LAPORAN TUGAS KECIL 1

WORD SEARCH PUZZLE

Diajukan sebagai salah satu tugas mata kuliah IF2211 Strategi Algoritma
Semester 4 Tahun Akademik 2021-2022

Oleh

Azka Syauqy Irsyad **13520107**

A. Algoritma *Brute Force*

Algoritma *Brute Force* adalah sebuah pendekatan yang sangat jelas (*straight forward*) untuk memecahkan suatu persoalan, biasanya didasarkan pada *problem statement* dan definisi konsep yang dilibatkan. Algoritma *brute force* memecahkan masalah dengan sangat sederhana, langsung, dan dengan cara yang jelas. Kelebihan algoritma ini adalah cukup mudah dipahami serta dapat diterapkan dihampir seluruh persoalan yang ada. Kekurangannya adalah pada masalah efisiensi, program ini membutuhkan banyak langkah sehingga membutuhkan waktu yang lama.

Algoritma *brute force* dimanfaatkan dalam program *word search puzzle* ini. Program *word search puzzle* ini menerima input file yang berisi matriks *puzzle* serta daftar *keyword* yang akan dicari di matriks tersebut. Pencarian *keyword* tersebut dilakukan dari kolom pertama baris pertama hingga kolom terakhir baris terakhir. Setelah ditemukan letak ditemukannya *keyword* tersebut pada matriks *puzzle*, akan ditampilkan matriks hasil yang menunjukan letak *keyword* tersebut.

Pertama, file untuk *puzzle* diinput ke dalam program. Program akan membaca matriks *puzzle* serta daftar *keyword* yang ada. Setelah semuanya siap, akan dimulai pencarian *keyword* dari yang paling pertama. *Keyword* tersebut akan diubah ke dalam bentuk *array of character*, dimana *array* pertama sama seperti *keyword* asli urutannya dan *array* kedua yang urutannya dibalik. Proses pencarian dimulai dari kiri atas, yaitu baris pertama (indeks nol) dan kolom pertama (indeks nol). Selama pencariannya, dicek karakter elemen yang sedang dikunjungi terhadap elemen pertama array pertama dan elemen pertama array kedua, jika ada yang sama atau keduanya sama maka akan dicek karakter-karakter selanjutnya dalam arah horizontal, vertikal, serong kanan bawah, dan serong kanan atas, horizontal yang urutannya dibalik, vertikal yang urutannya dibalik, serong kanan bawah yang urutannya dibalik, serta serong kanan atas yang urutannya dibalik. Jika tidak ada yang sama maka akan di cek ke baris atau kolom selanjutnya dalam matriks.

Setelah ditemukan, karakter-karakter yang terdapat dalam *keyword* akan menggantikan template dari matriks hasil yang awalnya berupa ‘-’. Matriks hasil tersebut akan ditampilkan kepada pengguna. Setelah semua keyword berhasil diproses, program akan menampilkan jumlah perbandingan kata yang terjadi serta waktu eksekusi proses menyelesaikan proses tersebut.

B. Source Code Program

Berikut merupakan *screenshot* kode program yang dibuat dalam file main.c

```c
#include "boolean.h"
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>

#define ROW_SIZE 1000
#define COL_SIZE 1000

typedef char ElType;
typedef struct {
    ElType contents[ROW_SIZE][COL_SIZE];
    int rowEff;
    int colEff;
} Matrix;
typedef struct {
    ElType contents[50][50];
    int neff;
} List;

#define ROW(M) (M).rowEff
#define COL(M) (M).colEff
#define ELMT(M, i, j) (M).contents[(i)][(j)]

#define NEFF(L) (L).neff
#define ELMTL(L, i) (L).contents[(i)]

FILE* openFile;
Matrix template;
Matrix result;
List keyword;
```

```
int count;
int length;
int kind;
char arr[50];
char reverse[50];
boolean found = false;

void readFile() {
    printf("Masukkan nama file yang diinginkan: ");
    char fileName[50];
    gets(fileName);

    openFile = fopen(fileName, "r");

    char tokenMap = fgetc(openFile);
    char beforeToken = tokenMap;
    ROW(template) = 0; COL(template) = 0;
    int baris = 0; int kolom = 0;
    while (tokenMap != '\n' || beforeToken != '\n') {
        putchar(tokenMap);
        if (tokenMap == '\n') {
            ROW(template)++;
            baris++;
            COL(template) = kolom;
            kolom = 0;
        }
        if (tokenMap != ' ' && tokenMap != '\n') {
            ELMT(template, baris, kolom) = tokenMap;
            kolom++;
        }
        beforeToken = tokenMap;
```

```
        tokenMap = fgetc(openFile);
    }

    printf("\nDaftar keyword yang perlu dicari:\n");
    NEFF(keyword) = 0;
    char line[100];
    while (fgets(line, sizeof(line), openFile)) {
        printf("%s", line);
        strcpy(ELMTL(keyword, NEFF(keyword)), line);
        NEFF(keyword)++;
    }

    for (int i = 0; i < NEFF(keyword)-1; i++) {
        ELMTL(keyword, i)[strlen(ELMTL(keyword, i))-1] = '\0';
    }

    fclose(openFile);
    count = 0;
}

void resultMatrix() {
    ROW(result) = ROW(template);
    COL(result) = COL(template);

    for (int i = 0; i < ROW(result); i++) {
        for (int j = 0; j < COL(result); j++) {
            ELMT(result, i, j) = '-';
        }
    }
}
```

```c
void printResMatrix() {
    for (int i = 0; i < ROW(result); i++) {
        for (int j = 0; j < COL(result); j++) {
            if (j+1 == COL(result)) {
                printf("%c\n", ELMT(result, i, j));
            } else {
                printf("%c ", ELMT(result, i, j));
            }
        }
    }
}

void charArray(int x) {
    length = strlen(ELMTL(keyword, x));
    for (int i = 0; i <= length; i++) {
        arr[i] = ELMTL(keyword, x)[i];
        reverse[i] = ELMTL(keyword, x)[i];
    }
}

void reverseCharArray() {
    char temp;
    int start = 0;
    int end = length-1;
    while (start <= end) {
        temp = reverse[start];
        reverse[start] = reverse[end];
        reverse[end] = temp;
        start++;
        end--;
    }
}
```

```
void searchHorizontal(int row, int col) {
    if (col+length > COL(template)) {

    } else {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row, col+iterator) != arr[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepCol = col+iterator;
        found = same;
        if (found) {
            resultMatrix();
            for (int writeCol = col; writeCol < keepCol; writeCol++) {
                ELMT(result, row, writeCol) = ELMT(template, row, writeCol);
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```
void searchHorizontalReverse(int row, int col) {
    if (col+length > COL(template)) {

    } else {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row, col+iterator) != reverse[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepCol = col+iterator;
        found = same;
        if (found) {
            resultMatrix();
            for (int writeCol = col; writeCol < keepCol; writeCol++) {
                ELMT(result, row, writeCol) = ELMT(template, row, writeCol);
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```
void searchVertical(int row, int col) {
    if (row+length > ROW(template)) {

    } else {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row+iterator, col) != arr[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepRow = row+iterator;
        found = same;
        if (found) {
            resultMatrix();
            for (int writeRow = row; writeRow < keepRow; writeRow++) {
                ELMT(result, writeRow, col) = ELMT(template, writeRow, col);
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```
void searchVerticalReverse(int row, int col) {
    if (row+length > ROW(template)) {

    } else {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row+iterator, col) != reverse[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepRow = row+iterator;
        found = same;
        if (found) {
            resultMatrix();
            for (int writeRow = row; writeRow < keepRow; writeRow++) {
                ELMT(result, writeRow, col) = ELMT(template, writeRow, col);
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```
void searchLTtoRB(int row, int col) {
    if (row+length <= ROW(template) && col+length <= COL(template)) {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row+iterator, col+iterator) != arr[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepCol = col+iterator;
        int keepRow = row+iterator;
        found = same;
        if (found) {
            resultMatrix();
            int writeRow = row;
            int writeCol = col;
            while (writeRow < keepRow) {
                ELMT(result, writeRow, writeCol) = ELMT(template, writeRow, writeCol);
                writeRow++;
                writeCol++;
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```
void searchLTtoRBreverse(int row, int col) {
    if (row+length < ROW(template) && col+length < COL(template)) {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row+iterator, col+iterator) != reverse[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepCol = col+iterator;
        int keepRow = row+iterator;
        found = same;
        if (found) {
            resultMatrix();
            int writeRow = row;
            int writeCol = col;
            while (writeRow < keepRow) {
                ELMT(result, writeRow, writeCol) = ELMT(template, writeRow, writeCol);
                writeRow++;
                writeCol++;
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```
void searchLBtoRT(int row, int col) {
    if (row-length >= -1 && col+length <= COL(template)) {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row-iterator, col+iterator) != arr[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepCol = col+iterator;
        int keepRow = row-iterator;
        found = same;
        if (found) {
            resultMatrix();
            int writeRow = row;
            int writeCol = col;
            while (writeCol < keepCol) {
                ELMT(result, writeRow, writeCol) = ELMT(template, writeRow, writeCol);
                writeRow--;
                writeCol++;
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```
void searchLBtoRTreverse(int row, int col) {
    if (row-length >= -1 && col+length <= COL(template)) {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row-iterator, col+iterator) != reverse[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepCol = col+iterator;
        int keepRow = row-iterator;
        found = same;
        if (found) {
            resultMatrix();
            int writeRow = row;
            int writeCol = col;
            while (writeCol < keepCol) {
                ELMT(result, writeRow, writeCol) = ELMT(template, writeRow, writeCol);
                writeRow--;
                writeCol++;
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

```c
int main() {
    readFile();
    printf("\n");
    clock_t onset = clock();
    for (int allkeys = 0; allkeys < NEFF(keyword); allkeys++) {
        kind = 1;
        charArray(allkeys);
        reverseCharArray();
        for (int i = 0; i < ROW(template); i++) {
            for (int j = 0; j < COL(template); j++) {
                if (ELMT(template, i, j) != arr[0] && ELMT(template, i, j) != reverse[0]) {
                    count++;
                }
                if (ELMT(template, i, j) == arr[0] || ELMT(template, i, j) == reverse[0]) {
                    searchHorizontal(i, j);
                    if (!found) {
                        searchVertical(i, j);
                    }
                    if (!found) {
                        searchLTtoRB(i, j);
                    }
                    if (!found) {
                        searchLBtoRT(i, j);
                    }
                    if (!found) {
                        searchHorizontalReverse(i, j);
                    }
                    if (!found) {
                        searchVerticalReverse(i, j);
                    }
                    if (!found) {
                        searchLTtoRBreverse(i, j);
                    }
                    if (!found) {
                        searchLBtoRTreverse(i, j);
                    }
                }
            }
        }
    }
    clock_t onfinish = clock();

    printf("\nLama eksekusi program adalah %f ms", (double) onfinish-onset);
    printf("\nJumlah total perbandingan karakter: %d", count);
}
```

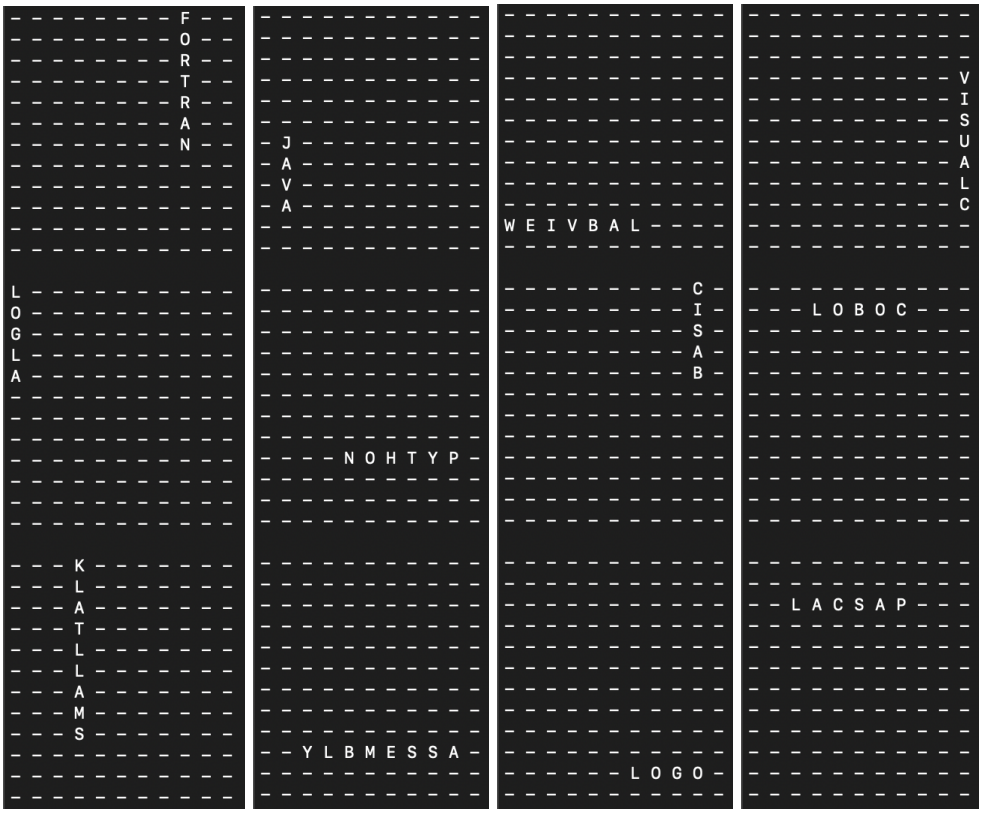
```
void searchLBtoRT(int row, int col) {
    if (row-length >= -1 && col+length <= COL(template)) {
        int iterator = 1;
        int counter = 1;
        boolean same = true;
        while (iterator < length && counter <= length && same) {
            if (ELMT(template, row-iterator, col+iterator) != arr[iterator]) {
                count++;
                same = false;
            } else {
                count++;
                counter++;
                iterator++;
            }
        }
        int keepCol = col+iterator;
        int keepRow = row-iterator;
        found = same;
        if (found) {
            resultMatrix();
            int writeRow = row;
            int writeCol = col;
            while (writeCol < keepCol) {
                ELMT(result, writeRow, writeCol) = ELMT(template, writeRow, writeCol);
                writeRow--;
                writeCol++;
            }
            printf("\n");
            printResMatrix();
        }
    }
}
```

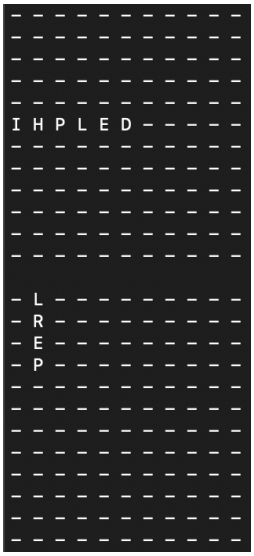
C. Screenshot Input dan *Output*

Berikut hasil percobaan program main.c

1. small1.txt
   Total perbandingan kata: 2633
   Lama waktu eksekusi: 1189 ms

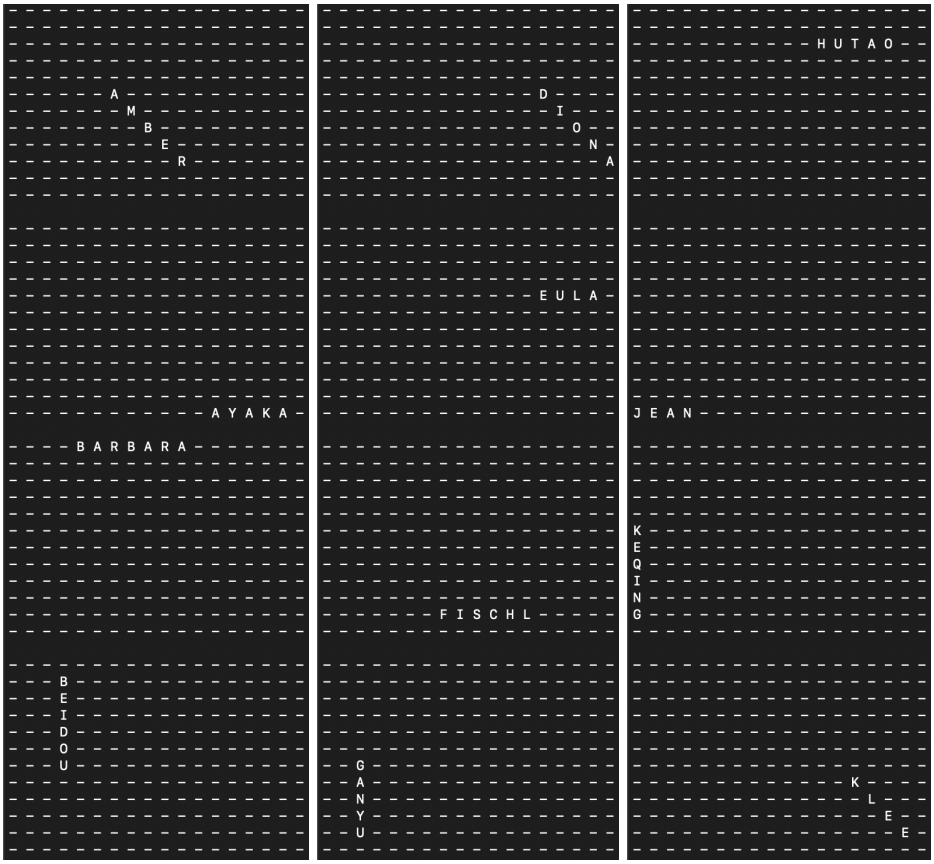
```
Masukkan nama file yang diinginkan: ../test/small1.txt
L L J K C A B L F C I
O R O L O B O C O I M
G E L A C S A P R S X
L P S T A H W V T A V
A N R L X L X Q R B I
I H P L E D O X A H S
K J Y A P H P Y N O U
F A B M A D A N Z J A
E V I S N O H T Y P L
A A Y L B M E S S A C
W E I V B A L O G O X
X Y Z D R V W U I O P

Daftar keyword yang perlu dicari:
FORTRAN
ALGOL
SMALLTALK
JAVA
PYTHON
ASSEMBLY
LABVIEW
BASIC
LOGO
VISUALC
COBOL
PASCAL
DELPHI
PERL
```

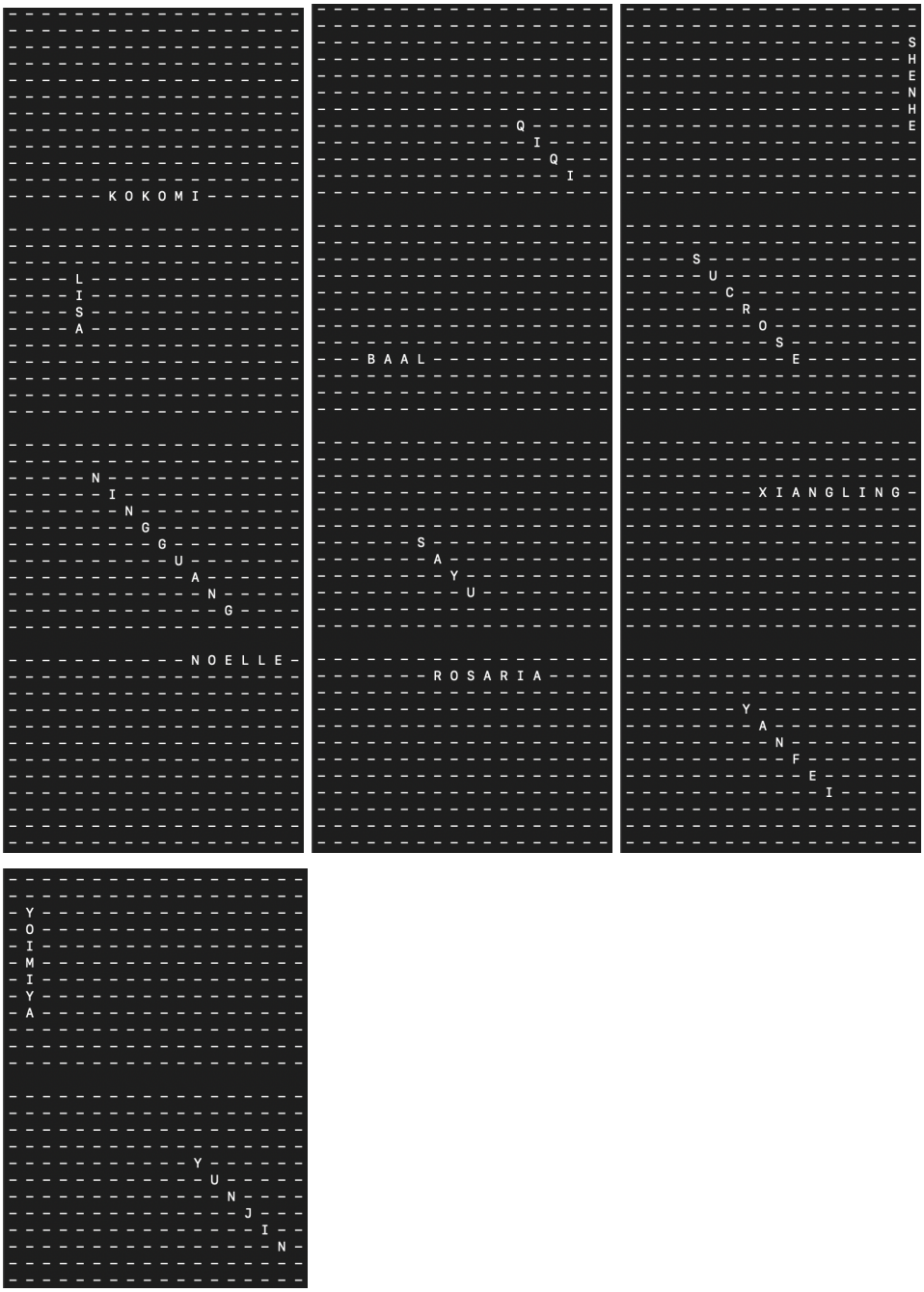




2. small2.txt
   Total perbandingan kata: 8032
   Lama waktu eksekusi: 3177 ms

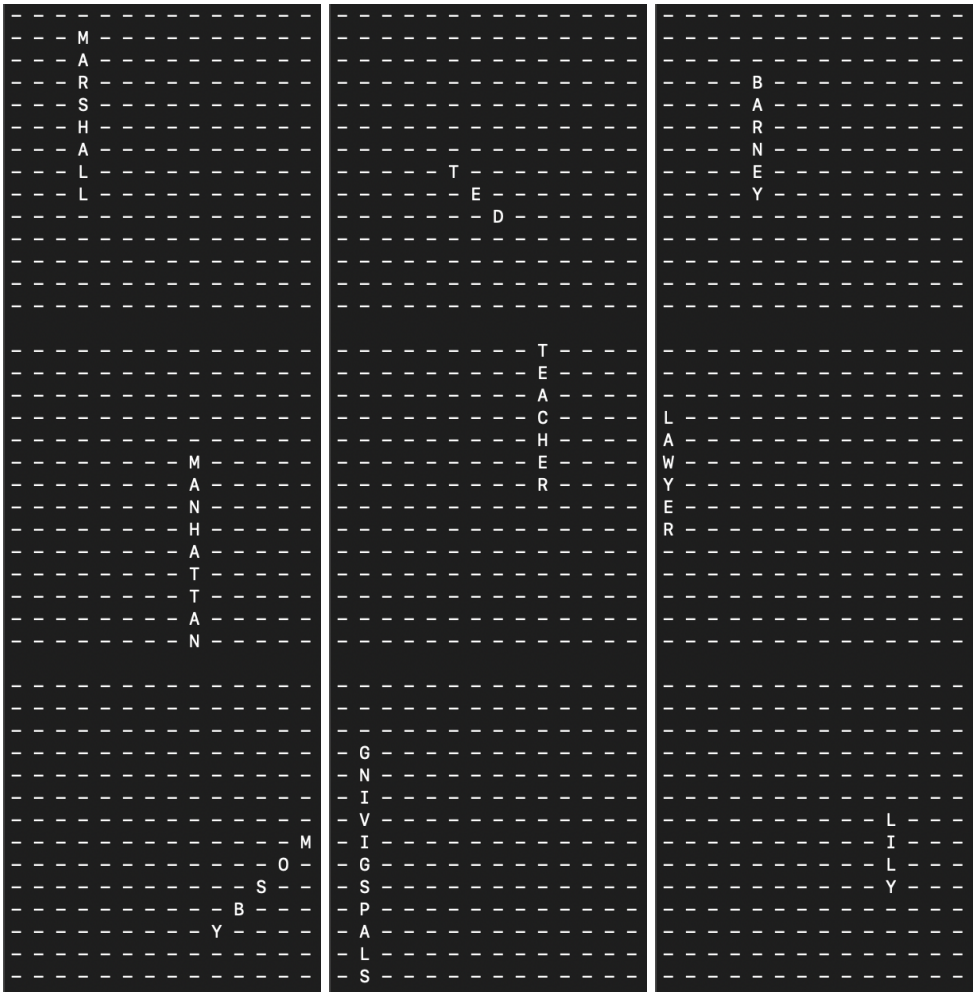
```
Masukkan nama file yang diinginkan: ../test/small2.txt
Y T I K B A R B A R A N O E L L E Q
Y I J B S S M R O S A R I A D Q M F
B Y V E S N K R O N Y H U T A O A S
I O Q I L U I Y X I A N G L I N G H
P I K D I N C N A B U Y R E U L A E
K M S O S B A R G N Q T U D U Z K N
E I G U A U S M O G F C H N I D M H
Q Y A W V Z H A B S U E Q K J O R E
I A N B A A L C Y E E A I I L I N S
N P Y N L J Y K Q U R O N K Q E N A
G I U S J O Z F I S C H L G C I E F
J E A N S K K O K O M I A Y A K A Z

Daftar keyword yang perlu dicari:
AMBER
AYAKA
BARBARA
BEIDOU
DIONA
EULA
FISCHL
GANYU
HUTAO
JEAN
KEQING
KLEE
KOKOMI
LISA
NINGGUANG
NOELLE
QIQI
BAAL
SAYU
ROSARIA
SHENHE
SUCROSE
XIANGLING
YANFEI
YOIMIYA
YUNJIN
```

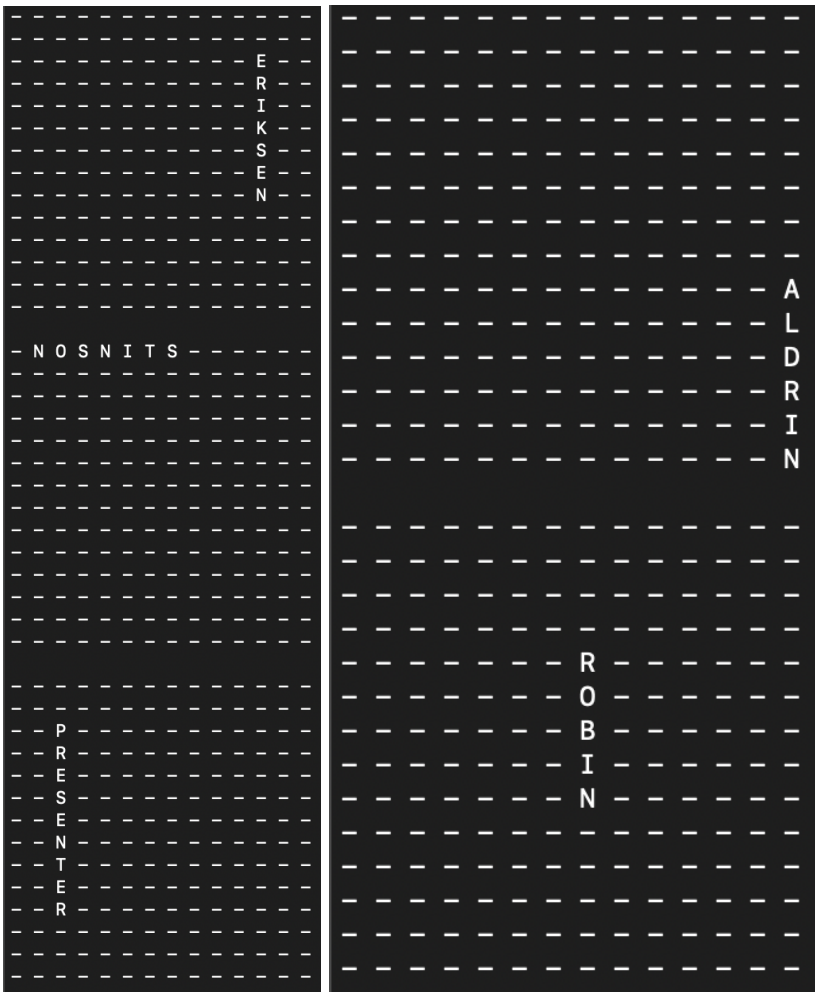

AMBER
DILUC
HUTAO
EULA
AYAKA
JEAN
BARBARA
KEQING
FISCHL
BEIDOU
GANYU
KLEE



3. small3.txt
   Total perbandingan kata: 3988
   Lama waktu eksekusi: 1710 ms

```
Masukkan nama file yang diinginkan: ../test/small3.txt
L N O S N I T S E T L A L A
M T S M N A E S T E B E R R
S L P A R T K N L A M E L I
L G R R B A I I I C N R R A
A N E S A S A R S H T I E E
W I S H R S E O M E H K Y L
Y V E A N B T B A R L S A B
E I N L E T D I N B I E E M
R G T L Y R E N H A L N O A
W S E Y E A O D A N Y S G L
A P R Y E R P R T A B O W D
P A N R B W R I T Y M T L R
B L N A A P M P A A B R I I
G S E L D A N E N E Y R E N

Daftar keyword yang perlu dicari:
MARSHALL
MANHATTAN
MOSBY
TED
TEACHER
SLAPSGIVING
BARNEY
LAWYER
LILY
ERIKSEN
STINSON
PRESENTER
ALDRIN
ROBIN
```

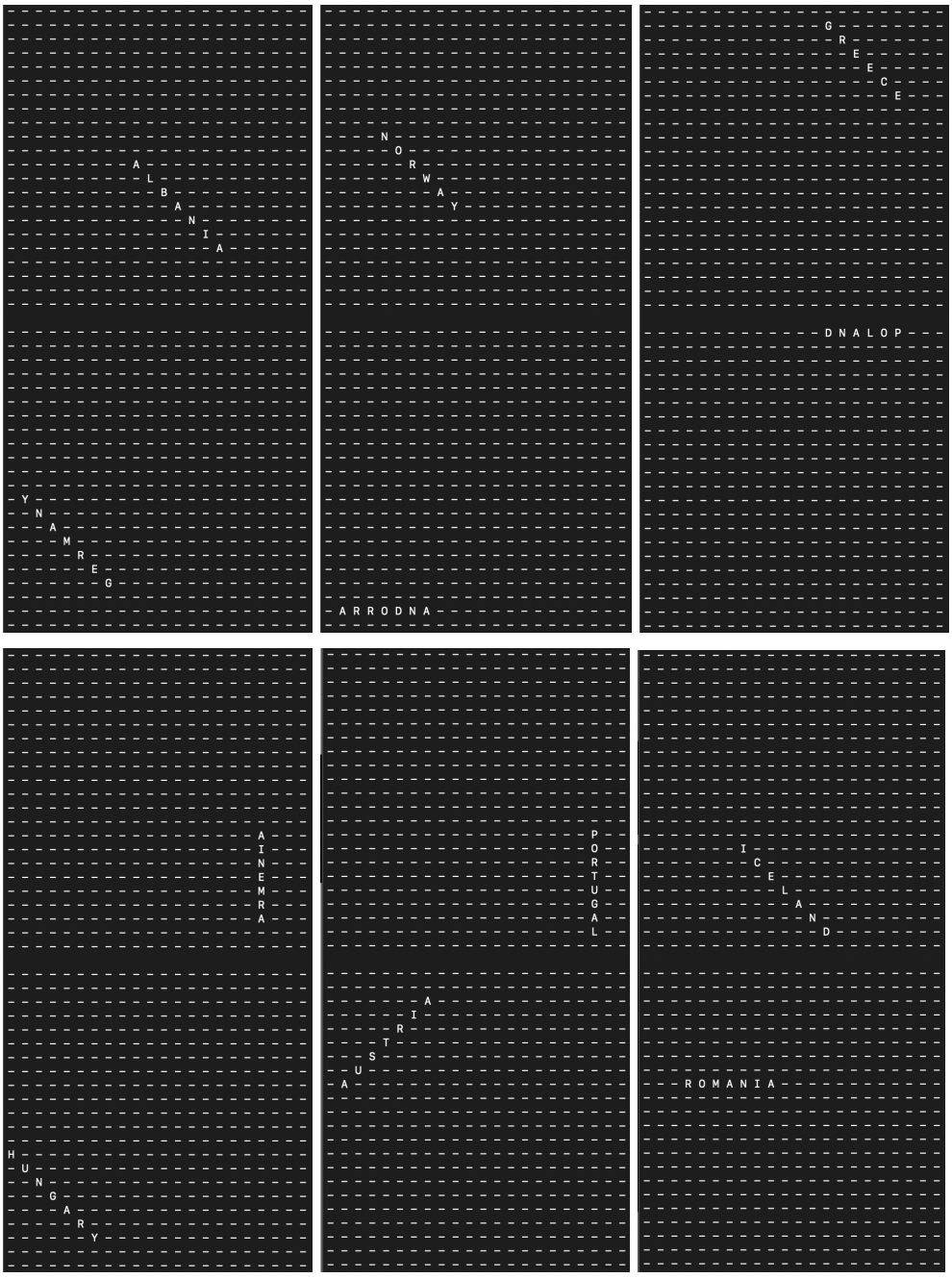

MARSHALL
MANHATTAN
BOSS
BY
TED
TEACHER
GNIVIGSPALS
BARNEY
LAWYER
LILY

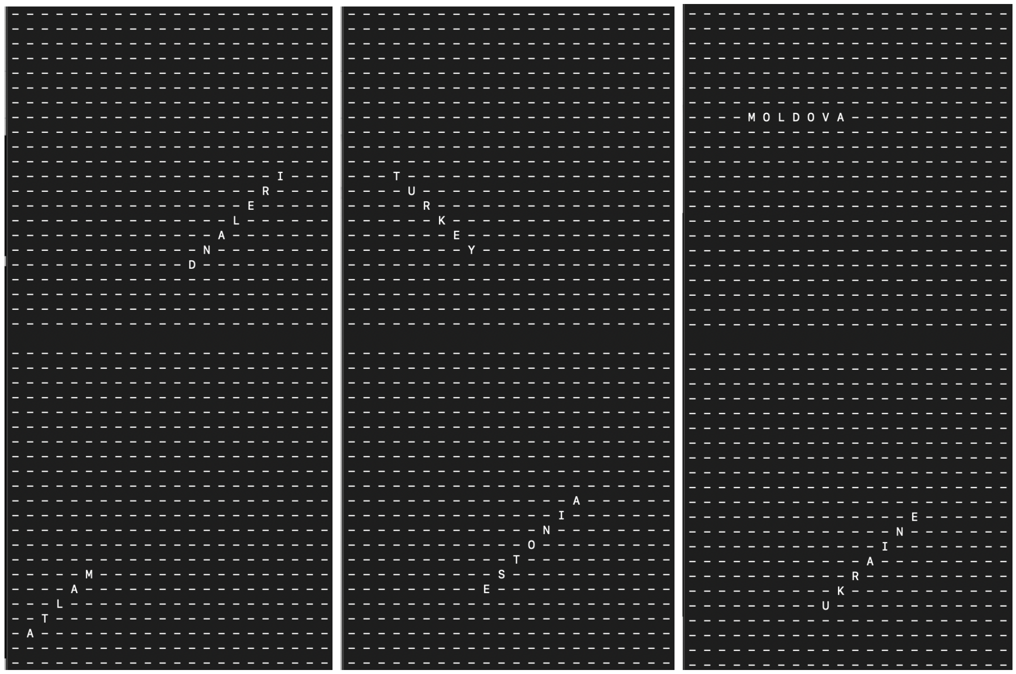


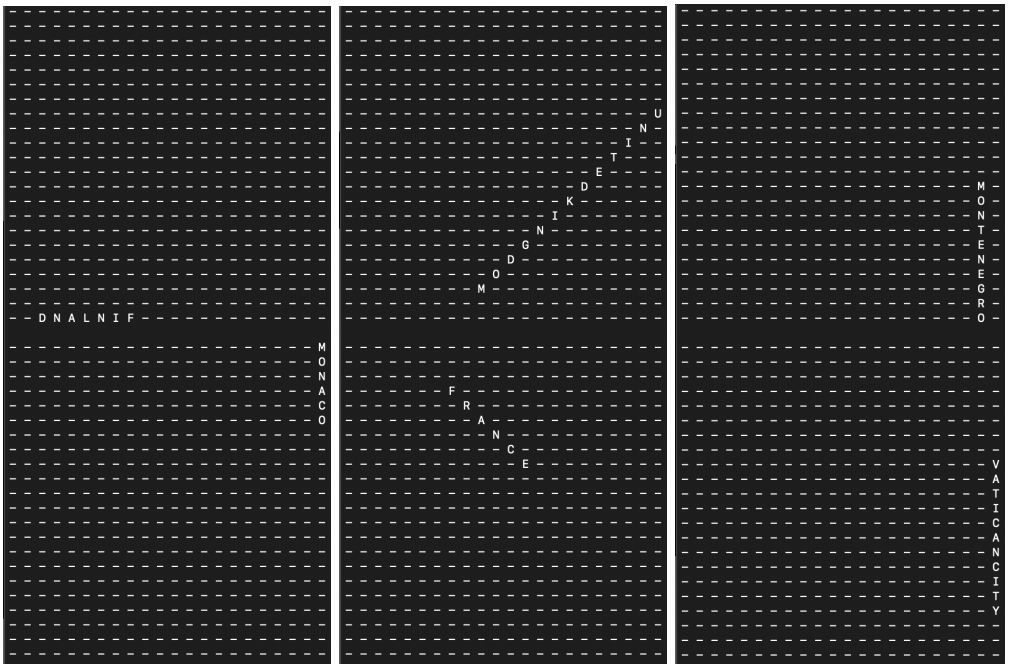
4. medium1.txt
   Total perbandingan kata: 18409

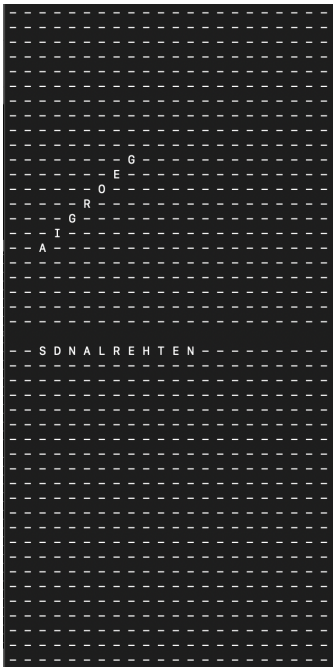
Lama waktu eksekusi: 5275 ms

```
Masukkan nama file yang diinginkan: ../test/medium1.txt
T C S D N A L R E H T E N D N A L O P Y S M
R R I A I N A U H T I L I G I D S K B D L O
Q S B Y V Q U A L D T B K T R E M N K N O N
M W E L B Y I F E B I R A R R E A H S A V A
A E L P A R A I R A G L U B A T E U O L A C
C D A O T B F K J A Y V I S S M R C P R K O
E E R S O N I R A M N A S H S P N U E E I R
D N U U M O L D O V A C K P Y I I E N Z A U
O A S R O M A N I A S A E C A X A E D T N V
N B O S N I A H E R Z E G O V I N A M I V A
I I W R D O R Q G A N O F K D A N U T W T T
A T E T W E R E K A C O E O I E I E I S O I
N Y T T U I O W P C L Z U N N G D R J V M C
H A N T S R X I A U H B O I L K E V A P O A
A U J A G N K I X Y L T A E I L C S I O N N
I B N I M M E E C F S R B N A A R L N R T C
H G A G A R M T Y E K H G N I T O O E T E I
C D U L A B E E H U L D D V E A A V M U N T
E M T Q O R R G X C O A T W F S T E R G E Y
Z A K U E S Y E A M E A N J Y G I N A A G A
C A R R O D N A Z H L I R D N U A I E L R W
A G D N A L N I F A U Q L Y S C P A K Z O N

Daftar keyword yang perlu dicari:
ALBANIA
GERMANY
NORWAY
ANDORRA
GREECE
POLAND
ARMENIA
HUNGARY
PORTUGAL
AUSTRIA
ICELAND
ROMANIA
IRELAND
MALTA
TURKEY
ESTONIA
MOLDOVA
UKRAINE
FINLAND
MONACO
UNITEDKINGDOM
FRANCE
MONTENEGRO
VATICANCITY
GEORGIA
NETHERLANDS
```


ALBANIA
NORWAY
GREECE
DNALOP
YNAMERG
ARRODNA
AINEMRA
PORTUGAL
HUNGARY
AIRTSUA
ICELAND
ROMANIA

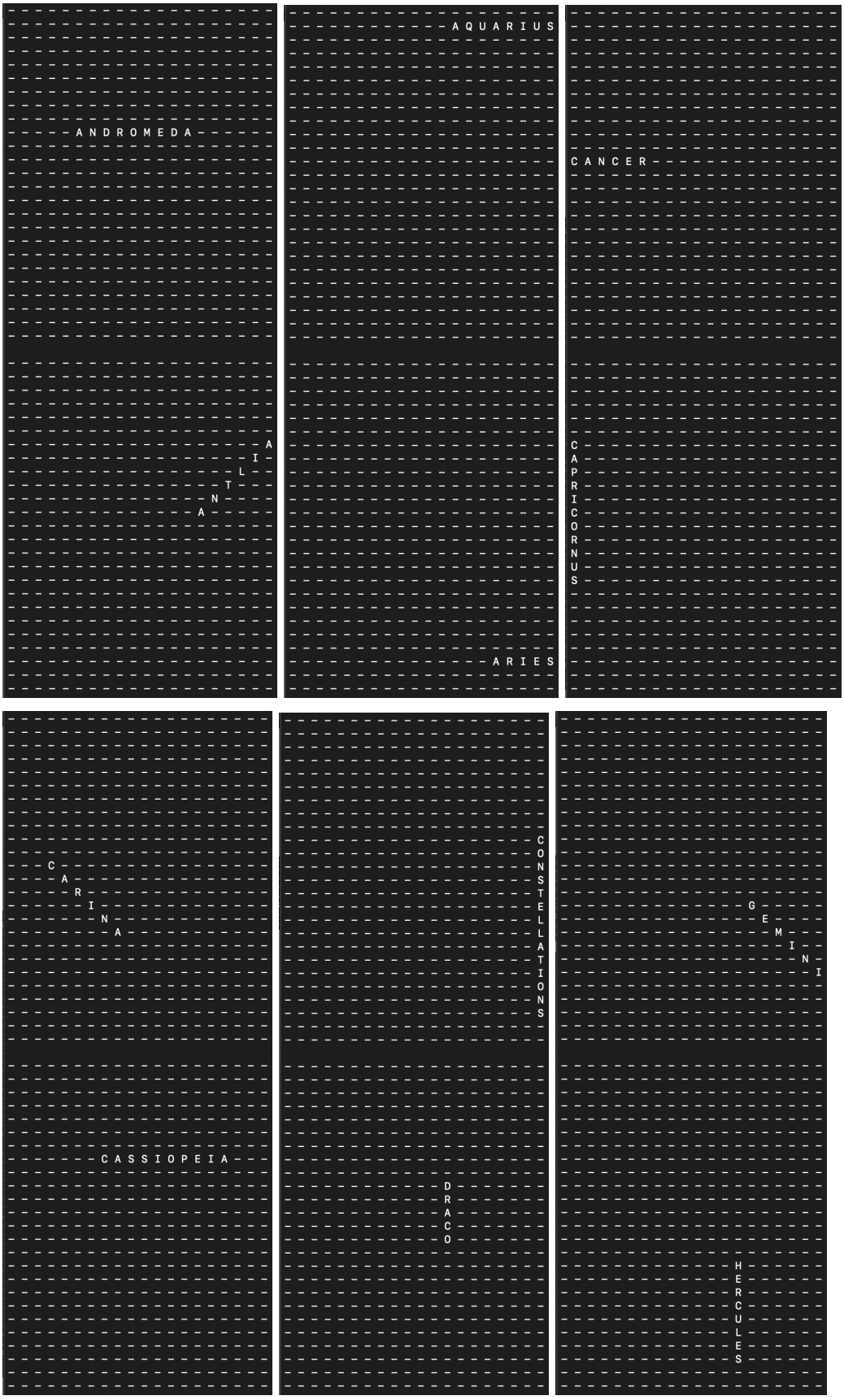

IRELAND
MALTA
TURKEY
ESTONIA
MOLDOVA
UKRAINE

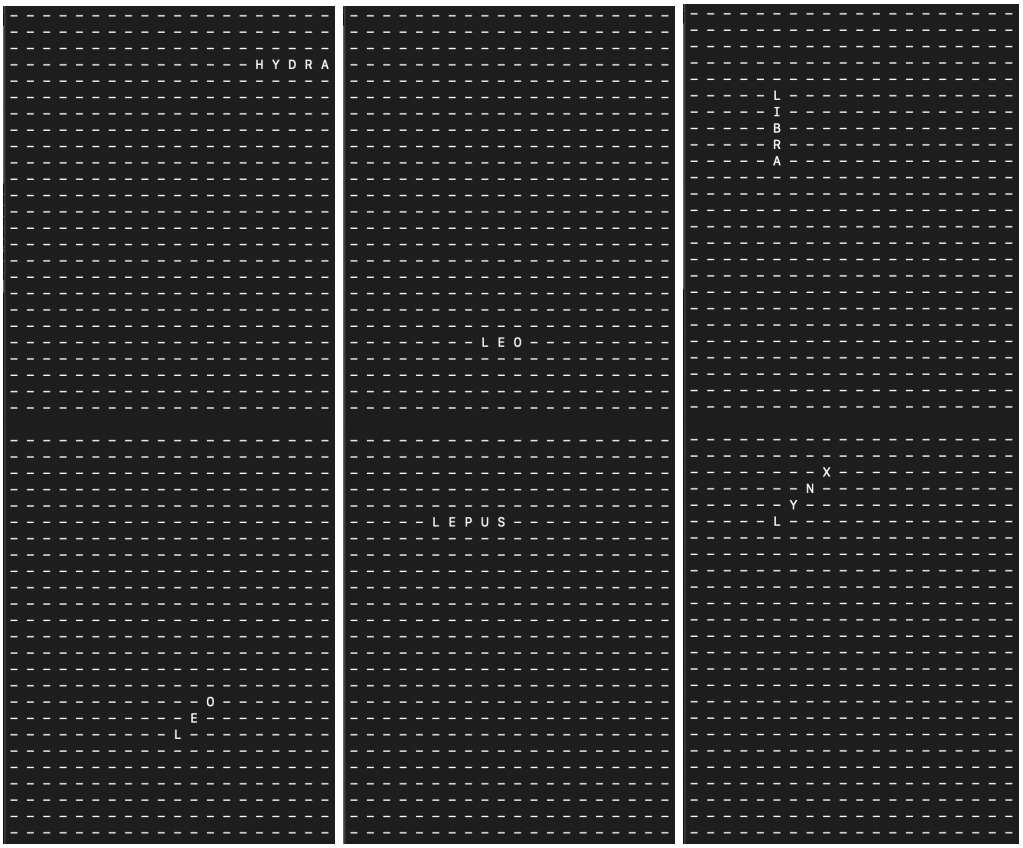


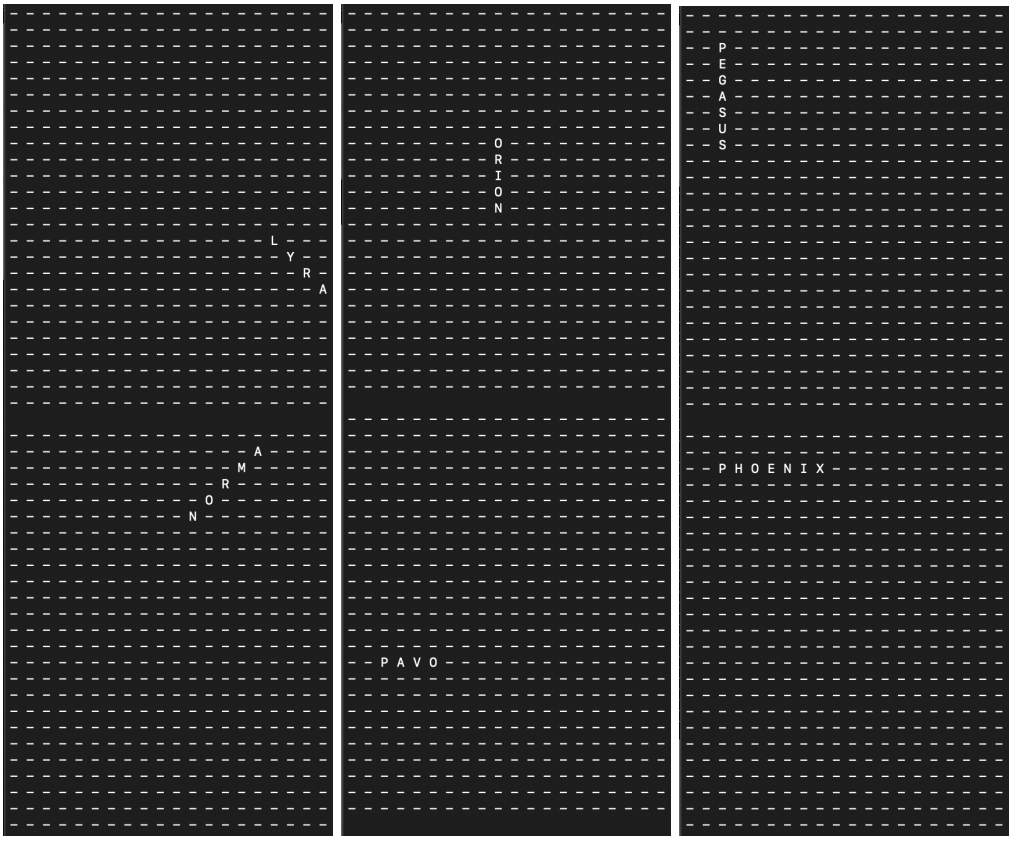


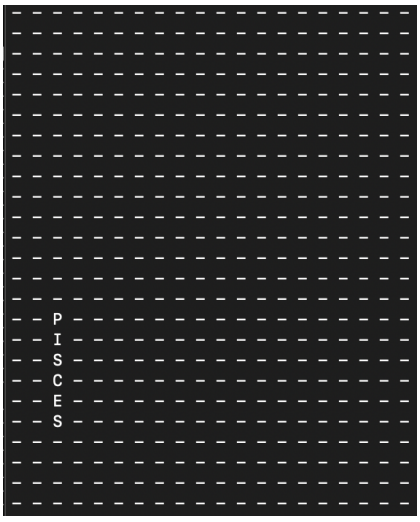
5. medium2.txt
   Total perbandingan kata: 23761
   Lama waktu eksekusi: 6273 ms

```
Masukkan nama file yang diinginkan: ../test/medium2.txt
A S O B S C U L P T O R I E S N O X A E
G K L I E S N O B V M J A Q U A R I U S
C F P H O E N I X O R I O M M I H A C F
O A E S V K L N J I O R S R S H Y D R A
X S G E M I Y A T Y E R O A O D K X O S
L T A B N L E P U S U N H S E R P E N S
C Y S S A I I T S C K L I E M J I O R A
A B U I N B D C A S S I O P E I A S I E
P U S G P R H F M J I O R S T A F L F O
R P A V I A N D R O M E D A T J T P T C
I B V M J I O R S R N U R B A N R A C O
C A N C E R U T R I A T A P A O T H E N
O A U S A G L E N O G U C T H S T A R S
R K L I E R V S U N S R O A F G E I N T
N J R T U E I J J I Z R S P G C L A P E
U A P A V O L N K L I E C H G E N Y M L
S O I T I B S U A T C A O E S A M F R L
T G S U R S E S P A F E R R D U R I A A
E U C H G X R E G U L I T C F R O L N T
T R E A O J P I E S N O I U K L I E S I
A J S D C M E C L E O F T L K E P U S O
U E D H X L N M N F O P D E X U R F T N
R F H I P U S T A P S G Z S T A R I E S
U W O P R I O W H J E S I E S N O H G I
S A G I T T A R I U S P S C O R P I U S

Daftar keyword yang perlu dicari:
ANDROMEDA
ANTLIA
AQUARIUS
ARIES
CANCER
CAPRICORNUS
CARINA
CASSIOPEIA
CONSTELLATIONS
DRACO
GEMINI
HERCULES
HYDRA
LEO
LEPUS
LIBRA
LYNX
LYRA
NORMA
ORION
PAVO
PEGASUS
PHOENIX
PISCES
```


ANDROMEDA
AQUARIUS
CANCER
ARIES
CAPRICORNUS
ALTAIR
CARINA
CASSIOPEIA
CONSTELLATIONS
DRACO
GEMINI
HERCULES

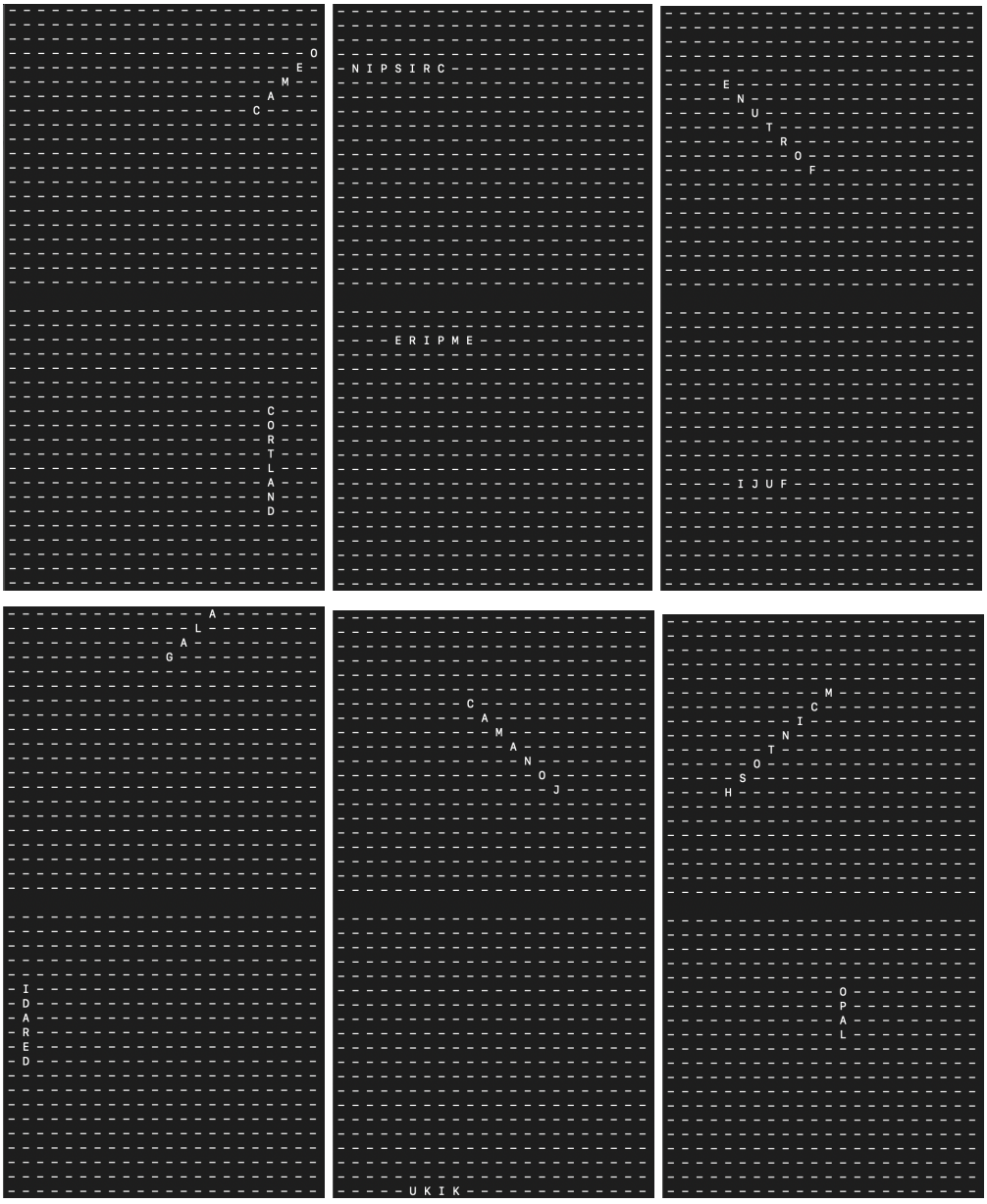

HYDRA
LEO
LEPUS
LIBRA
LYNX
LEO

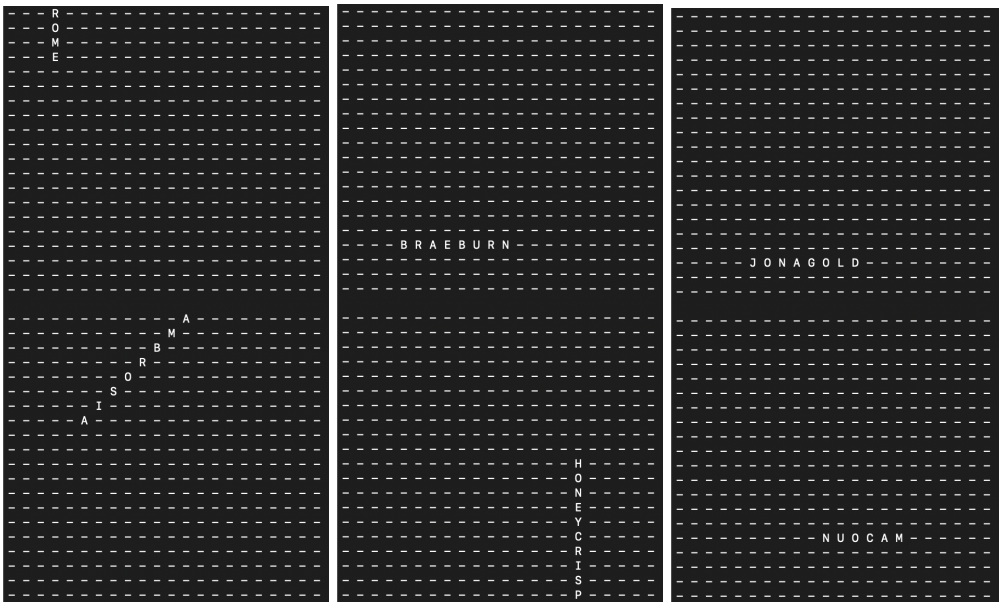




6. medium3.txt
   Total perbandingan kata: 11221
   Lama waktu eksekusi: 3461 ms

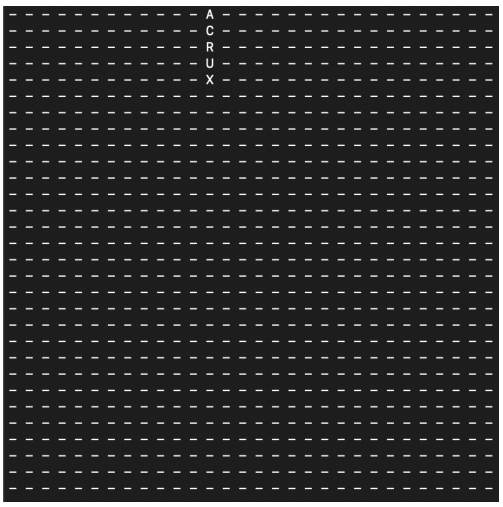
```
Masukkan nama file yang diinginkan: ../test/medium3.txt
A S R R F S B G C C O Y A A A E W H Q A H D
Q K D O A P K I B M K M Q L K X R G B V E G
G M H M E R I P M E B G A V V G Q I U S R M
O M C E S D B J S R Z G C I B C A N Y A X O
L N I P S I R C O Z G N X M N C P L N V E W
D I N O E A N S L N O M O D B Z A N H M N S
E D F T R N I K M C C J P S F B Y O A U Q E
N A S B P A U W R I A W A W C S P C C J Y S
D R U C C B Q T N E E M L Q M F G W O J A O
E E O R P W R T R J N U A I K G O G R K Y D
L D I Q S X O E V O I Y T N P F H G T I M T
I F C C V S O G Q A F H V C O J O K L B O R
C V I P H I J U F R W O B K X J N O A R A T
I L L L T Z Y V E L O C K A A W E A N W D W
O M E I V V F T U P I N K L A D Y L D X W Y
U X D H I J G C K N N U O C A M C H Y R P V
S R D B B R A E B U R N Y E E S R G Y K S B
Q Y E Z F J O N A G O L D C G X I F E O R Q
A I R N T D E R A L U A P V Q R S M I M T F
S R A V I U K I K J C B R D Z O P X G V G Z

Daftar keyword yang perlu dicari:
CAMEO
CORTLAND
CRISPIN
EMPIRE
FORTUNE
FUJI
GALA
IDARED
JONAMAC
KIKU
MCINTOSH
OPAL
ROME
AMBROSIA
BRAEBURN
HONEYCRISP
JONAGOLD
MACOUN
```

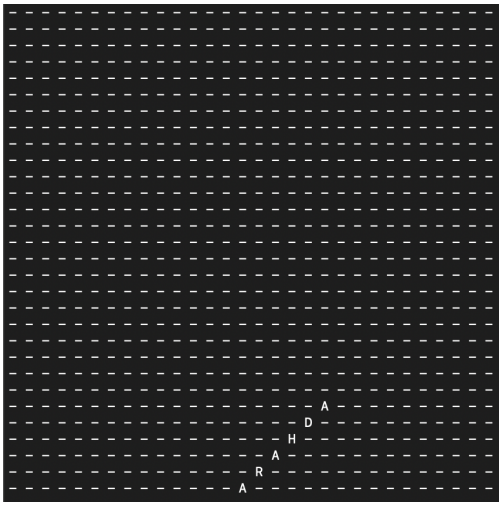

O
E
M
A
C
COURTLAND
NIPSIRC
ERIPME
E
N
U
T
R
O
F
IJUF
A
L
A
G
IDARED
C
A
M
A
N
O
J
UKIK
M
C
I
N
T
O
S
H
OPAL

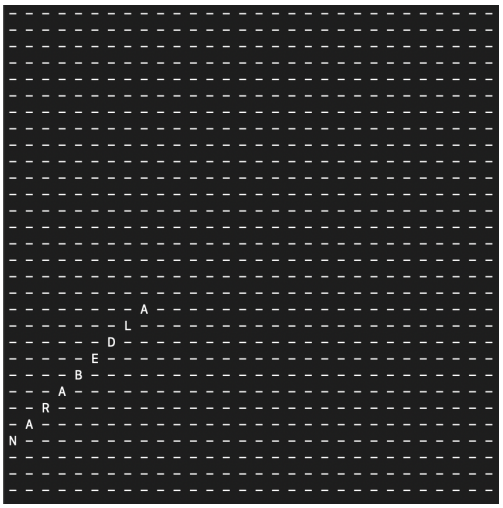


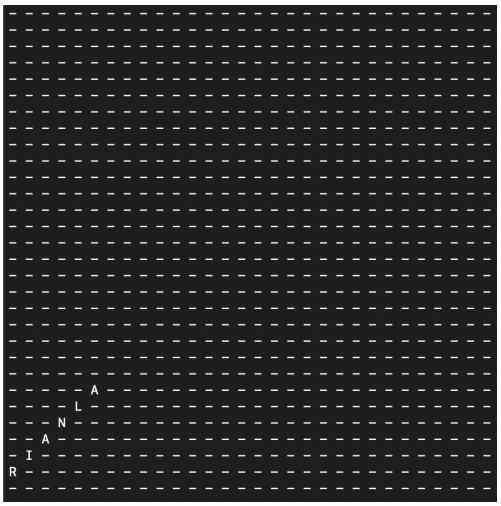
7. large1.txt
   Total perbandingan kata: 33176
   Lama waktu eksekusi: 6568 ms

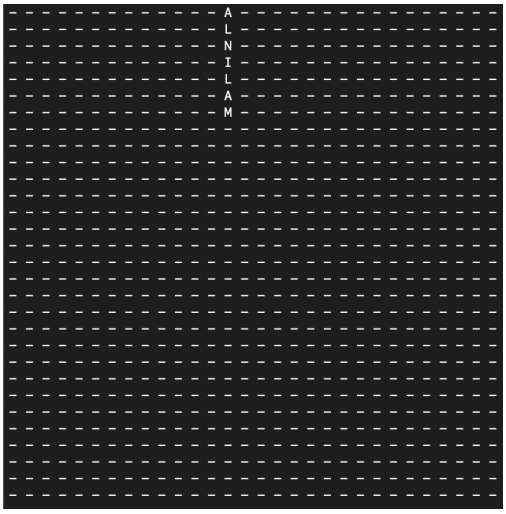
```
Masukkan nama file yang diinginkan: ../test/large1.txt
Z B E Y D A J N Y S P E A A E F N H E E I F S E B S N C O R
L V Z B M J C Z L R S Q C L N A N O S S T V F C U X N N R C
R I G I L K E N T U M S R N M B Y R Y U R E Q P N R P R Y K
K T Z A G S Y T E C C F U I E I E Y A C H U O A M R F N C I
X U Q T H I P G U X Z A X L H F A H B M O N W U H Q O Y C D
W L Q J Y Q L T I S D Y I A R F L P F P A R Z Q V P I T P Q
H G X Y G E T R F V G Q V M G A D E L C W X P F Z G V Y S T
K P A E T Q T O K Q F A N R M F N G T A T Q N D Q K M C L A
S J O E N A V B Z C D Q T O H T K Y O L C W E Z G A U J A P
E Z B C L B N F H Q M N F L W R O N W S W I A X J B N Y E V
J H D L G N Y F O G T H B K T B R O V I O U D R B E X L S J
G X E V B C U J J T X Q Z J A B M A R H C G F U Z Q L U Z L
B B S R W D T E B U O O R A M D F E X T W Y D D S A V H D K
S E R A T N A E L N A T H M K I G Y M A Y E K J L N Y L Z W
K Y S G U U M J C C S X D Y H G M S O N R S O Y N P P I Y S
M V E Q B W F F U G T V L E G I R O F W E N U R E X V U X S
U M R L P H Y R B A Z E C M O C R U S I G E Q O K U M T V H
V L P G Q D F A U U L G J Y T R A M I A U F A N U L C P G B
S A M Y D Q X H A K N A P T Q T N C T U L E Y Y S L Z N W U
U U R T Q F X L K X H D L Z K F U X I L U D Q B J O P E R M
I C Q C P W D S T H A D A R D B Q N Y P S A Q T V P T J U N
R S R C T E Q P O V Z N G G J S Q T R H S X L A L U A H S G
I Y B G B U A F H W H A T G A J M R G R P E X T H V V S N L
S Q S A Z A R Z V P M N Q S M R S L N X P W D P A D P X H W
C H R J L I J U T Y O J Q L S R M F P A V P L D R I L U Z R
W A W N P L R D S N N C C K R M E O D D K K K O F T R J B X
N G A U W F Y U W Q L C S W D B A H C P R E J V P S L B A R
U I L D M U V X S V V Q I Z P A A Z W K J Y Q Q X H A D E N
R D H U K R T S F U B M Y R Q R R K D Z Y R G U M M U U Y G
K M F H Z F W Z Q F W Z F W A H G E P O P X T O I O H D G G

Daftar keyword yang perlu dicari:
ACRUX
ADHARA
ALDEBARAN
ALNAIR
ALNILAM
ALTAIR
ANTARES
ARCTURUS
BELLATRIX
BETELGEUSE
CANOPUS
ELNATH
FOMALHAUT
HADAR
MIAPLACIDUS
MIMOSA
POLLUX
PROCYON
REGULUS
```

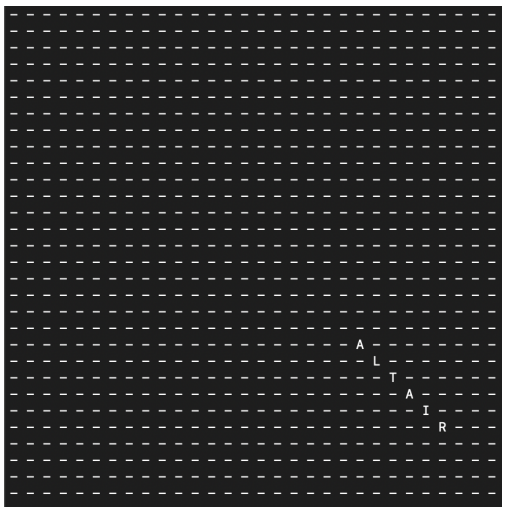

ACRUX

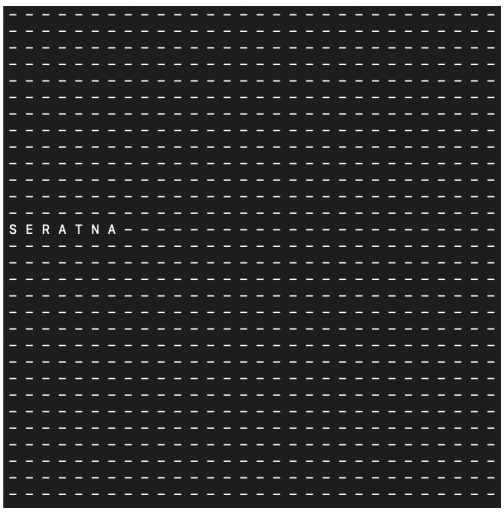

ADHARA

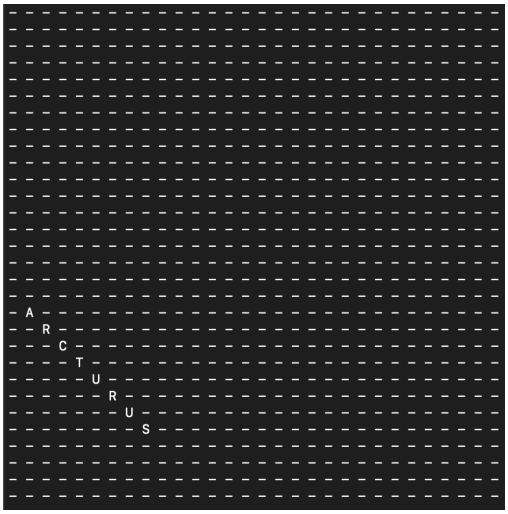

ALDEBARAN

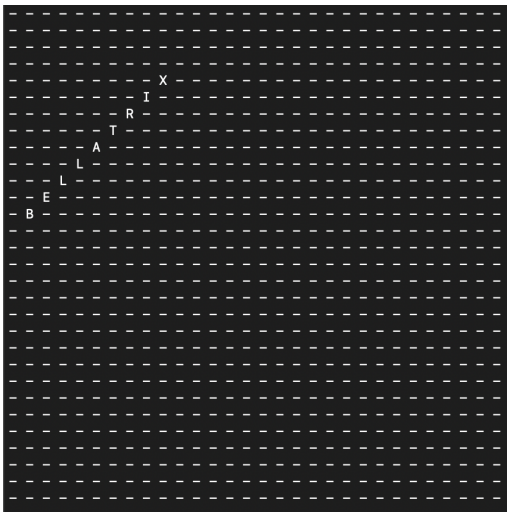

ALNAIR

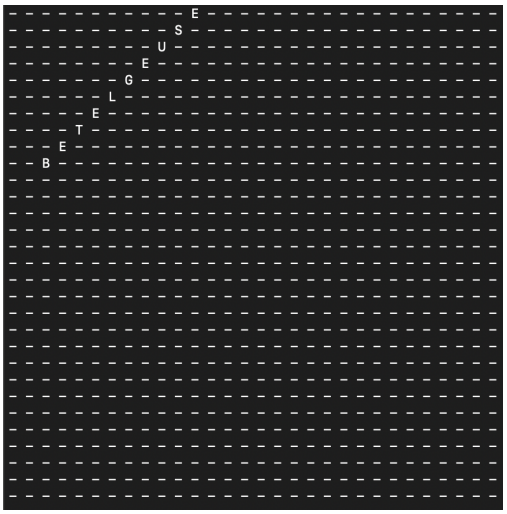

ALNILAM

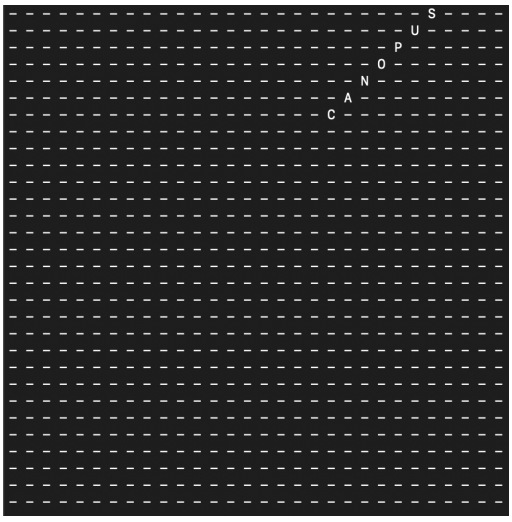

ALTAIR

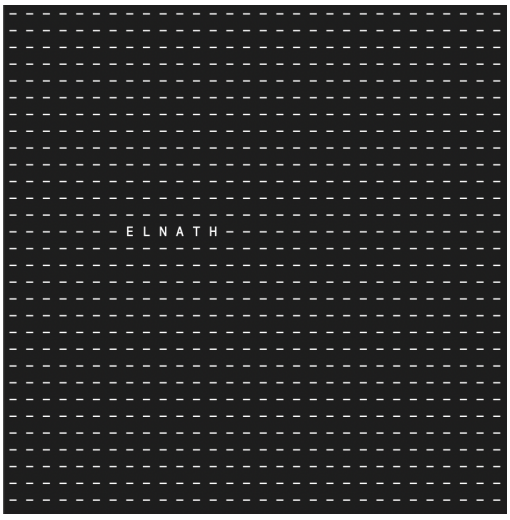

SERATNA

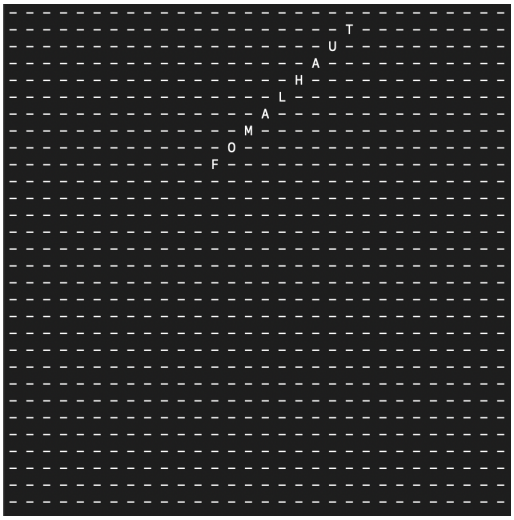

ARCTURUS

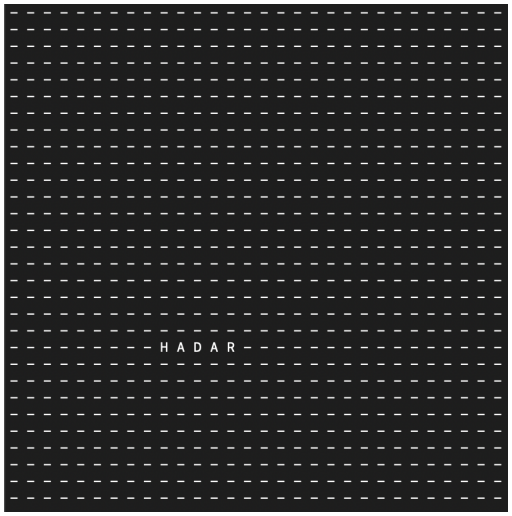

BELLATRIX

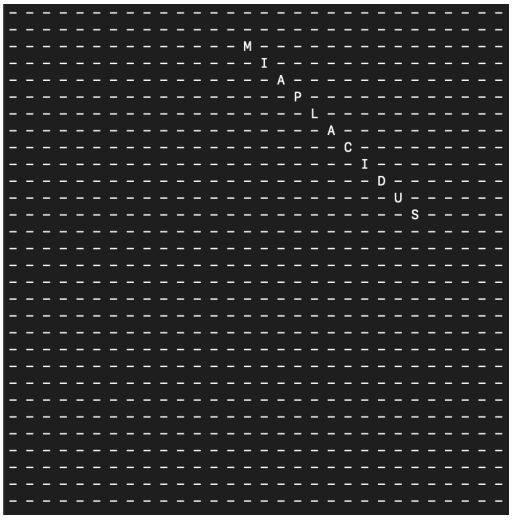

BETELGEUSE

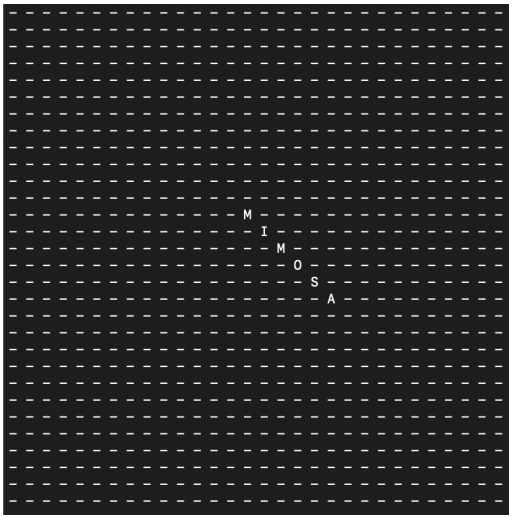

ACAMPUS

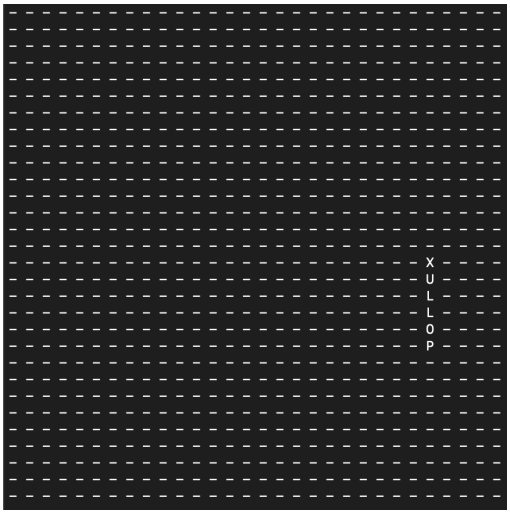

ELNATH

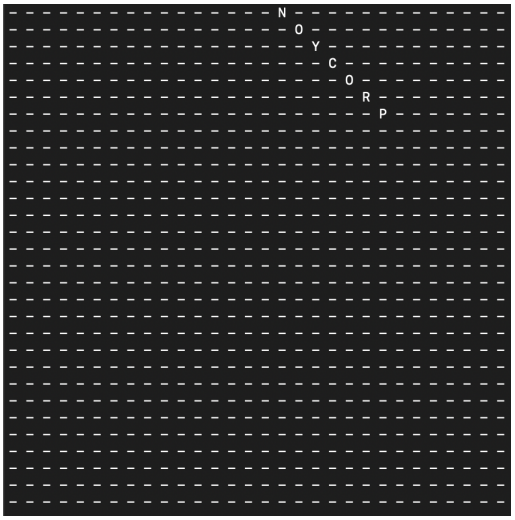

T
U
A
H
L
A
M
O
F

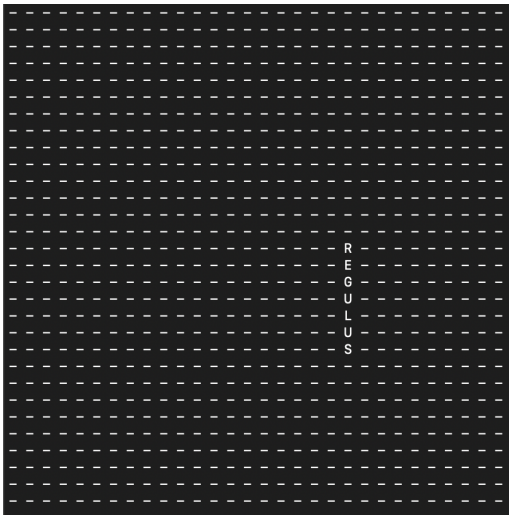

HADAR


M
I
A
P
L
A
C
I
D
U
S


M
I
M
O
S
A

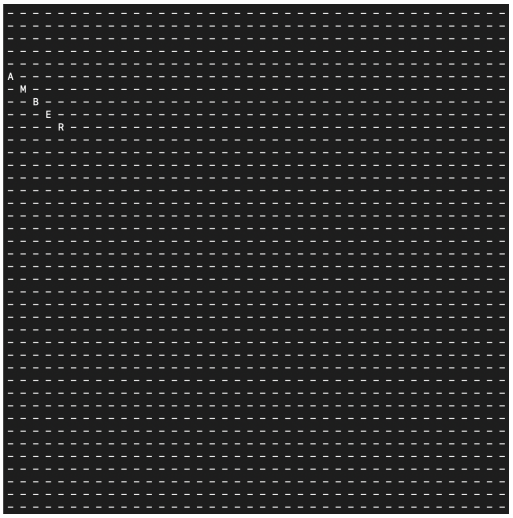


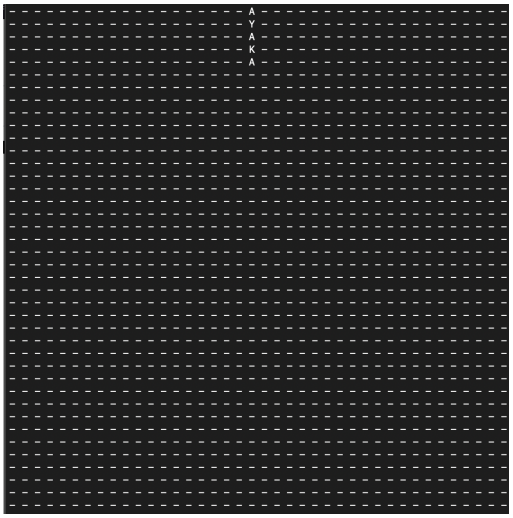


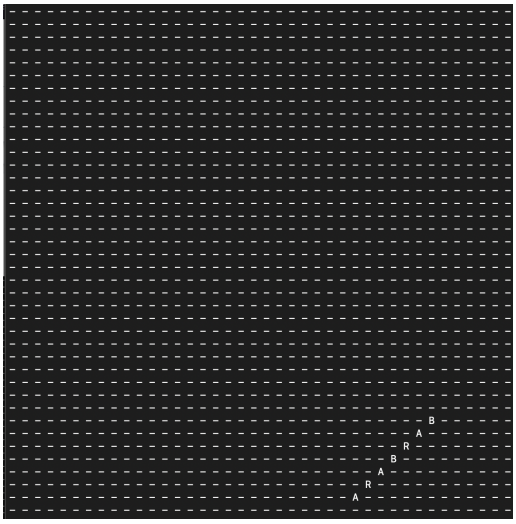


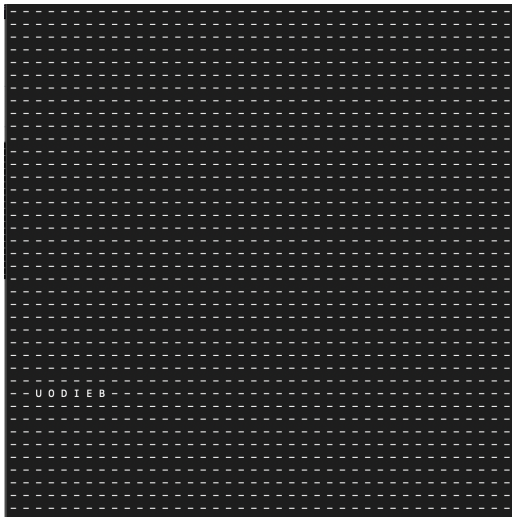
8. large2.txt
   Total perbandingan kata: 61623
   Lama waktu eksekusi: 10488 ms

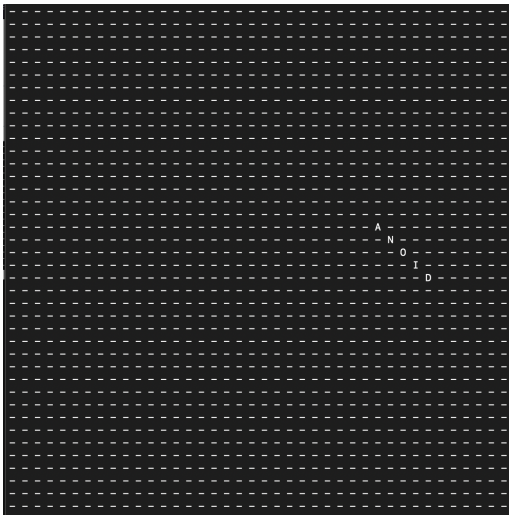
```
Masukkan nama file yang diinginkan: ../test/large2.txt
X Y G H V U O S G T I Q Y C J Z S J P A A E V R L U C J V G K Z T S D T S X D A
B Q I S U U J A I O M U Z W F G Y E U Y U G H H I C T G Z I O D A Z I N U P V Q
J D Y M X C J K S V T C C X Z X D A H A M S C N D H U E R M P Y C J W M Q L K U
K Y A M O S F D I D Q B W M W W W N Q K C S S R E W P T Q E U B F G H H W I D C
M A F J Q K B U H T E I N V X E Q N F A I O E O V H A H W X M H I Z B M Y I Q A
A Y G G B K O S P R V S G V E N K W R F S E N W M G S F I B X E L T S M X L M I
F M P Q U A O K S K S J D T D M D D L W A U S J A N X L Y R M F A X V E R A W V
U D B G F N C I D D C Q E V U U T L A A I R A S O R P B T Z C U I E Q U N L B C
R W I E S X R A H H B R R J D H C D U F Z W B X D I O R M T G L Z C T V K P B N
E V R V R T V P X F I B K X F S V I T X G R Y K L E K E D C Q H B T N A E X K A
H A U A W X T Z D E S I A N I X A R E A M T Q J Y K T S T U Y F E N T I E E L K
B V J L H C X R L X U M E X Z M M O B S T Q R D B O B O G G I J K N F U D O O T
E H F W S R S G G W D V S D D Z K L N A J P N G Y U R E E Q Y K Q R J Y X A I M
O A A Y H K O O J N P L O N I Z B N W U Y H M V Z Q H O I G L P S X M Q U L I J
V N R Y K Y A N F E I Z R Y H V Y O W A B Z C Z Q P V J L J S Q K N W A T R C C
U G U V T L T Z Y J F P C T I N Q S T G U X F N I J C L Q R F B X D T B A N N X
I B B U J Z W V Y O W A U R A S C X U X E W U K Y G F U S R H N M Y L W O L F A
C A N P J O P Z T H I Y S E T N P Y P I T E C V K G D R A A P H O X C Z G Z K J
E A H S L C V Y M I B M H J N Z O O J Z X O I X G Q S Z T T N V W O T N F R F C
W L G T C G X R D D Y W I R T K M U H J L Z E N M W N X J C Q O A U A B S Z K D
C S W T P M J D N D Z L V Y C J X F P U J X P I Z B N Z Z R F G I U A U K X C D
Z R R E B S C X M O B X U Z A Y S J B J W Q T A G A Q I D X D I G D B O N W P J
G P A F B W K B E O Z R J K R H T B R L Y K F G I B H M C R O G Z F E I O B D K
M V A F Z J Q E C V K G Z W S U O O G O K S M D J B Q V Z Y N V W H H Z W N Z S
O Y H T L E I N E P Q C Y L A P I Q G D K M S K H O R U S I K Z X Y B Y G N R P
L X T U T U F G D U I M N Y M Y A Y U A U M X I D T A D N E H G J F T N V V B S
O P H B T M R A T D S B Z P N A M V P J P B K F T G L C S S L E G Z I L T E T J
P K E P W A U P M J N F O A F W I L V L F L H P K Y U M P I O X A L S Q J L U B
K R Q J O S O O A B O I G Y V O H F J C Z W S G M X E G A N Y U G Y S S L J H J
A W F B C F N O X B P D M U N Z X W T P N K T D K Q C I P B A N K Z W V B K I Z
H P U O D I E B A K P V A N J J Z O O R E M A A F B R Y S J A X D F F J N V S I
S T G D A Z M O X U Z U M J Q D L C O Q J J L X Z S Y O C I H E X M D T C D J Q
I X O V S K D O W Q D O V I K Z K T I B Z A K T D Q A C X G T J P B U M B H E Q
H I Z I L P H C C O K B G N Z N Q N G E K X L B H U A W W U Y I A F Y G T R I C
N O E L L E F R B I Q D U G G M G B X Q I O F V X F S B Y X F R V N V S H T A Y
G W B A S Q I I M W I I K M U F Z U Y U B V L W X B I S I L B K Q K P E U I M I
Y Y F E B W P R S K U F R S I U E Z A Q M P S F S R L L S A I Q A T Z J N I W K
V P V W K I X X J Z R Z D N Z N Q E Y X Y L G M E W L C R L V N X T Q F C D C L
S L U N B U K U L R V Y U O W N X R T V P R T Y F Z N A W L F T Y B A N G Y M V
Q F H T Z O F M P A X C Q Q U A Y B V V H S C F E L R B A J W P I X H T G F X A

Daftar keyword yang perlu dicari:
AMBER
AYAKA
BARBARA
BEIDOU
DIONA
EULA
FISCHL
GANYU
HUTAO
```

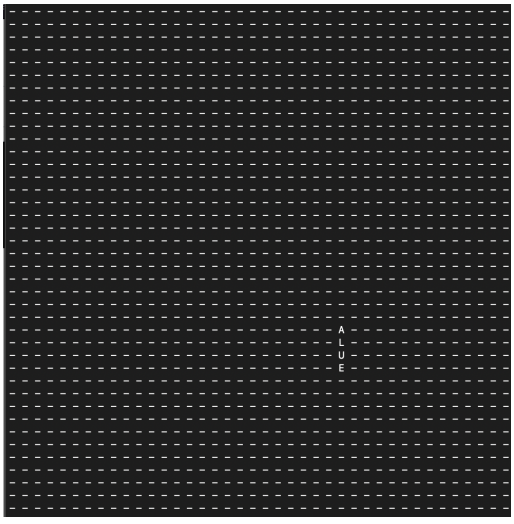

AMBER

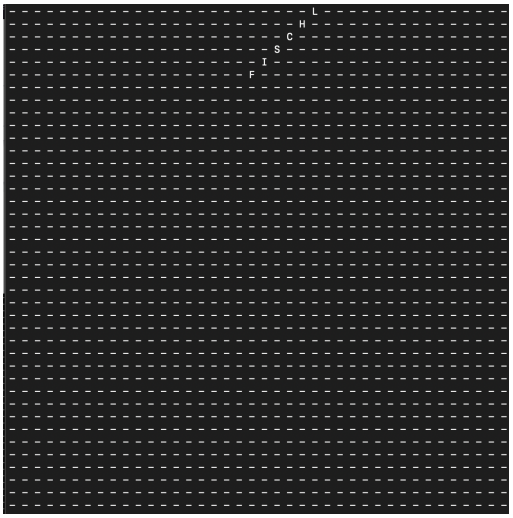

AYAKA

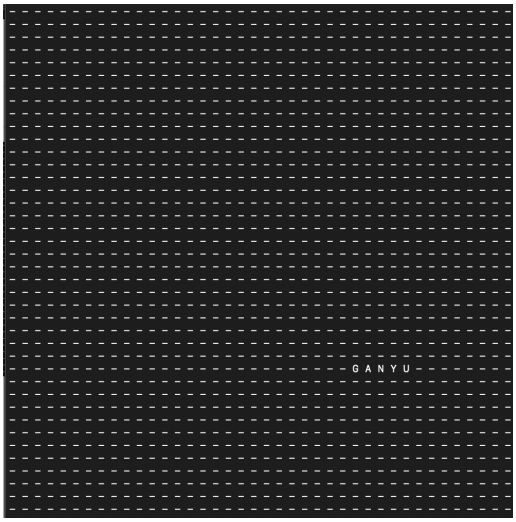

BARBARA

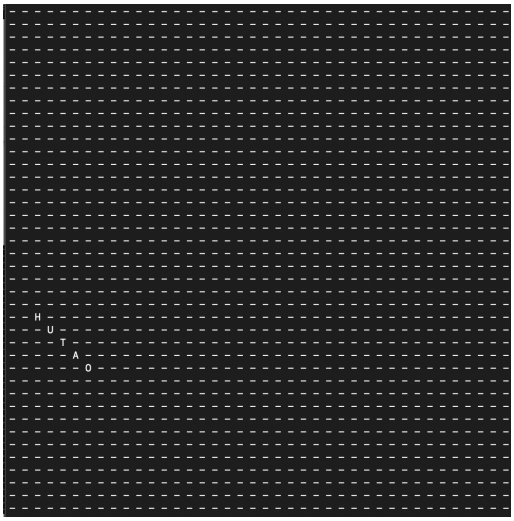

UODIEB

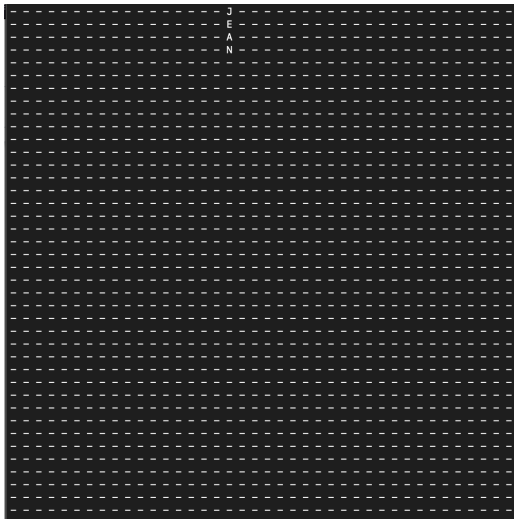

ANOID

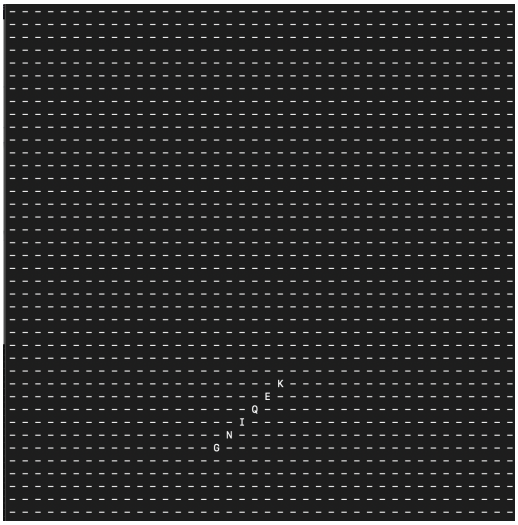

ALUE

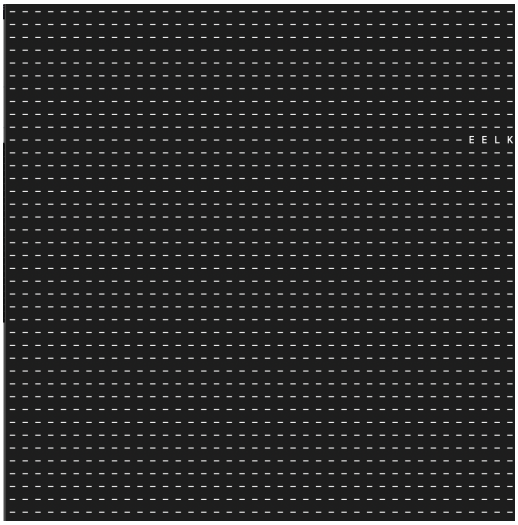

LICHSIF

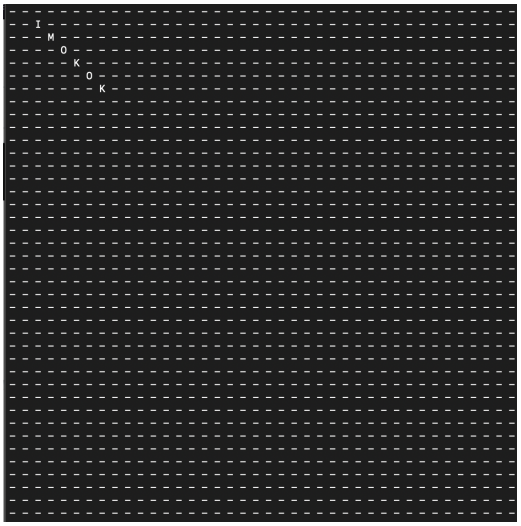

GANYU

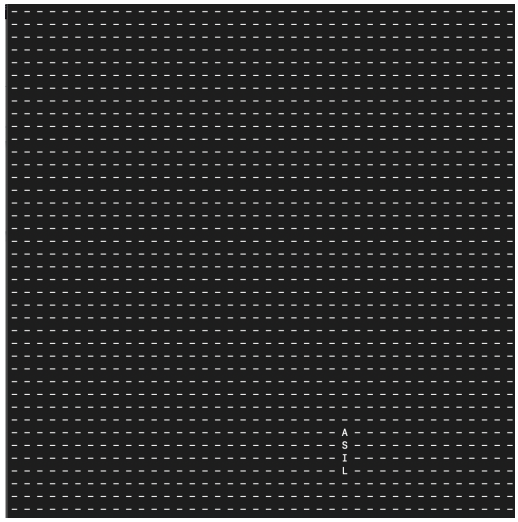

HUTAO

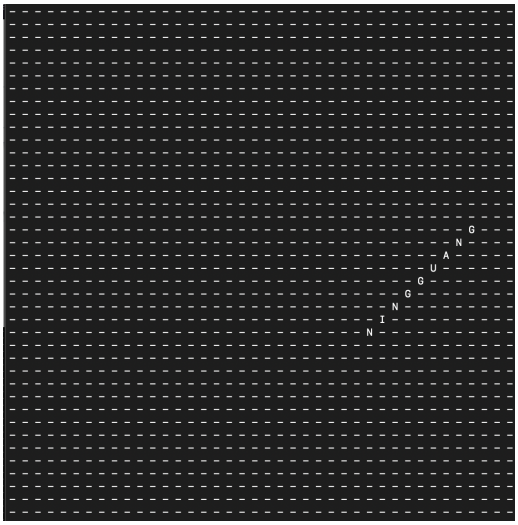

JEAN

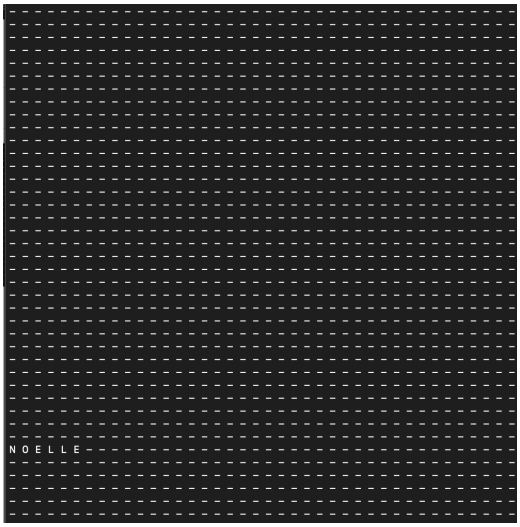

KEQING

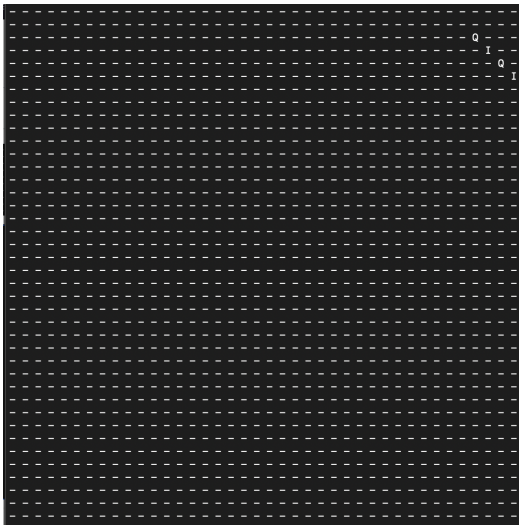

EELK

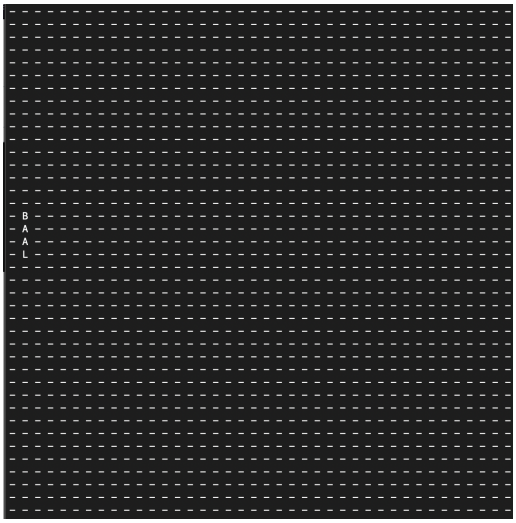

IMOKOK

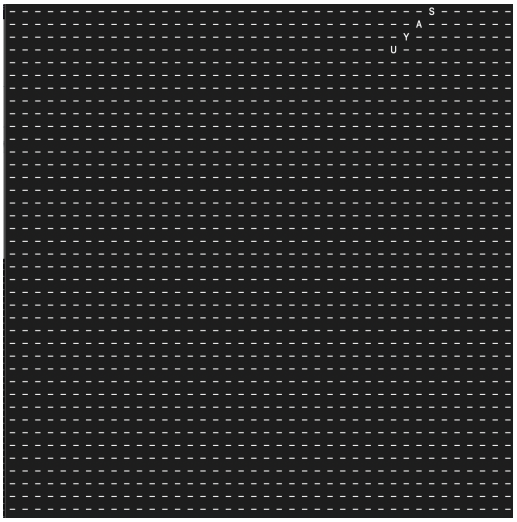

ASIL

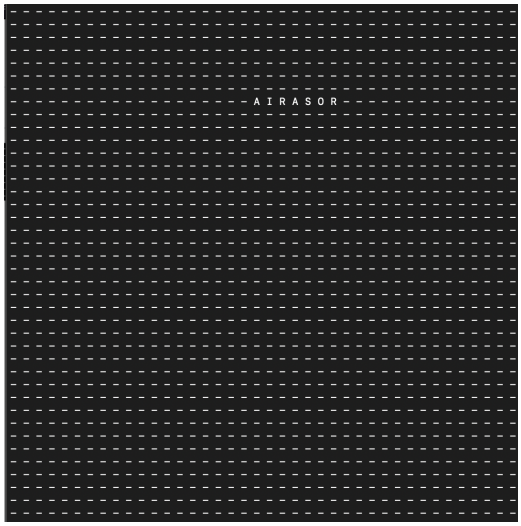

GNAUGNIN

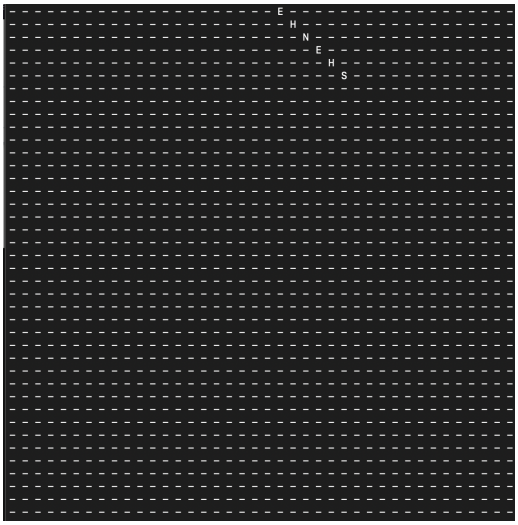

NOELLE

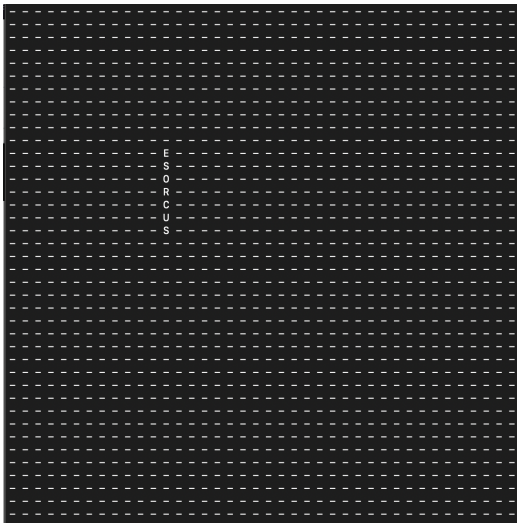

QIQI

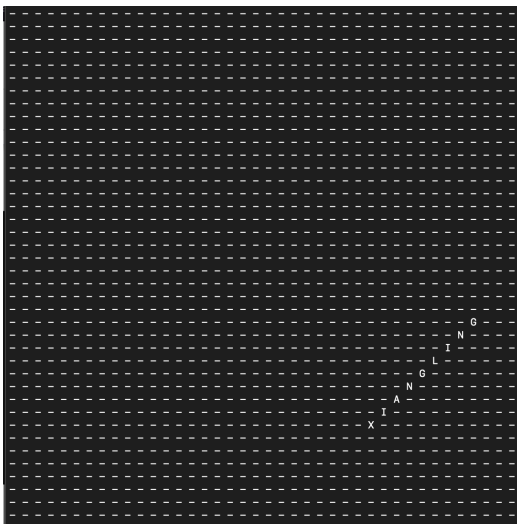

BAAL

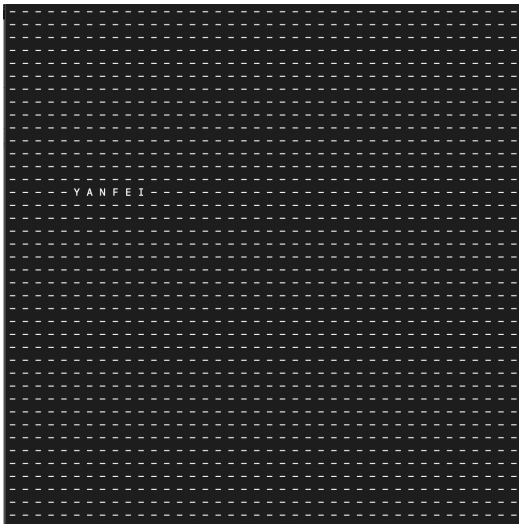

S
A
Y
U

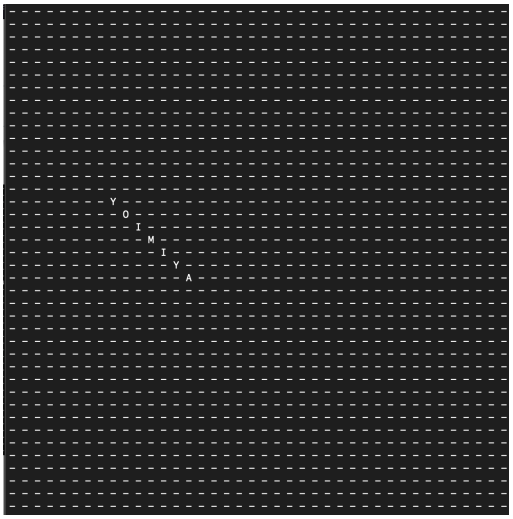

AIRASOR

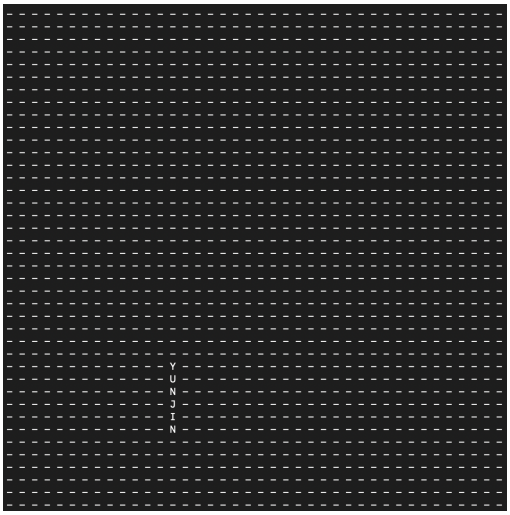

E
H
N
E
H
S


E
S
O
R
C
U
S



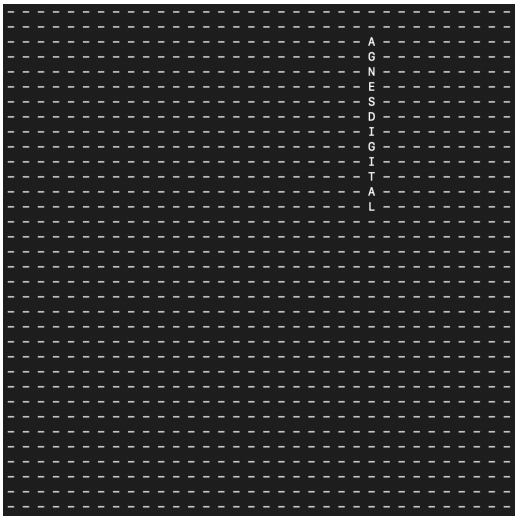


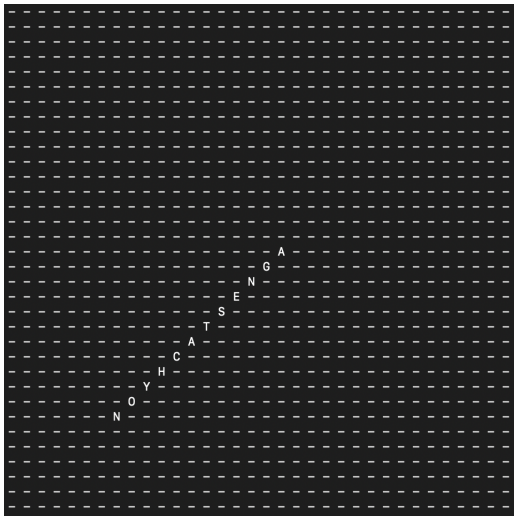


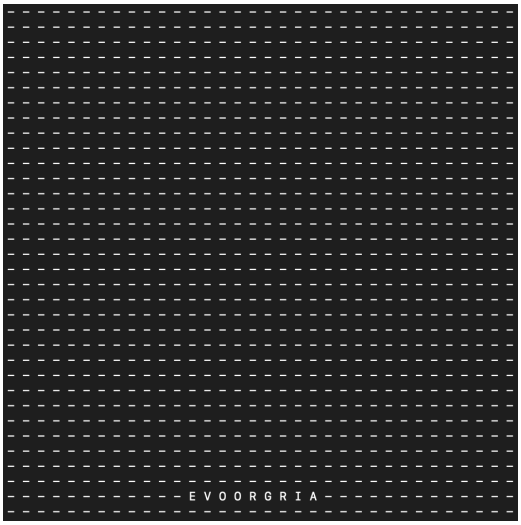


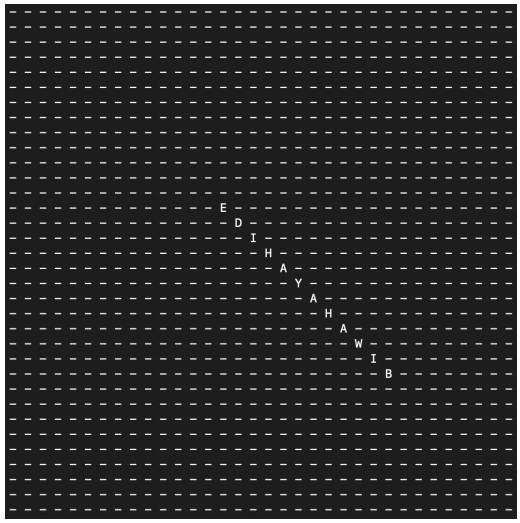
9. large3.txt
   Total perbandingan kata: 78695
   Lama waktu eksekusi: 14325 ms

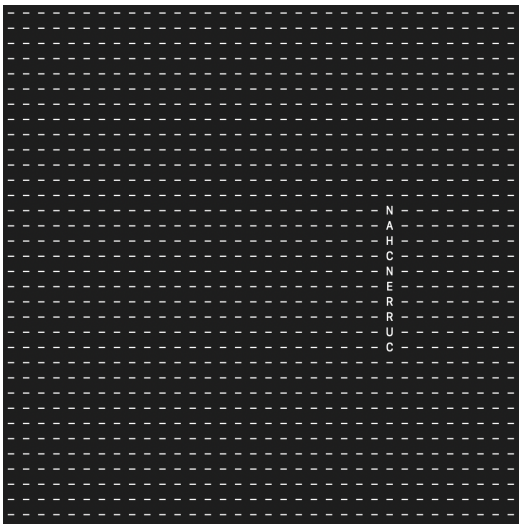
```
Masukkan nama file yang diinginkan: ../test/large3.txt
P J M M N B O X N A O V W S P W J R V W E Y C C U Q J I N I J I E A
N I S A H A Q A I C Q B S J Q T E X I G V L J L O S E J K J K Y R M
O P H F Y Z Y Y R D I O K R I S R Q F Y T P C O A V O K E P U A O E
B O B S I A R R F E R V F G U A S X E K Y T L O G D S D X K R V G I
R H N C D W N A O C P R Y A H I S H I A K E B O N O D O I U E C U S
U Y O O Z L Y O O R V O H A S O I B N Q V H V W E D Q N U U L Z R H
O Y F U Y X O M T R I N M N U Q O L A H G N I K S G O R N L T E I O
B K F M A I A G I O N J A T P K S E C G B J D H D B A R Y L T I C D
O S L B Z M H C Y A P B E S E S F T H Y E Y Q K I H O L P T U H A O
N N B X A J E C T K S G M M R G I A G J K E A J G S L F O A H A P T
O U W T N S G E A F G M U V C F N D Y Z U W I W I Q H D E N S K I O
H I X B H I N Q U R H B V N R H E N Z X A N Y Y T Y N I V I I A E X
I E N O U A Q J I E U A T F E X M Q I K T F K L A R Z M A C K X U W
M S W E K Y I X N D S K M Y E W O K A C B E L Y L N E O E M I U L I
O E Z I S K Y J E N J G A W K D T M Z Z E Z F B D A K C F P A F X N
R Q T O I S K N S O F F Q S N U I H F V N N P A D H Z Q P R T Z K N
J A U S W X S Z F W T Q L S Q P O H A T T W A G C C I T W W V X O I
M F E U Y U N D U S N J Z R R M N G A E O R P T M N C H E A U K C N
S K L J W E E X J S K N D I E B N A L Y F U I G U E A O V Z W Y H G
I R R C U S Z I I A N P N J T E K R Q G A G F V J R Z T H O A K E T
P E Q N C X U V N R L C I B S U A T O U E H Z B Z R E R T Q D J G I
M V H O A B R F C G E R A T Z C K L R X X T A E M U P V Z A U K A C
U S V I Y E A A X S O N A U S T T S Y X O E Z W O C K W B X H N A K
S Y P T X B M I S D S C S A R X N A R I T A T A I S H I N H A N D E
J L O E K A A I O C H E W T F K F L O D U R I L O B M Y S I C C A T
O F H R C Q K B I Y C I M A T I K A N E F U K U K I T A R U X B V M
Q D Q W N I E B O N A V W J N E E U Q C M O R I J E M B D P C Y J I
J M J E C R A N E D O N I H S U K A B A R U K A S T A A F I V S Q L
S M A R T F A L C O N W O K M R A O E S T O K A I T E I O S I L C C
Q V S A M F I M W Y O R B O R O N N E Z X A C V I G I F J Y S C A M
N W O O L S R V F E L A T V T L P A J L X C M R Q M H E N K N V W P
K P T N M P O H G H E J Z W Y T S U V E P I A F R N U Q P E D P A K
W Q N E T S V E J R Y K E V O O R G R I A N O Y T I C D L O G Y R A
P J J A C N O K D Q G F G Q B W F V E I S H I N F L A S H L K X W E

Daftar keyword yang perlu dicari:
AGNESDIGITAL
AGNESTACHYON
AIRGROOVE
BIWAHAYAHIDE
CURRENCHAN
DAIWASCARLET
EISHINFLASH
ELCONDORPASA
FINEMOTION
FUJIKISEKI
GOLDCITY
GOLDSHIP
GRASSWONDER
HARUURARA
HISHIAKEBONO
```

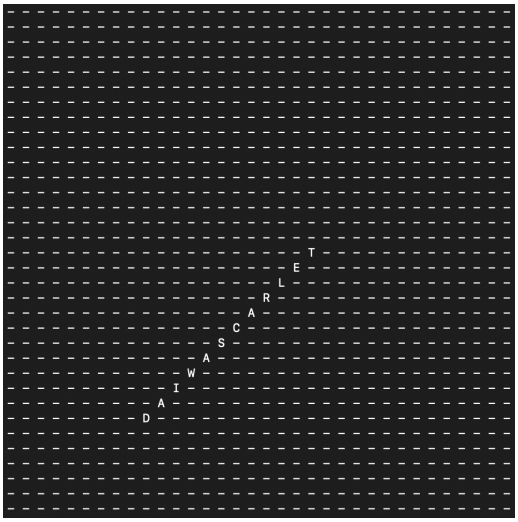

AGNESDIGITAL

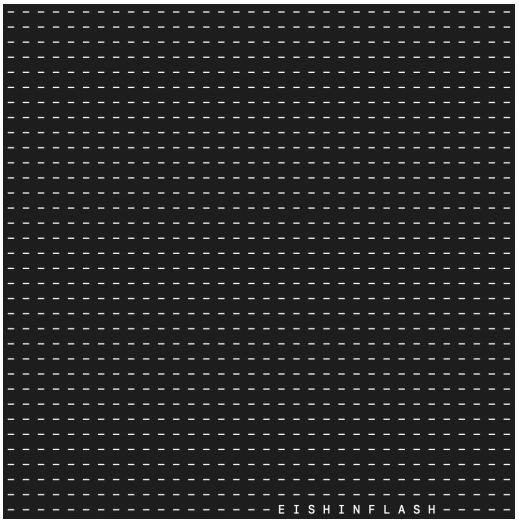

AGNESTACHYON

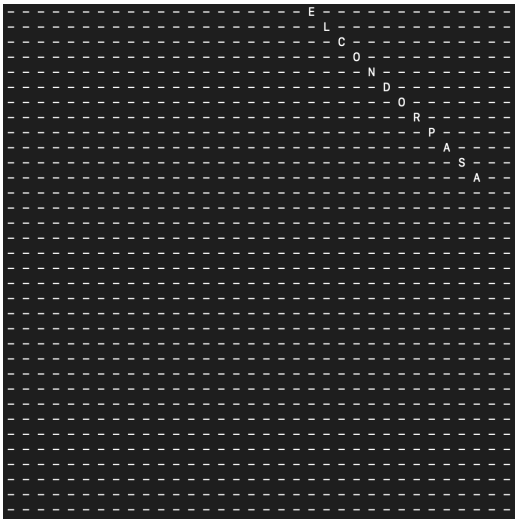

EVOORGRIA

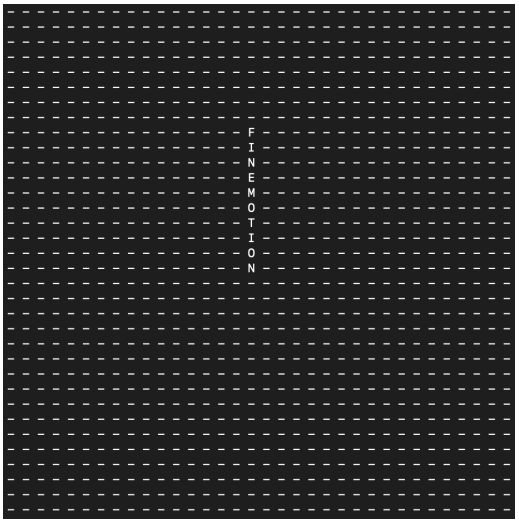

EDIHAYAHAWIB

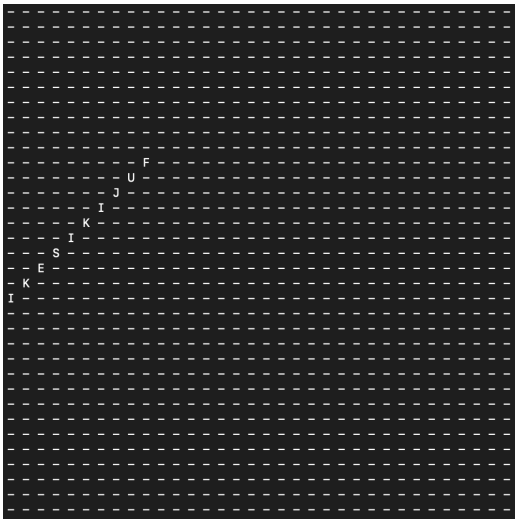

MACHINERY

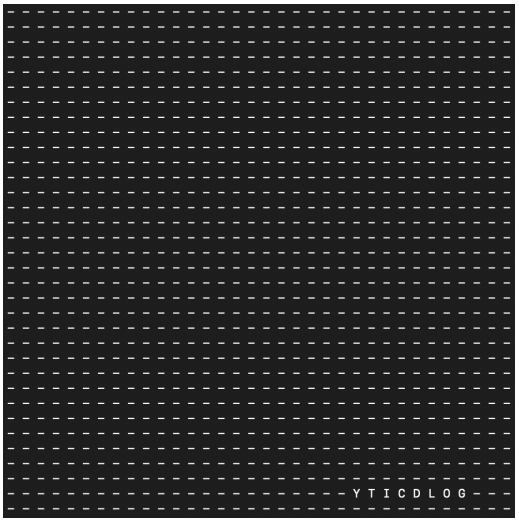

TELEVISION
WAIT

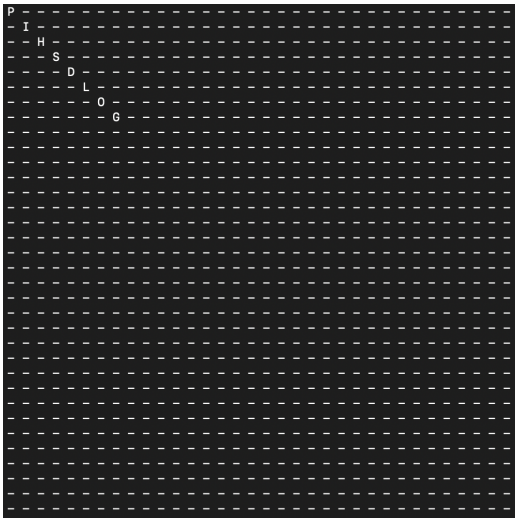

FISHING

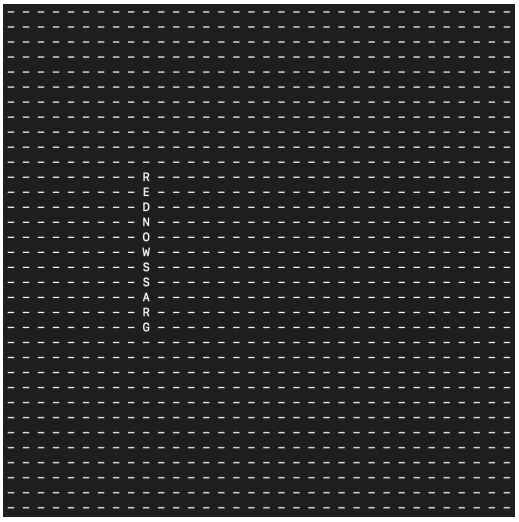

ELECTRONICS

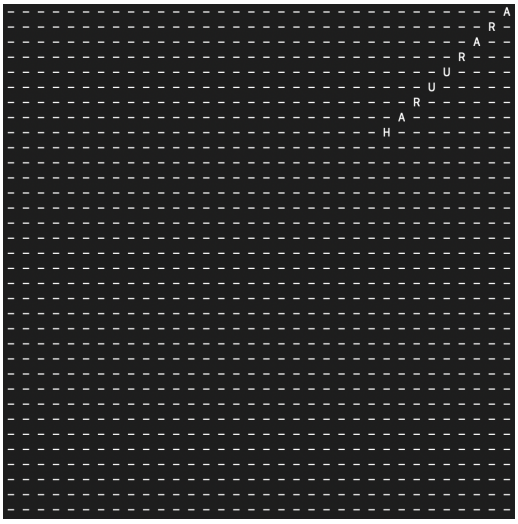

FINE
MOTION

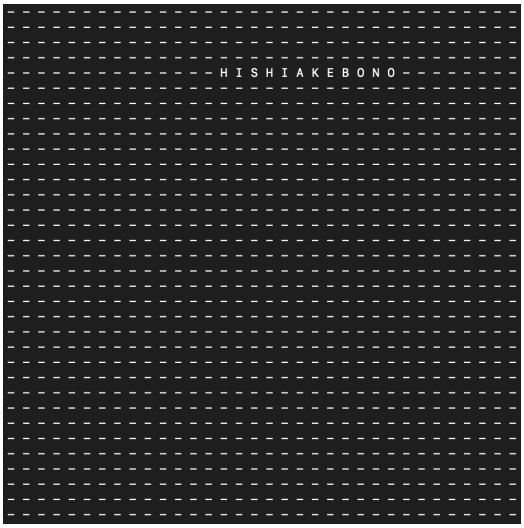

FUJI
SEIKO

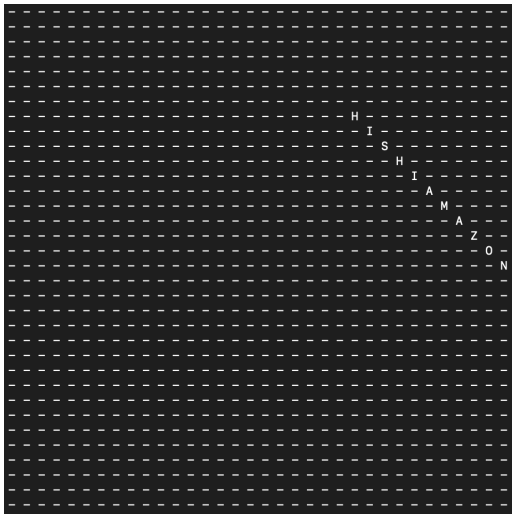

YTICDLOG

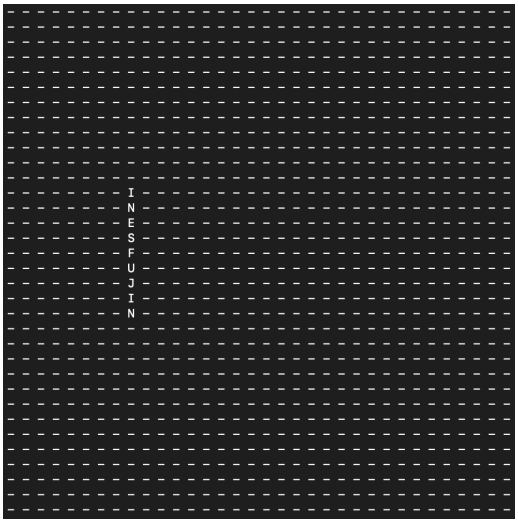

PIHSDLOG

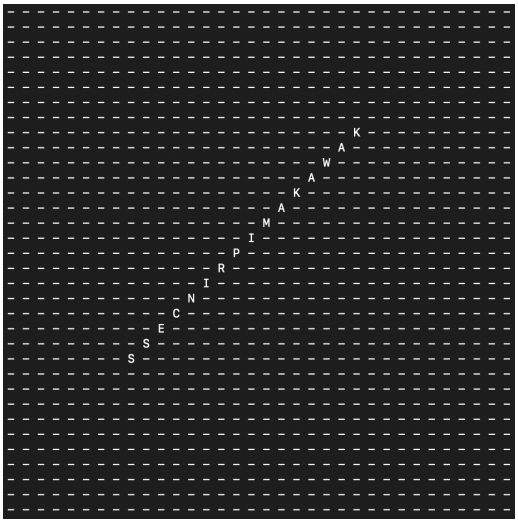

REDNOWSARG

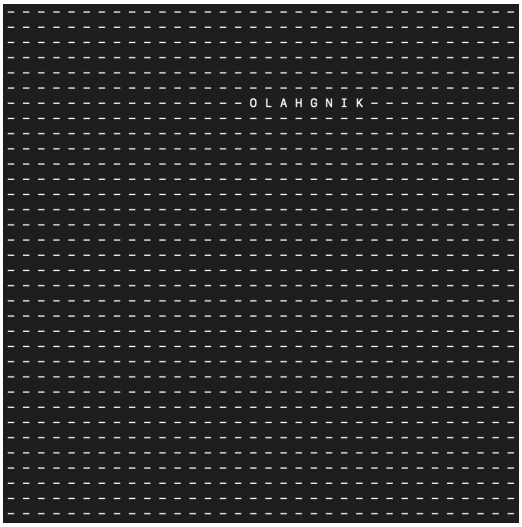

ARAURUARHA

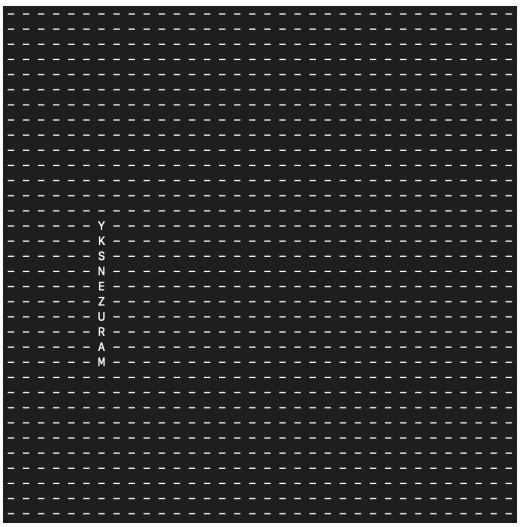

HISHIAKEBONO

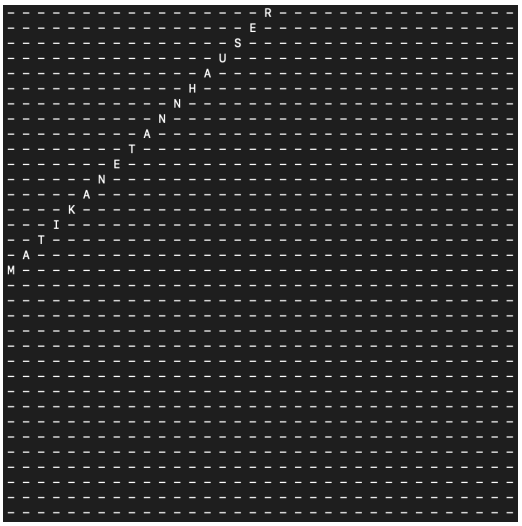

HISHIAMAZON

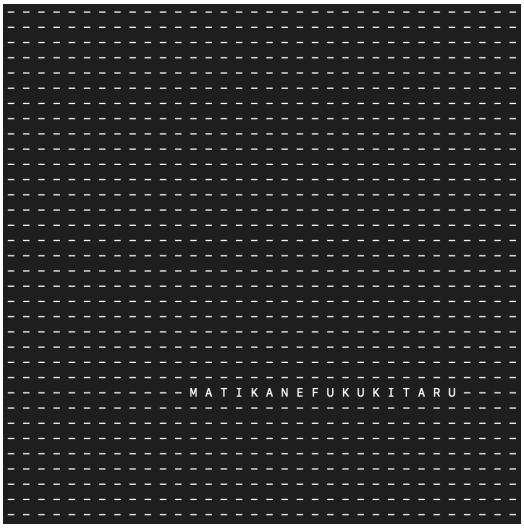

INESFUJIN

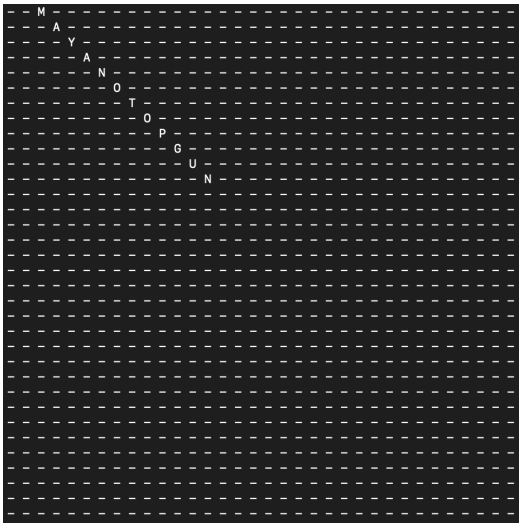

KAWAKAMIRACES

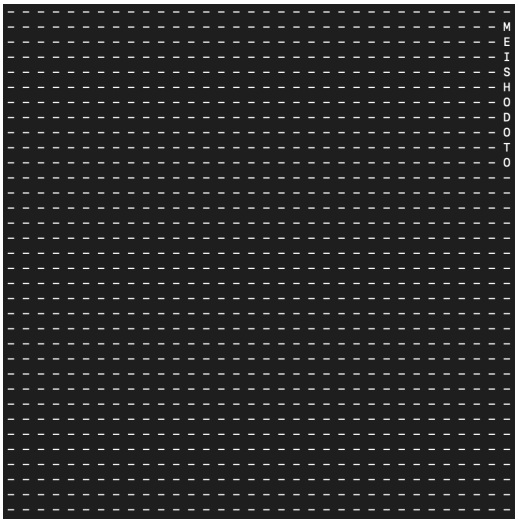

OLAHGNIK

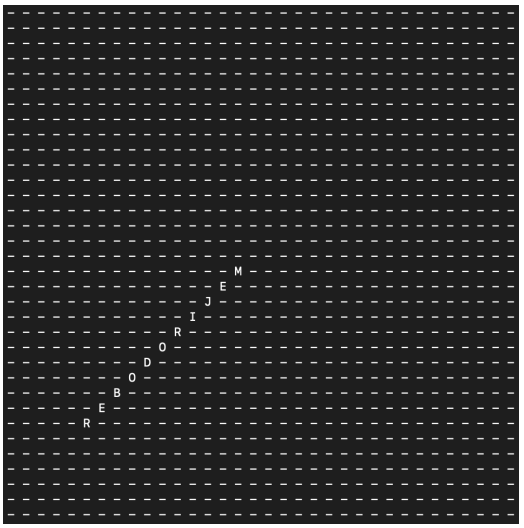

YKSNEZURAM

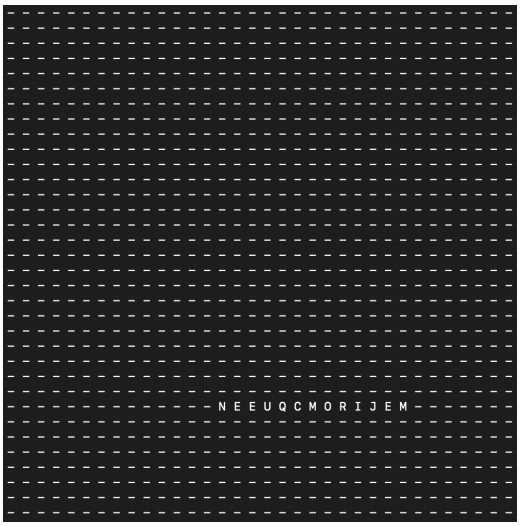

R
E
S
U
A
H
N
N
A
T
E
N
A
K
I
T
A
M

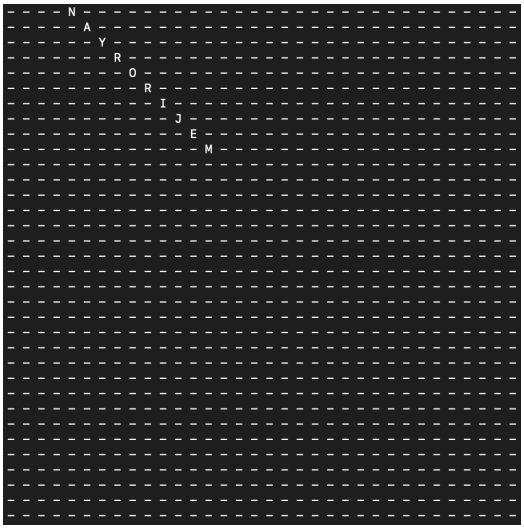

MATIKANEFUKUKITARU

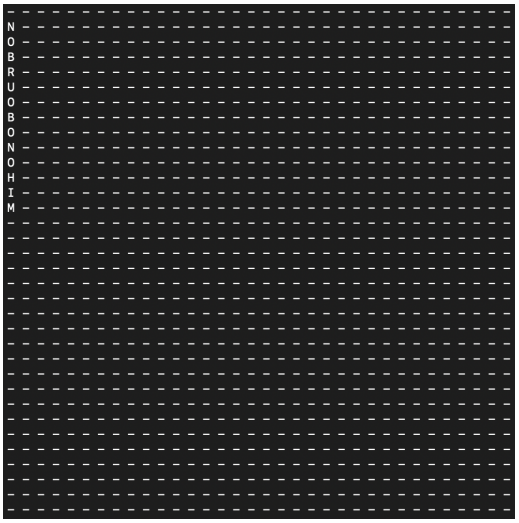

M
A
Y
A
N
O
T
O
P
G
U
N

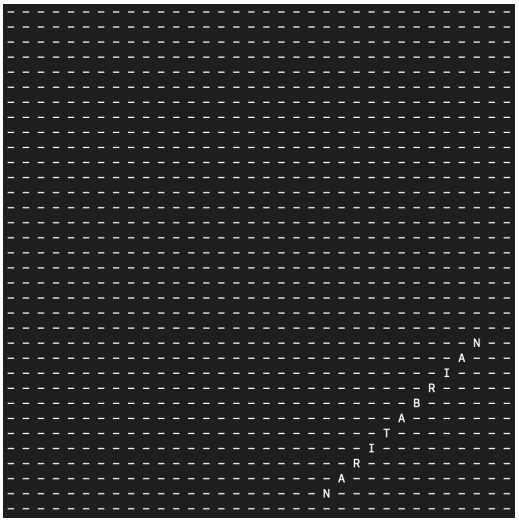

MEISHODOTO

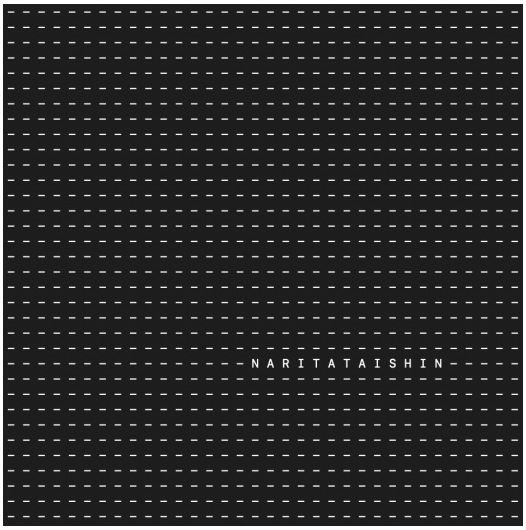

M
E
J
I
R
O
D
O
B
E
R

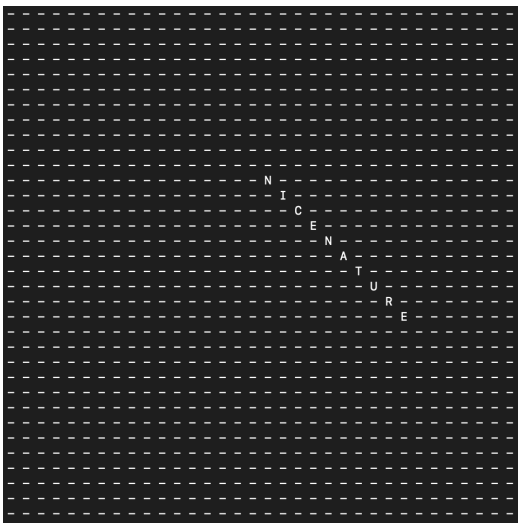

NEEUQCMORIJEM

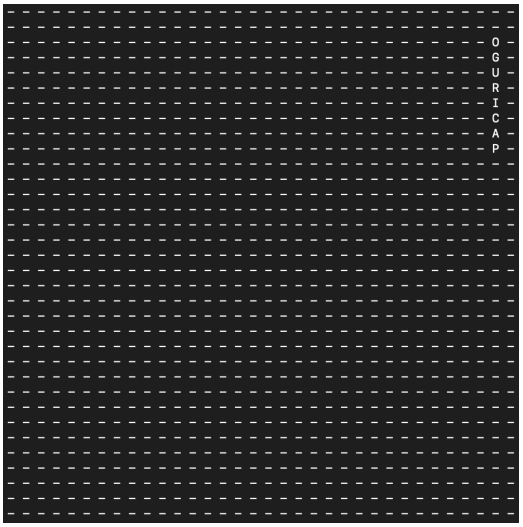

N
A
Y
R
O
R
I
J
E
M

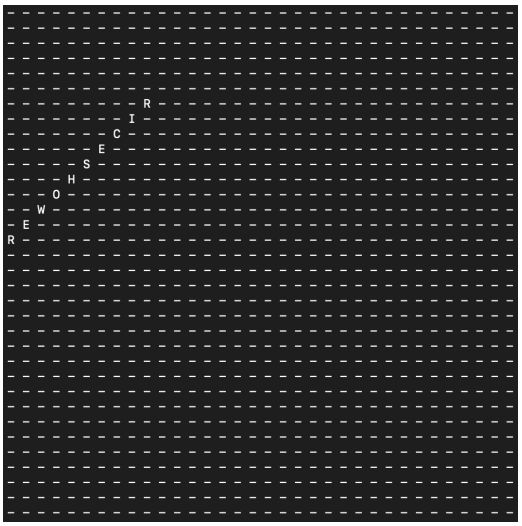

NOBRUOBONOHIM

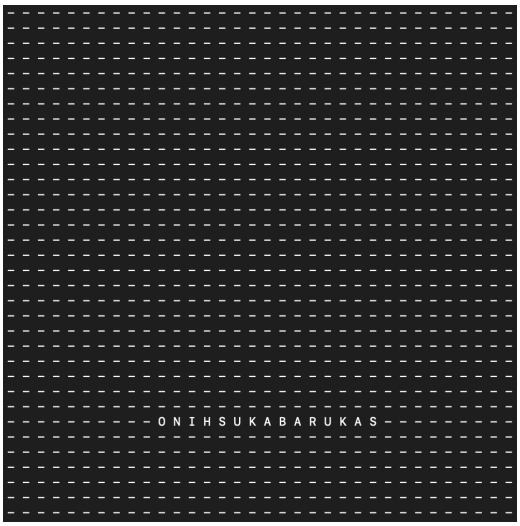

ANARITABARAIN

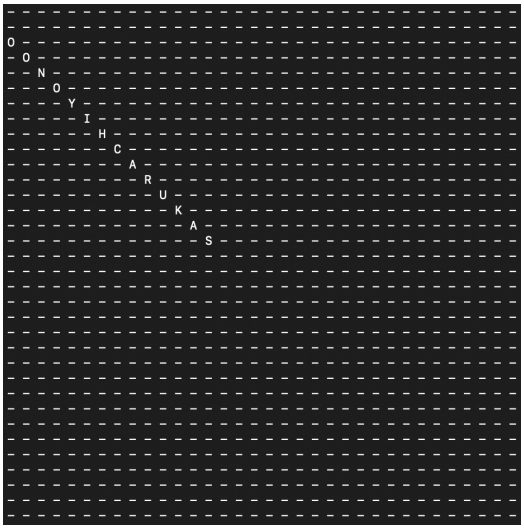

NARITATAISHIN

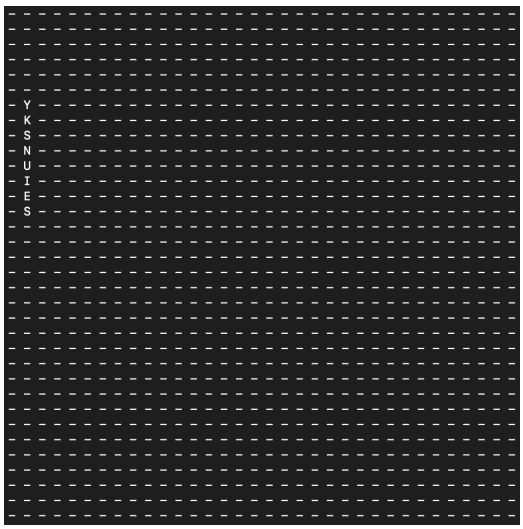

NICENATURE

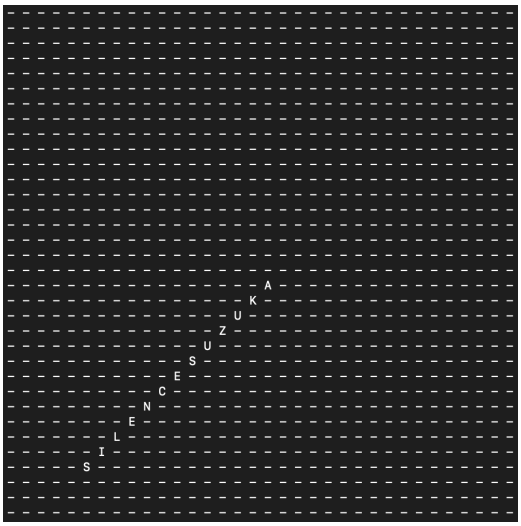

OGURICAP

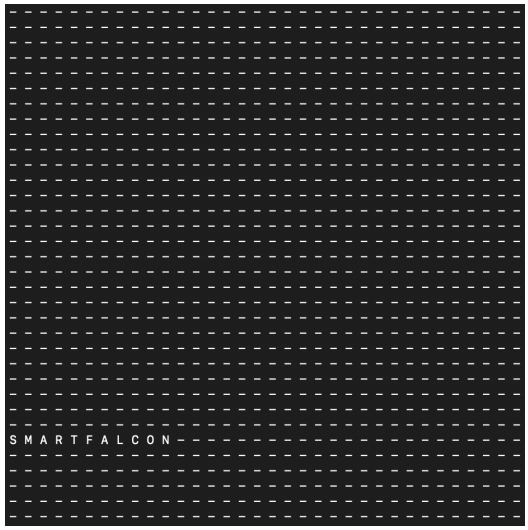

RICESHOWER

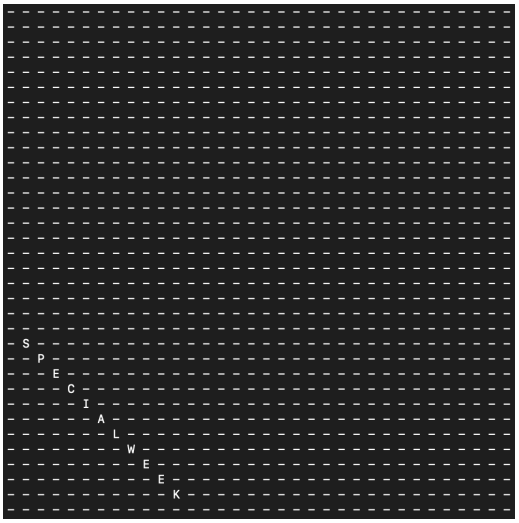

ONIHSUKABARUKAS

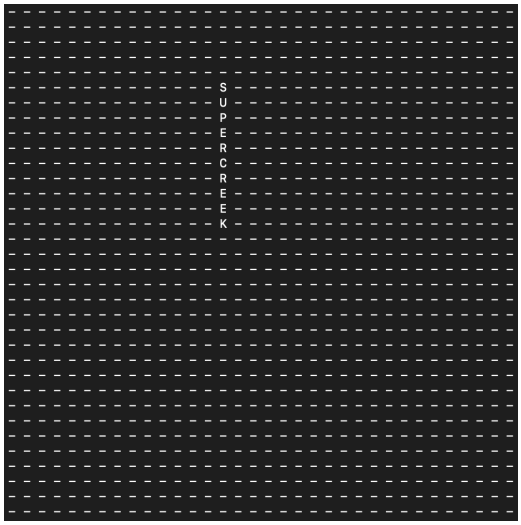

OONOYIHCARUKAS

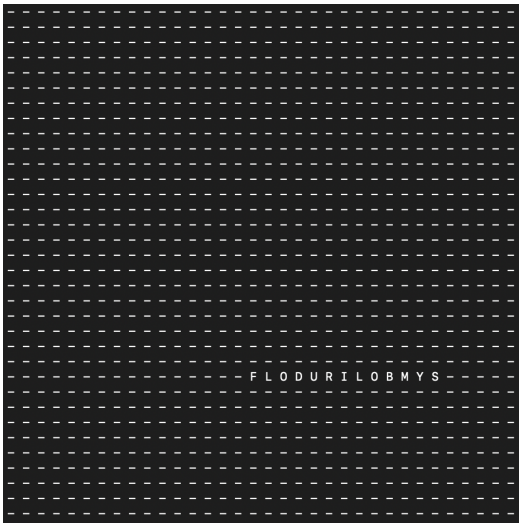

YKSNUIES

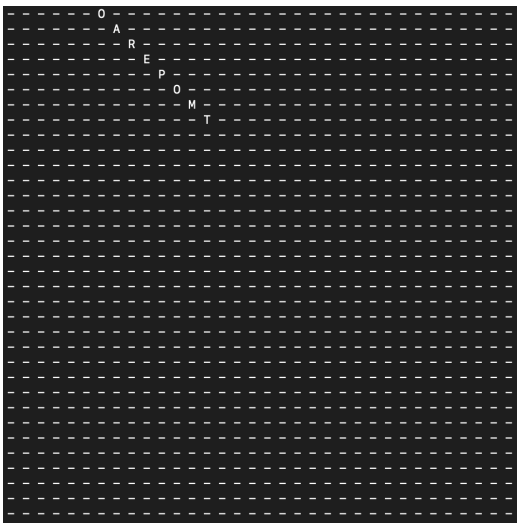

A
K
U
Z
U
S
E
C
N
E
L
I
S

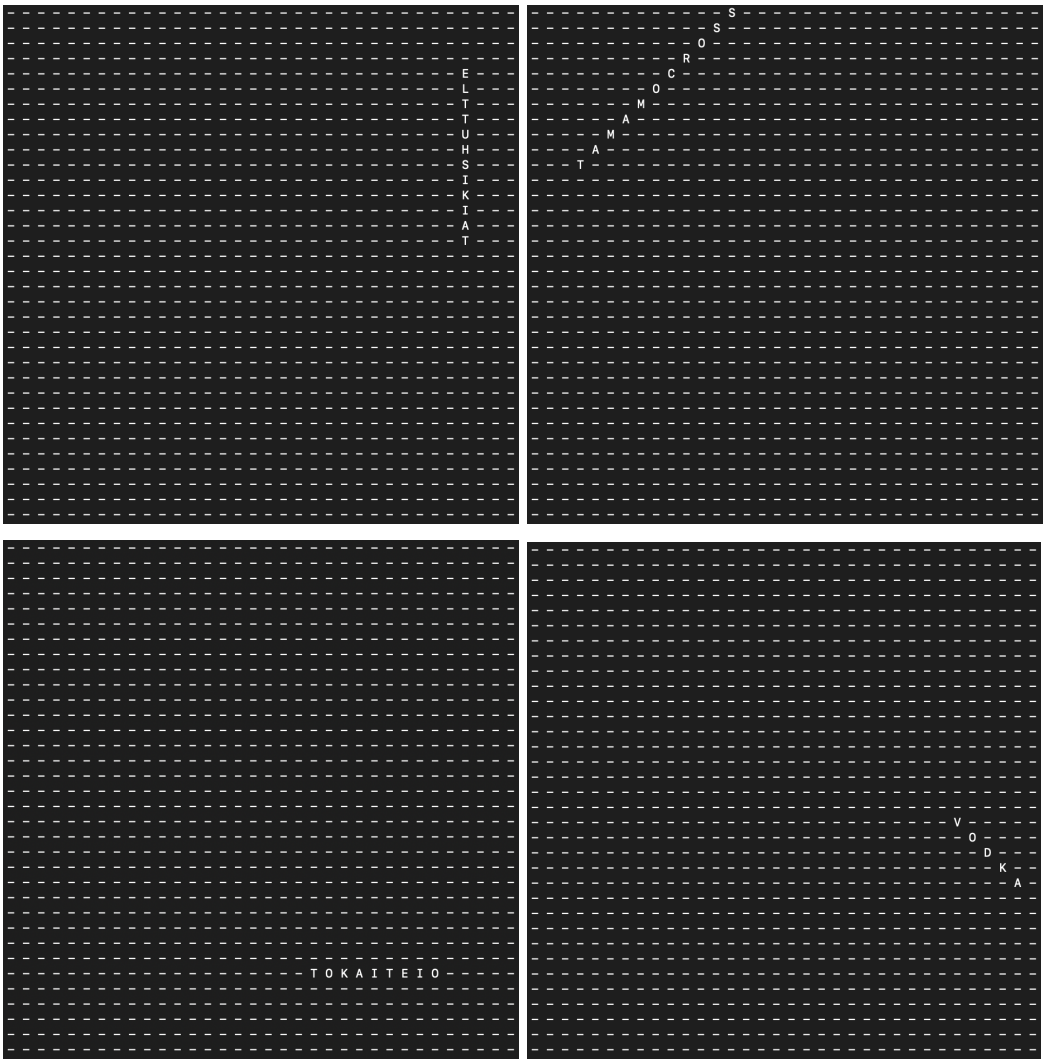

SMARTFALCON

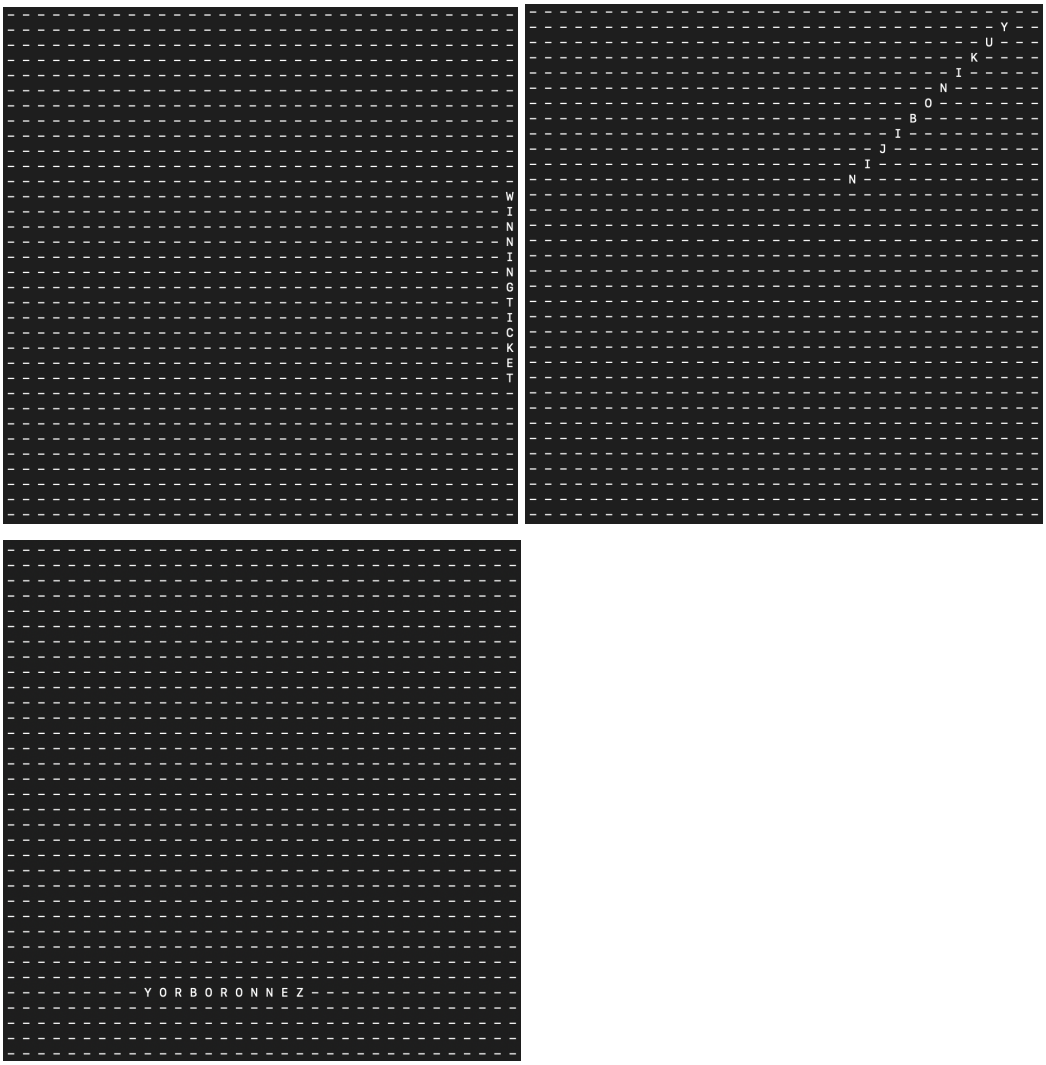

S
P
E
C
I
A
L
W
E
E
K


S
U
P
E
R
C
R
E
E
K


FLODURILOBMYS


O
A
R
E
P
O
M
T


ELTTUHSIKIAT
SSORCOMAMAT
TOKAITEIO
VODKA



D. Alamat *Source Code*

https://github.com/irsyadazka/wordsearchpuzzle

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.	√	