Laporan Tugas Kecil 1
Penyelesaian *Word Search Puzzle* dengan Algoritma *Brute Force*
IF2211 Strategi Algoritma

Disusun Oleh:
Farnas Rozaan Iraqee (13520067)

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2021/2022

Algoritma *Brute Force*

Berikut ini adalah deskripsi langkah-langkah algoritma *brute force* yang digunakan dalam program penyelesaian *word seacrh puzzle*. Pertama, *file* masukan dibaca dan disimpan di dua *nested array*, masing-masing untuk *layout puzzle* dan daftar kata yang hendak dicari di dalam *puzzle*. Setelah *file* masukan dibaca dan disimpan di *nested array*, kita melakukan pencocokan setiap huruf dari setiap kata yang hendak dicari dengan huruf yang ada di *puzzle*. Upaya pencocokan dimulai dari ujung kiri atas sampai dengan ujung kanan bawah *puzzle*. Arah pencocokan dilakukan searah jarum jam, yaitu mulai dari atas, kanan atas, kanan, kanan bawah, bawah, kiri bawah, kiri, sampai kiri atas. Apabila seluruh kata dari suatu huruf berhasil ditemukan di dalam *puzzle*, maka upaya pencocokan dihentikan dan kita beralih ke kata yang hendak dicari selanjutnya. Karena proses pencocokan ini dilakukan per huruf, maka meskipun kita sudah sampai ke huruf terakhir yang hendak dicocokkan, tetapi ternyata tidak cocok, upaya pencocokan diulangi lagi dari awal dengan arah yang berbeda. Namun, sebelum melakukan upaya pencocokan tersebut, program ini terlebih dahulu mengecek apakah panjang huruf yang hendak dicari tidak menyebabkan upaya pencocokan melewati batas dari *layout puzzle* yang ada sehingga kita tidak perlu melakukan upaya pencocokan ke arah tersebut dan langsung berpindah ke arah yang lain. Setelah semua kata sudah berhasil ditemukan di dalam *puzzle*, maka akan ditampilkan letak masing-masing kata di dalam *puzzle* dan jumlah perbandingan huruf yang dilakukan dalam upaya pencocokan sebelumnya serta waktu yang dibutuhkan program dalam melakukan upaya pencocokan terhadap seluruh kata yang terdapat dalam daftar kata yang hendak dicari dalam *puzzle*..

Source Code Program dalam Bahasa C

Program ini memiliki dua *file* .c dan satu *file* header. Berikut adalah *source code program* yang tertulis di ketiga *file* tersebut:

1. wsp.h (header)

```
#ifndef WSP_H
#define WSP_H

#include<stdio.h>
#include<stdlib.h>
#include<string.h>
```

```c
#include<time.h>

char word[2000][30];
char puzzle[40][40];

/* Membaca file puzzle*/
void readFile(int *rowPuzzle, int *totalWords);
/* Menyelesaikan word search puzzle */
void solve(int rowPuzzle, int totalWords);
/* Cek apakah panjang kata cukup di puzzle */
int checkLength(int row, int col);
/* String matching ke atas */
int searchUp(int row, int col, int wordLength, int x, int
*comparison);
/* String matching ke kanan atas */
int searchUpRight(int row, int col, int wordLength, int x, int
*comparison);
/* String matching ke kanan */
int searchRight(int row, int col, int wordLength, int x, int
*comparison);
/* String matching ke kanan bawah */
int searchDownRight(int row, int col, int wordLength, int x,
int *comparison);
/* String matching ke bawah */
int searchDown(int row, int col, int wordLength, int x, int
*comparison);
/* String matching ke kiri bawah */
int searchDownLeft(int row, int col, int wordLength, int x,
int *comparison);
/* String matching ke kiri */
```

```
int searchLeft(int row, int col, int wordLength, int x, int
*comparison);
/* String matching ke kiri atas */
int searchUpLeft(int row, int col, int wordLength, int x, int
*comparison);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
atas)*/
void displaySolutionUp(int rowPuzzle, int row, int col, int
wordLength);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
atas kanan)*/
void displaySolutionUpRight(int rowPuzzle, int row, int col,
int wordLength);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
kanan)*/
void displaySolutionRight(int rowPuzzle, int row, int col, int
wordLength);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
bawah kanan)*/
void displaySolutionDownRight(int rowPuzzle, int row, int col,
int wordLength);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
bawah)*/
void displaySolutionDown(int rowPuzzle, int row, int col, int
wordLength);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
bawah kiri)*/
void displaySolutionDownLeft(int rowPuzzle, int row, int col,
int wordLength);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
kiri)*/
```

```
void displaySolutionLeft(int rowPuzzle, int row, int col, int
wordLength);
/* Menampilkan solusi (kata berada di puzzle dengan posisi ke
atas kiri)*/
void displaySolutionUpLeft(int rowPuzzle, int row, int col,
int wordLength);

#endif
```

2. wsp.c (berisi fungsi-fungsi yang digunakan dalam program)

```
#include "wsp.h"

void readFile(int *rowPuzzle, int *totalWords) {
    FILE *file;
    char fileName[20];
    printf("\nInput file name (../test/namafile.txt): ");
    scanf("%s", fileName);
    file = fopen(fileName, "r");

    if (file == NULL)
    {
        printf("Error in opening file!!!\n");
        exit(1);
    }
    else {
        char line[100], line2[40];
        int i, rows=0, cols;

        do
        {
            fgets(line,sizeof(line),file);
```

```
            cols = 0;
            for (i = 0; i < strlen(line)-1; i++)
            {
                if (line[i] != 32)
                {
                    puzzle[rows][cols] = line[i];
                    cols++;
                }
            }
            rows++;
        } while (puzzle[rows-1][0] != 0);
        *rowPuzzle = rows-1;

        rows = 0;
        while (!feof(file))
        {
            fgets(line2, sizeof(line2), file);
            for (i = 0; i < strlen(line2)-1; i++)
            {
                word[rows][i] = line2[i];
            }
            rows++;
        }
                            word[rows-1][strlen(line2)-1]    =
line2[strlen(line2)-1];

        *totalWords = rows;
    }
}

void solve(int rowPuzzle, int totalWords) {
```

```
    int i, j, k, wordLength, comparison;
        int  wordsDir[totalWords],  listComparison[totalWords],
listWordLength[totalWords];
    int listRow[totalWords], listCol[totalWords];
    int colPuzzle = strlen(puzzle[0]);
    clock_t start = clock();

    for (i = 0; i < totalWords; i++)
    {
        comparison = 0;
        for (j = 0; j < rowPuzzle; j++)
        {
            for (k = 0; k < colPuzzle; k++)
            {
                wordLength = strlen(word[i]);
                listWordLength[i] = wordLength;
                        if  (checkLength(j-wordLength+1,  k)&&
searchUp(j,k,wordLength,i,&comparison))
                {
                    wordsDir[i] = 1;
                    listRow[i] = j;
                    listCol[i] = k;
                    listComparison[i] = comparison;
                }
                        else  if  (checkLength(j-wordLength+1,
k+wordLength-1)                                                 &&
searchUpRight(j,k,wordLength,i,&comparison))
                {
                    wordsDir[i] = 2;
                    listRow[i] = j;
                    listCol[i] = k;
```

```
                    listComparison[i] = comparison;
                }
                    else if (checkLength(j, k+wordLength-1) &&
searchRight(j,k,wordLength,i,&comparison))
                {
                    wordsDir[i] = 3;
                    listRow[i] = j;
                    listCol[i] = k;
                    listComparison[i] = comparison;
                }
                        else if (checkLength(j+wordLength-1,
k+wordLength-1)                                             &&
searchDownRight(j,k,wordLength,i,&comparison))
                {
                    wordsDir[i] = 4;
                    listRow[i] = j;
                    listCol[i] = k;
                    listComparison[i] = comparison;
                }
                    else if (checkLength(j+wordLength-1, k) &&
searchDown(j,k,wordLength,i,&comparison))
                {
                    wordsDir[i] = 5;
                    listRow[i] = j;
                    listCol[i] = k;
                    listComparison[i] = comparison;
                }
                        else if (checkLength(j+wordLength-1,
k-wordLength+1)                                             &&
searchDownLeft(j,k,wordLength,i,&comparison))
                {
```

```
                    wordsDir[i] = 6;
                    listRow[i] = j;
                    listCol[i] = k;
                    listComparison[i] = comparison;
                }
                    else if (checkLength(j, k-wordLength+1) &&
searchLeft(j,k,wordLength,i,&comparison))
                {
                    wordsDir[i] = 7;
                    listRow[i] = j;
                    listCol[i] = k;
                    listComparison[i] = comparison;
                }
                        else if (checkLength(j-wordLength+1,
k-wordLength+1) && searchUpLeft(j,k,wordLength,i,&comparison))
                {
                    wordsDir[i] = 8;
                    listRow[i] = j;
                    listCol[i] = k;
                    listComparison[i] = comparison;
                }
            }
        }
    }
    clock_t end = clock();
    double seconds = ((double) end - start)/CLOCKS_PER_SEC;
    double ms = 1000 * seconds;

    for (i = 0; i < totalWords; i++)
    {
        printf("\n");
```

```
        if (wordsDir[i] == 1) {
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
                    displaySolutionUp(rowPuzzle,  listRow[i],
listCol[i], listWordLength[i]);
        }
        else if (wordsDir[i] == 2) {
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
                displaySolutionUpRight(rowPuzzle,  listRow[i],
listCol[i], listWordLength[i]);
        }
        else if (wordsDir[i] == 3) {
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
                  displaySolutionRight(rowPuzzle,  listRow[i],
listCol[i], listWordLength[i]);
        }
        else if (wordsDir[i] == 4) {
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
               displaySolutionDownRight(rowPuzzle, listRow[i],
listCol[i], listWordLength[i]);
        }
        else if (wordsDir[i] == 5) {
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
                   displaySolutionDown(rowPuzzle,  listRow[i],
listCol[i], listWordLength[i]);
        }
        else if (wordsDir[i] == 6) {
```

```
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
                displaySolutionDownLeft(rowPuzzle, listRow[i],
listCol[i], listWordLength[i]);
        }
        else if (wordsDir[i] == 7) {
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
                    displaySolutionLeft(rowPuzzle, listRow[i],
listCol[i], listWordLength[i]);
        }
        else if (wordsDir[i] == 8) {
            printf("%d. %s - %d comparisons \n", i+1, word[i],
listComparison[i]);
                  displaySolutionUpLeft(rowPuzzle, listRow[i],
listCol[i], listWordLength[i]);
        }
    }

    printf("\nTime elapsed = %f s or %f ms\n", seconds, ms);
}

int checkLength(int row, int col) {
    if (puzzle[row][col] != 0)
    {
        return 1;
    }
    else
    {
        return 0;
    }
```

```
}

int searchUp(int row, int col, int wordLength, int x, int
*comparison) {
    int match = 1;
    int i = 0;
    while (wordLength != 0 && match == 1)
    {
        if (puzzle[row][col] == word[x][i])
        {
            row--;
            i++;
        }
        else
        {
            match = 0;
        }
        *comparison += 1;
        wordLength--;
    }
    return match;
}

int searchUpRight(int row, int col, int wordLength, int x, int
*comparison) {
    int match = 1;
    int i = 0;
    while (wordLength != 0 && match == 1)
    {
        if (puzzle[row][col] == word[x][i])
        {
```

```
            row--;
            col++;
            i++;
        }
        else
        {
            match = 0;
        }
        *comparison += 1;
        wordLength--;
    }
    return match;
}

int searchRight(int row, int col, int wordLength, int x, int
*comparison) {
    int match = 1;
    int i = 0;
    while (wordLength != 0 && match == 1)
    {
        if (puzzle[row][col] == word[x][i])
        {
            col++;
            i++;
        }
        else
        {
            match = 0;
        }
        *comparison += 1;
        wordLength--;
```

```
    }
    return match;
}

int searchDownRight(int row, int col, int wordLength, int x,
int *comparison) {
    int match = 1;
    int i = 0;
    while (wordLength != 0 && match == 1)
    {
        if (puzzle[row][col] == word[x][i])
        {
            row++;
            col++;
            i++;
        }
        else
        {
            match = 0;
        }
        *comparison += 1;
        wordLength--;
    }
    return match;
}

int searchDown(int row, int col, int wordLength, int x, int
*comparison) {
    int match = 1;
    int i = 0;
    while (wordLength != 0 && match == 1)
```

```
    {
        if (puzzle[row][col] == word[x][i])
        {
            row++;
            i++;
        }
        else
        {
            match = 0;
        }
        *comparison += 1;
        wordLength--;
    }
    return match;
}

int searchDownLeft(int row, int col, int wordLength, int x,
int *comparison) {
    int match = 1;
    int i = 0;
    while (wordLength != 0 && match == 1)
    {
        if (puzzle[row][col] == word[x][i])
        {
            row++;
            col--;
            i++;
        }
        else
        {
            match = 0;
```

```
        }
        *comparison += 1;
        wordLength--;
    }
    return match;
}

int searchLeft(int row, int col, int wordLength, int x, int
*comparison) {
    int match = 1;
    int i = 0;
    while (wordLength != 0 && match == 1)
    {
        if (puzzle[row][col] == word[x][i])
        {
            col--;
            i++;
        }
        else
        {
            match = 0;
        }
        *comparison += 1;
        wordLength--;
    }
    return match;
}

int searchUpLeft(int row, int col, int wordLength, int x, int
*comparison) {
    int match = 1;
```

```
    int i = 0;
    while (wordLength != 0 && match == 1)
    {
        if (puzzle[row][col] == word[x][i])
        {
            row--;
            col--;
            i++;
        }
        else
        {
            match = 0;
        }
        *comparison += 1;
        wordLength--;
    }
    return match;
}

void displaySolutionUp(int rowPuzzle, int row, int col, int
wordLength) {
    int upLimit = row-wordLength+1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
    {
        for (int j = 0; j < colPuzzle; j++)
        {
            if (i >= upLimit && i <= row && j == col)
            {
                printf("%c ", puzzle[i][j]);
            }
```

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

void displaySolutionUpRight(int rowPuzzle, int row, int col,
int wordLength) {
    int x = 0;
    int upLimit = row-wordLength+1;
    int rightLimit = col+wordLength-1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
    {
        for (int j = 0; j < colPuzzle; j++)
        {
            if (i == upLimit+x && i <= row && j >= col && j ==
rightLimit-x)
            {
                printf("%c ", puzzle[i][j]);
                x++;
            }
            else
            {
                printf("- ");
            }
        }
        printf("\n");
```

```
    }
}

void displaySolutionRight(int rowPuzzle, int row, int col, int
wordLength) {
    int rightLimit = col+wordLength-1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
    {
        for (int j = 0; j < colPuzzle; j++)
        {
            if (i == row && j >= col && j <= rightLimit)
            {
                printf("%c ", puzzle[i][j]);
            }
            else
            {
                printf("- ");
            }
        }
        printf("\n");
    }
}

void displaySolutionDownRight(int rowPuzzle, int row, int col,
int wordLength) {
    int x = 0;
    int lowLimit = row+wordLength-1;
    int rightLimit = col+wordLength-1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
```

```
    {
        for (int j = 0; j < colPuzzle; j++)
        {
            if (i == row+x && i <= lowLimit && j == col+x && j
<= rightLimit)
            {
                printf("%c ", puzzle[i][j]);
                x++;
            }
            else
            {
                printf("- ");
            }
        }
        printf("\n");
    }
}

void displaySolutionDown(int rowPuzzle, int row, int col, int
wordLength) {
    int lowLimit = row+wordLength-1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
    {
        for (int j = 0; j < colPuzzle; j++)
        {
            if (i >= row && i <= lowLimit && j == col)
            {
                printf("%c ", puzzle[i][j]);
            }
            else
```

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

void displaySolutionDownLeft(int rowPuzzle, int row, int col,
int wordLength) {
    int x = 0;
    int lowLimit = row+wordLength-1;
    int leftLimit = col-wordLength+1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
    {
        for (int j = 0; j < colPuzzle; j++)
        {
             if (i == row+x && i <= lowLimit && j >= leftLimit
&& j == col-x)
            {
                printf("%c ", puzzle[i][j]);
                x++;
            }
            else
            {
                printf("- ");
            }
        }
        printf("\n");
    }
```

```
}

void displaySolutionLeft(int rowPuzzle, int row, int col, int
wordLength) {
    int leftLimit = col-wordLength+1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
    {
        for (int j = 0; j < colPuzzle; j++)
        {
            if (i == row && j >= leftLimit && j <= col)
            {
                printf("%c ", puzzle[i][j]);
            }
            else
            {
                printf("- ");
            }
        }
        printf("\n");
    }
}

void displaySolutionUpLeft(int rowPuzzle, int row, int col,
int wordLength) {
    int x = 0;
    int upLimit = row-wordLength+1;
    int leftLimit = col-wordLength+1;
    int colPuzzle = strlen(puzzle[0]);
    for (int i = 0; i < rowPuzzle; i++)
    {
```

```
        for (int j = 0; j < colPuzzle; j++)
        {
            if (i == upLimit+x && i <= row && j == leftLimit+x
&& j <= col)
            {
                printf("%c ", puzzle[i][j]);
                x++;
            }
            else
            {
                printf("- ");
            }
        }
        printf("\n");
    }
}
```

3. main.c (berisi program inti)

```
#include "wsp.h"

int main() {
    int rowPuzzle, totalWords;
    readFile(&rowPuzzle, &totalWords);
    solve(rowPuzzle, totalWords);
    return 0;
}
```

Screenshot Input dan *Output*

Untuk *screenshot output* ukuran *medium-large* hanya akan ditampilkan beberapa saja dengan tujuan menghemat tempat dan karena untuk ukuran *large* seluruh hasil tidak dapat ditampilkan secara bersamaan ketika *program* dijalankan karena ruang terminal yang terbatas sehingga tidak memungkinkan untuk semuanya disertakan di laporan.

1. small1.txt

   Input:

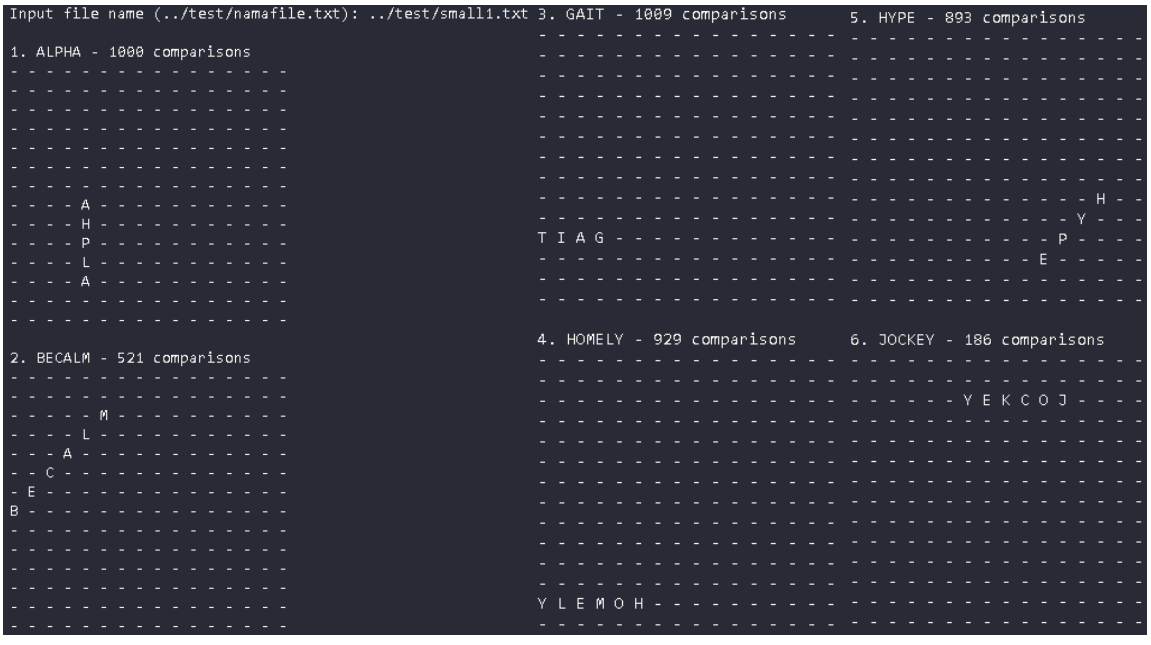
Puzzle

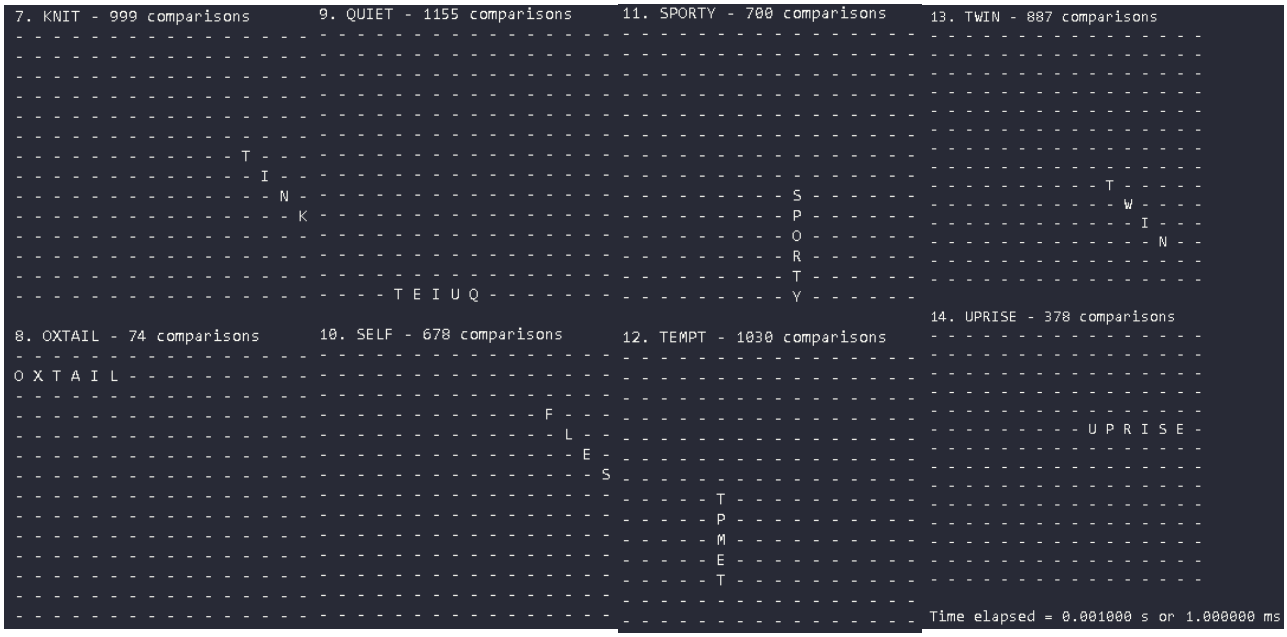
```
M P Q W V Q S O J E T O X G E P
O X T A I L S P J M C T P Q R U
D O W X P M Y E K C O J S B R Z
I O H E L C W W T E I C F B Q V
O L K A H M N D I T I V Y L W S
H Y C D B U N P Y U P R I S E B
W E R I C N F H X O R C T M J S
B M U N A T J R N P K A S I C Q
W M T M H P J U O S T J Y H N Q
E J D T P 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
```

Word

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

   Output:





2. small2.txt
   Input:

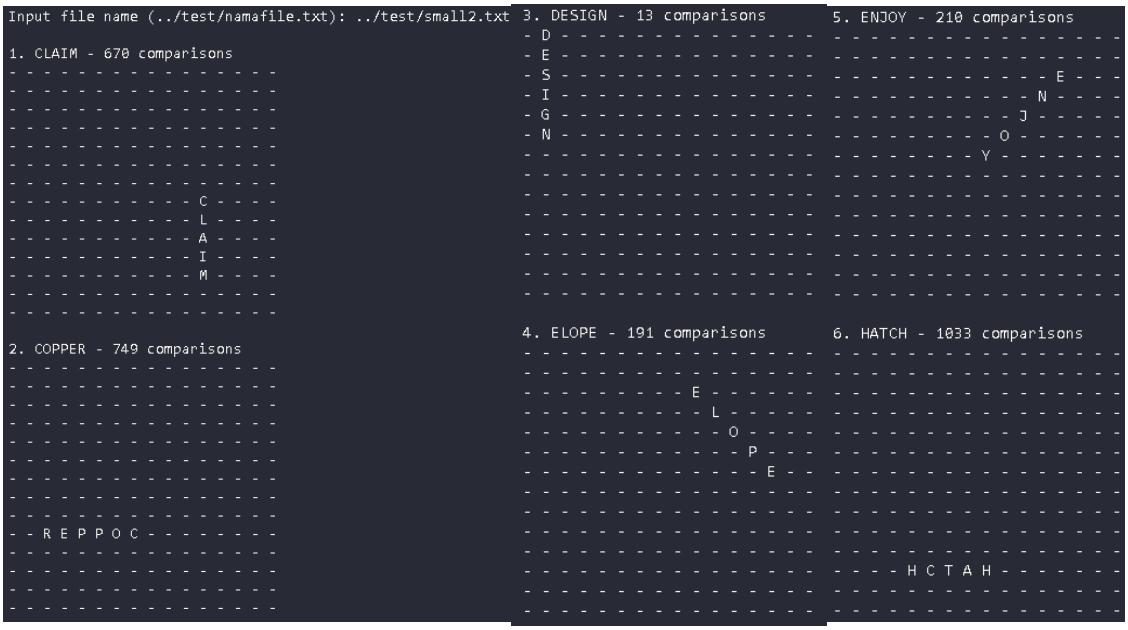
Puzzle

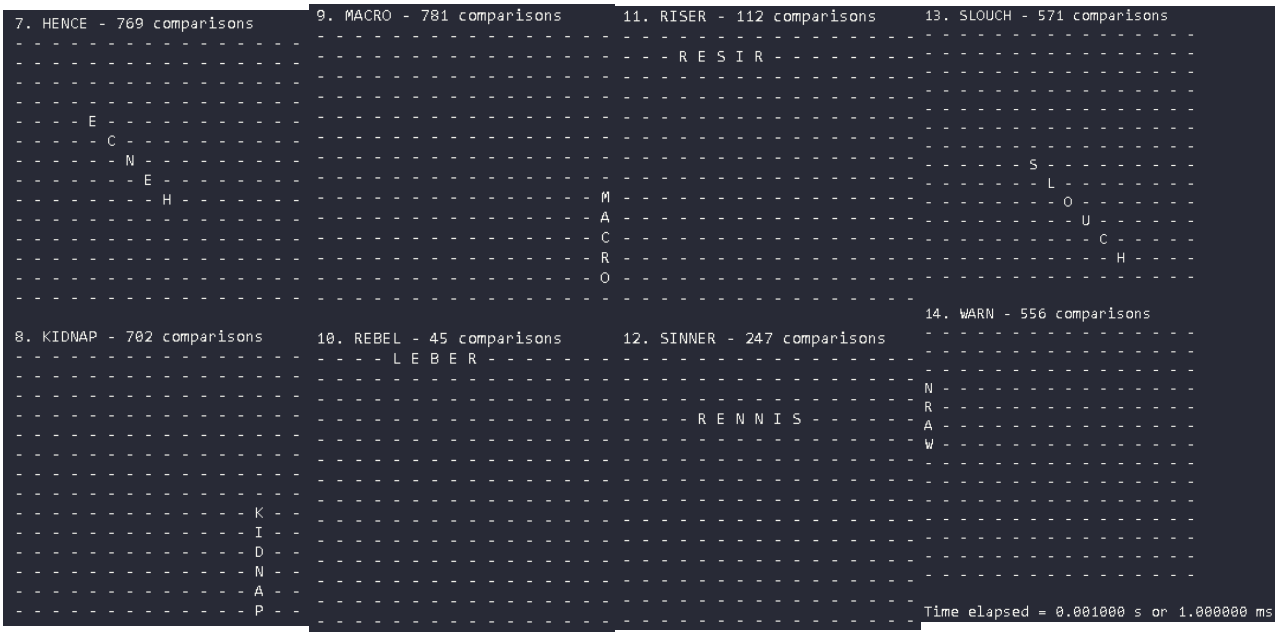
J	D	I	Y	L	E	B	E	R	A	Y	U	U	T	P
B	E	B	R	E	S	I	R	O	S	D	H	T	I	J	N
X	S	Y	Y	W	O	Y	R	H	E	Z	L	E	Y	P	M
N	I	E	S	R	E	N	N	I	S	L	N	B	I	N	F
R	G	A	E	E	N	H	W	O	J	J	O	A	D	H	O
A	N	A	T	B	C	X	U	F	O	J	L	P	U	W	X
W	O	Q	G	P	Q	N	S	Y	Y	O	D	J	E	W	U
T	Y	U	C	D	A	S	E	Y	L	F	C	Q	F	I	T
Q	Y	E	Q	E	K	B	L	H	K	I	L	I	K	V	M
R	K	R	E	P	P	O	C	O	M	Q	A	C	I	H	A
E	G	Q	A	Z	I	Z	N	B	U	O	I	K	D	D	C
K	D	H	F	H	C	T	A	H	B	C	M	G	N	P	R
W	I	P	X	T	A	U	Q	C	C	N	H	P	A	L	O
A	R	Z	K	B	I	V	C	X	P	S	E	P	P	B	H

Word

CLAIM
COPPER
DESIGN
ELOPE
ENJOY
HATCH
HENCE
KIDNAP
MACRO
REBEL
RISER
SINNER
SLOUCH
WARN

Output:


Input file name (../test/namafile.txt): ../test/small2.txt
1. CLAIM - 670 comparisons
2. COPPER - 749 comparisons
3. DESIGN - 13 comparisons
4. ELOPE - 191 comparisons
5. ENJOY - 210 comparisons
6. HATCH - 1033 comparisons


7. HENCE - 769 comparisons
8. KIDNAP - 702 comparisons
9. MACRO - 781 comparisons
10. REBEL - 45 comparisons
11. RISER - 112 comparisons
12. SINNER - 247 comparisons
13. SLOUCH - 571 comparisons
14. WARN - 556 comparisons
Time elapsed = 0.001000 s or 1.000000 ms

3. small3.txt

Input:

Puzzle

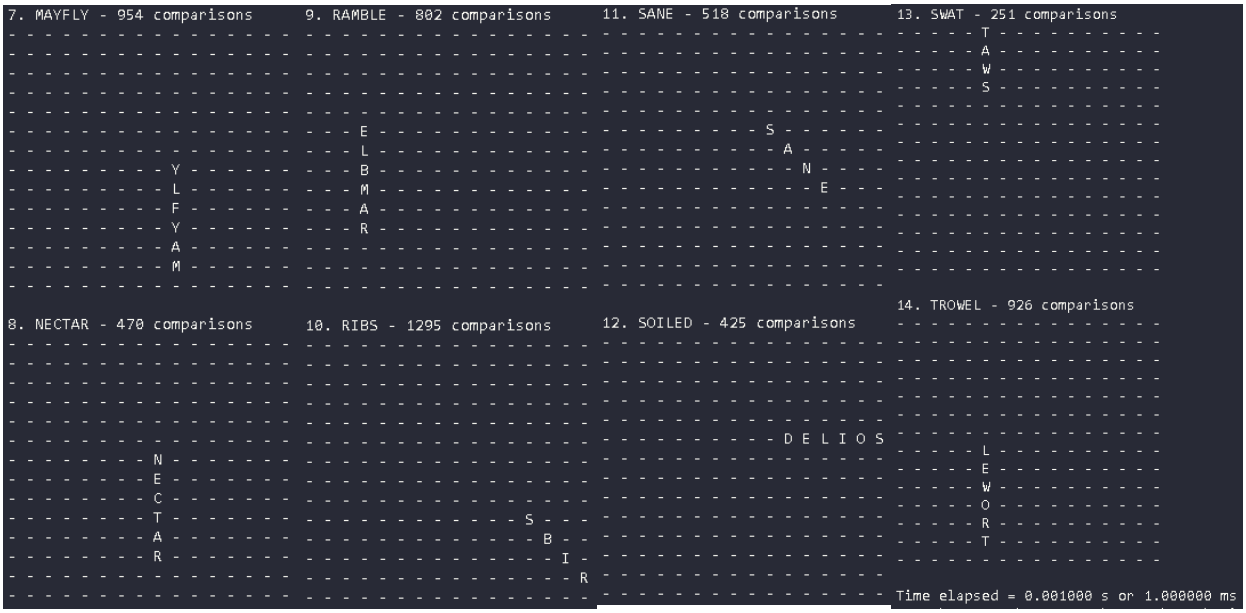
```
R Z C W S T J J Z Z X W X P G J
L K E L S A B I U O C I V P W B
U C O M U W Y V Y W F W D P A T
P I Y U C S P J P Q L D D R V X
I F W B D A G A L L E Y R A A O
H X T E M R N L X S D E L I O S
Y X E L F E O H N W A T C L P X
J U T B M L N P E Y B N O Y S K
S M T M V E M W C L R R E W P H
G H G A E W E C T F E Q S B Z R
O E E R G O B Y A Y W I S B P U
C L I W R R N U R A H H I L I A
M J A Y E T M W Q M W E I F U R
B M A J I S R M C I T H S T G E
```

Word

```
BREW
COGS
CUSS
DROP
EGRET
GALLEY
MAYFLY
NECTAR
RAMBLE
RIBS
SANE
SOILED
SWAT
TROWEL
```

Output:

```
Input file name (../test/namafile.txt): ../test/small3.txt

1. BREW - 752 comparisons
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - B - - - -
- - - - - - - - - - R - - - -
- - - - - - - - - - E - - - -
- - - - - - - - - - W - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -

2. COGS - 1134 comparisons
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
S - - - - - - - - - - - - - -
G - - - - - - - - - - - - - -
O - - - - - - - - - - - - - -
C - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -

3. CUSS - 242 comparisons
- - - - S - - - - - - - - - -
- - - - S - - - - - - - - - -
- - - - U - - - - - - - - - -
- - - - C - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -

4. DROP - 372 comparisons
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - D - - - - - - - - - -
- - - - - - R - - - - - - - -
- - - - - - - O - - - - - - -
- - - - - - - - P - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -

5. EGRET - 867 comparisons
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - E - - - - - - - - - -
- - - - G - - - - - - - - - -
- - - - R - - - - - - - - - -
- - - - E - - - - - - - - - -
- - - - T - - - - - - - - - -

6. GALLEY - 269 comparisons
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - G A L L E Y - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
- - - - - - - - - - - - - - -
```



4. medium1.txt

   Input:

Puzzle

D	Z	N	F	B	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

Word

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

Output:

```
Input file name (../test/namafile.txt): ../test/medium1.txt

1. ADHERENCE - 1236 comparisons
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - E - - - - - - - - -
- - - - - - - - - - C - - - - - - - - -
- - - - - - - - - - N - - - - - - - - -
- - - - - - - - - - E - - - - - - - - -
- - - - - - - - - - R - - - - - - - - -
- - - - - - - - - - E - - - - - - - - -
- - - - - - - - - - H - - - - - - - - -
- - - - - - - - - - D - - - - - - - - -
- - - - - - - - - - A - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
```

```
2. AMUSE - 2349 comparisons
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - E S U M A - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
```

```
3. BIBLICAL - 1437 comparisons
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - B -
- - - - - - - - - - - - - - - - - - I -
- - - - - - - - - - - - - - - - - - B -
- - - - - - - - - - - - - - - - - - L -
- - - - - - - - - - - - - - - - - - I -
- - - - - - - - - - - - - - - - - - C -
- - - - - - - - - - - - - - - - - - A -
- - - - - - - - - - - - - - - - - - L -
```

```
18. SPATULA - 1961 comparisons
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - A - -
- - - - - - - - - - - - - - - - - L - -
- - - - - - - - - - - - - - - - - U - -
- - - - - - - - - - - - - - - - - T - -
- - - - - - - - - - - - - - - - - A - -
- - - - - - - - - - - - - - - - - P - -
- - - - - - - - - - - - - - - - - S - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
```

```
20. URUGUAY - 1433 comparisons
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - U R U G U A Y - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - -
Time elapsed = 0.001000 s or 1.000000 ms
```

5. medium2.txt

Input:

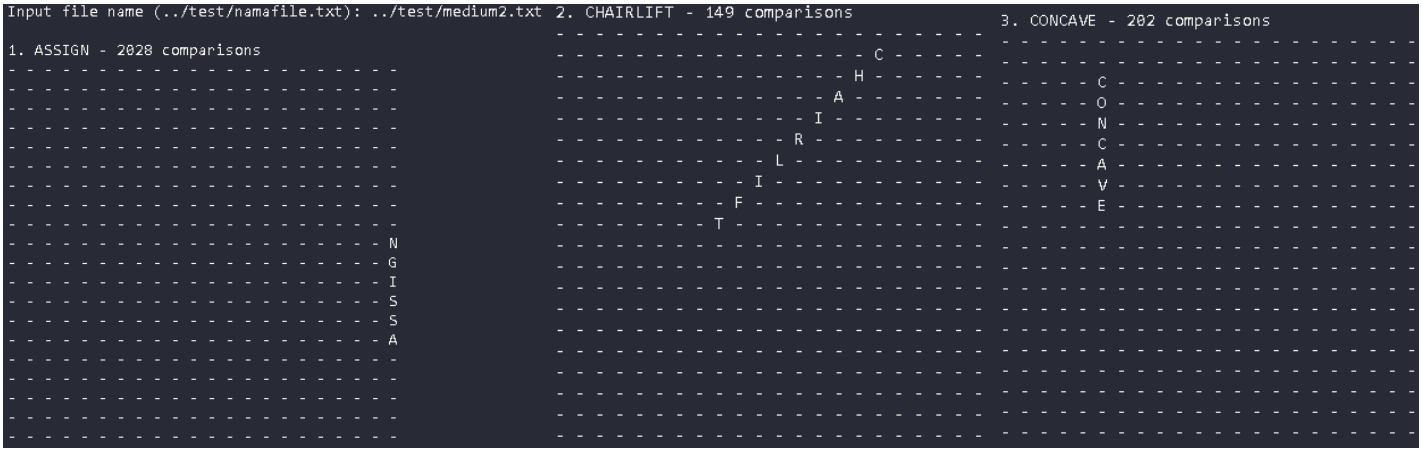
Puzzle

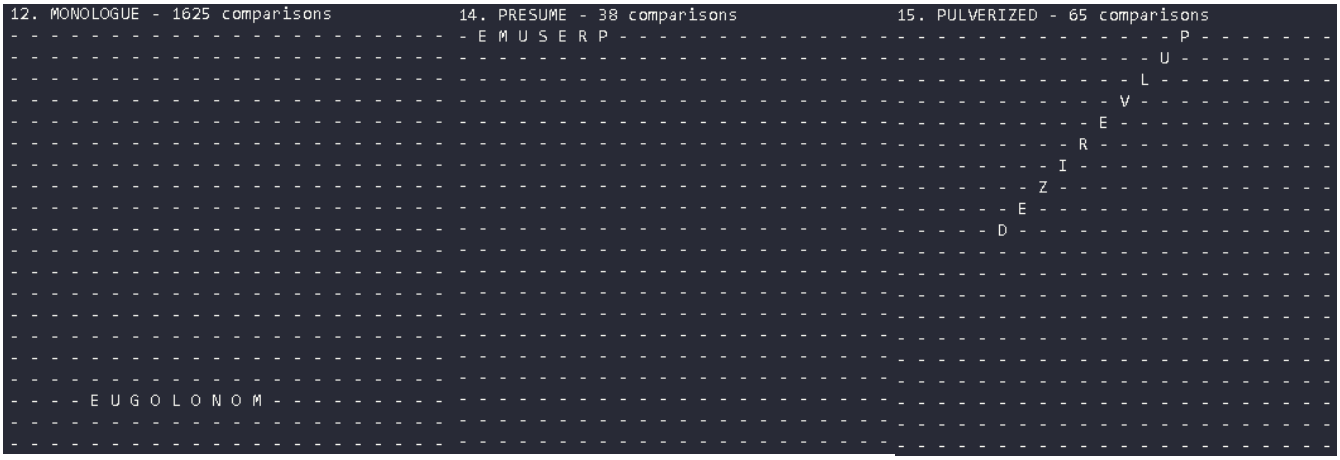
```
F E M U S E R P Q N V V S R P Q K W Z N G P
R W O N N I M Q Y O H N F U W E C D S R O Q
Z Z P N N C D B V X O G L A Y H V C D W J T
B I A B J O O A F O T V J L A C V K I Y P O
W P G I G N C O R R E T M I N E K M Z X A B
C K I Y V C T O M R K X R N C W D K T Z F A
M D J I Y A O A I P Y L U U U T E P B B O S
D E O B O V R Z O H I T I Y J N H X Y Y W O
Y R H Z L E E L Y F D P M E L S G B I N F A
E N H W O D L J T E A D H O A L U T B X U N
F J L U W A X O T Q G P O S Y O A D J W U G
T Y U C W D A R Y L F Q S F I G L C T Q Y I
E Q E K B K A T S S T V O E R A K M I Q C S
H E G Q A P Z N T I Z N B V T M B O K P D S
K D H F E B U O G P W I E L S I P X T A Y A
U Q C D C G P N P L A R R O E N Z K B I V T
C X P S N P E P B H O Q O V T G V O Q O Y I
B R F E E U G O L O N O M N T U V G W M O T
A T S D M H L Q C Y A F T I A N U L O I A X
S S H U B F G O A D N J L W F A K E T J J E
```

Word

```
ASSIGN
CHAIRLIFT
CONCAVE
DEPARTED
DOCTOR
FATTEST
GAMING
INVOLVE
LAUGHED
LUNA
MINNOW
MONOLOGUE
OBOE
PRESUME
PULVERIZED
SNUGNESS
SOBER
STOPPED
TYPICALLY
WALLOP
```

Output:





6. medium3.txt

Input:

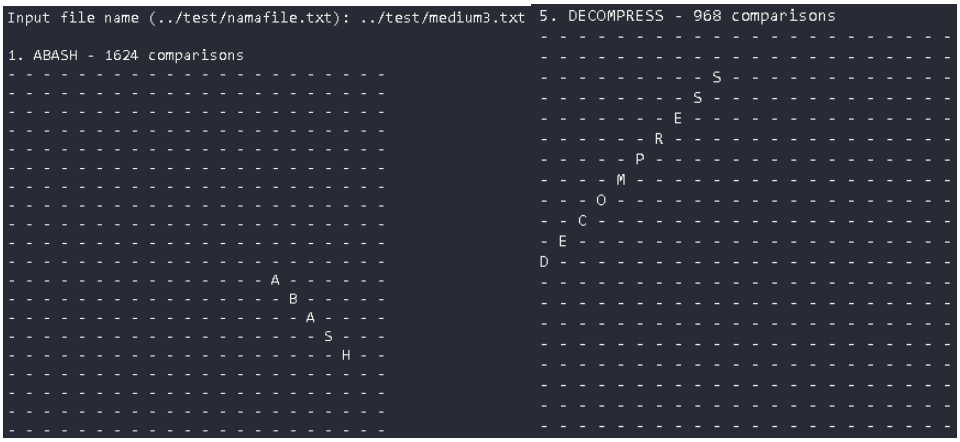
Puzzle

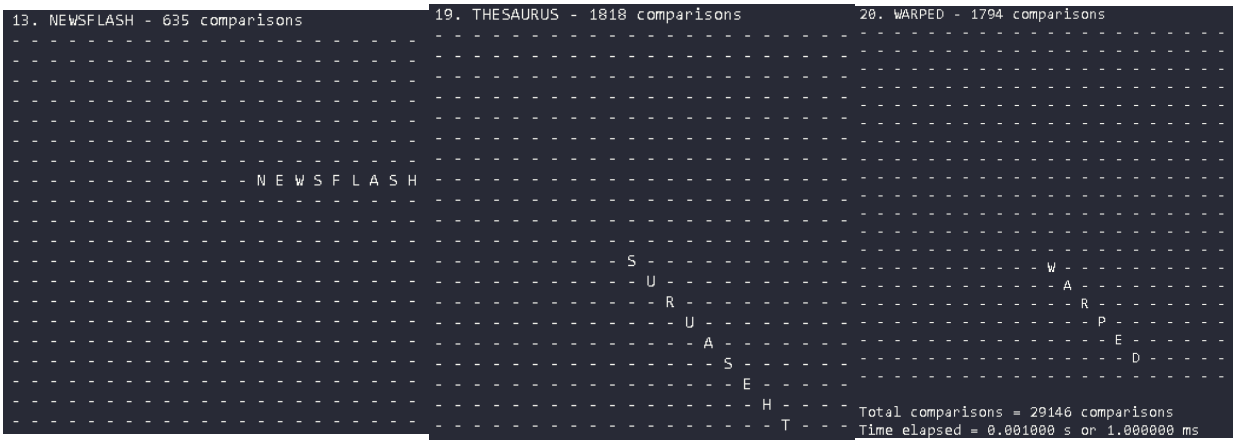
O	H	E	C	W	W	T	E	I	C	B	Q	V	O	L	K	H	M	N	D	I	T
A	S	S	E	R	T	I	N	G	I	V	Y	W	S	H	Y	D	B	U	N	P	Y
G	B	W	R	I	C	N	F	H	S	X	O	R	C	M	J	M	U	N	J	R	N
P	L	K	A	S	C	Q	S	S	E	L	E	C	R	U	O	S	W	M	T	M	J
U	O	A	J	Y	R	Q	E	E	J	G	N	I	S	I	P	S	E	D	D	T	Y
P	A	I	N	A	B	R	Q	D	W	W	P	S	W	A	D	S	Q	F	B	E	Q
I	I	J	C	C	P	R	G	U	U	N	P	U	X	D	Q	K	G	D	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

Word

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

Output:

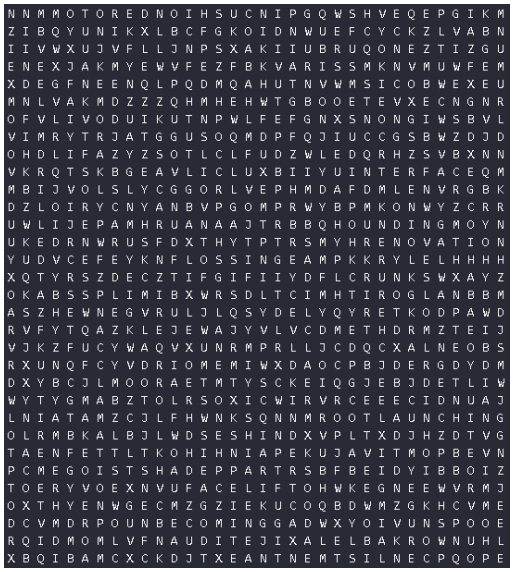




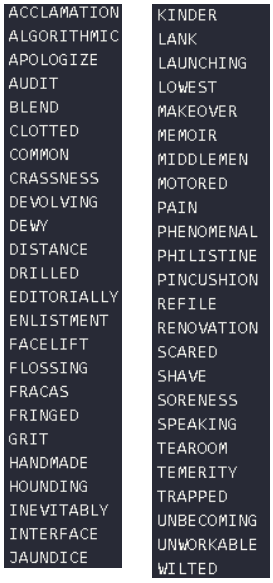
7. large1.txt

Input:

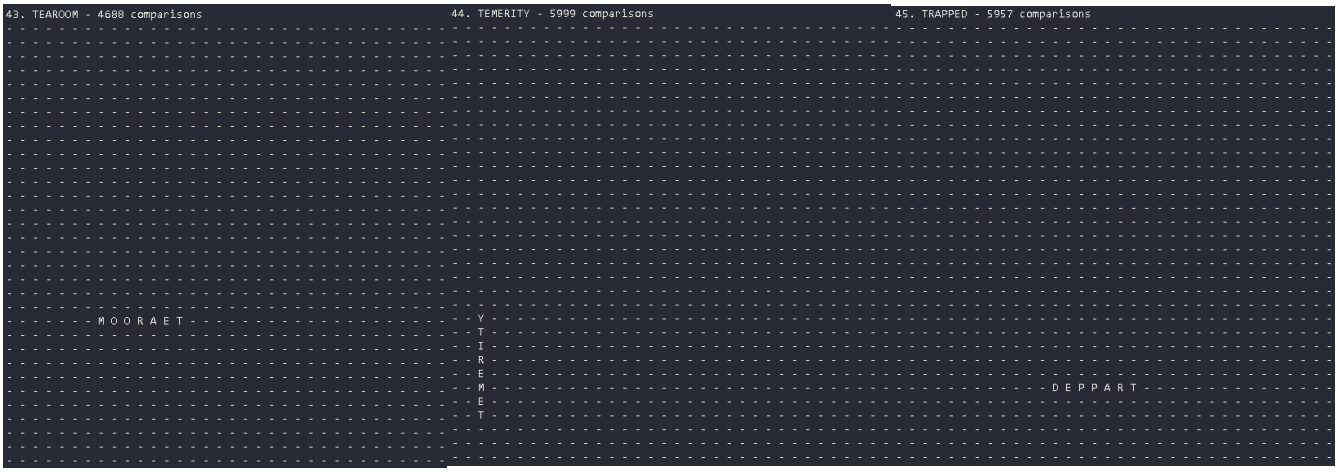
Puzzle

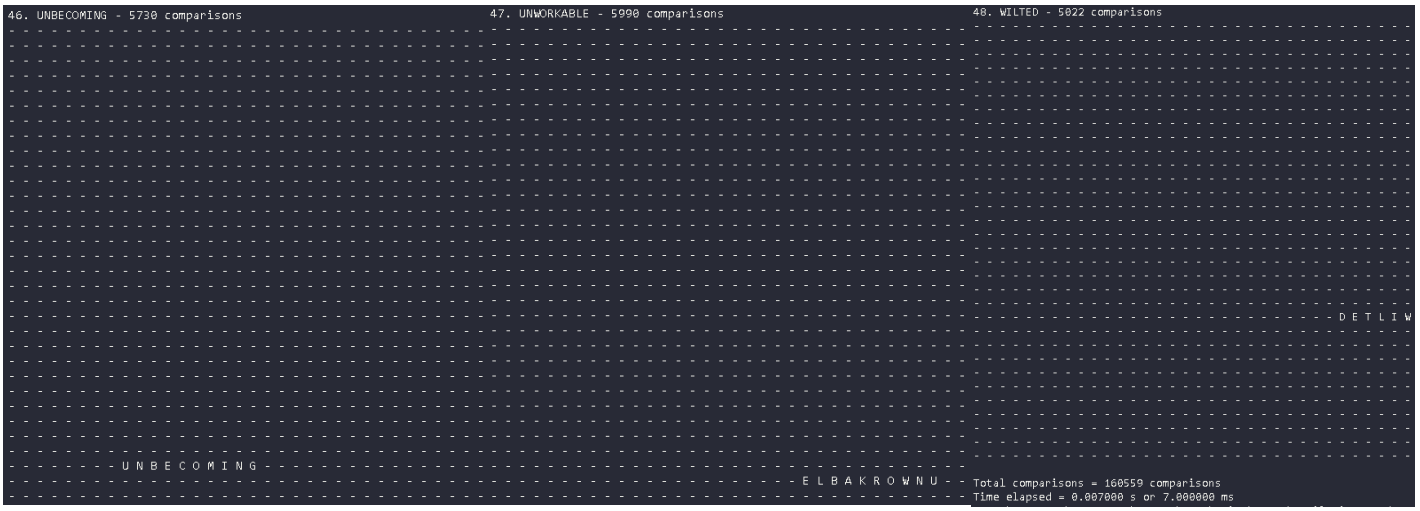


Word



Output:





8. large2.txt

Input:

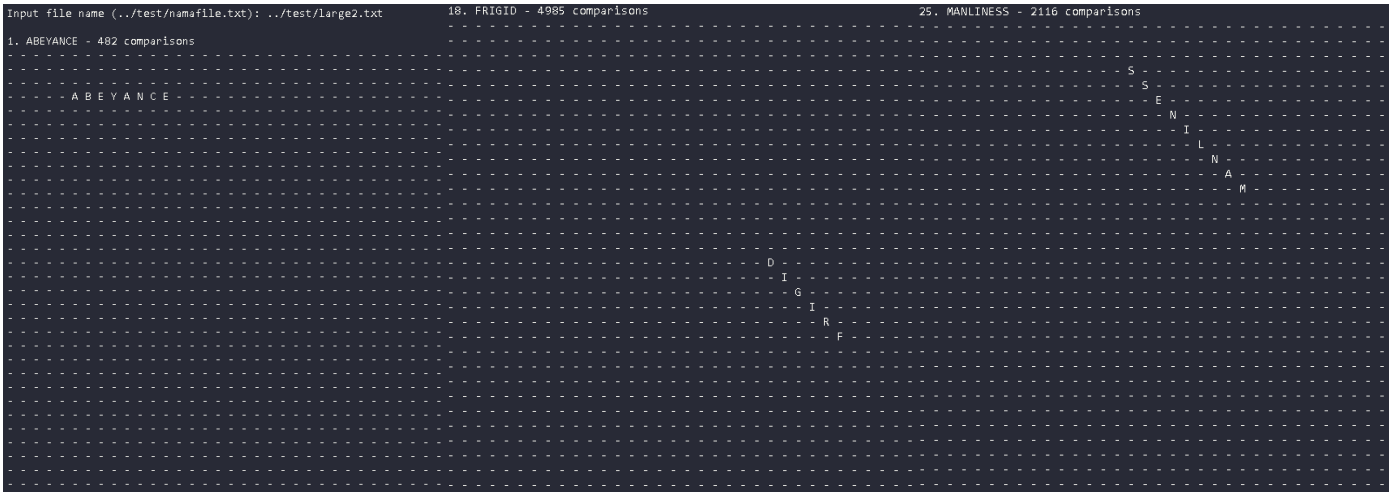
Puzzle

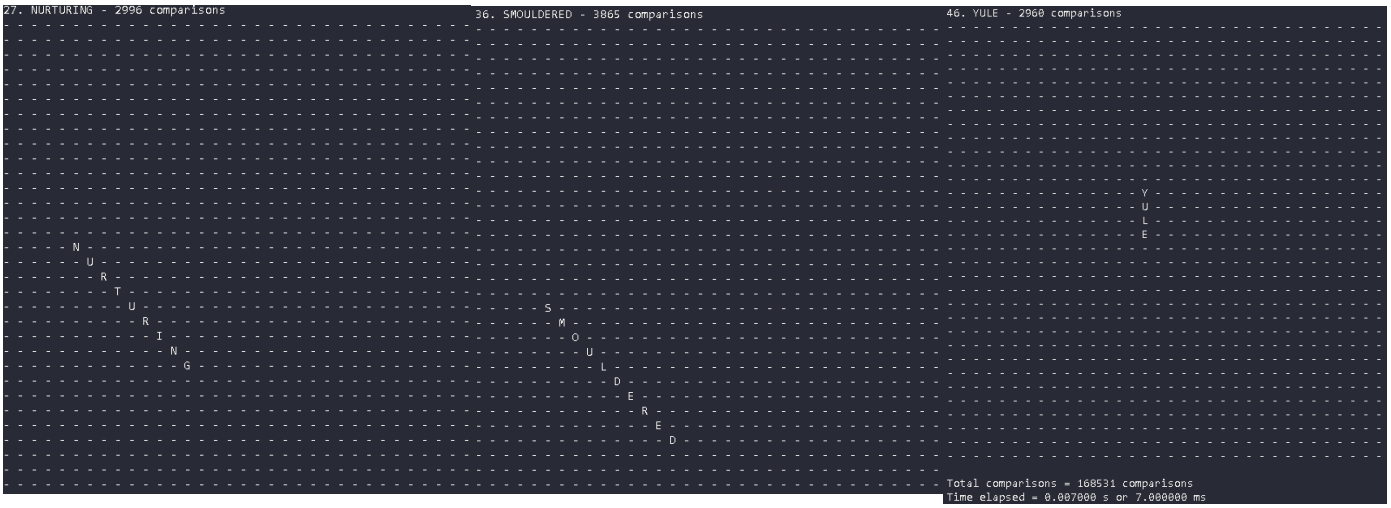
```
XBLOKNHNAXLQDWQIXDZBCYVRLTVTZKRPVI
KVSREENSTTXGDBSARROPSMTJEIRSSXRZHQ
HVCNXMZYKHRYHLNKIVEUUOAXQAHUVIYTPI
ZORHUABEYANCEDASOIHVOKEJFZCHNPEPVK
CWVXASMEBOVCZGKDSURHHLTFBBCTVEPRBC
BLZRRRNBTQBNTOEEOENLWLIZPRACTICEEB
EVHYLVTVOEPDSXDRSLNDLCNJTBKGXOPBBC
SKFFJGYPNNJWIDIROXRIYMPXLOIJFNLAAI
CDTMARSIAONXCCRUURSQLIEEOUWEJGPTJG
STRMFKWSJKOCIVECARVLENYANUKKPMEEPI
GGNOVSZZMGIJRYFGGTLIXNAHGSXTBPXMHL
CONDIMENTNTSIPYAIWINPTDMPKDKPAPOZE
EQRZISFFCIAVPUTYLSPECTIFYAMOKJEIPK
SKNBZBXAFRRIMHQUGZFASTGMLJRUXDRDIC
ZPWOKDHABAGUEHDLQIJRFPXDMGAGSETIOG
FPQHENOTTEEMBQFEYKIBRHIJECRROUVVKN
ECRDKRUIGHTCZWWMUITHTUZDODEMQTRVFI
DAODLAERCHNHQUICEOETALIMISSABYOBWN
YONNKNTHTJIBRD0WNSTAGEMAXGWJEBTHXE
YKPITSOMWURWFNLLSPISIGDJQSIMEDYXPT
AYRSSAMKNDRHXUSDDOBQTDZKTQLRXAMCDH
XDUFPEIOLNFIDLCEEICTHRNRZDPQFNBJRG
KZSEGGENUABXNBLTDXBSKYSCRAPERQEXEI
DIUMNAWDMLOLEGLERWPDYZMVHPCEMIRCHT
DOEAIRZGCEDZDCTRAMS ZXEFIREPRISEPD
ELCQZYDWURNESNLMFGQMCBEHZXAPUVOYEU
MAOWIVBKEUDTRBTIPHGERMANYQWJLUGJHG
ARNYTBAOBTQVNERNZADDRESSBNUOLGYASY
HBSFPRSUDSSJWGDIHKBGQZQNPBOOPNRLRM
SJUVAKIYFAEEQZLSBIDEDTOMARTINGALEV
AJLWBLNJSPFCOWLTSYTPBWWECRBZWZZXIJ
UKTNBMUTICKERNCMFZXRUKPMVEDNFEEDLF
```

Words

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

Output:





Repository **Github**

Berikut adalah link *repository* github yang akan diatur ke *public* setelah pengumpulan:
https://github.com/OjaanIr/TucilStima-WordSearchPuzzle

Status 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	✓	