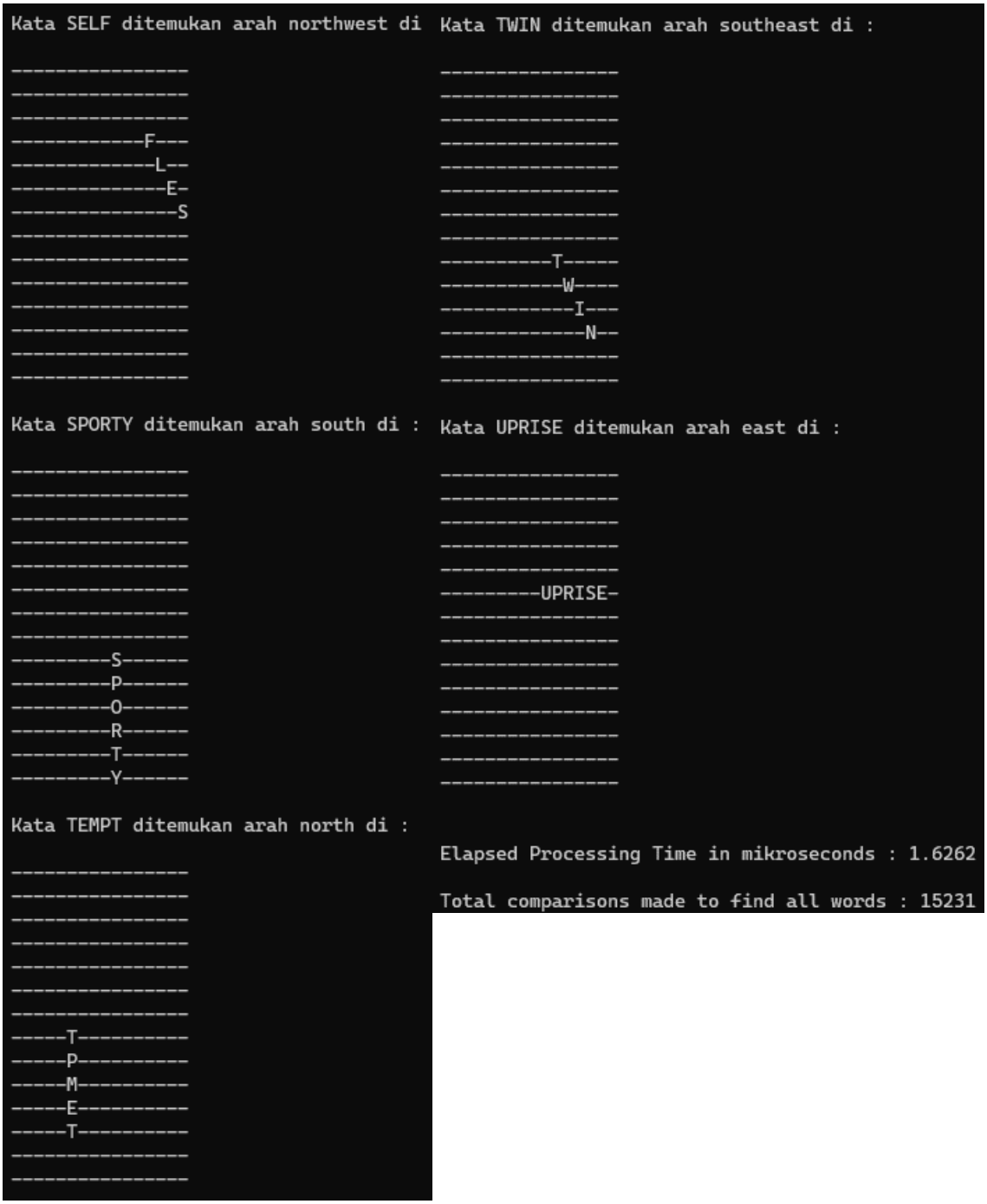
```
Kata KNIT ditemukan arah northwest di :

---------------
---------------
---------------
---------------
---------------
---------------
----------T---
---------I--
--------N-
-------K
---------------
---------------
---------------
---------------
```

```
Kata HYPE ditemukan arah southwest di :

---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
----------H--
---------Y---
--------P----
-------E-----
---------------
---------------
```

```
Kata OXTAIL ditemukan arah east di :

---------------
OXTAIL----------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
```

```
Kata JOCKEY ditemukan arah west di :

---------------
---------------
------YEKCOJ----
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
```

```
Kata QUIET ditemukan arah west di :

---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
---------------
----TEIUQ-------
```



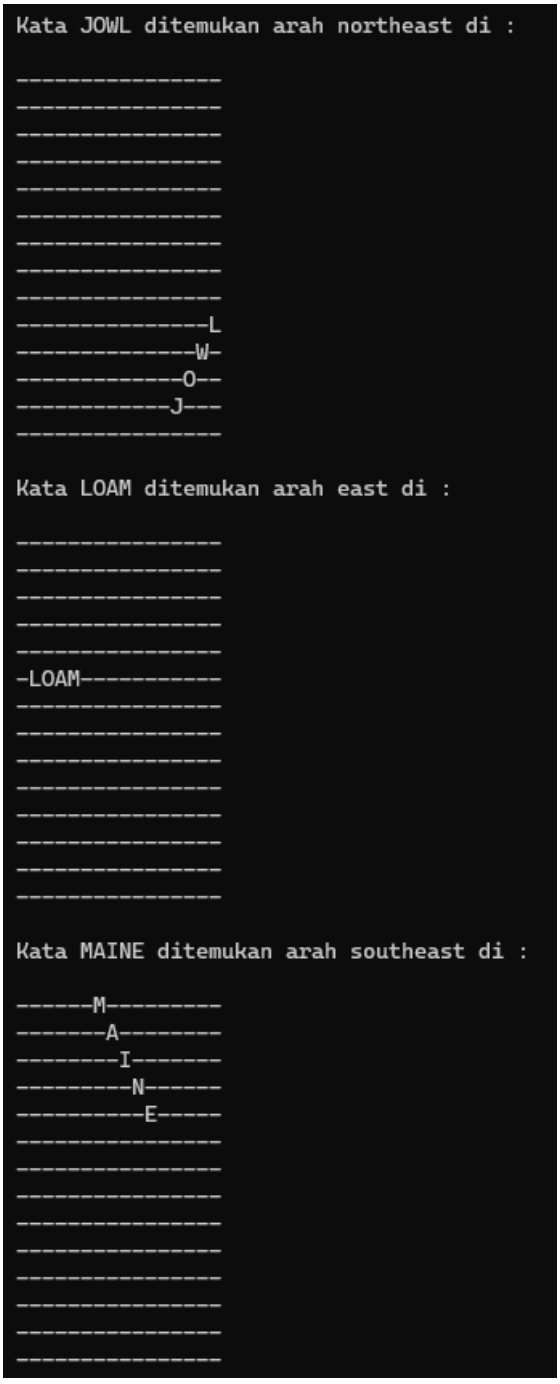
Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 1.6262 mikrodetik dan dibutuhkan 15,232 kali perbandingan huruf secara keseluruhan.

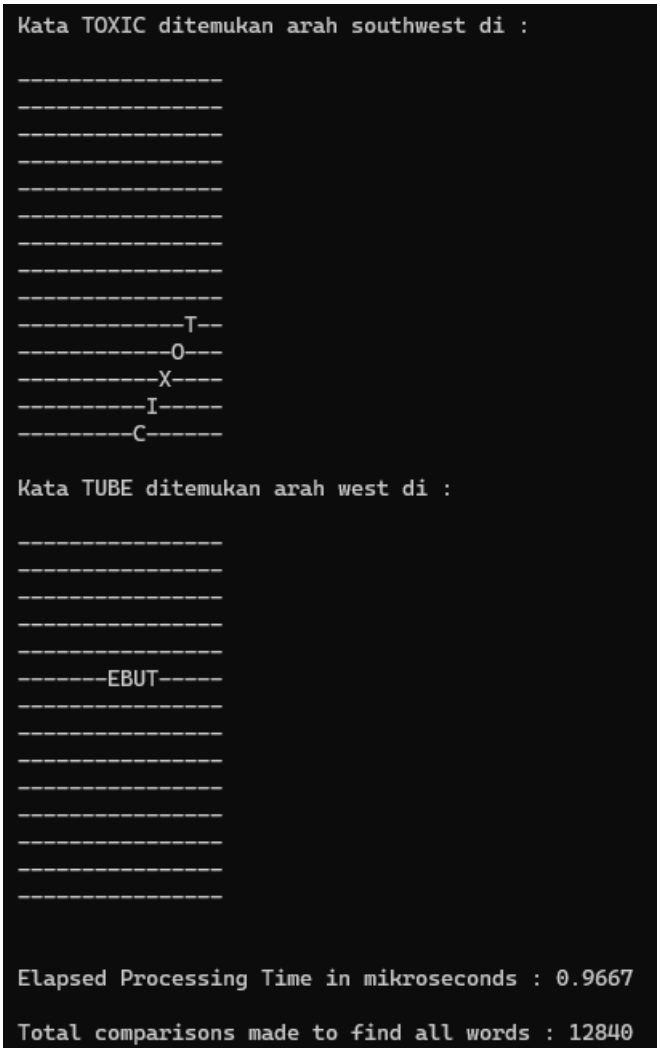
Uji coba kedua dilakukan dengan menggunakan file “small_2.txt” dengan dimensi matriks hurufnya 16x14 dan jumlah kata kunci yang harus dicari sebanyak 14 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat small_2.txt
P F Y W W K M Z E X O L Q S B H
G K E X S S X A K K S X I K O J
T N T T T D F H I U R X J N C X
A F N R L R C C I N B A J H S D
K B R S F S Q T R O E C O B Z M
L L O A M Z E E B U T C O C K Y
S E N R S J G K B V L E B I U F
I M C A L N N P T L K J V Y Q Z
B O W X I K H H I A P L A C O F
K A A L E L W R E Y K S P T V L
W P D R P C H U N M P P O O W V
R I O I A T G I I G U X J O W D
S M Q X E N X U R Z I F J X U Y
E W X Q E Q G G B C T U E S M C

ANEW
BRINE
COCKY
EXCEL
FOCAL
IDLING
JOWL
LOAM
MAINE
ROME
SUITE
THRILL
TOXIC
TUBE
```

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : small_2.txt

Kata ANEW ditemukan arah northeast di :

---W------------
--E------------
-N-------------
A---------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------

Kata BRINE ditemukan arah north di :

----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
--------E-------
--------N-------
--------I-------
--------R-------
--------B-------

Kata COCKY ditemukan arah east di :

----------------
----------------
----------------
----------------
----------------
----------COCKY
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
```


Kata EXCEL ditemukan arah northwest di :
L
-E
--C
---X
----E
Kata FOCAL ditemukan arah west di :
----------LACOF
Kata IDLING ditemukan arah northeast di :
------G
-----N
----I
---L
--D
-I


Kata JOWL ditemukan arah northeast di :
----------L
----------W-
----------O--
----------J---
Kata LOAM ditemukan arah east di :
-LOAM-----
Kata MAINE ditemukan arah southeast di :
------M---------
-------A--------
--------I-------
---------N------
----------E-----



Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 0.9667 mikrodetik dan dibutuhkan 12,480 kali perbandingan huruf secara keseluruhan.

Uji coba ketiga dilakukan dengan menggunakan file “small_3.txt” dengan dimensi matriks hurufnya 16x14 dan jumlah kata kunci yang harus dicari sebanyak 14 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat small_3.txt
P L U T O F Q X P G N O S X N D
M L X Y E G F S Y Q B Q A O I O
L B Z D P B E R I I Q K C V L F
R E M N U L A I J D V R I T X Q
Q N T T Y M J C X V A N L C W T
B K Y E L T T I L V E B E F D P
S H I N E D Q P A R T X T R H V
J Z U V F E K T E N P F C Y S E
J H O D B F B U T W U A E L H L
U B L M T E G O C X X T P W A B
A M Q T T G C X U F O B S N D U
S A T O X W U A E L O M A H Y K
M B Q A T T F E M T S O L B R A
I S U U P S H O T E L Z Q H M W

ABOVE
ASPECT
BECAME
CRAVAT
DIVINE
LITTLE
LOST
MOLE
PLUTO
SHADY
SHINE
SONG
STUN
UPSHOT
```

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : small_3.txt

Kata ABOVE ditemukan arah northeast di :

----------------
----------------
----------------
----------------
----------------
----------------
----E-----------
---V-----------
--O------------
-B-------------
A---------------
----------------
----------------
----------------

Kata ASPECT ditemukan arah north di :

----------------
----------------
----------------
----------------
----------------
----------------
----------T---
----------C---
----------E---
----------P---
----------S---
----------A---
----------------
----------------

Kata BECAME ditemukan arah southeast di :

----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----B-----------
-----E----------
------C---------
-------A--------
-------M--------
--------E-------
```

```
Kata CRAVAT ditemukan arah southwest di :

----------------
----------------
----------C---
----------R----
----------A-----
---------V------
--------A-------
-------T--------
----------------
----------------
----------------
----------------
----------------
----------------

Kata DIVINE ditemukan arah southwest di :

---------------D
--------------I-
-------------V--
------------I---
-----------N----
----------E-----
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------

Kata LITTLE ditemukan arah west di :

----------------
----------------
----------------
----------------
----------------
---ELTTIL-------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
```

```
Kata LOST ditemukan arah west di :

----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
---------TSOL---
----------------

Kata MOLE ditemukan arah west di :

----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
--------ELOM----
----------------
----------------

Kata PLUTO ditemukan arah east di :

PLUTO-----------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
----------------
```

Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 1.1486 mikrodetik dan dibutuhkan 15,077 kali perbandingan huruf secara keseluruhan.

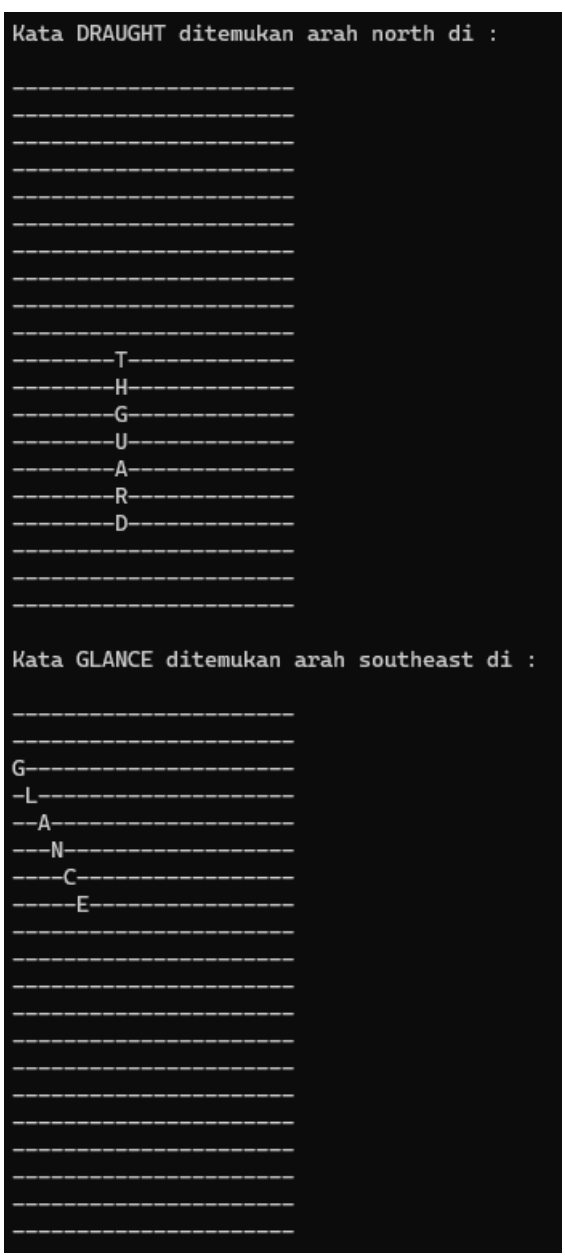
Uji coba keempat dilakukan dengan menggunakan file “medium_1.txt” dengan dimensi matriks hurufnya 22x20 dan jumlah kata kunci yang harus dicari sebanyak 20 kata.

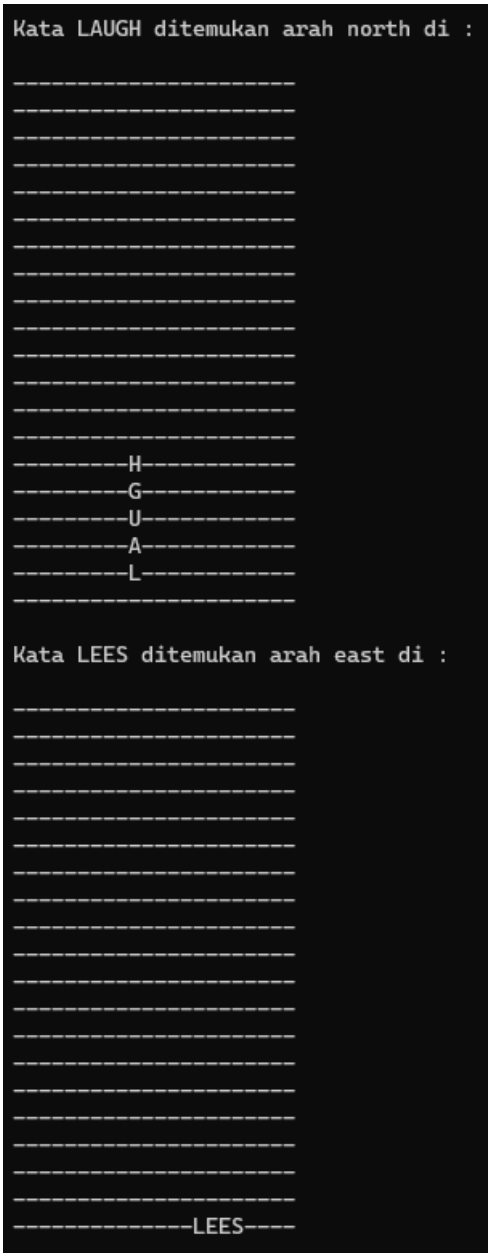
```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat medium_1.txt
O H E C W W T E I C B Q V O L K H M N D I T
A S S E R T I N G I V Y W S H Y D B U N P Y
G B W R I C N F H S X O R C M J M U N J R N
P L K A S C Q S S E L E C R U O S W M T M J
U O A J Y R Q E E J G N I S I P S E D D T Y
P A I N A B R Q D W W P S W A D S Q F B E Q
I I J C C P R G U U N P U X D Q K G D W B L
O H S O M E F U I C V D H N E W S F L A S H
U O X O U T P M J W F W F G O Z B C N L C G
X G C D D Y T S C B B I X K L Y V Y N V S K
G E A N Y K R G T P G R Y L X A Y P T U O D
D F J A A M O K H C S R W B V A R R U F N R
D R S U M I P A G W Q U S C O Y B B H D E Z
Q H X J H G S S U F E W R G N X S A C X L L
C U E Z T H S U A H W F A U L A L T S C K I
Z B G E S T A G R G D Q X R A Y I F U H Q R
X L X R A D P Q D U O K Y O P S M P N G K J
A U F B K P B F E A Q L P S L E E R K V P F
Y W K Z E X O L Q L S B H G K X D H S S X K
K X I K O S U I S L E C J T L E E S T T T T

ABASH
ASSERTING
ASTHMA
CELSIUS
DECOMPRESS
DESPISING
DRAUGHT
GLANCE
LAUGH
LEES
LEFTY
MIGHT
NEWSFLASH
OSCAR
PASSPORT
SLIME
SMUG
SOURCELESS
THESAURUS
WARPED
```

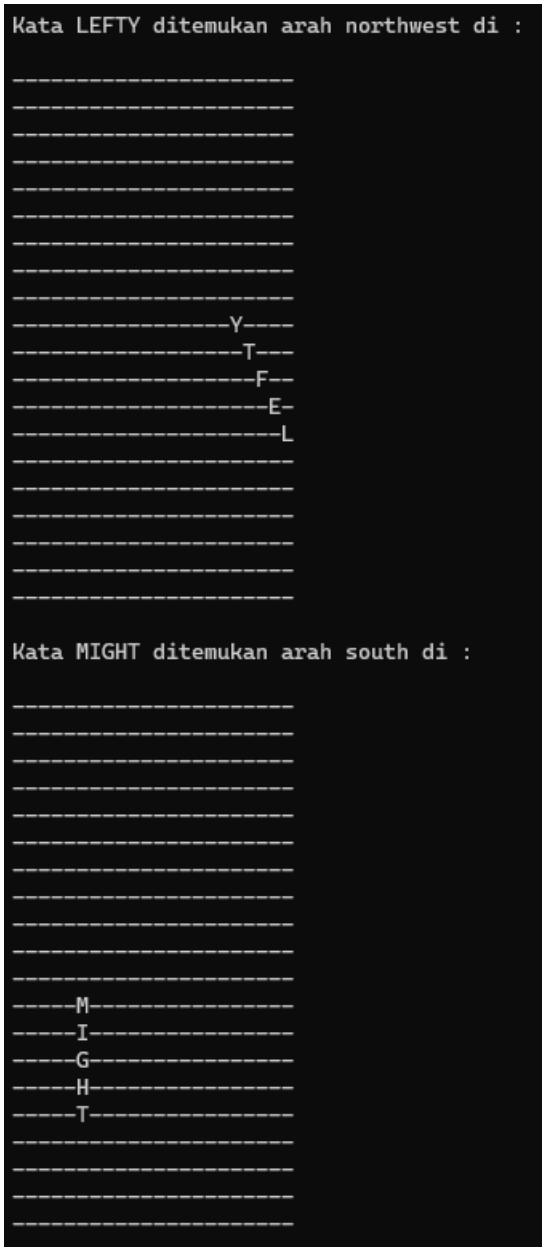
```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : medium_1.txt

Kata ABASH ditemukan arah southeast di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
---------------A------
----------------B-----
-----------------A----
------------------S---
-------------------H--
----------------------
----------------------
----------------------
----------------------

Kata ASSERTING ditemukan arah east di :

----------------------
ASSERTING------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

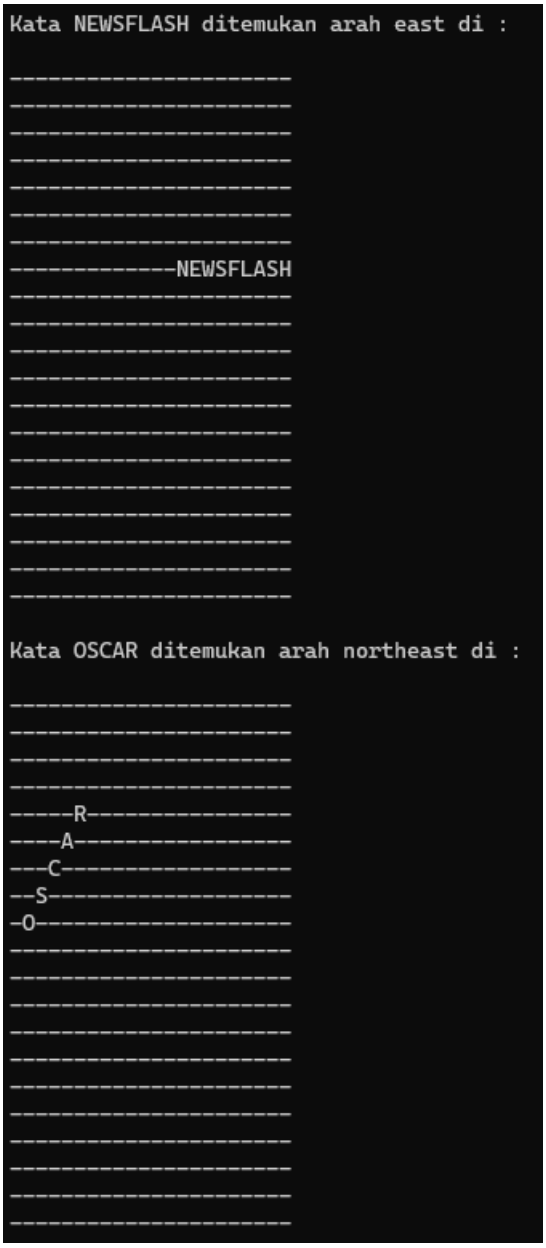
```
Kata ASTHMA ditemukan arah north di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----A-----------------
----M-----------------
----H-----------------
----T-----------------
----S-----------------
----A-----------------
----------------------
----------------------
----------------------

Kata CELSIUS ditemukan arah west di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
-----SUISLEC-----------
```

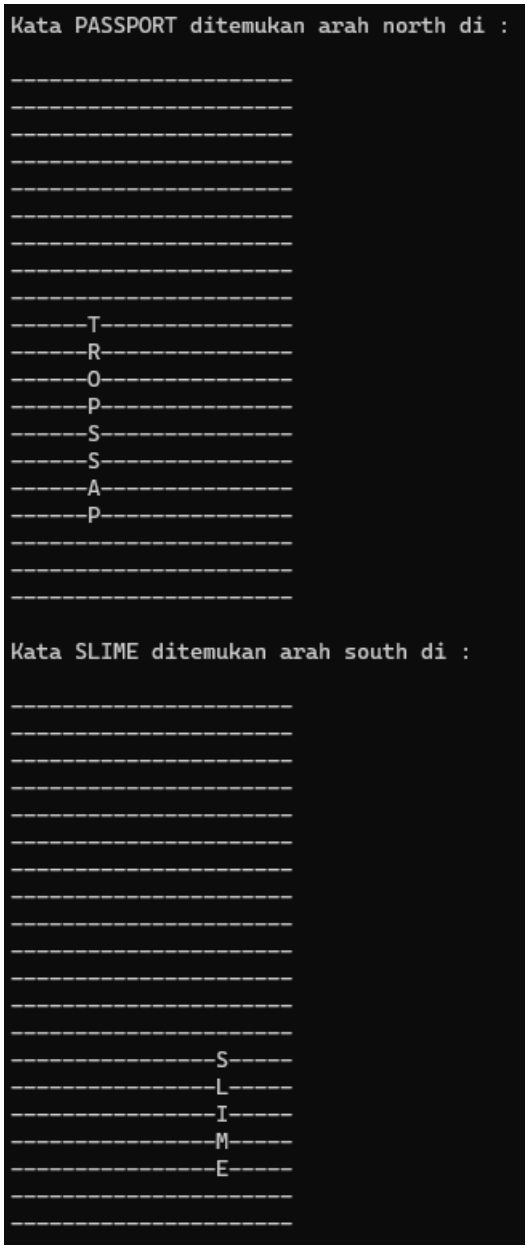
```
Kata DECOMPRESS ditemukan arah northeast di :

----------------------
----------------------
---------S------------
--------S-------------
-------E--------------
------R---------------
-----P----------------
----M-----------------
---O------------------
--C-------------------
-E--------------------
D---------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------

Kata DESPISING ditemukan arah west di :

----------------------
----------------------
----------------------
----------------------
----------GNISIPSED---
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

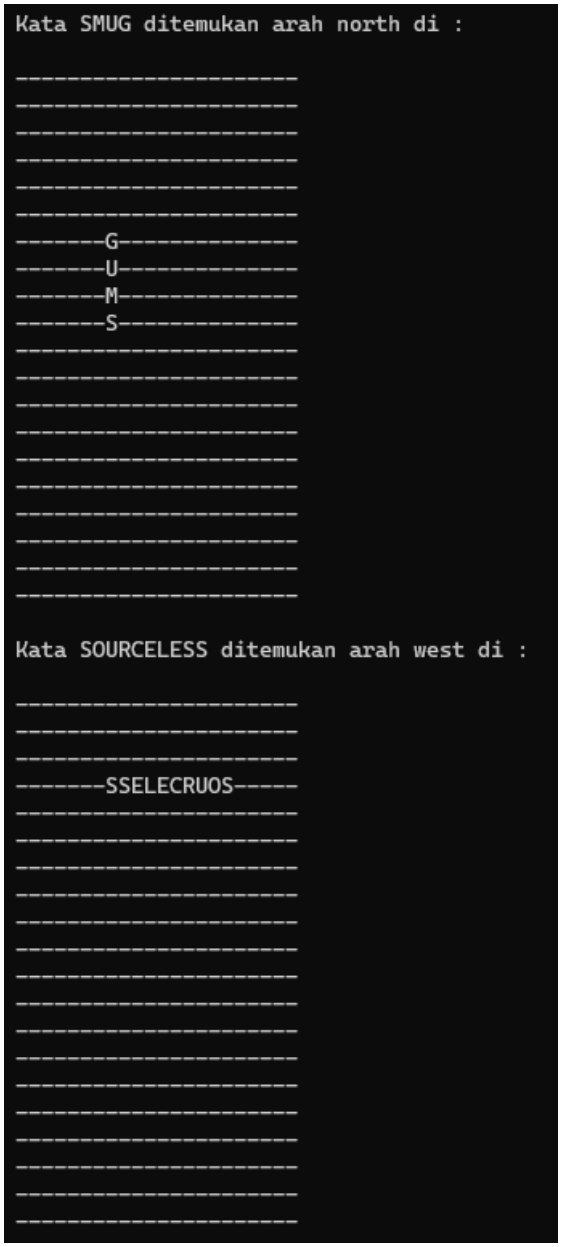

Kata DRAUGHT ditemukan arah north di :
Kata GLANCE ditemukan arah southeast di :

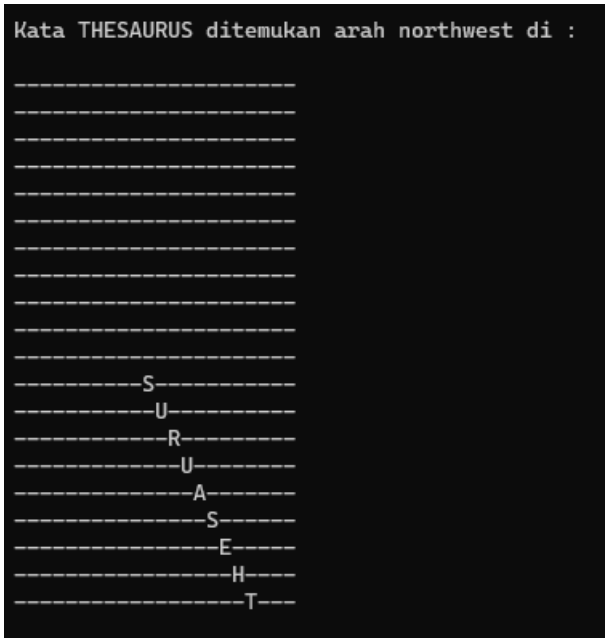

Kata LAUGH ditemukan arah north di :
Kata LEES ditemukan arah east di :

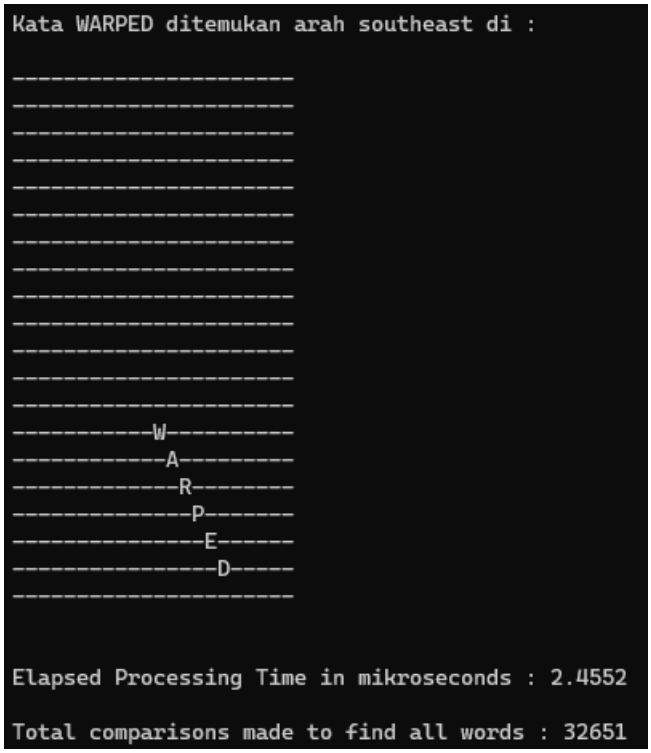

Kata LEFTY ditemukan arah northwest di :
Y
T
F
E
L
Kata MIGHT ditemukan arah south di :
M
I
G
H
T


Kata NEWSFLASH ditemukan arah east di :
NEWSFLASH
Kata OSCAR ditemukan arah northeast di :
R
A
C
S
O


Kata PASSPORT ditemukan arah north di :
Kata SLIME ditemukan arah south di :


Kata SMUG ditemukan arah north di :
Kata SOURCELESS ditemukan arah west di :



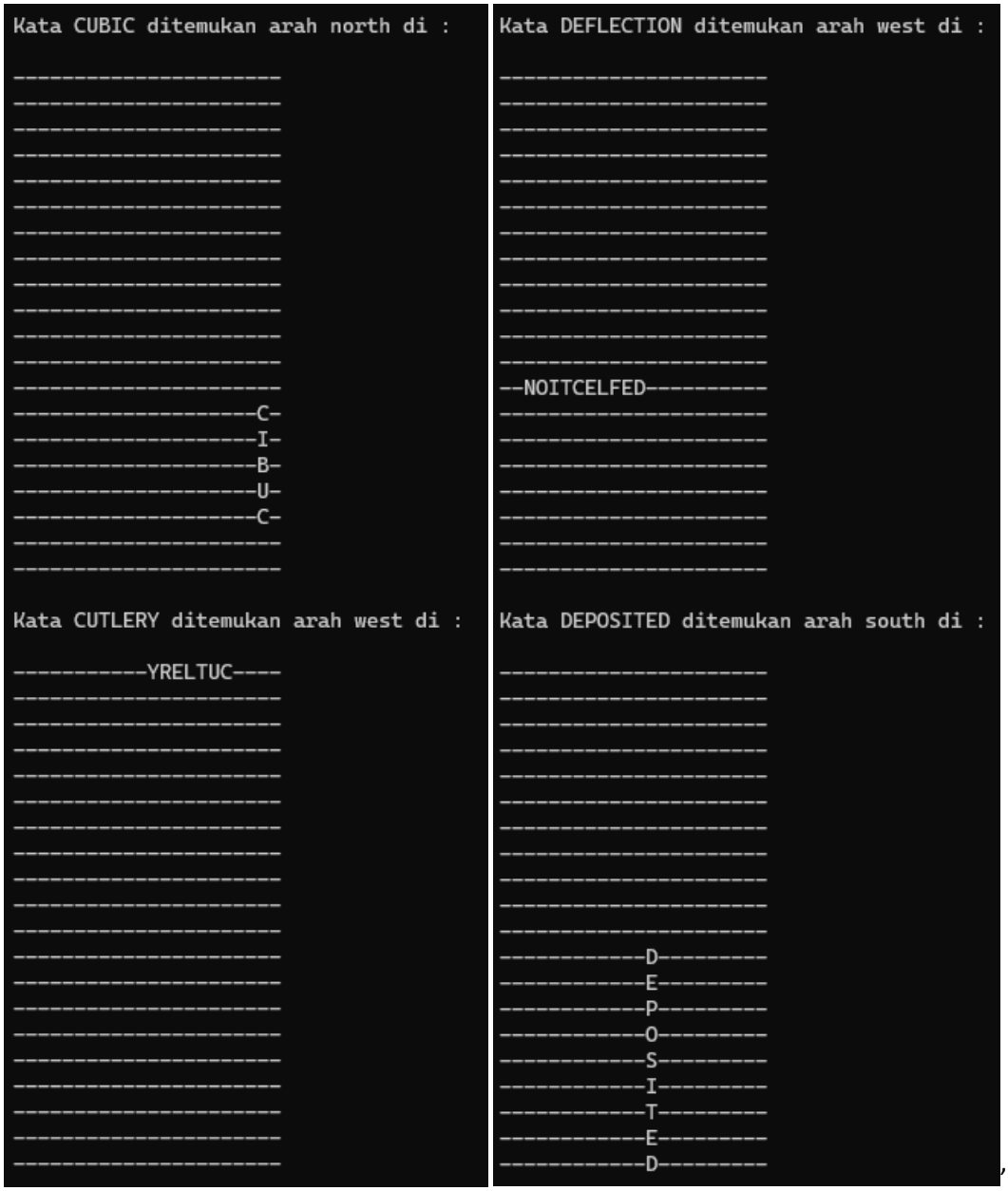


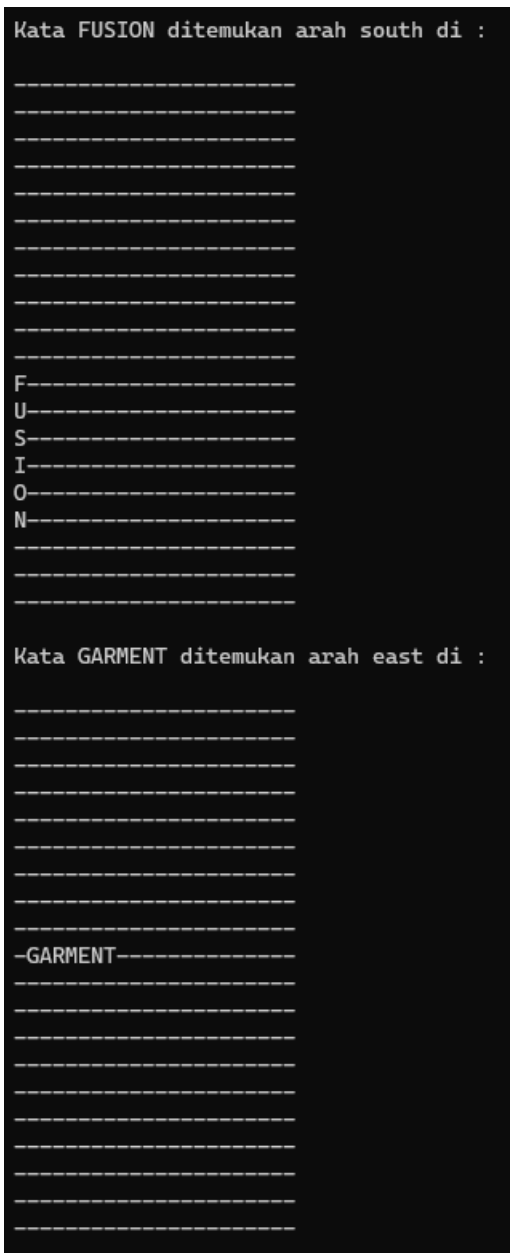
Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 2.4552 mikrodetik dan dibutuhkan 32,651 kali perbandingan huruf secara keseluruhan.

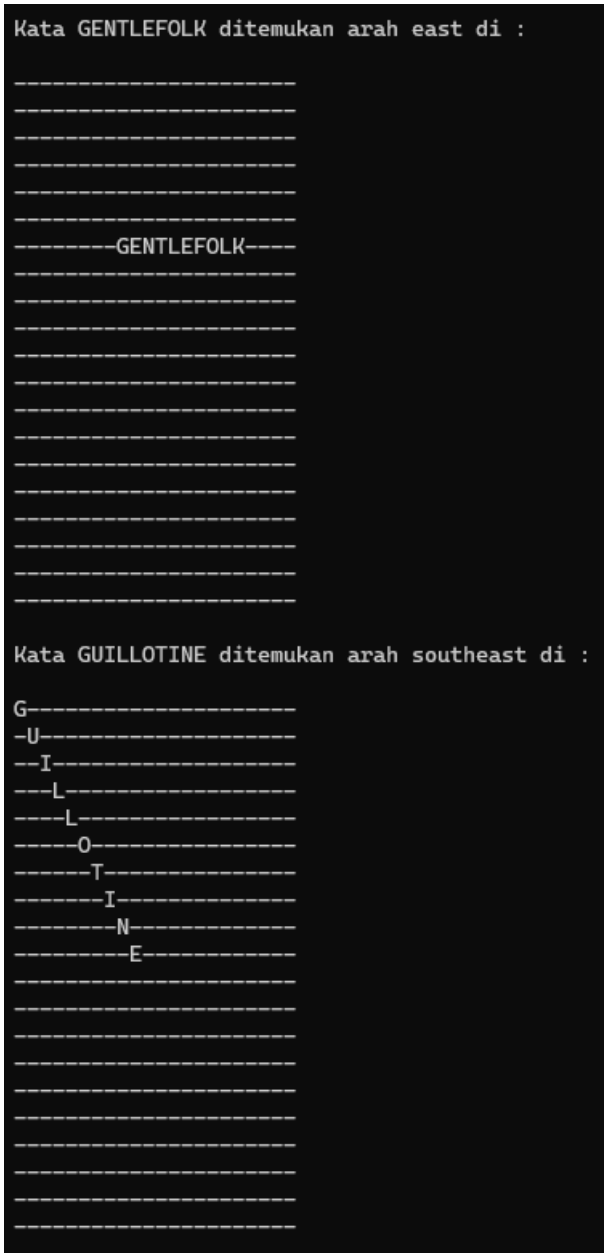
Uji coba kelima dilakukan dengan menggunakan file “medium_2.txt” dengan dimensi matriks hurufnya 22x20 dan jumlah kata kunci yang harus dicari sebanyak 20 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat medium_2.txt
G H A Y F E V E R U L Y R E L T U C A F K E
X U N S P R I N G I E S T G T X F A U F O U
W I I O Z W Q F V J J A O K D I X F Y D W X
L F Q L O Q L A W D G H R O Y P U K V T D N
A A T B L A S P H E M E R I X M T B Q A R Z
N S G V Q O P G V L T J A R U E F U L L Y M
Q K J A H P T W G E N T L E F O L K L Y J N
A Z W L S N O I L I V A P I P M O B J G Y W
Z P P B K H M C N W H U Y P F H O W G H S W
G G A R M E N T E E L X O A Z B Q G E D G M
Y B C I H N M K Y P H M L K Y O V E U U P W
F I Y G Y S R D U M E A D A B L W L D S A F
U V N O I T C E L F E D E U R U R B H P U X
S V G M A B A J G S K E P H B O E A T L C D
I B N J U E X D A U E V O G E O M I J I I O
O R W Y Y C D O U X L B S W E V K L O T B R
N L H Y Z S P H D F P A I M P J F E M T U R
Y R Z G N G X M I N O I T J E E H R F I C X
T U Q R D E C N U O R T E E R H T B S N W I
B M A V H A Z I N E S S D Q O P O U Q G S T

BEEPER
BLASPHEMER
CUBIC
CUTLERY
DEFLECTION
DEPOSITED
FUSION
GARMENT
GENTLEFOLK
GUILLOTINE
HAYFEVER
HAZINESS
MORAL
PAVILION
REGULATE
RELIABLE
RUEFULLY
SPLITTING
SPRINGIEST
TROUNCED
```

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : medium_2.txt

Kata BEEPER ditemukan arah south di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------B-------
----------E-------
----------E-------
----------P-------
----------E-------
----------R-------
----------------------

Kata BLASPHEMER ditemukan arah east di :

----------------------
----------------------
----------------------
----------------------
---BLASPHEMER---------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

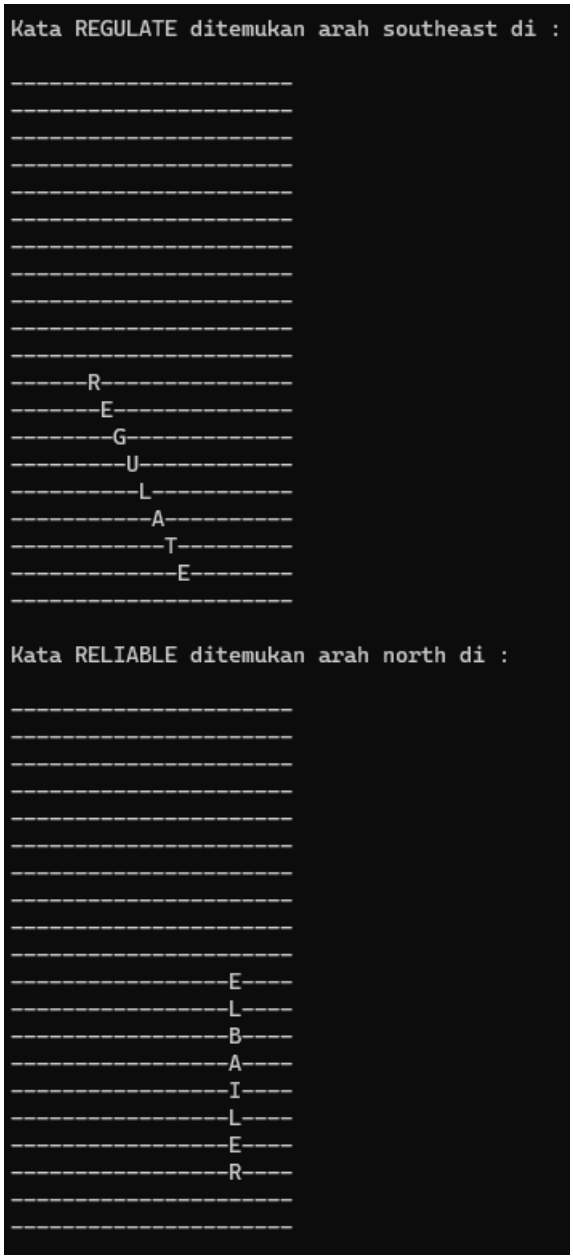

Kata CUBIC ditemukan arah north di :
Kata DEFLECTION ditemukan arah west di :
Kata CUTLERY ditemukan arah west di :
Kata DEPOSITED ditemukan arah south di :


Kata FUSION ditemukan arah south di :
F
U
S
I
O
N
Kata GARMENT ditemukan arah east di :
GARMENT


Kata GENTLEFOLK ditemukan arah east di :
GENTLEFOLK
Kata GUILLOTINE ditemukan arah southeast di :
G
U
I
L
L
O
T
I
N
E

```
Kata HAYFEVER ditemukan arah east di :

-HAYFEVER----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------

Kata HAZINESS ditemukan arah east di :

----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----HAZINESS----------
```

```
Kata MORAL ditemukan arah northwest di :

----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------L---------
----------A--------
----------R-------
----------O------
----------M-----
----------
----------
----------
----------
----------

Kata PAVILION ditemukan arah west di :

----------
----------
----------
----------
----------
----------
----------
-----NOILIVAP---------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
----------
```


Kata REGULATE ditemukan arah southeast di :
R
E
G
U
L
A
T
E


Kata RUEFULLY ditemukan arah east di :
RUEFULLY

Kata RELIABLE ditemukan arah north di :
E
L
B
A
I
L
E
R

Kata SPLITTING ditemukan arah south di :
S
P
L
I
T
T
I
N
G

Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 2.6449 mikrodetik dan dibutuhkan 36,024 kali perbandingan huruf secara keseluruhan.

Uji coba keenam dilakukan dengan menggunakan file “medium_3.txt” dengan dimensi matriks hurufnya 22x20 dan jumlah kata kunci yang harus dicari sebanyak 20 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat medium_3.txt
D Z N F B F F F F Y P T H R U X Y B R I N Y
V G C B X D X I R R E G U L A R J Y J I R R
X J L R G M N U U T M V O H J B X T J H F E
I B W G K D W P U S O L I D F L T L Q U E C
I I N E R T N E S S Z E S F W B T D A D C O
O O N F C H B L K D G C V N G Z X E D H D V
S M H I N V N A B V O N C N Y S M B L X I E
O J W R D N C N H O U E J W T M J I L P A R
J Q O M V E A I C B F R G K S L D R O W R U
F Y E N U J I F D P A E Y Y N K A C G X B P
P R E S C R I B E D D H A R A N X S Q F K Z
C K N I I Q G H N X M D I U L R L N P K X D
R U R U G U A Y D G C A C J U R V I M L Q B
K Q C V J W N Z K S Q J V R T W B T J G V I
U H O E D T K H E O G I F E A R S R Q B I B
Y I H E R E W O H S J H Y P P G G Q T P H L
S I S A S Q D E S U M A S J S N W P M A R I
O J O T T O T A L L Y A A O T H U V S H O C
K C K C R E R R T X T O G G N I K C A S C A
T S E N O H S I D G T Z M F U P J M Z J Y L

ADHERENCE
AMUSE
BIBLICAL
BRAID
BRINY
DISHONEST
FINAL
INERTNESS
INSCRIBED
IRREGULAR
JUNE
PERJURY
PRESCRIBED
RECOVER
SACKING
SHOWER
SOLID
SPATULA
TOTALLY
URUGUAY
```

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : medium_3.txt

Kata ADHERENCE ditemukan arah north di :

----------------------
----------------------
----------------------
----------------------
----------E-----------
----------C-----------
----------N-----------
----------E-----------
----------R-----------
----------E-----------
----------H-----------
----------D-----------
----------A-----------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------

Kata AMUSE ditemukan arah west di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
-------ESUMA----------
----------------------
```

```
Kata BIBLICAL ditemukan arah south di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------B
----------I
----------B
----------L
----------I
----------C
----------A
----------L

Kata BRAID ditemukan arah north di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------D-
----------I-
----------A-
----------R-
----------B-
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

```
Kata BRINY ditemukan arah east di :

----------------BRINY
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------

Kata DISHONEST ditemukan arah west di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
TSENOHSID------------
```

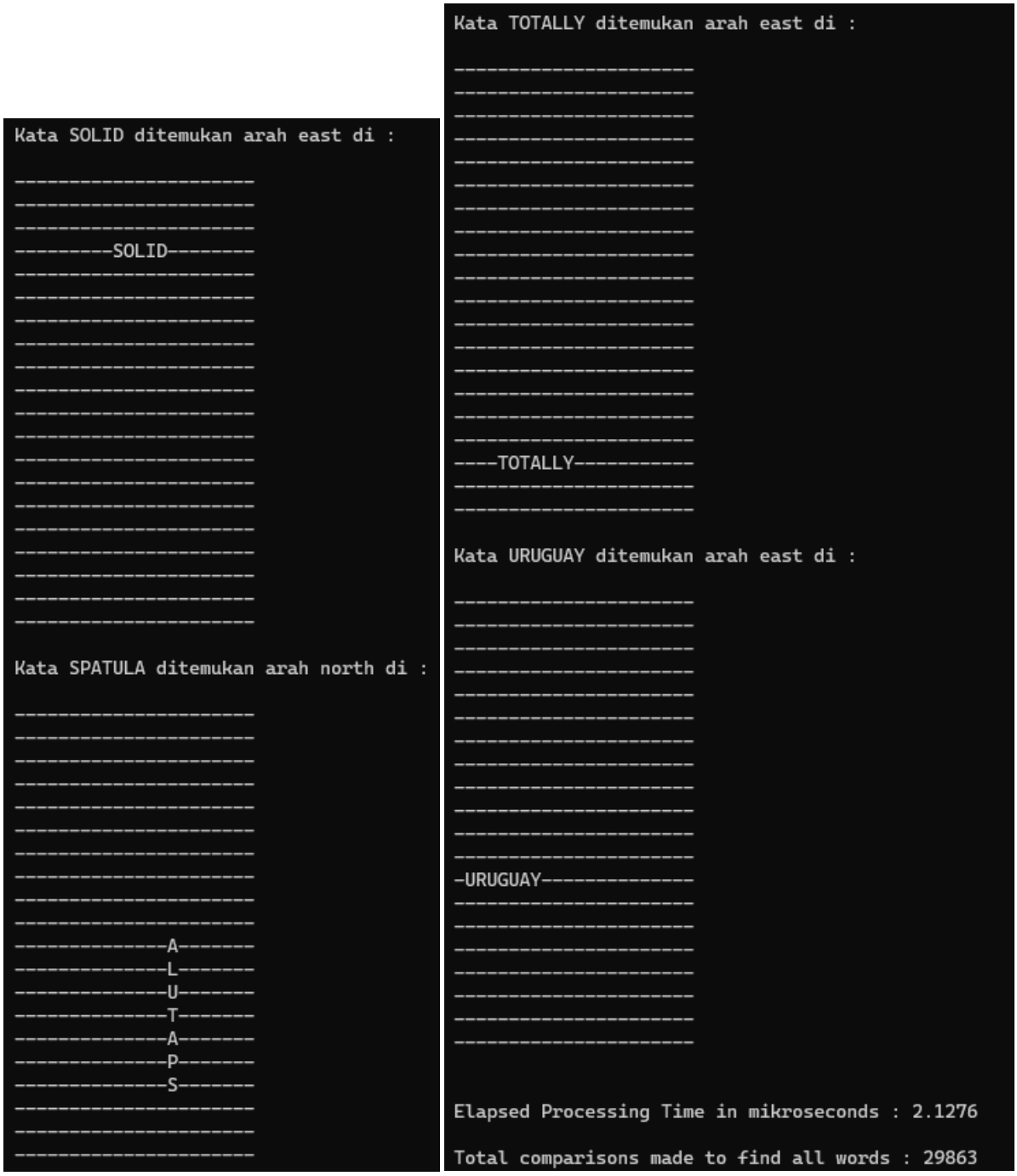
```
Kata FINAL ditemukan arah north di :

----------------------
----------------------
----------------------
----------------------
----------------------
-------L--------------
-------A--------------
-------N--------------
-------I--------------
-------F--------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------

Kata INERTNESS ditemukan arah east di :

----------------------
----------------------
----------------------
----------------------
-INERTNESS------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

```
Kata INSCRIBED ditemukan arah north di :

----------------------
----------------------
----------------------
----------------------
----------------D-----
----------------E-----
----------------B-----
----------------I-----
----------------R-----
----------------C-----
----------------S-----
----------------N-----
----------------I-----
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------

Kata IRREGULAR ditemukan arah east di :

----------------------
-------IRREGULAR------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

```
Kata JUNE ditemukan arah west di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
--ENUJ----------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------

Kata PERJURY ditemukan arah north di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
-------------Y--------
-------------R--------
-------------U--------
-------------J--------
-------------R--------
-------------E--------
-------------P--------
----------------------
----------------------
----------------------
----------------------
```

```
Kata PRESCRIBED ditemukan arah east di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
PRESCRIBED------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------

Kata RECOVER ditemukan arah south di :

----------------------
---------------------R
---------------------E
---------------------C
---------------------O
---------------------V
---------------------E
---------------------R
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
```

```
Kata SACKING ditemukan arah west di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
-------------GNIKCAS--
----------------------

Kata SHOWER ditemukan arah west di :

----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----------------------
----REWOHS------------
----------------------
----------------------
----------------------
----------------------
```



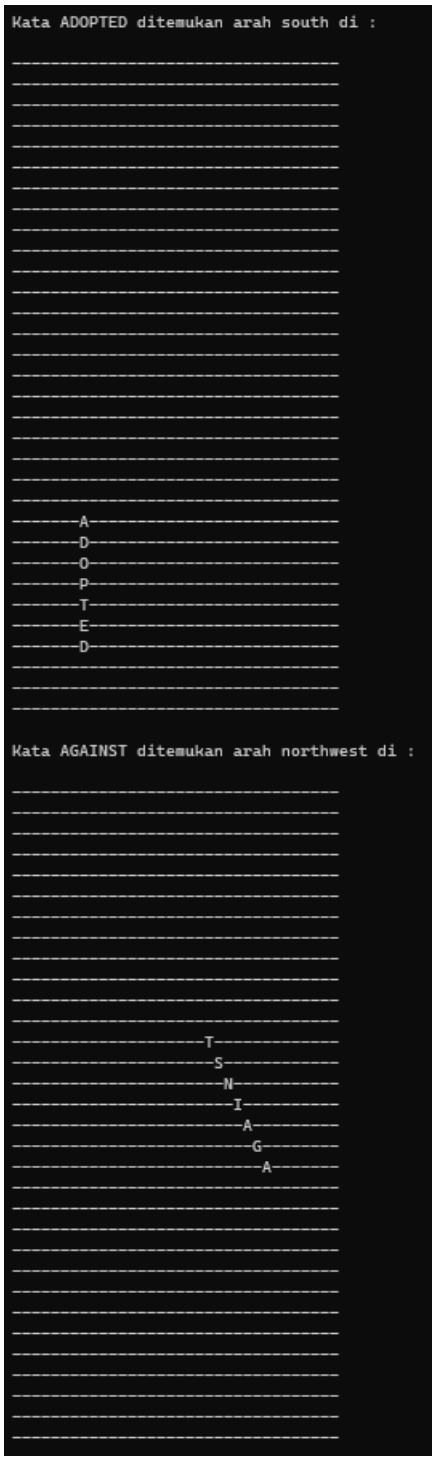
Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 2.1276 mikrodetik dan dibutuhkan 29,863 kali perbandingan huruf secara keseluruhan.

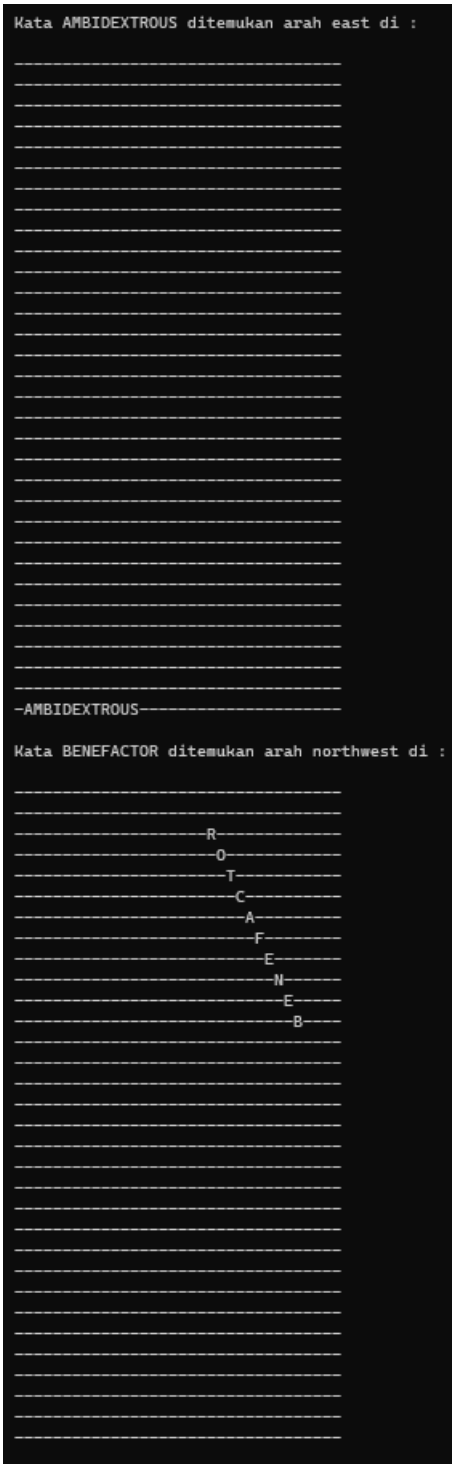
Uji coba ketujuh dilakukan dengan menggunakan file “large_1.txt” dengan dimensi matriks hurufnya 34x32 dan jumlah kata kunci yang harus dicari sebanyak 48 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat large_1.txt
A S Q K K Y R E N N A C Y E I K I S X D V T J K H O N L Y Z T Q Z Z
M B V G K B P A Z H K C E R U T A E F C M D B S C A P E G O A T B Q
Q C A V S D C F L C Q F H B O Y L Z W B R T C F M A W B I U U I I N
W R F T R V J M K N M S P F U Y P G O Q H O P I A G D I U F D D F U
P E C A E P O Q R Q P U D K N J R Y T S I M T Y P K F T N Z U I W T
Q A T Y H M H G B K E P A C S Y T I C A K R P C F U G T H D M J A T
E K W D L Q E R E L J H T M U G U T F E E B L Y A N M Q F H U R S P
Y I B O J D I N H Z F S K D E V B B P V R P E Q X F T W Z I C C F S
M N R C M O A J T W A P K X F W I C G I S B Y S X R E V O U O O E B
A G S Y N C H R O N I Z E P Z H F D N G W E A H F L E N A O I C F D
T P I S U Y W X Y A B R E L G X O E I J V I N U R Y E E E C D F Z S
P F M D E M I O M G T E T I B B C R D X Y U J Y P D R F T B I Y V E
N U H Q A S S H F I J M A H H O A I E D T L I J P U G P Q M G K T D
F E E S T R U W N Y D U V I E V L A E L H S T A B H E E A R F A Y S
Q N E K I L A G R T O T T E I C E P C U C F N F T W D E E O B Q C E
W E W C T R J P O B O H D X K X K S X C G C P I S H Y L Y R Q Z F D
M P U N I S H I N G R K I C B N L E E E I V V D A O C R E J H O Q X
C E P A M D N I H T K C R H O W S D A V W B N P C G U C E P J G R T
U D A F Z S Q L B V N W I Z Y W O J T P I I Z C K X A Y Z F N H P U
R E I Q J B R P Y I O R S E R S X Z Z O W E A U S X Y L B I I W B S
C J J I X E K O O S B R Z R T V M K U O T N E M E M U Z K E H N A G
K H A E G W J W T K C Z P F R A Y E N N D H T A W T A A Q N O R E E
V R G V O Q C A Y C Y X O W E E T K H D Z A J H U L W M H I S P Z R
Q E N I S P M D I S O Q R H P U K I U U S I N C E R I T Y T R A T R
Y S I S X C Z O G P X R Q W R B N Q V H L G O Q D R E C W N Z Y C F
Z E L S U H T P D A A A P G O J E V V A O J C C G H E Y F E S H Q E
M M D E E G N T X R D X F R V E Z H Q K R W L L E B F J U P B F C H
H B N S V R E E N T P Y V A E J F M M R I G I N X R Z X R R Q S A C
V L O B F V R D K A L X V V T U Y J H O J P Y G G W E K Y U Q U J W
I E F O Z W M G T N Z S I Y V J D F V P I D U J I R C V B T B U O D
F W G S S A P R U S S Y Q G N I Z I T A M A R D D R E E E Q V O J M
N A M B I D E X T R O U S Q K S B B I V F R Y X K F V Q F R H J S T
```

```
ABATEMENT
ADOPTED
AGAINST
AMBIDEXTROUS
BENEFACTOR
BIFOCAL
BUREAUCRAT
CANNERY
CHEF
CITYSCAPE
CLERGY
CREAKING
DEGREE
DESPAIRED
DOORKNOB
DRAMATIZING
DRAT
EXACERBATE
EXCEEDING
EXERTING
FEATURE
FEEBLY
FONDLING
GRAVITATE
GRAVY
HIND
INDUCED
JEEP
LIKEN
MEMENTO
MISTY
OBSESSIVE
PARADIGM
PILGRIM
PROCTORS
PUNISHING
REFINER
REPROVE
RESEMBLE
REVERE
SCAPEGOAT
SINCERITY
SPARTAN
SURPASS
SYNCHRONIZEP
TURPENTINE
WAKING
WINDSWEPT
```

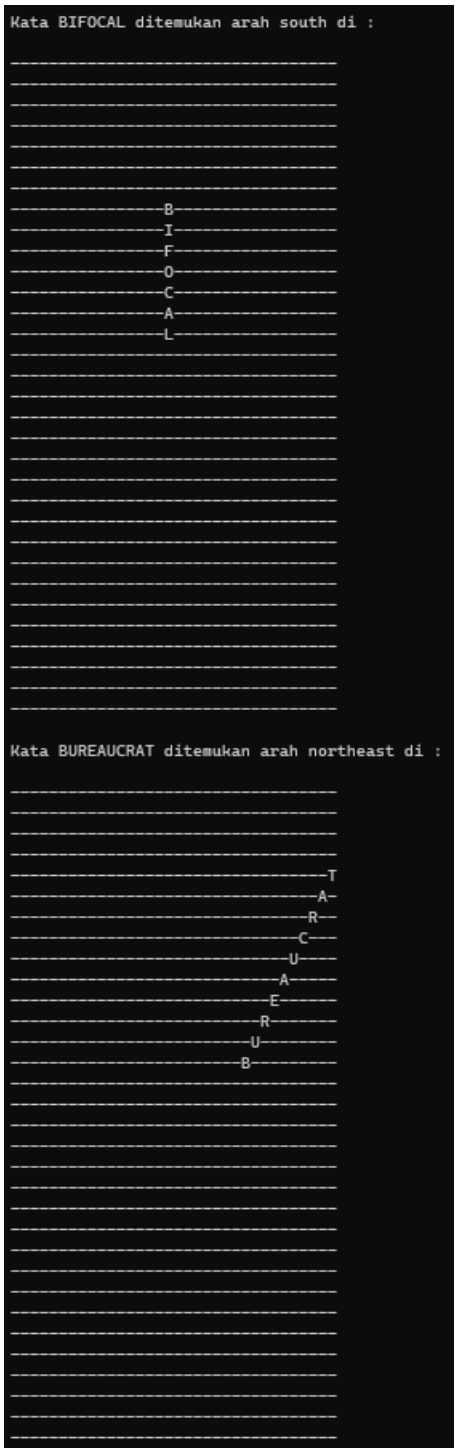
```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : large_1.txt

Kata ABATEMENT ditemukan arah southeast di :

A--------------------------------
-B--------------------------------
--A-------------------------------
---T------------------------------
----E-----------------------------
-----M----------------------------
------E---------------------------
-------N--------------------------
--------T-------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
--------------------------------
```


Kata ADOPTED ditemukan arah south di :
A
D
O
P
T
E
D


Kata AMBIDEXTROUS ditemukan arah east di :
AMBIDEXTROUS

Kata AGAINST ditemukan arah northwest di :
T
S
N
I
A
G
A

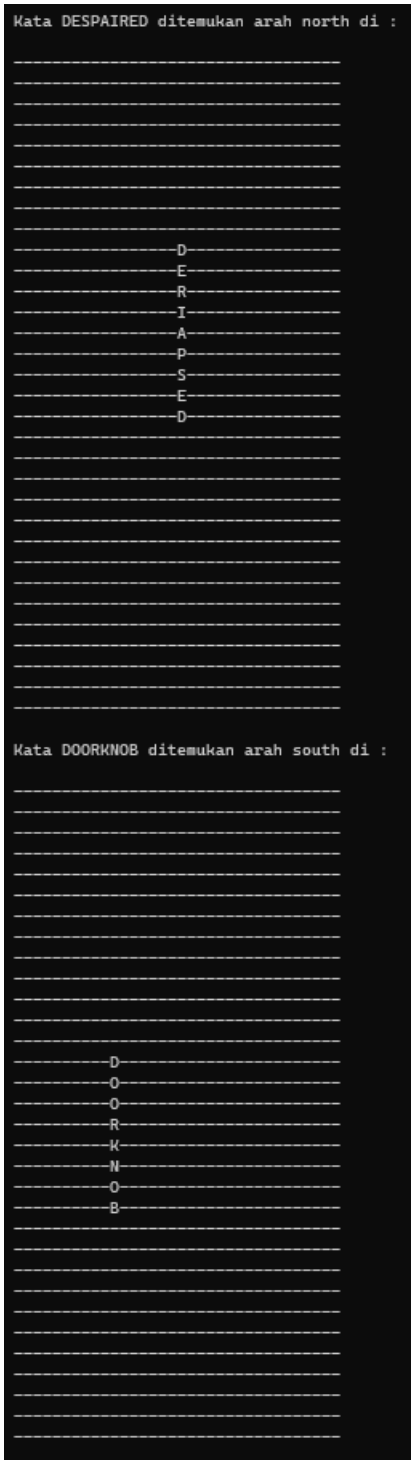
Kata BENEFACTOR ditemukan arah northwest di :
R
O
T
C
A
F
E
N
E
B


Kata BIFOCAL ditemukan arah south di :
B
I
F
O
C
A
L
Kata BUREAUCRAT ditemukan arah northeast di :
T
A
R
C
U
A
E
R
U
B


Kata CANNERY ditemukan arah west di :
YRENNAC
Kata CHEF ditemukan arah north di :
F
E
H
C


Kata CITYSCAPE ditemukan arah west di :
EPACSYTIC
Kata CLERGY ditemukan arah northeast di :
Y
G
R
E
L
C


Kata CREAKING ditemukan arah south di :
C
R
E
A
K
I
N
G
Kata DEGREE ditemukan arah north di :
E
E
R
G
E
D

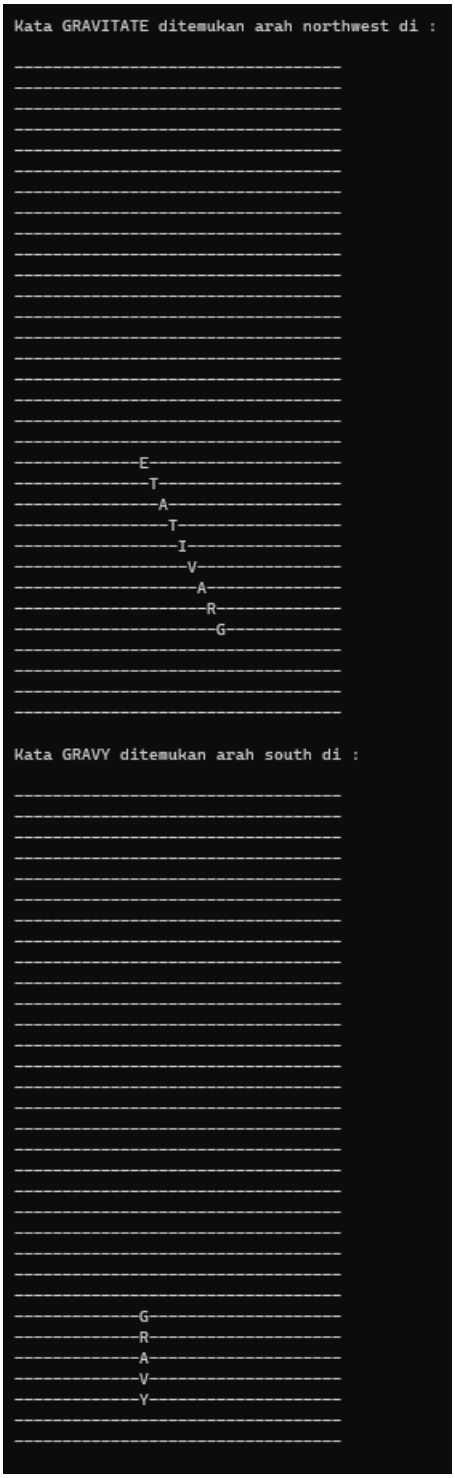

Kata DESPAIRED ditemukan arah north di :
D
E
R
I
A
P
S
E
D
Kata DOORKNOB ditemukan arah south di :
D
O
O
R
K
N
O
B

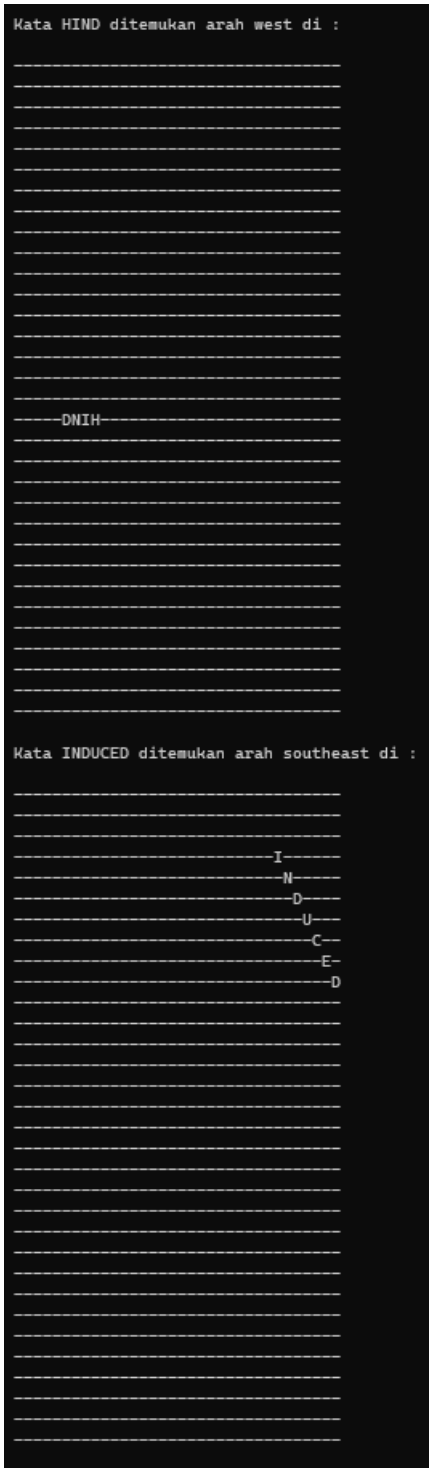

Kata DRAMATIZING ditemukan arah west di
GNIZITAMARD
Kata DRAT ditemukan arah southwest di :
D
R
A
T


Kata EXACERBATE ditemukan arah northeast di :
E
T
A
B
R
E
C
A
X
E
Kata EXCEEDING ditemukan arah north di :
G
N
I
D
E
E
C
X
E


Kata EXERTING ditemukan arah southwest di :
E
X
E
R
T
I
N
G
Kata FEATURE ditemukan arah west di :
ERUTAEF


Kata FEEBLY ditemukan arah east di :
FEEBLY
Kata FONDLING ditemukan arah north di :
G
N
I
L
D
N
O
F

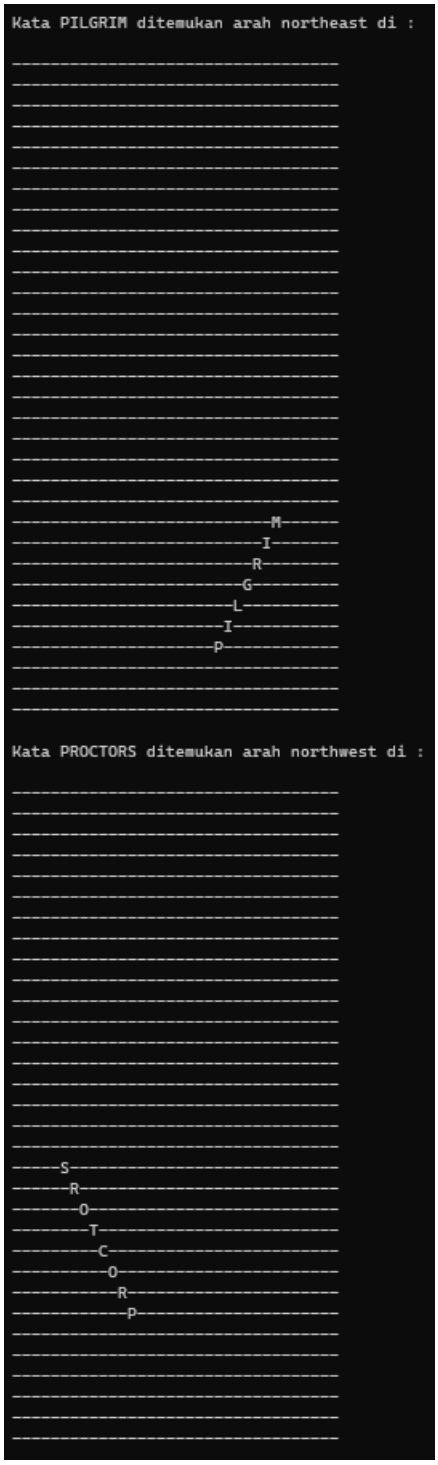

Kata GRAVITATE ditemukan arah northwest di :
E
T
A
T
I
V
A
R
G
Kata GRAVY ditemukan arah south di :
G
R
A
V
Y


Kata HIND ditemukan arah west di :
-----DNIH----------
Kata INDUCED ditemukan arah southeast di :
I------
-N-----
--D----
---U---
----C--
-----E-
------D

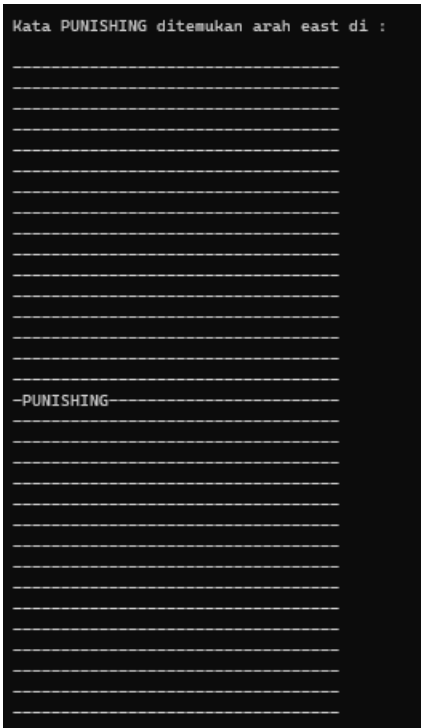

Kata JEEP ditemukan arah northeast di :
----------P----------
---------E-----------
--------E------------
-------J-------------
Kata LIKEN ditemukan arah west di :
-NEKIL-------------

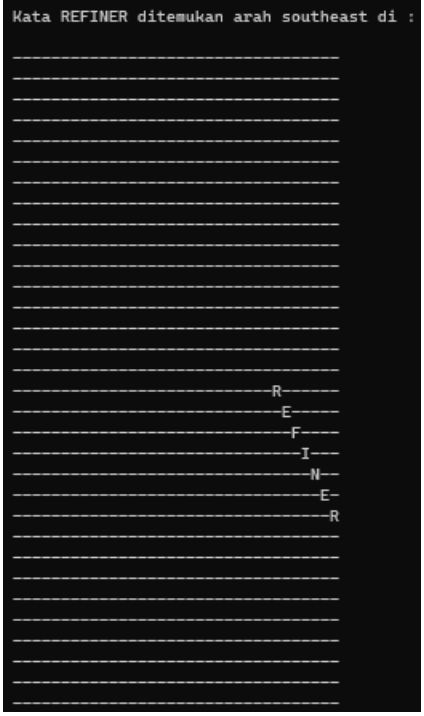

Kata MEMENTO ditemukan arah west di :
--------OTNEMEM--------
Kata MISTY ditemukan arah west di :
----------YTSIM----------

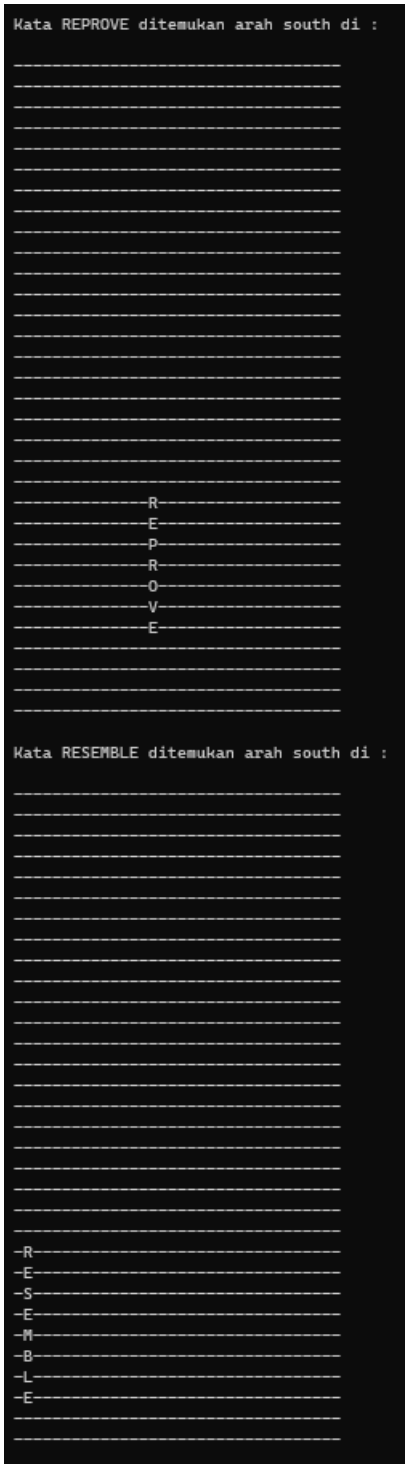

Kata OBSESSIVE ditemukan arah north di :
---E
---V
---I
---S
---S
---E
---S
---B
---O
Kata PARADIGM ditemukan arah northwest di :
M
-G
--I
---D
----A
-----R
------A
-------P

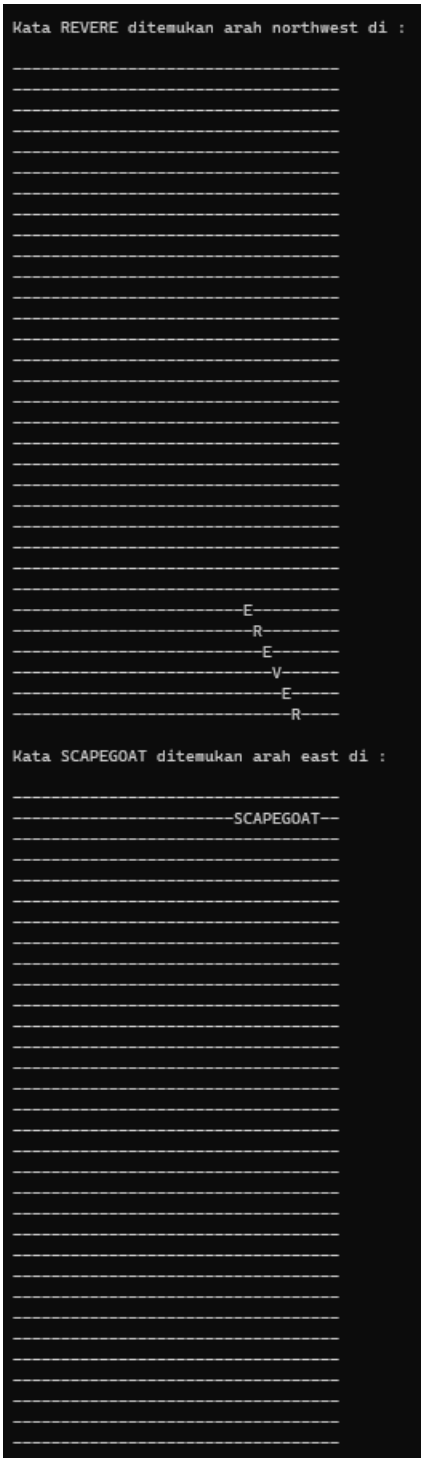

Kata PILGRIM ditemukan arah northeast di :
M
I
R
G
L
I
P

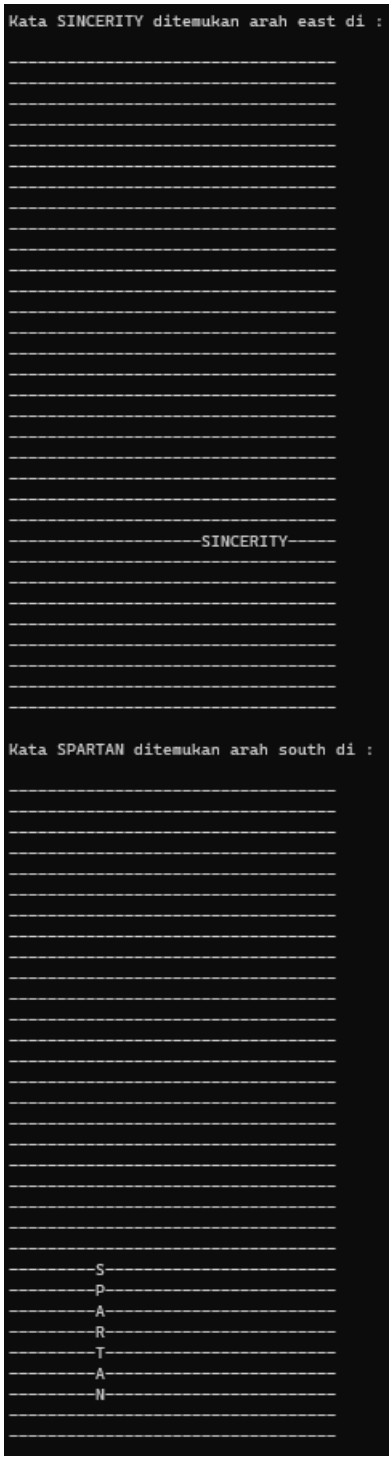
Kata PROCTORS ditemukan arah northwest di :
S
R
O
T
C
O
R
P


Kata PUNISHING ditemukan arah east di :
PUNISHING

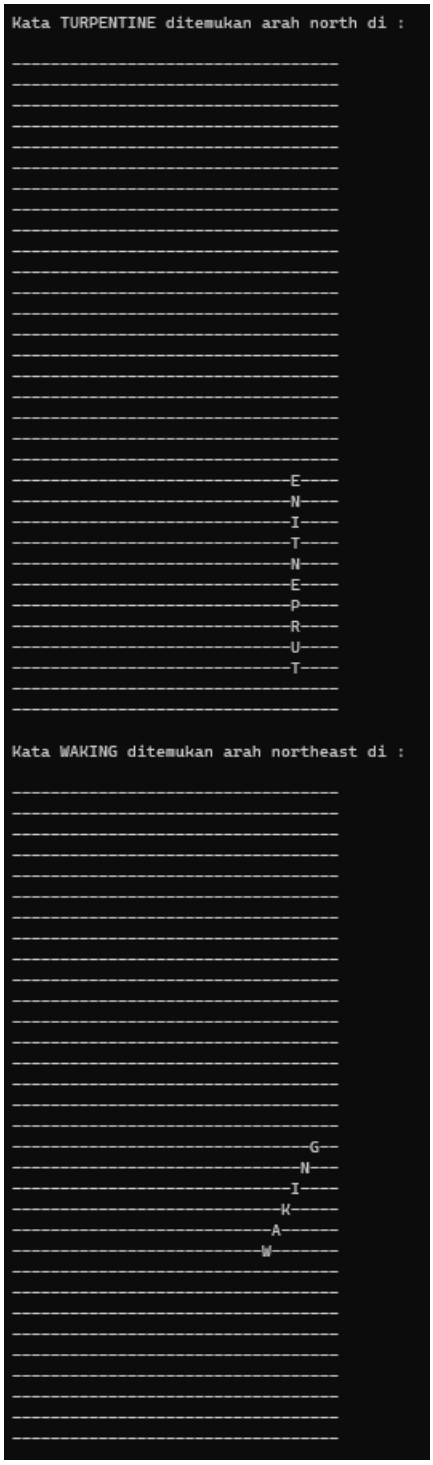

Kata REFINER ditemukan arah southeast di :
R
E
F
I
N
E
R

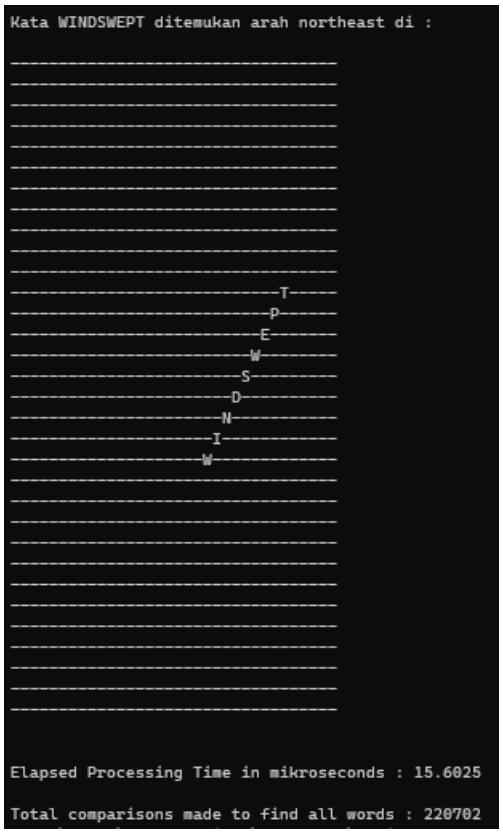

Kata REPROVE ditemukan arah south di :
R
E
P
R
O
V
E
Kata RESEMBLE ditemukan arah south di :
R
E
S
E
M
B
L
E


Kata REVERE ditemukan arah northwest di :
E
R
E
V
E
R
Kata SCAPEGOAT ditemukan arah east di :
SCAPEGOAT


Kata SINCERITY ditemukan arah east di :
SINCERITY
Kata SPARTAN ditemukan arah south di :
S
P
A
R
T
A
N


Kata SURPASS ditemukan arah west di :
SSAPRUS
Kata SYNCHRONIZEP ditemukan arah east di :
SYNCHRONIZEP





Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 15.6025 mikrodetik dan dibutuhkan 220,702 kali perbandingan huruf secara keseluruhan.

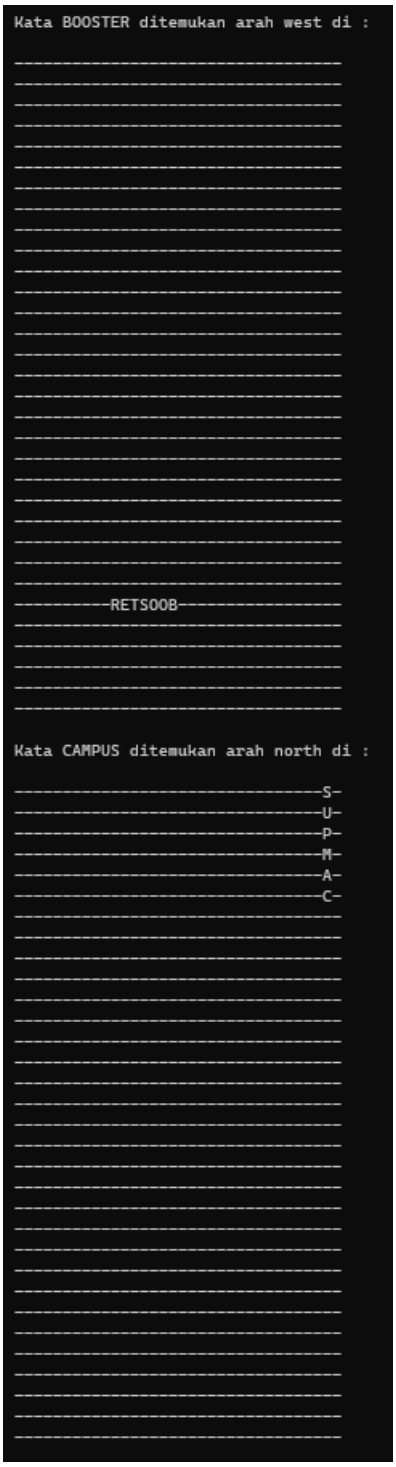
Uji coba kedelapan dilakukan dengan menggunakan file “large_2.txt” dengan dimensi matriks hurufnya 22x20 dan jumlah kata kunci yang harus dicari sebanyak 20 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat large_2.txt
T Y O C Y D M T I A G G W Z D T Q N N W H I F F Z S V V Q Q Y S S P
I K X B D D Z P P F U C N R Q U P X V F S F N V E U W Q Z P S L U E
G N D V S C E J O Q G S T H B C U B F K B R C T D U Z O N O Y X P T
L Q B N H E W G G Z X N C H L W G V G S G N M S N E D P O F C Z M C
C H N L T I S X N B Q U I A I I Q D E V I A T E Y E H K H G Z D A J
V L A H P Z M S A I G K H S N Y F F E X I N I Z C A M C W X X Q C B
A X S D G W N J I J U W U L U D A O T K A Y H L R O H H I V F I Z F
D R K H F Y C J Z O Z S M C C F A H W I F T A N N R U O J O S G N Q
M M I F R O Z C N H N S A M X F E L A Y M I O V I M J R C O L C B M
I M D E L B A R E L O T N I X I I R I L M B E S X D L M E K N I U H
T U D E C I U J O G D M I E E T R M U Z L I T N V D O Q N G R M N J
T B I M H X Q O I M P H Z P Q N X P Y I E E B G B N J S Y D M A P E
I O N G E A N Y S I X S E M E O R S V J N D N X O N X L S Z S Z R F
N L G H L D A G Z S P I D L N P G I Q E T I E T F P L O K Q Z E C L
G Q W Z P M U Z Z J U U T Z D X O E D Y Y W H N A J N U H Z C M R O
Z W R P L Z Y V Z U Q G D O U U L N U T H E Z D A G W Z X O T E C E
P K O D E T W F O D N N B E S M M C S M I C D M K F E C L J D G S A
A G I L S E V N C G G I H W P P I K N S A R P S R F O L I H U B W T
R A W Z S L P G O M L T N E Y P R Q M U L H G B I I E R E P X E L Q
E S C U L M M K J E K S X I K P I O P I B F A O E C N A P H E A J A
T W L Q Y U W A A N Y I G C K W Y U S I Y N P T T S D X S J W Y N S
A Z L Q U V F B I T D D N O N H P J Q E T A Y Z R X U N U T W N D R
M X E M D I Z Y I E C V P Y S J Y M Z F C C S I V F K O M A U Z I V
I D A J C M S N O C N P F A C E T I O U S U H T M G O F H A X V H R
C A P K B H X I P D E L E E N K O X J P N Y T F P Y K L T N U M U B
E Y E O F L A T T E R Y Q K R A Q G P A R N V I O Y K X Y L E Y S C
D L D H J G M T A I R E T S O O B R N T B U Q J O R O S E E S E Y R
P A R A B L E Y Y H O S C E Z R P N O I P M A T S N K T M Z P Z R O
W B W I N F J V O A R N K T S V Q G L E J O O R A G N A K H L Y A G
S G X Q O W M U L T I P L E E Z I G L D A I S A N A H T U E U T V D
U O M S D T R E R V K E K K L B K S Q J T K O S Z W J E E S Y A P Y
T R N N L U N C L E A N L Y G V U P T N S A V I C A R I O U S L Y D
```

```
ACQUISITION
ADMITTING
BIGOTRY
BIRDSONG
BOOSTER
CAMPUS
CHRISTENED
DECIMATE
DECLAIM
DEVIATE
DISTINGUISH
EUTHANASIA
FACETIOUS
FLATTERY
GAIT
GREENHOUSE
HELPLESSLY
HUMANIZED
INTOLERABLE
JINGO
JUICE
KANGAROO
KNEELED
LEAPED
MISJUDGMENT
MONOTHEISM
MULTIPLE
OBLIVIOUS
OBSESSION
PARABLE
PITCHFORK
PONTIFF
PROFANE
PROSECUTION
QUIPPED
RECOLLECT
REDHEAD
REFUSING
RIVULET
SCANDALIZED
SKIDDING
SOJOURN
STAMP
SUING
TYING
UNCLEANLY
VICARIOUSLY
WHIFF
```

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\bin> java MainProgram
Insert your path to the textfile here (please watch the formatting) : large_2.txt

Kata ACQUISITION ditemukan arah southeast di :

----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
-A--------------------------------
--C-------------------------------
---Q------------------------------
----U-----------------------------
-----I----------------------------
------S---------------------------
-------I--------------------------
--------T-------------------------
---------I------------------------
----------O-----------------------
-----------N----------------------
----------------------------------
----------------------------------
----------------------------------

Kata ADMITTING ditemukan arah south di :

----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
A---------------------------------
D---------------------------------
M---------------------------------
I---------------------------------
T---------------------------------
T---------------------------------
I---------------------------------
N---------------------------------
G---------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
----------------------------------
```


Kata BIGOTRY ditemukan arah northeast di :
Y
R
T
O
G
I
B
Kata BIRDSONG ditemukan arah southwest di :
B
I
R
D
S
O
N
G


Kata BOOSTER ditemukan arah west di :
RETSOOB
Kata CAMPUS ditemukan arah north di :
S
U
P
M
A
C

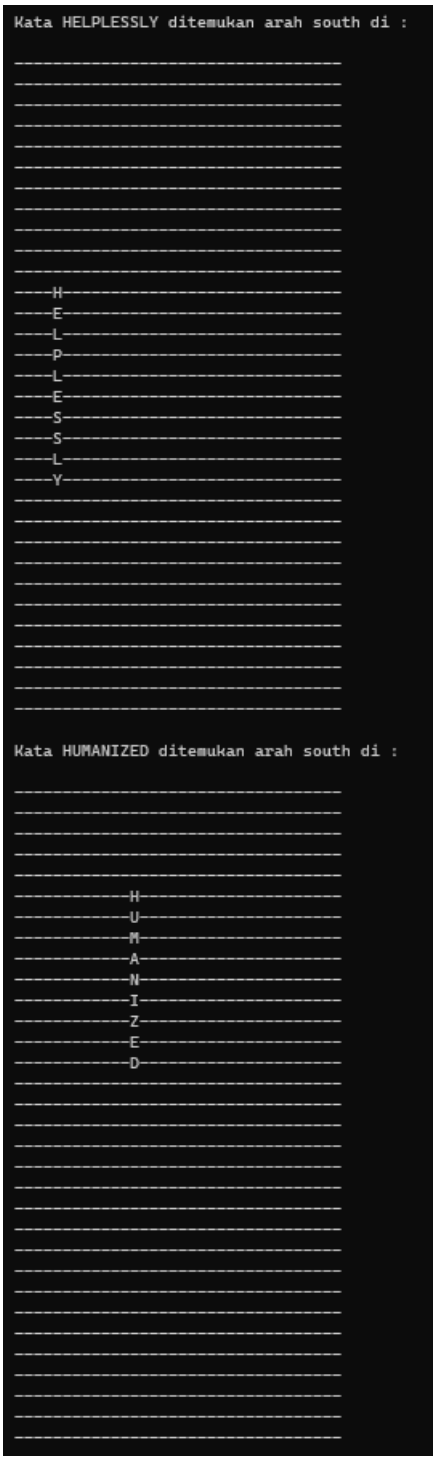

Kata CHRISTENED ditemukan arah southwest di :
C
H
R
I
S
T
E
N
E
D
Kata DECIMATE ditemukan arah north di :
E
T
A
M
I
C
E
D

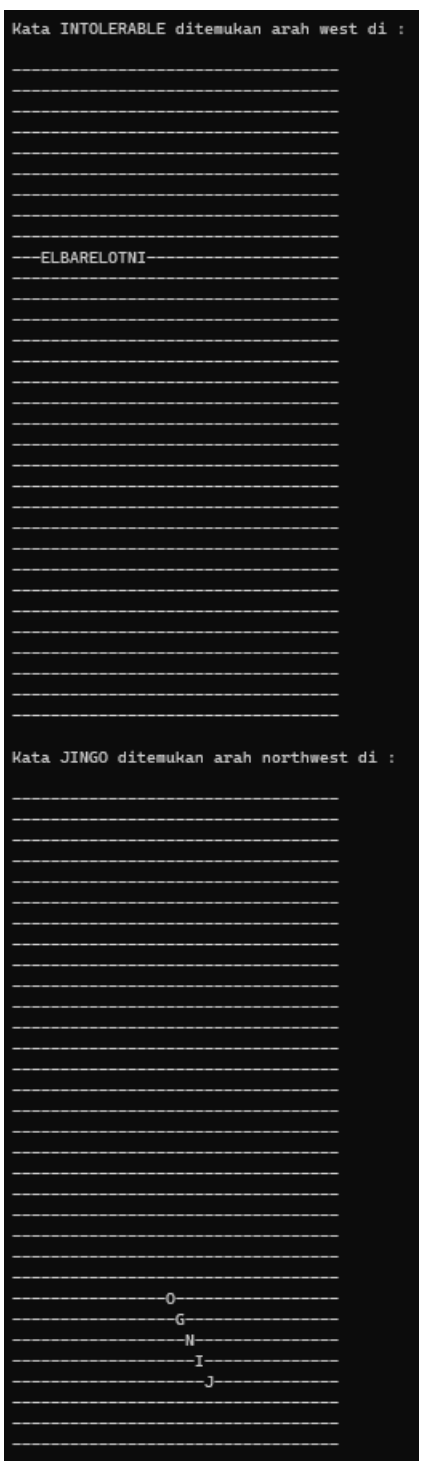

Kata DECLAIM ditemukan arah southwest di :
D
E
C
L
A
I
M
Kata DEVIATE ditemukan arah east di :
DEVIATE


Kata DISTINGUISH ditemukan arah north di :
H
S
I
U
G
N
I
T
S
I
D
Kata EUTHANASIA ditemukan arah west di :
AISANAHTUE

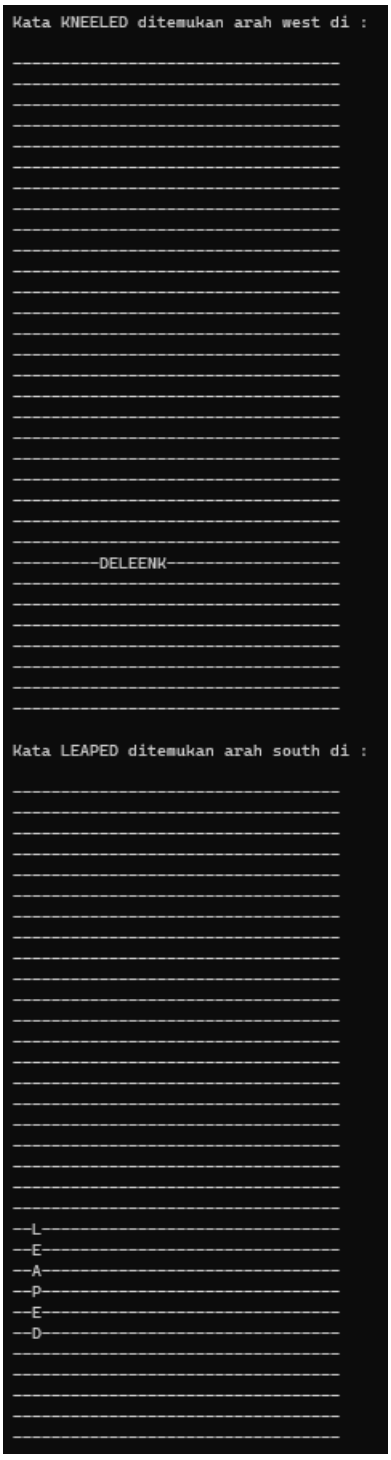

Kata FACETIOUS ditemukan arah east di :
FACETIOUS
Kata FLATTERY ditemukan arah east di :
FLATTERY


Kata GAIT ditemukan arah west di :
TIAG
Kata GREENHOUSE ditemukan arah northwest di :
E
S
U
O
H
N
E
E
R
G


Kata HELPLESSLY ditemukan arah south di :
H
E
L
P
L
E
S
S
L
Y
Kata HUMANIZED ditemukan arah south di :
H
U
M
A
N
I
Z
E
D


Kata INTOLERABLE ditemukan arah west di :
---ELBARELOTNI----------
Kata JINGO ditemukan arah northwest di :
O
G
N
I
J


Kata JUICE ditemukan arah west di :
---ECIUJ----------
Kata KANGAROO ditemukan arah west di :
----------OORAGNAK-----


Kata KNEELED ditemukan arah west di :
----------DELEENK----------
Kata LEAPED ditemukan arah south di :
L
E
A
P
E
D


Kata MISJUDGMENT ditemukan arah south di :
M
I
S
J
U
D
G
M
E
N
T
Kata MONOTHEISM ditemukan arah southwest di :
M
O
N
O
T
H
E
I
S
M

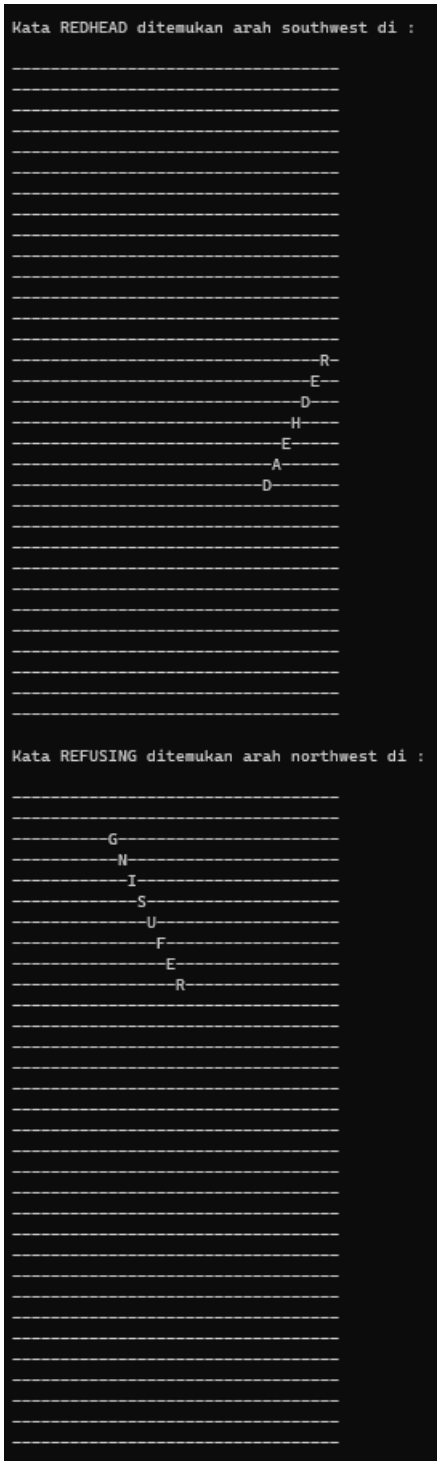

Kata MULTIPLE ditemukan arah east di :
MULTIPLE
Kata OBLIVIOUS ditemukan arah southwest di :
O
B
L
I
V
I
O
U
S

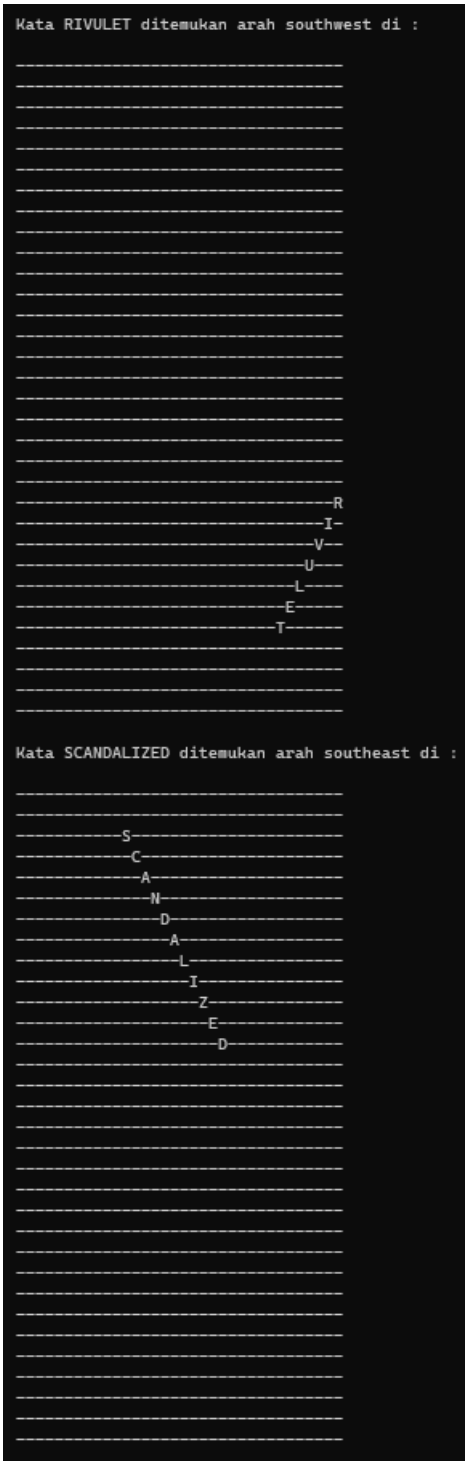

Kata OBSESSION ditemukan arah southeast di :
Kata PARABLE ditemukan arah east di :
PARABLE

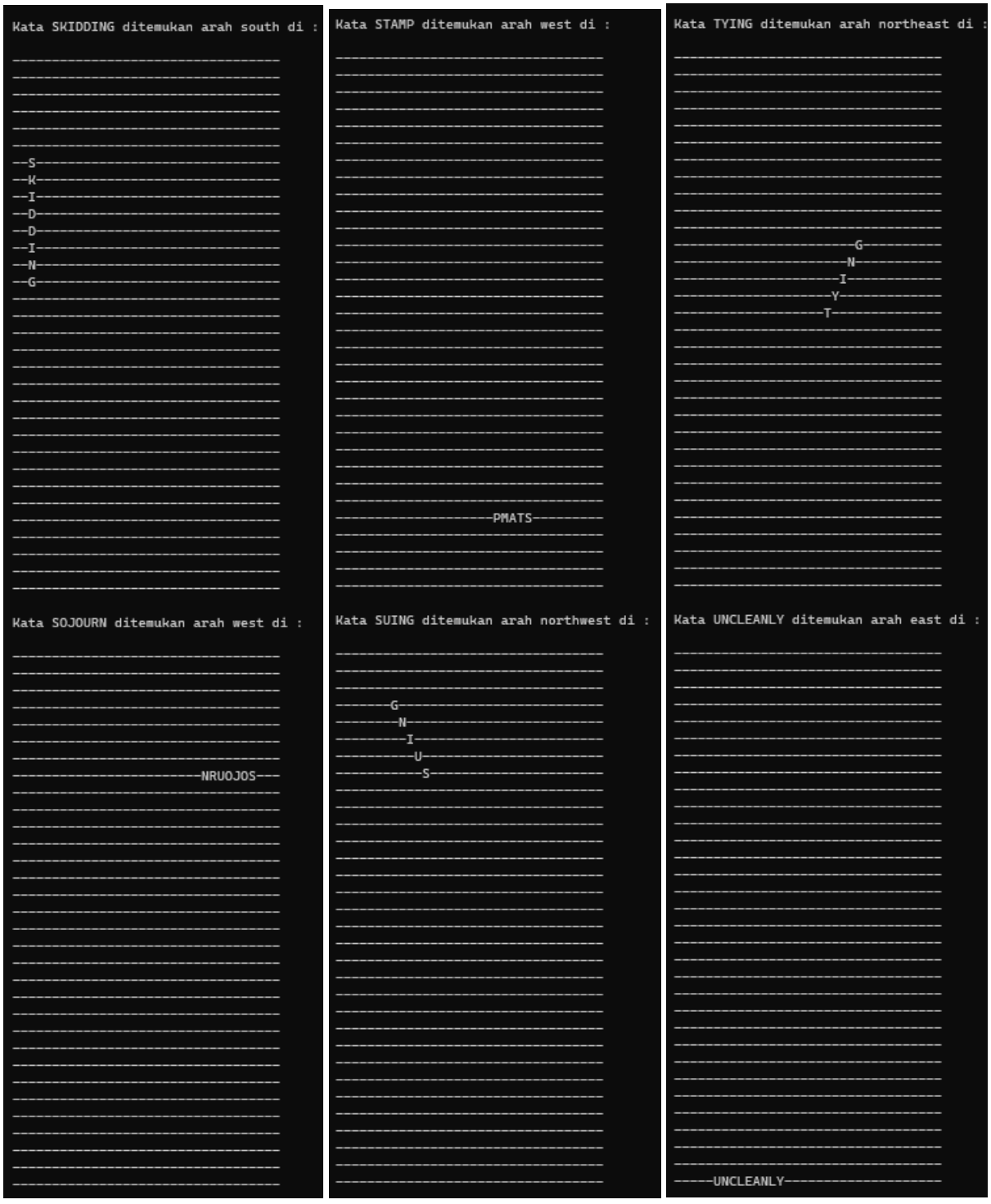

Kata PITCHFORK ditemukan arah southeast di :
Kata PONTIFF ditemukan arah north di :


Kata PROFANE ditemukan arah northwest di :
E
N
A
F
O
R
P
Kata PROSECUTION ditemukan arah southeast di :
P
R
O
S
E
C
U
T
I
O
N


Kata QUIPPED ditemukan arah northwest di :
D
E
P
P
I
U
Q
Kata RECOLLECT ditemukan arah southwest di :
R
E
C
O
L
L
E
C
T


Kata REDHEAD ditemukan arah southwest di :
Kata REFUSING ditemukan arah northwest di :


Kata RIVULET ditemukan arah southwest di :
Kata SCANDALIZED ditemukan arah southeast di :


Kata SKIDDING ditemukan arah south di :
S
K
I
D
D
I
N
G
Kata STAMP ditemukan arah west di :
PMATS
Kata TYING ditemukan arah northeast di :
G
N
I
Y
T
Kata SOJOURN ditemukan arah west di :
NRUOJOS
Kata SUING ditemukan arah northwest di :
G
N
I
U
S
Kata UNCLEANLY ditemukan arah east di :
UNCLEANLY

Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 18.1447 mikrodetik dan dibutuhkan 249,060 kali perbandingan huruf secara keseluruhan.

Uji coba kesembilan dilakukan dengan menggunakan file “large_3.txt” dengan dimensi matriks hurufnya 22x20 dan jumlah kata kunci yang harus dicari sebanyak 20 kata.

```
PS C:\Users\Gede Prasidha\Documents\Kuliah Semester 4\Startegi Algoritma\Bruteforce-Word-Puzzle-Java-\testfile> cat large_3.txt
X B L O K N H N A X L Q D W Q I X D Z B C Y V R L T V T Z K R P V I
K V S R E E N S T T X G D B S A R R O P S M T J E I R S S X R Z H Q
H V C N X M Z Y K H R Y H L N K I V E U U O A X Q A H U V I Y T P I
Z O R H U A B E Y A N C E D A S O I H V O K E J F Z C H N P E P V K
C W V X A S M E B O V C Z G K D S U R H H L T F B B C T V E P R B C
B L Z R R R N B T Q B N T O E E O E N L W L I Z P R A C T I C E E B
E V H Y L V T V O E P D S X D R S L N D L C N J T B K G X O P B B C
S K F F J G Y P N N J W I D I R O X R I Y M P X L O I J F N L A A I
C D T M A R S I A O N X C C R U U R S Q L I E E O U W E J G P T J G
S T R M F K W S J K O C I V E C A R V L E N Y A N U K K P M E E P I
G G N Q V S Z Z M G I J R Y F G G T L I X N A H G S X T B P X M H L
C O N D I M E N T N T S I P Y A I W I N P T D M P K D K P A P O Z E
E Q R Z I S F F C I A V P U I Y L S P E C T T F Y A M O K J E I P K
S K N B Z B X A F R R I M H Q U G Z F A S T G M L J R U X D R D I C
Z P W O K D H A B A G U E H D L Q I J R F P X D M G A G S E T I O G
F P Q H E N O T T E E M B Q F E Y K I B R H I J E C R R O U V V K N
E C R D K R U I G H T C Z W W M U I T H T U Z D O D E M Q T R V F I
D A O D L A E R C H N H Q U I C E O E T A L I M I S S A B Y O B W N
Y O N N K N T H T J I B R D O W N S T A G E M A X G W J E B T H X E
Y K P I T S O M W U R W F N L L S P I S I G D J Q S I M E D Y X P T
A Y R S S A M K N D R H X U S D D O B Q T D Z K T Q L R X A M C D H
X D U F P E I O L N F I D L C E E I C T H R N R Z D P Q F N B J R G
K Z S E G G E N U A B X N B L T D X B S K Y S C R A P E R Q E X E I
D I U M N A W D M L O L E G L E R W P D Y Z M V H P C E M I R C H T
D O E A I R Z G C E D Z D C T R A M S F Z X E F I R E P R I S E P D
E L C Q Z Y D W U R N E S N L M F G Q M C B E H Z X A P U V O Y E U
M A O W I V B K E U D T R B T I P H G E R M A N Y Q W J L U G J H G
A R N Y T B A O B T Q V N E R N Z A D D R E S S B N U O L G Y A S Y
H B S F P R S U D S S J W G D I H K B G Q Z Q N P B O O P N R L R M
S J U V A K I Y F A E E Q Z L S B I D E D T O M A R T I N G A L E V
A J L W B L N J S P F C O W L T S Y T P B W W E C R B Z W Z Z X I J
U K T N B M U T I C K E R N C M F Z X R U K P M V E D N F E E D L F
```

```
ABEYANCE
ADDRESS
ANISEED
ASHAMED
ASSIMILATE
BAPTIZING
BASIN
BIDED
CHART
CONDIMENT
CONSULT
CONTAINMENT
CRIME
DETERMINIST
DOWNSTAGE
EMPIRICIST
EXPERT
FEED
FRIGID
GERMANY
HEARING
HOOT
IDIOM
INTEGRATION
LINEAR
MANLINESS
MARTINGALE
NURTURING
PASTURELAND
PHOTOGRAPHY
PRACTICE
PRINTABLE
REBATE
REPRISE
SHEPHERD
SKYSCRAPER
SMART
SMOULDERED
SNAKED
SNEER
SUNS
SWINE
TICKER
TIGHTENING
TRAFFIC
USURP
WHEREON
YULE
```

```
Insert your path to the textfile here (please watch the formatting) : large_3.txt

Kata ABEYANCE ditemukan arah east di :

__________________________________
__________________________________
__________________________________
_____ABEYANCE____________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________

Kata ADDRESS ditemukan arah east di :

__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________________________________
__________ADDRESS________________
__________________________________
__________________________________
__________________________________
__________________________________
```


Kata ANISEED ditemukan arah southeast di :
A
N
I
S
E
E
D


Kata ASSIMILATE ditemukan arah west di :
ETALIMISSA

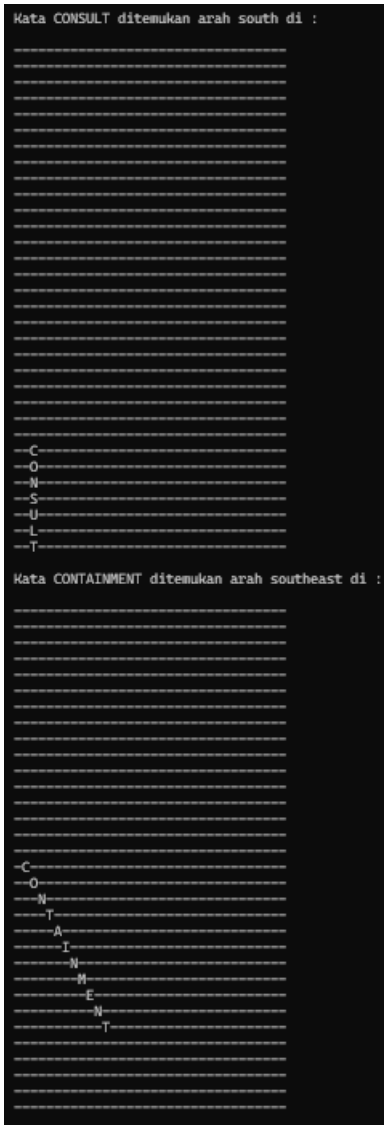

Kata BASIN ditemukan arah south di :
B
A
S
I
N

Kata ASHAMED ditemukan arah north di :
D
E
M
A
H
S
A

Kata BAPTIZING ditemukan arah north di :
G
N
I
Z
I
T
P
A
B

Kata BIDED ditemukan arah east di :
BIDED


Kata CHART ditemukan arah southeast di :
C
H
A
R
T


Kata CONSULT ditemukan arah south di :
C
O
N
S
U
L
T


Kata CRIME ditemukan arah west di :
EMIRC

Kata CONDIMENT ditemukan arah east di :
CONDIMENT

Kata CONTAINMENT ditemukan arah southeast di :
C
O
N
T
A
I
N
M
E
N
T

Kata DETERMINIST ditemukan arah south di :
D
E
T
E
R
M
I
N
I
S
T


Kata DOWNSTAGE ditemukan arah east di :
DOWNSTAGE


Kata EXPERT ditemukan arah south di :
E
X
P
E
R
T
Kata FEED ditemukan arah east di :
FEED


Kata FRIGID ditemukan arah northwest di :
D
I
G
I
R
F
Kata GERMANY ditemukan arah east di :
GERMANY

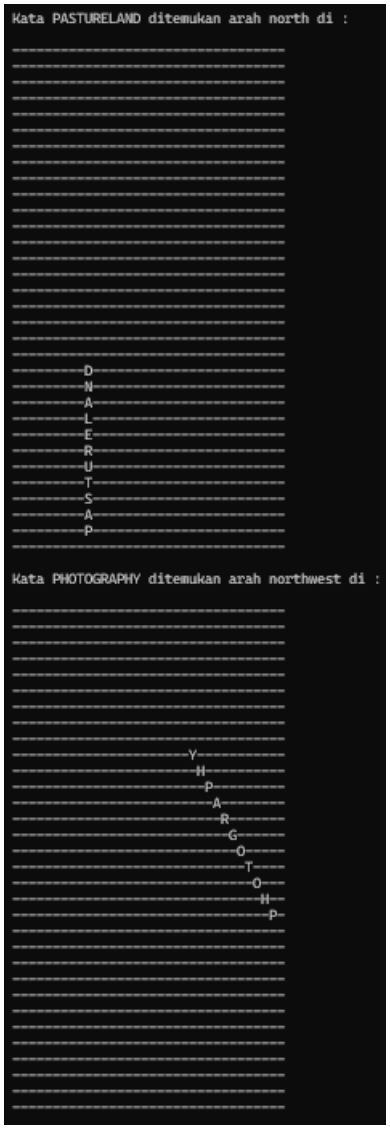
Kata EMPIRICIST ditemukan arah north di :
T
S
I
C
I
R
I
P
M
E

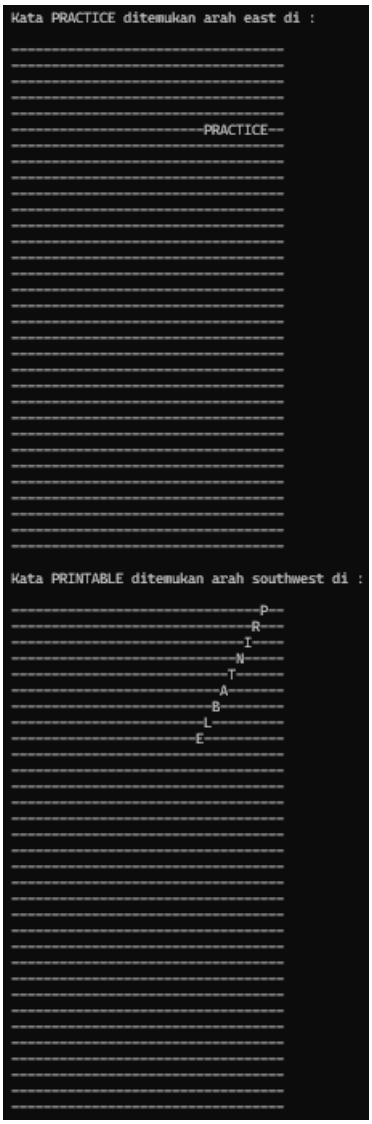

Kata HEARING ditemukan arah north di :
G
N
I
R
A
E
H
Kata HOOT ditemukan arah northeast di :
T
O
O
H


Kata IDIOM ditemukan arah north di :
M
O
I
D
I
Kata INTEGRATION ditemukan arah north di :
N
O
I
T
A
R
G
E
T
N
I


Kata LINEAR ditemukan arah south di :
L
I
N
E
A
R
Kata MANLINESS ditemukan arah northwest di :
S
S
E
N
I
L
N
A
M


Kata MARTINGALE ditemukan arah east di :
MARTINGALE
Kata NURTURING ditemukan arah southeast di :
N
U
R
T
U
R
I
N
G


Kata PASTURELAND ditemukan arah north di :
D
N
A
L
E
R
U
T
S
A
P
Kata PHOTOGRAPHY ditemukan arah northwest di :
Y
H
P
A
R
G
O
T
O
H
P


Kata PRACTICE ditemukan arah east di :
PRACTICE
Kata PRINTABLE ditemukan arah southwest di :
P
R
I
N
T
A
B
L
E


Kata REBATE ditemukan arah south di :
R
E
B
A
T
E
Kata REPRISE ditemukan arah east di :
REPRISE

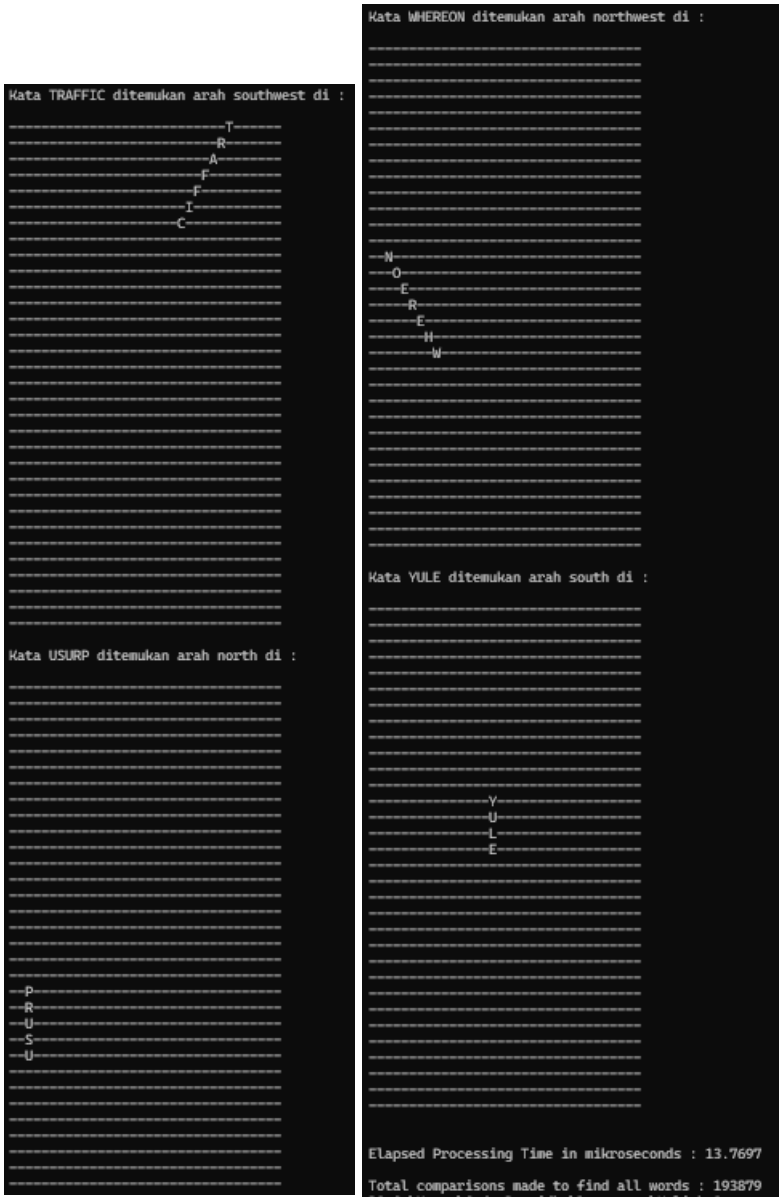

Kata SHEPHERD ditemukan arah north di :
D
R
E
H
P
E
H
S
Kata SKYSCRAPER ditemukan arah east di :
SKYSCRAPER


Kata SMART ditemukan arah west di :
TRAMS
Kata SMOULDERED ditemukan arah southeast di :
S
M
O
U
L
D
E
R
E
D


Kata SNAKED ditemukan arah south di :
S
N
A
K
E
D
Kata SNEER ditemukan arah west di :
REENS


Kata SUNS ditemukan arah northwest di :
S
N
U
S
Kata SWINE ditemukan arah northeast di :
E
N
I
W
S


Kata TICKER ditemukan arah east di :
TICKER
Kata TIGHTENING ditemukan arah north di :
G
N
I
N
E
T
H
G
I
T



Dapat dilihat bahwa untuk menemukan semua kata kunci di dalam matriks huruf dibutuhkan waktu 13.7697 mikrodetik dan dibutuhkan 193,879 kali perbandingan huruf secara keseluruhan.

IV. Alamat (*Link*) GitHub Source Code

https://github.com/LordGedelicious/Bruteforce-Word-Puzzle-Java-

V. *Checklist* Pencapaian dan Progress Program

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan (*no syntax error*)	√	
2. Program berhasil running	√	
3. Program dapat membaca file masukan dan menuliskan luaran	√	
4. Program berhasil menemukan semua kata di dalam puzzle	√	