Laporan

Tugas Kecil 1 IF2211 Strategi Algoritma

Penyelesaian *Word Search Puzzle* dengan Algoritma *Brute Force*

Disusun Oleh:

Suryanto 13520059

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

DAFTAR ISI

ALGORITMA BRUTE FORCE

Tugas ini diprogram dengan bahasa pemrograman C dengan memanfaatkan beberapa library yang telah tersedia. *Library* yang digunakan adalah stdio.h untuk keperluan I/O program, time.h digunakan untuk menghitung waktu eksekusi, dan string.h untuk mengkonkat string (digunakan untuk menggabungkan nama file dan direktori file test case).

Program dijalankan dengan algoritma *brute force* yang mengikuti tahapan-tahapan berikut ini:

1. Program dimulai dengam melakukan pengecekan jumlah kolom/baris yang ada cukup untuk membentuk *keyword* yang diperiksa. Hal ini dilakukan untuk mengoptimalkan pencarian (misalnya pada baris pertama, program tidak akan melakukan pencarian ke arah atas),
2. Selanjutnya program akan melakukan pencocokan huruf pada tiap kolom di tiap baris dengan huruf pertama *keyword* yang sedang diperiksa,
3. Jika huruf pada matriks dengan indeks (i,j) sesuai dengan huruf pertama pada kata yang dicek maka program akan melakukan pencocokan ke delapan arah dimulai dari vertikal atas dan dilanjutin dengan arah lain sesuai arah jarum jam,
4. Apabila kedelapan arah telah diperiksa dan tidak ditemukan arah yang membentuk kata yang diuji, maka pencarian gagal dan program kembali ke tahap nomor 1 (tetapi dengan pencarian indeks kolom selanjutnya atau baris selanjutnya apabila telah mencapai kolom terakhir),
5. Jika ditemukan kata yang cocok, program akan mencatat arah, titik awal, dan titik akhir dari kata untuk digunakan pada keluaran solusi (pencarian berhasil).

SOURCE CODE

Bahasa Pemrograman: C

```c
#include <stdio.h>
#include <time.h>
#include <string.h>

typedef struct {
    char contents[100];
    int length;
    int arah;
    int xAwal;
    int yAwal;
    int xAkhir;
    int yAkhir;
} word;



//KAMUS FILE//
char currentChar;
static FILE * tapeFile;
char puzzle[100][100];
word kata[100];

#define True 1
#define False 0
#define newLine '\n'

void adv(){
    /* KAMUS LOKAL */
    static int * retval;

    /* ALGORITMA */
    retval = fscanf(tapeFile,"%c",&currentChar);
}

int startFile(char *fileName){

    /* Algoritma */
    tapeFile = fopen(fileName,"r");
    if (tapeFile == NULL){
        printf("Error: could not open file\n");
        return False;
    }
    else{
        adv();
        return True;
```

```
    }
}


void readMatrix(int *row,int *col){
    /* KAMUS LOKAL */
    int i,j;

    /* ALGORITMA */
    i = 0;
    while (currentChar != newLine){
        j = 0;
        while (currentChar != newLine){
            puzzle[i][j] = currentChar;
            j++;
            adv();
            if (currentChar == ' ') adv();
        }
        adv();
        i++;
    }
    *row = i;
    *col = j;
}

void readKata(word solusi[100], int *banyakKata){
    /* KAMUS LOKAL */
    int i,j;

    /* ALGORITMA */
    adv();
    i = 0;
    while ((!feof(tapeFile))){
        j=0;
        while (currentChar != newLine && (!feof(tapeFile))){
            solusi[i].contents[j] = currentChar;
            adv();
            j++;
        }
        solusi[i].length=j;
        i++;
        adv();
    }
    *banyakKata = i;
}

void tulisKata(word sample){
    /* KAMUS LOKAL */
    int i;
```

```
    /* ALGORITMA */
    for (i = 0; i< sample.length; i++){
        printf("%c", sample.contents[i]);
    }
    printf("\n");
}


void tulisMatriks(word sample,int row, int col){
    /* KAMUS LOKAL */
    int i,j,k;

    /* ALGORITMA */
    tulisKata(sample);

    //vertikal-atas
    if (sample.arah == 1){
        k = sample.length-1;
        for (i = 0; i< row; i++){
            for ( j = 0; j< col; j++){
                if (j == sample.xAwal && i == sample.yAkhir){
                    printf("%c ", sample.contents[k]);
                    k--;
                    if (k >=0) sample.yAkhir++;
                }
                else{
                    printf("- ");
                }
            }
            printf("\n");
        }
    }

    //menyamping  atas-kanan
    else if (sample.arah == 2){
        k = sample.length-1;
        for (i = 0; i< row; i++){
            for ( j = 0; j< col; j++){
                if (j == sample.xAkhir && i == sample.yAkhir){
                    printf("%c ", sample.contents[k]);

                    k--;
                    if (k >=0){
                        sample.yAkhir++;
                        sample.xAkhir--;
                    }

                }
```

```
            else{
                printf("- ");
            }
        }
        printf("\n");
    }
}

//mendatar kanan
else if (sample.arah == 3){
    k = 0;
    for (i = 0; i< row; i++){
        for ( j =0; j< col; j++){
            if (j == sample.xAwal && i == sample.yAwal){
                printf("%c ", sample.contents[k]);
                k++;
                if (k < sample.length){
                    sample.xAwal++;
                }
            }
            else{
                printf("- ");
            }
        }
        printf("\n");
    }

}

//menyamping kanan-bawah
else if (sample.arah == 4){
    k = 0;
    for (i = 0; i< row; i++){
        for ( j =0; j< col; j++){
            if (j == sample.xAwal && i == sample.yAwal){
                printf("%c ", sample.contents[k]);
                k++;
                if (k < sample.length){
                    sample.xAwal++;
                    sample.yAwal++;
                }
            }
            else{
                printf("- ");
            }
        }
        printf("\n");
    }
```

```
    }

    //vertikal bawah
    else if (sample.arah == 5){
        k = 0;
        for (i = 0; i< row; i++){
            for ( j =0; j< col; j++){
                if (j == sample.xAwal && i == sample.yAwal){
                    printf("%c ", sample.contents[k]);
                    k++;
                    if (k < sample.length){
                        sample.yAwal++;
                    }
                }
                else{
                    printf("- ");
                }
            }
            printf("\n");
        }
    }

    //menyamping kiri-bawah
    else if (sample.arah == 6){
        k = 0;
        for (i = 0; i< row; i++){
            for ( j =0; j< col; j++){
                if (j == sample.xAwal && i == sample.yAwal){
                    printf("%c ", sample.contents[k]);
                    k++;
                    if (k < sample.length){
                        sample.xAwal--;
                        sample.yAwal++;
                    }
                }
                else{
                    printf("- ");
                }
            }
            printf("\n");
        }
    }

    //mendatar kiri
    else if (sample.arah == 7){
        k = sample.length-1;
        for (i = 0; i< row; i++){
            for ( j =0; j< col; j++){
```

```
                if (j == sample.xAkhir && i == sample.yAwal){
                    printf("%c ", sample.contents[k]);
                    k--;
                    if (k >=0 ){
                        sample.xAkhir++;
                    }
                }
                else{
                    printf("- ");
                }
            }
            printf("\n");
        }
    }

    //menyamping atas-kiri
    else if (sample.arah == 8){
        k = sample.length-1;
        for (i = 0; i< row; i++){
            for ( j =0; j< col; j++){
                if (j == sample.xAkhir && i == sample.yAkhir){
                    printf("%c ", sample.contents[k]);
                    k--;
                    if (k >=0 ){
                        sample.xAkhir++;
                        sample.yAkhir++;
                    }
                }
                else{
                    printf("- ");
                }
            }
            printf("\n");
        }
    }
}


int solver(word sample, int row, int col){
    /* KAMUS LOKAL */
    int i, j, k,ii,jj,found,temp,counter;

    /* ALGORITMA */
    //pengecekan searah jarum jam dimulai dari atas-kanan
    found = False;
    i = 0;
    counter = 0;
    while ((!found) && (i < row || j <col)){
        j = 0;
```

```
        while (j<col && !found){
            temp = sample.contents[0] == puzzle[i][j];
            counter++;
            if (temp){

                sample.xAwal = j;
                sample.yAwal = i;

                k = 1;
                if (i>=sample.length-1){
                    ii = i-1;
                    while (temp && !found){
                        counter++;
                        temp = (sample.contents[k] == puzzle[ii][j]);
                        k++;
                        if (k == sample.length && temp){
                                found = True;
                                sample.xAkhir = j;
                                sample.yAkhir = ii;
                                sample.arah = 1;
                        }
                        ii--;
                    }

                    //menyamping atas-kanan
                    if (!temp){         //apabila pencarian menurun sudah
dilakukan dan gagal mendapatkan solusi
                        ii = i-1;
                        k = 1;

                        if (j<= col -sample.length-1){
                            temp = True;
                            // printf("hello %d", j);
                            jj = j+1;

                            while (temp && !found){
                                counter++;
                                temp = (sample.contents[k] == puzzle[ii][jj]);
                                k++;
                                if (k == sample.length && temp){
                                    found = True;
                                    sample.xAkhir = jj;
                                    sample.yAkhir = ii;
                                    sample.arah = 2;
                                }
                                ii--;
                                jj++;
```

```
                        }
                    }
                }

                //menyamping atas-kiri
                if (!temp){         //apabila pencarian menurun sudah
dilakukan dan gagal mendapatkan solusi
                    ii = i-1;
                    k = 1;

                    if (j>= sample.length-1){

                        temp = True;
                        jj = j-1;

                        while (temp && !found){
                            counter++;
                            temp = (sample.contents[k] == puzzle[ii][jj]);
                            k++;

                            if (k == sample.length && temp){
                                found = True;
                                sample.xAkhir = jj;
                                sample.yAkhir = ii;
                                sample.arah = 8;
                            }
                            ii--;
                            jj--;
                        }
                    }
                }

            }

            //cek datar kanan
            if (!found && (j<= col -sample.length))
            {
                jj = j+1;
                k = 1;
                temp = True;
                while (temp && !found){
                    counter++;
                    temp = (sample.contents[k] == puzzle[i][jj]);
                    k++;

                    if (k == sample.length && temp){
                        found = True;
                        sample.xAkhir = jj;
```

```
                        sample.yAkhir = i;
                        sample.arah = 3;
                    }
                    jj++;
                }

            }


            //cek datar kiri
            if (!found && (j>=sample.length-1))
            {
                temp = True;
                jj = j-1;
                k = 1;

                while (temp && !found){
                    counter++;
                    temp = (sample.contents[k] == puzzle[i][jj]);
                    k++;

                    if (k == sample.length && temp){
                        found = True;
                        sample.xAkhir = jj;
                        sample.yAkhir = i;
                        sample.arah = 7;
                    }
                    jj--;
                }

            }

            //cek menurun-bawah, menyamping bawah-kanan, menyamping bawah-
kiri
            if (!found && i<=row - sample.length){
                ii = i+1;
                k = 1;
                temp = True;
                //menurun-bawah
                while (temp && !found){
                    counter++;
                    temp = (sample.contents[k] == puzzle[ii][j]);
                    k++;

                    if (k == sample.length && temp){
                            found = True;
                            sample.xAkhir = j;
                            sample.yAkhir = ii;
```

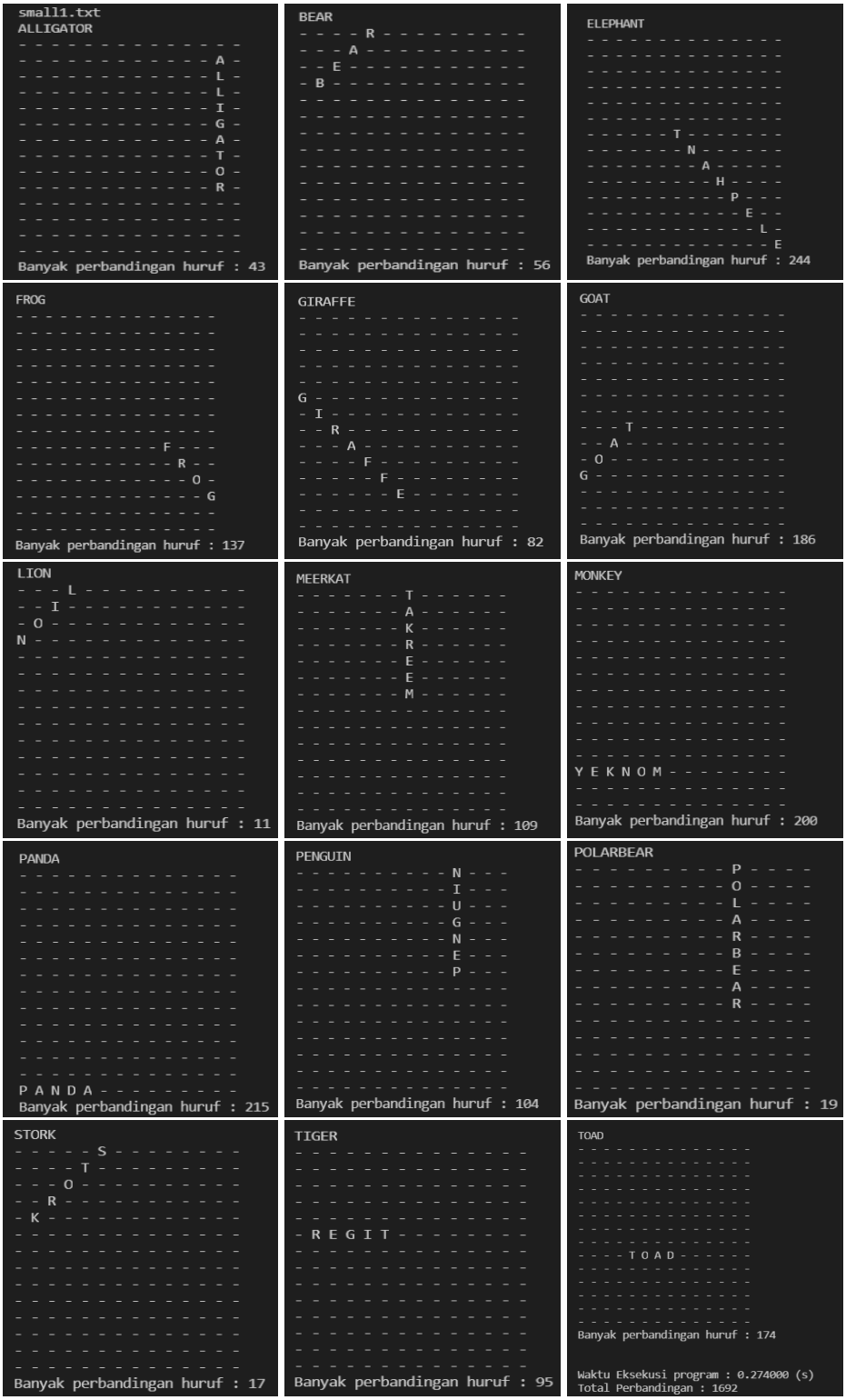
```
                            sample.arah = 5;
                        }
                        ii++;
                    }

                    //menyamping bawah-kanan
                    if (!temp){         //apabila pencarian menurun sudah
dilakukan dan gagal mendapatkan solusi
                        ii = i+1;
                        k = 1;
                        if (j<= col -sample.length){
                            temp = True;
                            jj = j+1;

                            while (temp && !found){
                                counter++;
                                temp = (sample.contents[k] == puzzle[ii][jj]);
                                k++;

                                if (k == sample.length && temp){
                                    found = True;
                                    sample.xAkhir = jj;
                                    sample.yAkhir = ii;
                                    sample.arah = 4;
                                }
                                ii++;
                                jj++;
                            }
                        }
                    }

                    //menyamping bawah-kiri
                    if (!temp){         //apabila pencarian menurun sudah
dilakukan dan gagal mendapatkan solusi

                        ii = i+1;
                        k = 1;

                        if (j>= sample.length-1){

                            temp = True;
                            jj = j-1;

                            while (temp && !found){
                                counter++;
                                temp = (sample.contents[k] == puzzle[ii][jj]);
                                k++;
```

```
                                if (k == sample.length && temp){
                                    found = True;
                                    sample.xAkhir = jj;
                                    sample.yAkhir = ii;
                                    sample.arah = 6;
                                }
                                ii++;
                                jj--;
                            }
                        }
                    }
                }
            }
            j++;
        }
        i++;

    }
    if (found){
        tulisMatriks(sample,row,col);
    }
    else{
        printf("Kata tidak ditemukan\n");
    }
    printf("Banyak perbandingan huruf : %d\n", counter);
    return counter;
}




int main(){
    /* KAMUS  */
    int i,j;
    char filename[50];

    int row,col;
    int banyakKata ;
    int sumPerbandingan;
    double cpu_time_used;
    clock_t start, end;

    /* ALGORITMA */

    scanf("%s", filename);
    char dir[] = "../test/";
    strcat(dir,filename);

    if (startFile(dir)){
```

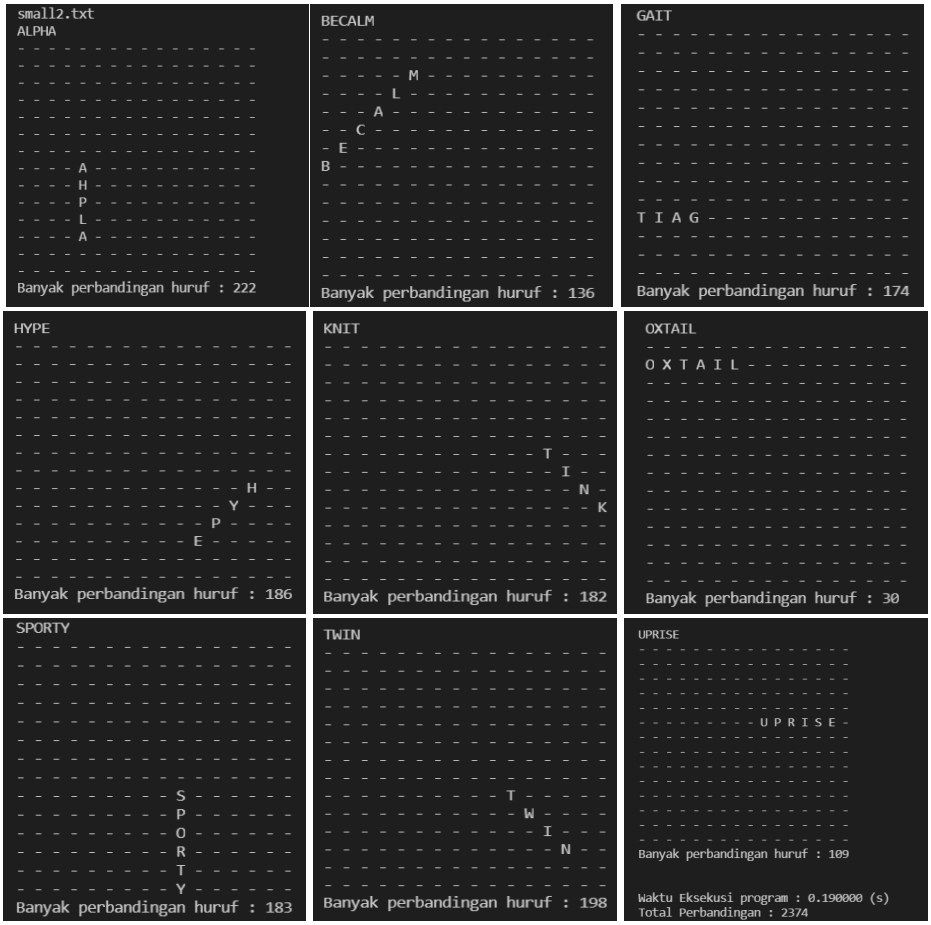
```
        readMatrix(&row,&col);

        readKata(kata, &banyakKata);

        start = clock();
        sumPerbandingan = 0;
        for ( i = 0; i <banyakKata; i++){
            sumPerbandingan += solver(kata[i], row,col);
            printf("\n\n");
        }

        end = clock();
        cpu_time_used = ((double) (end - start)) / CLOCKS_PER_SEC;
        printf("Waktu Eksekusi program : %lf (s)\n", cpu_time_used);
        printf("Total Perbandingan : %d\n", sumPerbandingan);

    }
    else{
        // do nothing
    }


}
```

SCREENSHOT INPUT OUTPUT

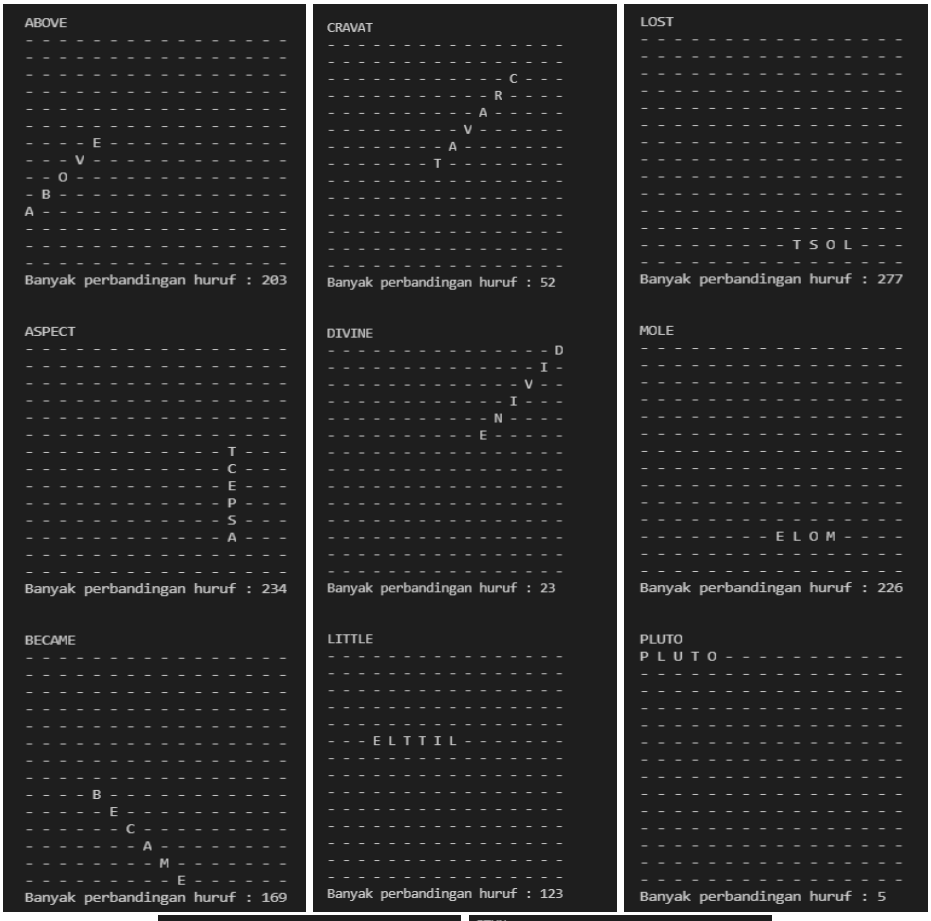
1. small1.txt

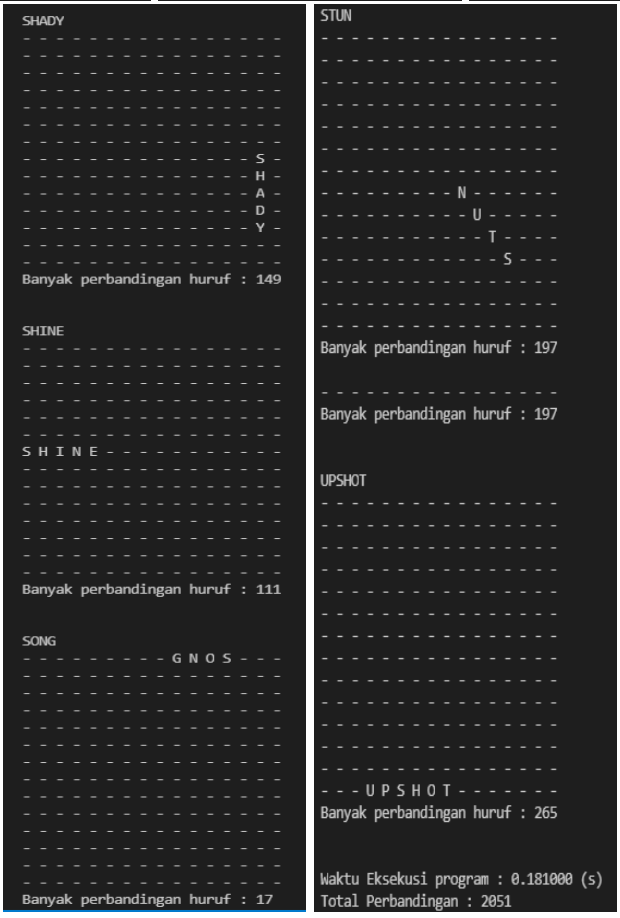


2. small2.txt

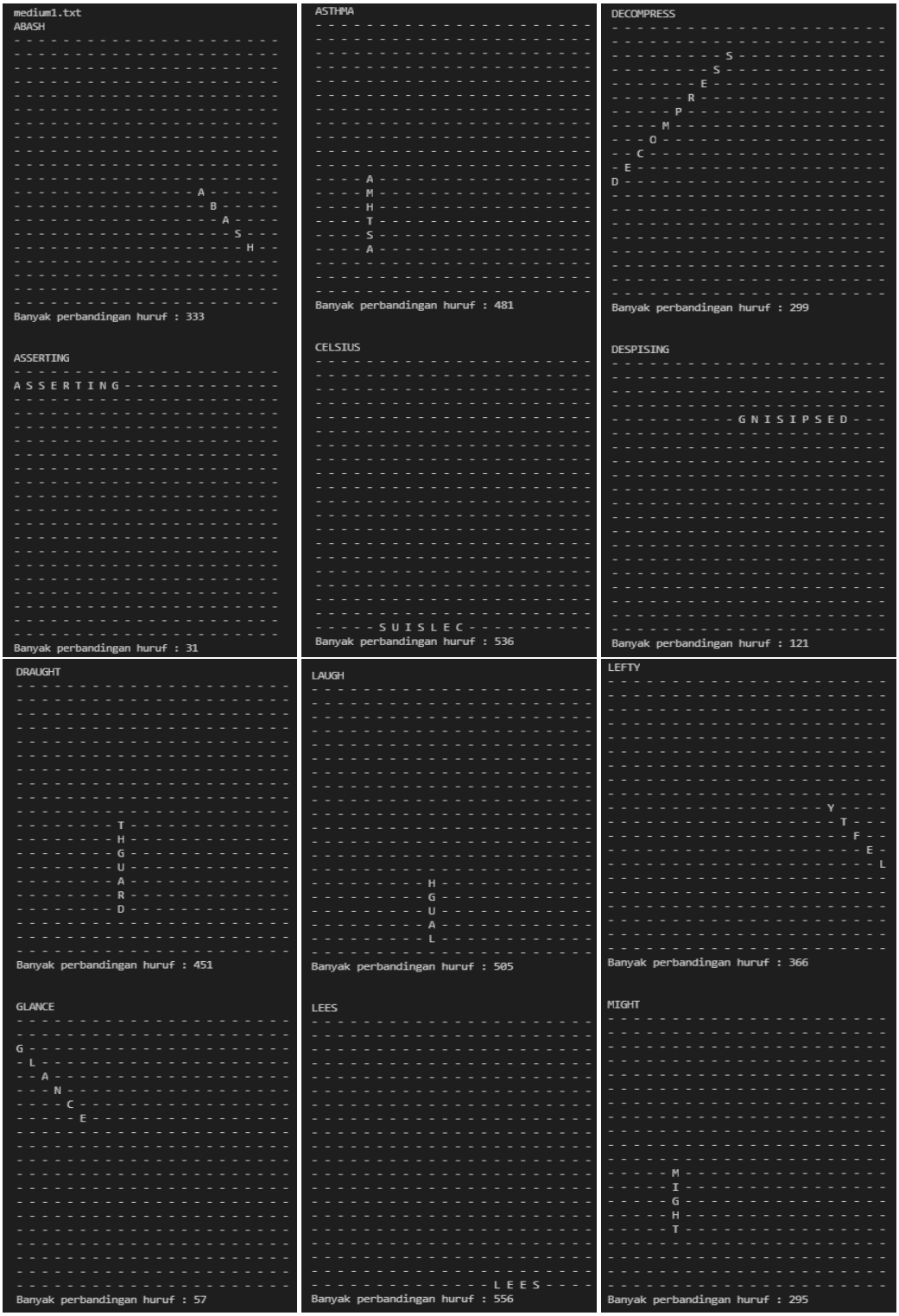

small2.txt
ALPHA
Banyak perbandingan huruf : 222
BECALM
Banyak perbandingan huruf : 136
GAIT
Banyak perbandingan huruf : 174
HYPE
Banyak perbandingan huruf : 186
KNIT
Banyak perbandingan huruf : 182
OXTAIL
Banyak perbandingan huruf : 30
SPORTY
Banyak perbandingan huruf : 183
TWIN
Banyak perbandingan huruf : 198
UPRISE
Banyak perbandingan huruf : 109
Waktu Eksekusi program : 0.190000 (s)
Total Perbandingan : 2374

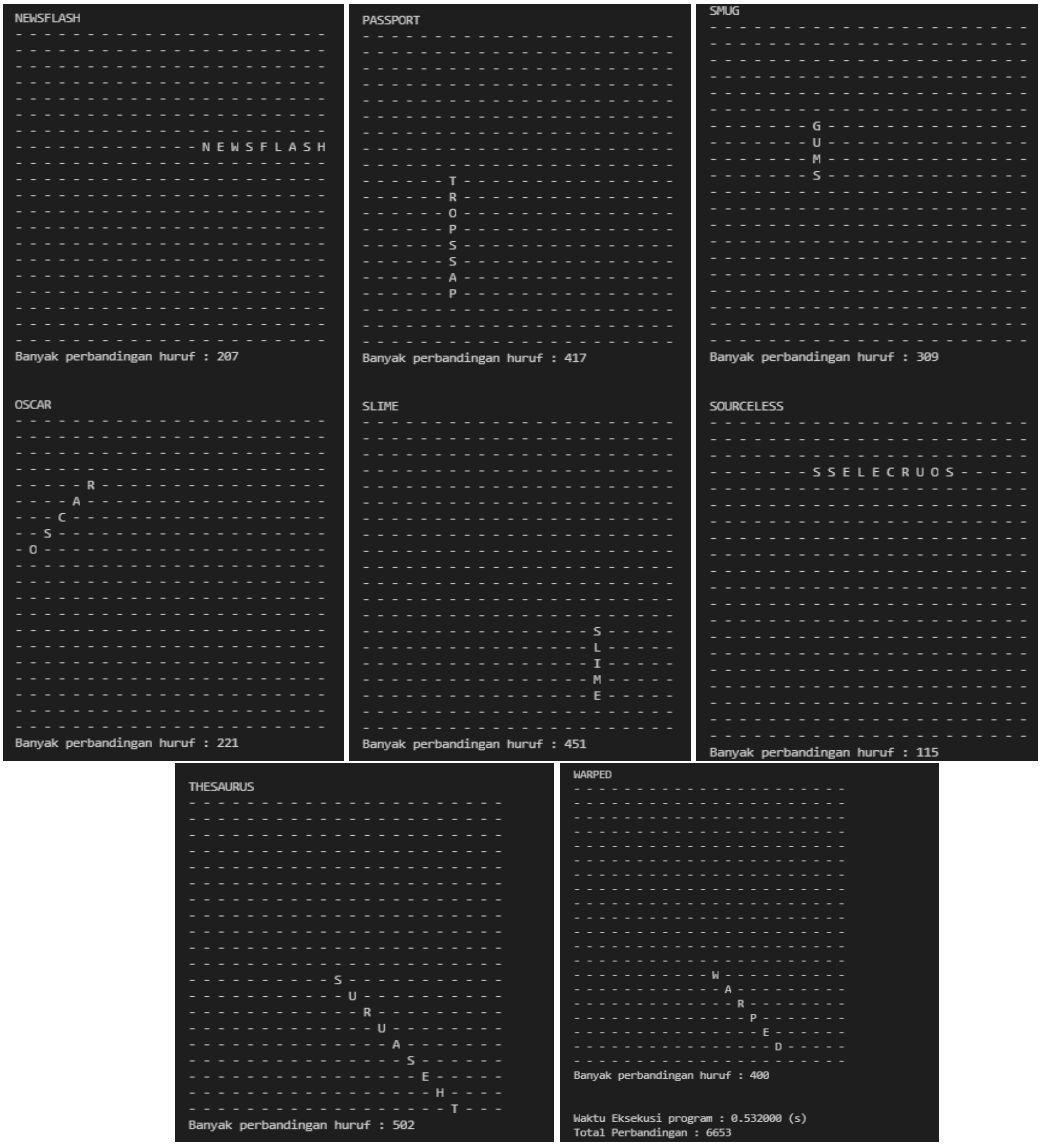
3. small3.txt



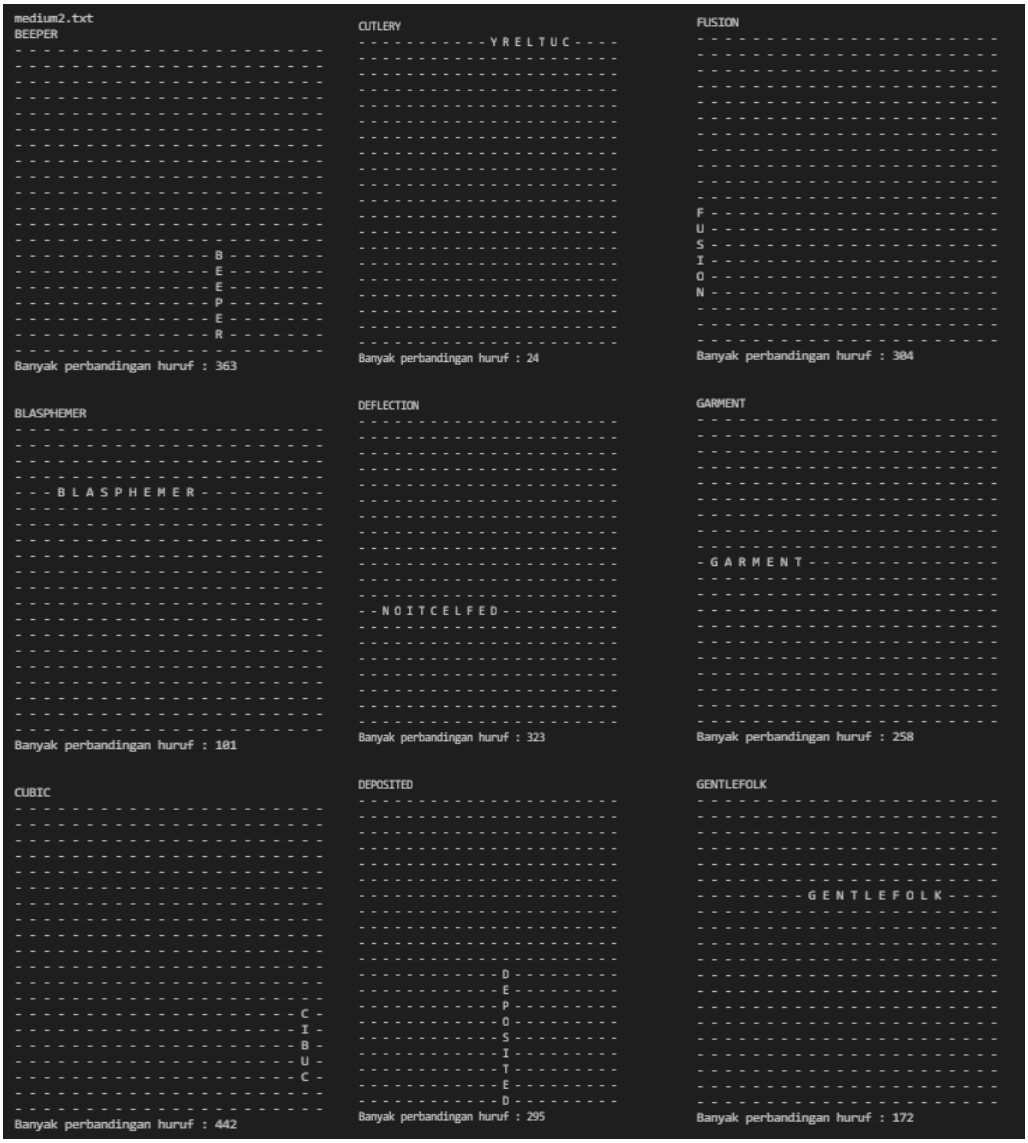


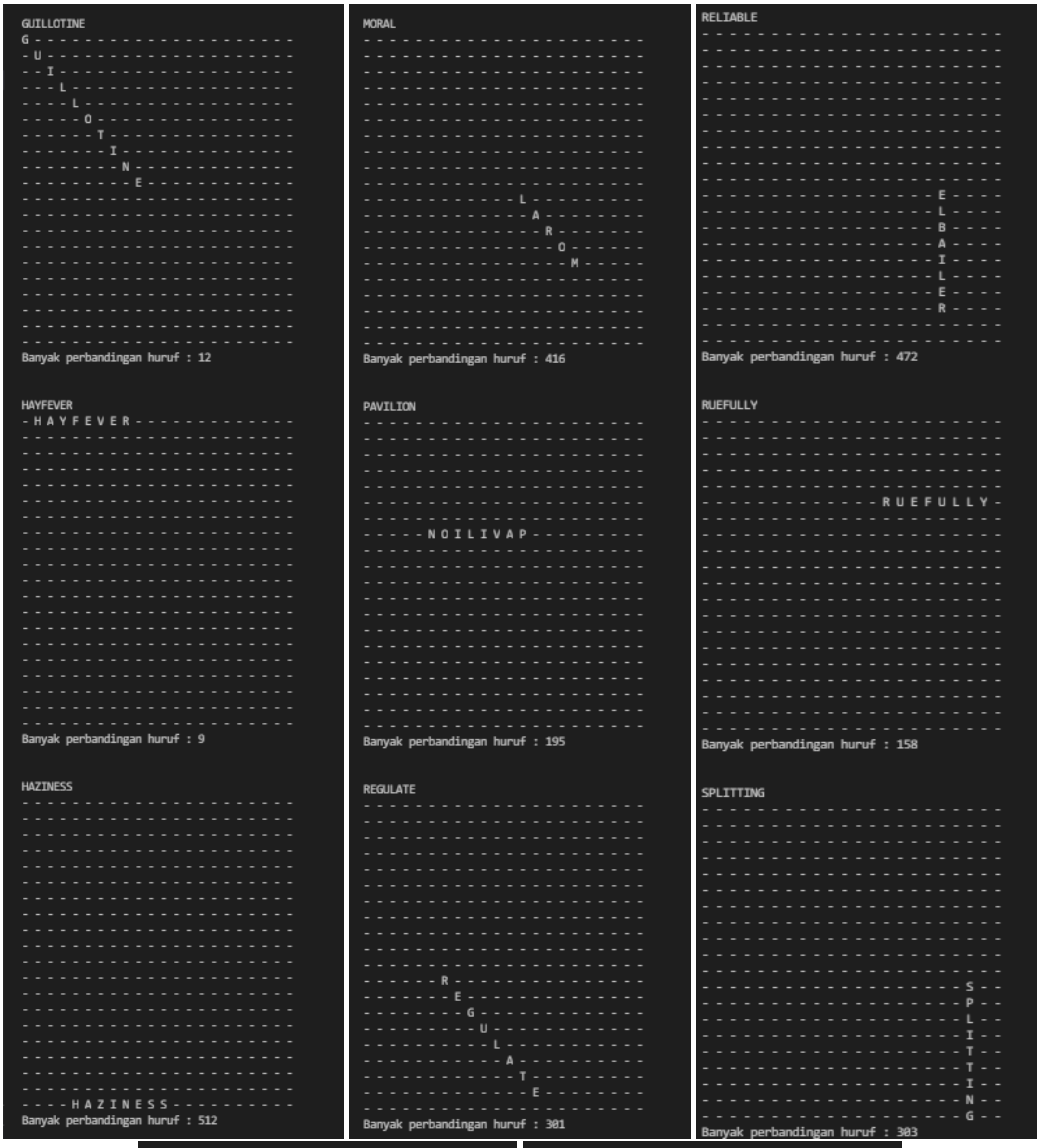
4. medium1.txt


medium1.txt
ABASH
Banyak perbandingan huruf : 333
ASSERTING
A S S E R T I N G
Banyak perbandingan huruf : 31
ASTHMA
Banyak perbandingan huruf : 481
CELSIUS
S U I S L E C
Banyak perbandingan huruf : 536
DECOMPRESS
Banyak perbandingan huruf : 299
DESPISING
G N I S I P S E D
Banyak perbandingan huruf : 121
DRAUGHT
Banyak perbandingan huruf : 451
GLANCE
Banyak perbandingan huruf : 57
LAUGH
Banyak perbandingan huruf : 505
LEES
L E E S
Banyak perbandingan huruf : 556
LEFTY
Banyak perbandingan huruf : 366
MIGHT
Banyak perbandingan huruf : 295

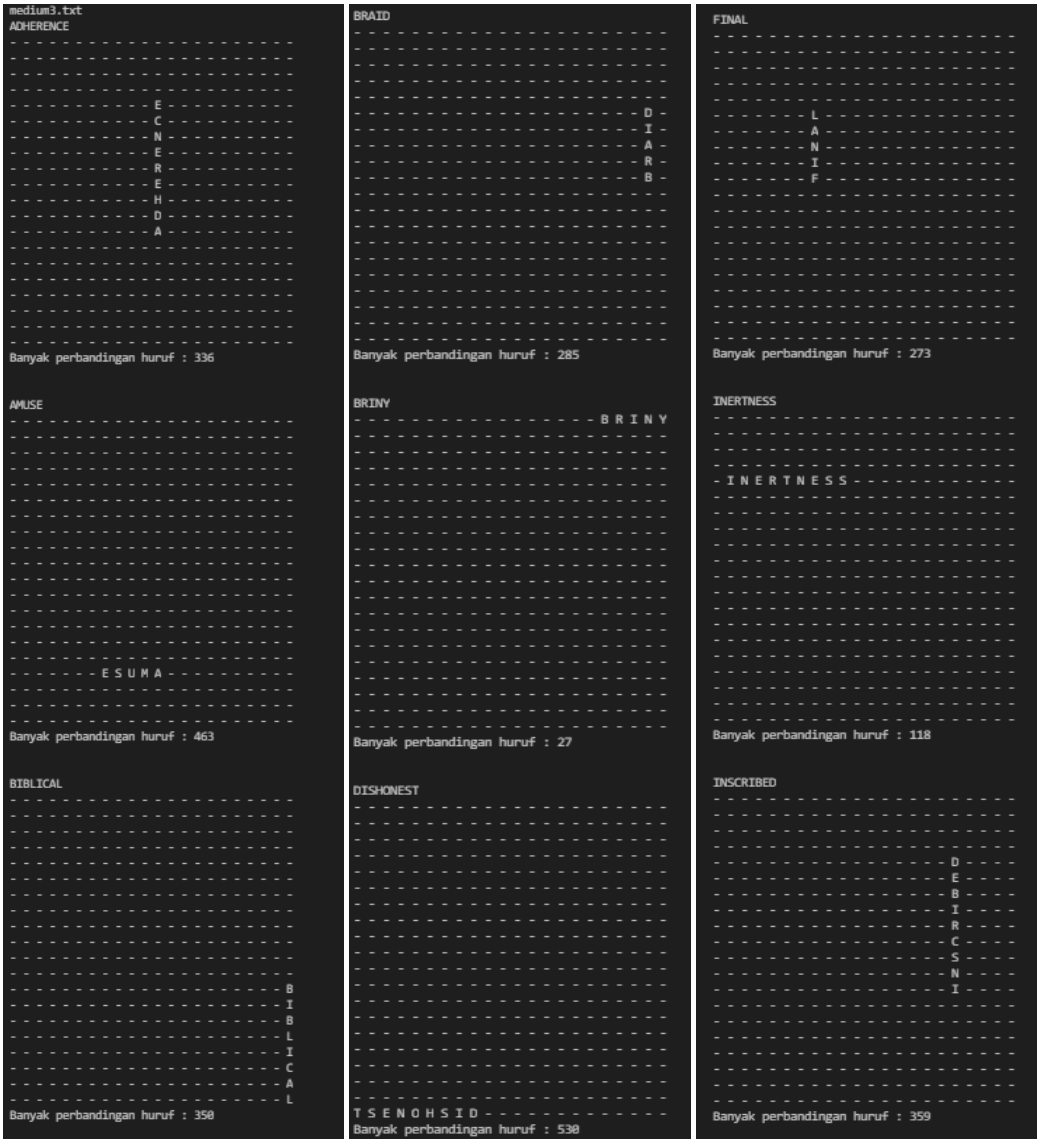


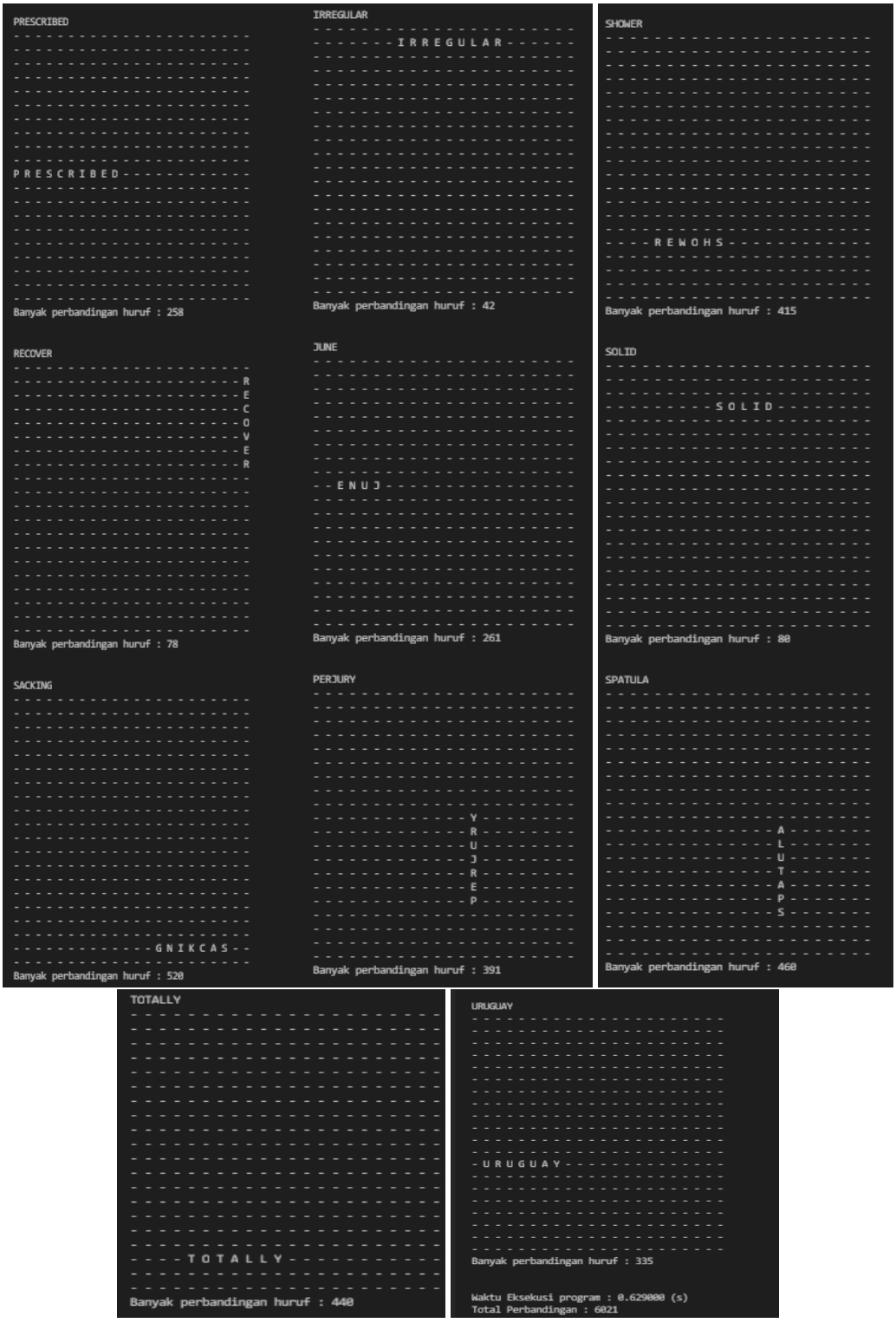
5. medium2.txt


medium2.txt
BEEPER
Banyak perbandingan huruf : 363
CUTLERY
Banyak perbandingan huruf : 24
FUSION
Banyak perbandingan huruf : 304
BLASPHEMER
Banyak perbandingan huruf : 101
DEFLECTION
Banyak perbandingan huruf : 323
GARMENT
Banyak perbandingan huruf : 258
CUBIC
Banyak perbandingan huruf : 442
DEPOSITED
Banyak perbandingan huruf : 295
GENTLEFOLK
Banyak perbandingan huruf : 172

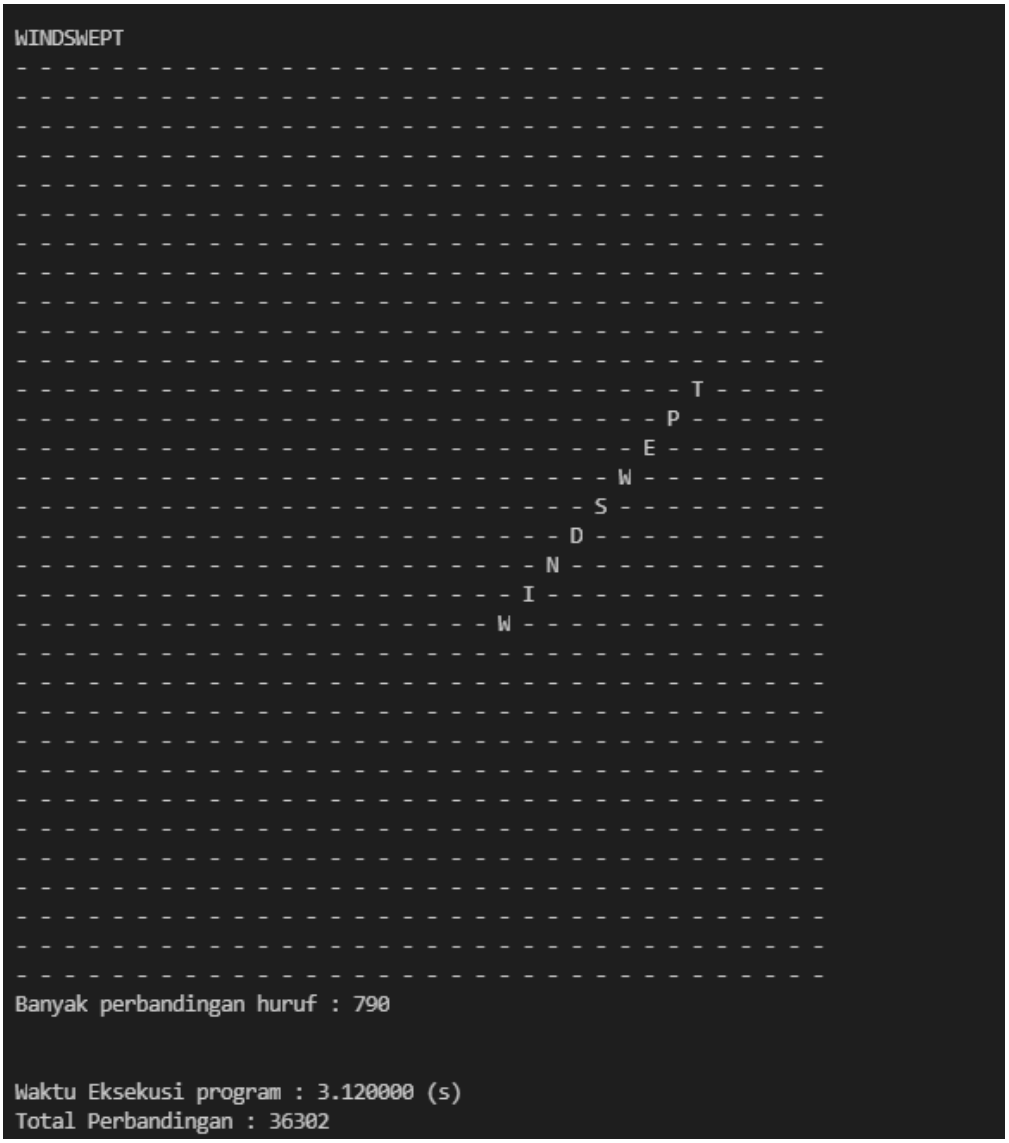

GUILLOTINE
Banyak perbandingan huruf : 12
MORAL
Banyak perbandingan huruf : 416
RELIABLE
Banyak perbandingan huruf : 472
HAYFEVER
Banyak perbandingan huruf : 9
PAVILION
Banyak perbandingan huruf : 195
RUEFULLY
Banyak perbandingan huruf : 158
HAZINESS
Banyak perbandingan huruf : 512
REGULATE
Banyak perbandingan huruf : 301
SPLITTING
Banyak perbandingan huruf : 303
SPRINGIEST
Banyak perbandingan huruf : 35
TROUNCED
Banyak perbandingan huruf : 486
Waktu Eksekusi program : 0.669000 (s)
Total Perbandingan : 5181

6. medium3

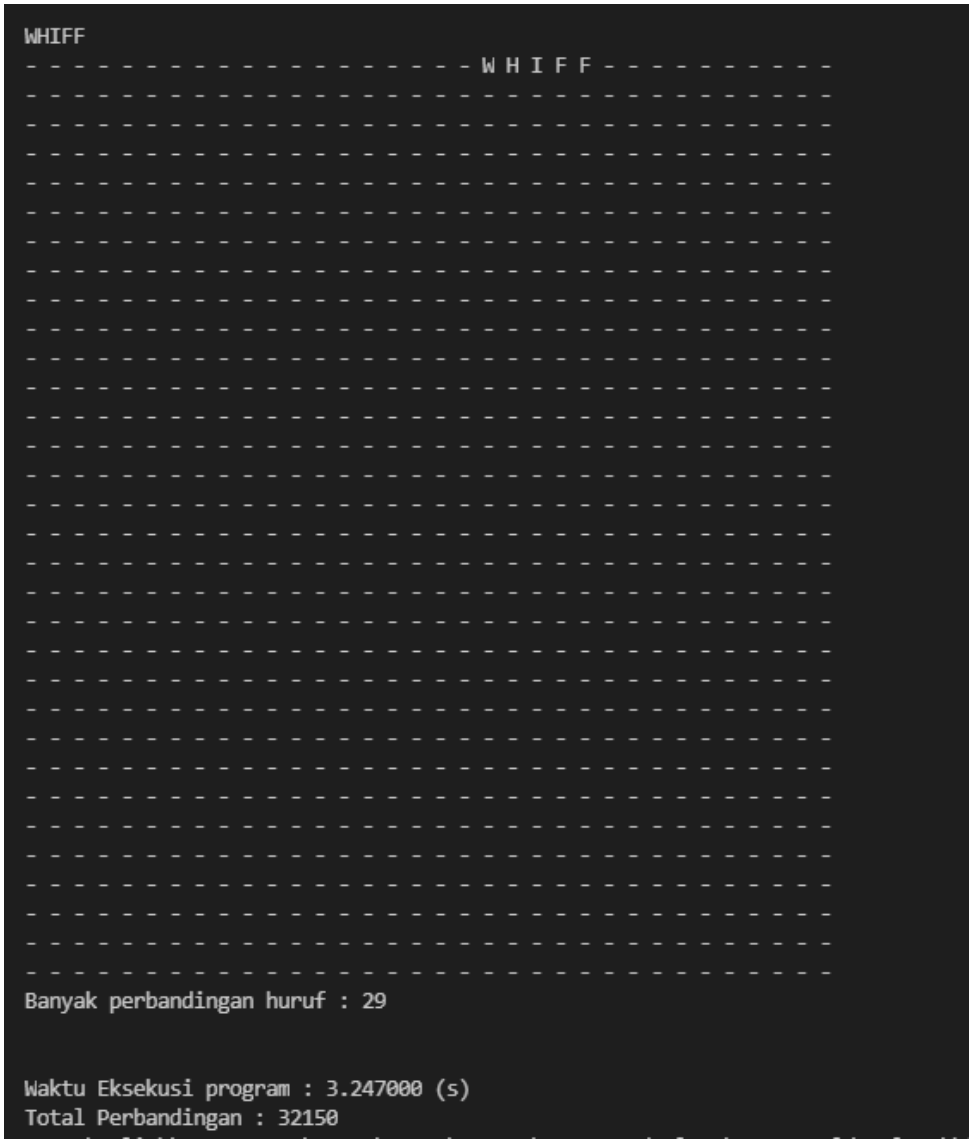

medium3.txt
ADHERENCE
Banyak perbandingan huruf : 336
AMUSE
Banyak perbandingan huruf : 463
BIBLICAL
Banyak perbandingan huruf : 350
BRAID
Banyak perbandingan huruf : 285
BRINY
Banyak perbandingan huruf : 27
DISHONEST
Banyak perbandingan huruf : 530
FINAL
Banyak perbandingan huruf : 273
INERTNESS
Banyak perbandingan huruf : 118
INSCRIBED
Banyak perbandingan huruf : 359


PRESCRIBED
PRESCRIBED
Banyak perbandingan huruf : 258
IRREGULAR
IRREGULAR
Banyak perbandingan huruf : 42
SHOWER
REWOHS
Banyak perbandingan huruf : 415
RECOVER
RECOVER
Banyak perbandingan huruf : 78
JUNE
ENUJ
Banyak perbandingan huruf : 261
SOLID
SOLID
Banyak perbandingan huruf : 80
SACKING
GNIKCAS
Banyak perbandingan huruf : 520
PERJURY
YRUJREP
Banyak perbandingan huruf : 391
SPATULA
ALUTAPS
Banyak perbandingan huruf : 460
TOTALLY
TOTALLY
Banyak perbandingan huruf : 440
URUGUAY
URUGUAY
Banyak perbandingan huruf : 335
Waktu Eksekusi program : 0.629000 (s)
Total Perbandingan : 6021

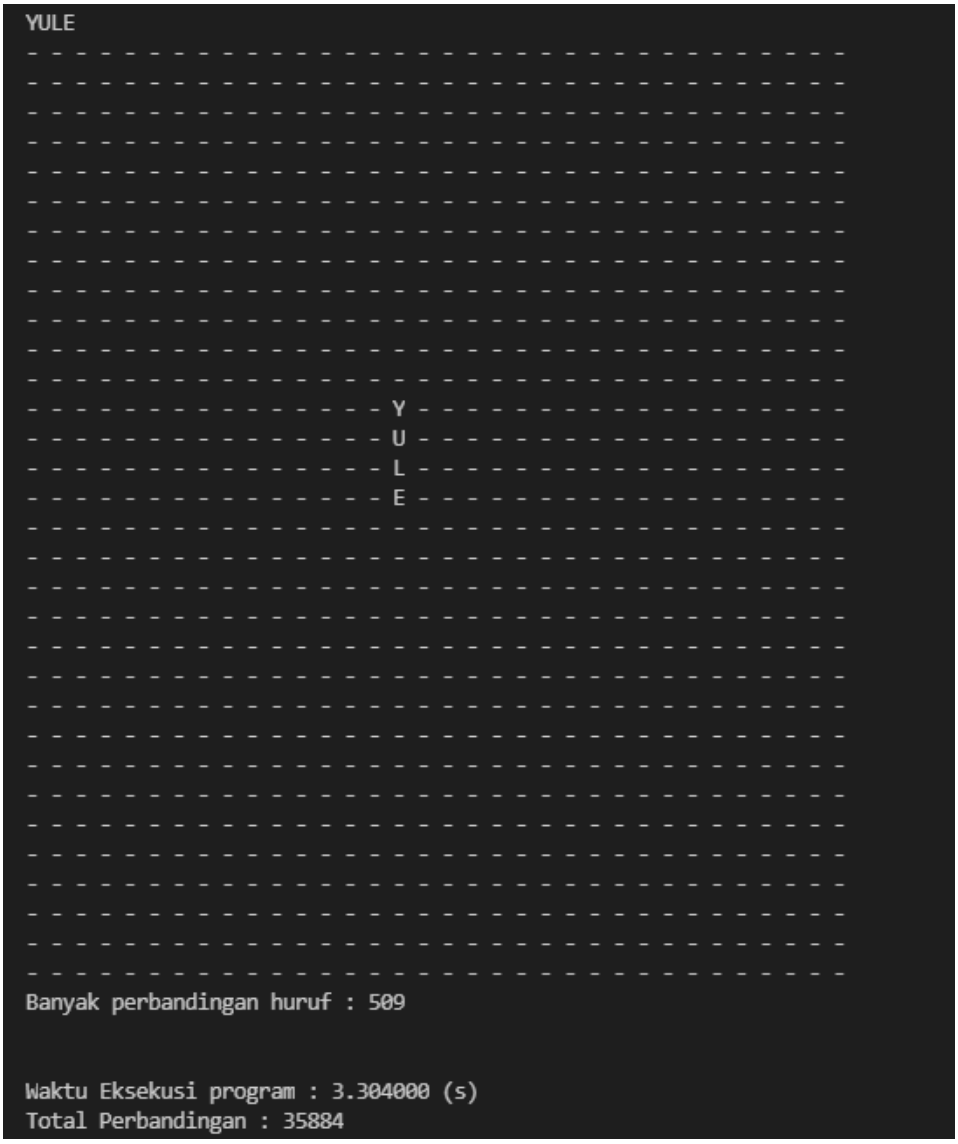
7. large1.txt


WINDSWEPT
Banyak perbandingan huruf : 790
Waktu Eksekusi program : 3.120000 (s)
Total Perbandingan : 36302

8. large2.txt


WHIFF
W H I F F
Banyak perbandingan huruf : 29
Waktu Eksekusi program : 3.247000 (s)
Total Perbandingan : 32150

9. large3.txt



Pada puzzle berukuran besar, output dari program tidak dapat ditampilkan seluruhnya karena keterbasan perangkat dan terminal. Oleh karena itu hanya ditampilkan output berupa keyword terakhir yang diperiksa, waktu eksekusi, dan total perbandingan pada puzzle berukuran *large*.

SOURCE CODE FILE

https://github.com/SurTan02/WordSearch-Puzzle-Solver

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.	√	