

Laporan Tugas Kecil 1
IF2211 Strategi Algoritma
Penyelesaian *Word Search Puzzle* dengan Algoritma *Brute Force*
Semester II Tahun 2021/2022



Disusun oleh:

Jevant Jedidia Augustine
13520133

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2022

Algoritma *Brute Force*

Untuk menyelesaikan permasalahan *word search puzzle*, digunakan algoritma *brute force*. *Puzzle* huruf yang berukuran $M \times N$ akan disimpan di dalam bentuk matriks, kemudian akan dicari semua kata pada daftar kata satu persatu.

Iterasi kemudian dilakukan pada matriks huruf acak. Apabila ditemukan huruf yang sama dengan huruf pertama dari kata yang dipilih, dilakukan iterasi pada 8 arah, kiri atas, atas, kanan atas, kiri, kanan, kiri bawah, bawah, dan kanan bawah. Diperiksa terlebih dahulu jika dilakukan iterasi pada arah tertentu sebanyak huruf pada kata yang dipilih, iterasi akan *out of bounds* atau tidak. Jika iya, iterasi arah dilanjutkan ke arah selanjutnya, jika tidak, pada arah tersebut dilakukan iterasi untuk membandingkan apakah huruf pada arah yang diiterasi sama dengan huruf pada kata yang dipilih. Bila setiap huruf sama, pencarian kata yang dipilih selesai, tetapi bila tidak, iterasi arah dilanjutkan ke arah yang berikutnya. Saat semua 8 arah telah diperiksa dan tidak ada arah yang memiliki solusi dari kata yang dipilih, iterasi pada matriks huruf acak akan dilakukan ke huruf selanjutnya.

Source Code Program

Fungsi display2DVector

```
void display2DVector(vector<vector<char>> vec){  
    for (int i=0;i<vec.size();i++){  
        for (int j=0;j<vec[0].size();j++){  
            cout << vec[i][j];  
            if (j < vec[0].size()-1){  
                cout << " ";  
            }  
        }  
        cout << endl;  
    }  
}
```

Fungsi findSolution

```
void findSolution (vector<vector<char>> puzzle, int row, int col, int length, int direction){  
    vector<vector<char>> solution(puzzle.size(), vector<char>(puzzle[0].size(), '-'));  
    switch (direction){  
        case 1:  
            while (length > 0){  
                solution[row][col] = puzzle[row][col];  
                row--;  
                col--;  
                length--;  
            }  
            break;  
        case 2:  
            while (length > 0){  
                solution[row][col] = puzzle[row][col];  
                row--;  
                length--;  
            }  
            break;  
        case 3:  
            while (length > 0){  
                solution[row][col] = puzzle[row][col];  
                row--;  
                col++;  
                length--;  
            }  
            break;  
        case 4:  
            while (length > 0){  
                solution[row][col] = puzzle[row][col];  
                col--;  
                length--;  
            }  
            break;  
    }
```

```
case 6:
    while (length > 0){
        solution[row][col] = puzzle[row][col];
        col++;
        length--;
    }
    break;
case 7:
    while (length > 0){
        solution[row][col] = puzzle[row][col];
        row++;
        col--;
        length--;
    }
    break;
case 8:
    while (length > 0){
        solution[row][col] = puzzle[row][col];
        row++;
        length--;
    }
    break;
case 9:
    while (length > 0){
        solution[row][col] = puzzle[row][col];
        row++;
        col++;
        length--;
    }
    break;
}
display2DVector(solution);
}
```

Fungsi wordSearch

```
void wordSearch (vector<vector<char>> puzzle, string word, int *ctr) {
    bool found = false;
    int length = word.length(), row, col, len, direction, i, j;

    i = 0;
    while ((i < puzzle.size()) and !found){
        j = 0;
        while ((j < puzzle[0].size()) and !found){
            *ctr += 1;
            if (puzzle[i][j] == word[0]){
                if ((j+1-length >= 0) and (i+1-length >= 0) and !found){
                    found = true;
                    direction = 1;
                    row = i;
                    col = j;
                    for (int len=0; len<length; len++){
                        *ctr += 1;
                        if (puzzle[row][col] != word[len]){
                            found = false;
                            direction = 0;
                            break;
                        }
                    }
                    else{
                        row -= 1;
                        col -= 1;
                    }
                }
            }
            if ((i+1-length >= 0) and !found){
                found = true;
                direction = 2;
                row = i;
                col = j;
                for (int len=0; len<length; len++){
                    *ctr += 1;
                    if (puzzle[row][col] != word[len]){
                        found = false;
                        direction = 0;
                        break;
                    }
                }
                else{
                    row -= 1;
                }
            }
        }
        if ((i+1-length >= 0) and (j+length-1 < puzzle[0].size()) and !found){
            found = true;
            direction = 3;
            row = i;
            col = j;
            for (int len=0; len<length; len++){
                *ctr += 1;
                if (puzzle[row][col] != word[len]){
                    found = false;

```

```

        direction = 0;
        break;
    }
    else{
        row -= 1;
        col += 1;
    }
}
}
if ((j+1-length >= 0) and !found){
    found = true;
    direction = 4;
    row = i;
    col = j;
    for (int len=0; len<length; len++){
        *ctr += 1;
        if (puzzle[row][col] != word[len]){
            found = false;
            direction = 0;
            break;
        }
        else{
            col -= 1;
        }
    }
}
if ((j+length-1 < puzzle[0].size()) and !found){
    found = true;
    direction = 6;
    row = i;
    col = j;
    for (int len=0; len<length; len++){
        *ctr += 1;
        if (puzzle[row][col] != word[len]){
            found = false;
            direction = 0;
            break;
        }
        else{
            col += 1;
        }
    }
}
if ((j+1-length >= 0) and (i+length-1 < puzzle.size()) and !found){
    found = true;
    direction = 7;
    row = i;
    col = j;
    for (int len=0; len<length; len++){
        *ctr += 1;
        if (puzzle[row][col] != word[len]){
            found = false;
            direction = 0;
            break;
        }
    }
}
}

```

```

        else{
            row += 1;
            col -= 1;
        }
    }
}
if ((i+length-1 < puzzle.size()) and !found){
    found = true;
    direction = 8;
    row = i;
    col = j;
    for (int len=0; len<length; len++){
        *ctr += 1;
        if (puzzle[row][col] != word[len]){
            found = false;
            direction = 0;
            break;
        }
        else{
            row += 1;
        }
    }
}
if ((i+length-1 < puzzle.size()) and (j+length-1 < puzzle[0].size()) and !found){
    found = true;
    direction = 9;
    row = i;
    col = j;
    for (int len=0; len<length; len++){
        *ctr += 1;
        if (puzzle[row][col] != word[len]){
            found = false;
            direction = 0;
            break;
        }
        else{
            row += 1;
            col += 1;
        }
    }
}
j++;
}
i++;
}
findSolution(puzzle,i-1,j-1,length,direction);
}
}

```

Fungsi main

```
int main() {
    char text;
    int ctr = 0;
    string filename, fileloc, key;
    vector<char> rows;
    vector<vector<char>> puzzle;

    using clock = chrono::system_clock;
    using sec = chrono::duration<double>;

    cout << "Input file name (in .txt): ";
    cin >> filename;
    fileloc = "../test/" + filename;
    ifstream ifile;
    ifile.open(fileloc);
    if (!ifile){
        fileloc = "../test/" + filename;
    }

    ifile.close();

    ifstream file(fileloc);
    while (file.get(text)){
        if (text == '\n'){
            if (rows.empty()){
                break;
            }
            puzzle.push_back(rows);
            rows.clear();
        }
        else if (text != ' '){
            rows.push_back(text);
        }
    }

    const auto before = clock::now();

    while(getline(file, key)){
        cout << key << endl;
        wordSearch(puzzle, key, &ctr);
        cout << endl;
    }

    const sec duration = clock::now() - before;
    cout << "Waktu eksekusi program: " << duration.count() << "s" << endl;
    cout << "Jumlah total perbandingan huruf: " << ctr << endl;

    file.close();
}
```


Hasil Percobaan


small1.txt

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

CATALOGUE
CENTRE
COLOUR
DEFENCE
FLAVOUR
HUMOUR
JEWELLERY
LABOUR
LICENCE
LITRE
NEIGHBOUR
PARALYSE
THEATRE
TRAVELLER
YOGHURT

CATALOGUE

CENTRE



ERTNEC

COLOUR

COLOUR

DEFENCE


A 10x10 grid of dots. The word "HUMOUR" is written vertically on the right side of the grid, with each letter aligned with a row of dots. The letters are: H (row 1), U (row 2), M (row 3), O (row 4), R (row 5), U (row 6), R (row 7), O (row 8), U (row 9), and H (row 10).

JEWELLERY

LICENCE

LITRE

NEIGHBOUR
- - - R U O B H G I E N - - -

A large solid black square occupies the right half of the page, positioned next to the title and subtitle.

PARALYSE

THEATRE

TRAVELLER

YOGHURT

Waktu eksekusi program: 0.33556s
Jumlah total perbandingan huruf: 2318

small2.txt

B	P	E	J	R	S	I	L	K	Y	K	C	A	T	S	BLIND
O	Q	J	U	U	J	X	V	X	L	A	I	N	V	X	BLUE
L	S	D	W	L	H	T	T	I	B	R	B	W	E	D	BULL
K	E	A	Q	J	B	I	M	N	L	J	F	Y	F	N	CAT
Z	N	T	Q	P	Q	G	O	X	I	T	O	W	Z		FOSSIL
Q	Z	B	I	B	Z	E	C	V	N	M	S	G	E	V	HOUND
C	H	D	F	H	E	R	W	B	D	S	S	Q	A	V	LEMON
H	Y	M	C	G	W	Z	T	P	I	M	L	L	S	D	MILK
A	D	D	Z	M	K	J	I	L	L	M	I	N	E	M	PIGEYE
G	R	Q	K	I	M	G	P	L	V	G	T	U	L	F	SILKY
L	I	B	X	P	E	R	U	M	C	M	E	F	W	N	SLITEYE
L	W	U	E	Y	R	B	G	D	E	N	Y	M	O	Y	TIGER
Z	D	M	E	Z	V	R	I	N	O	M	E	L	S	H	WEASEL
X	C	R	E	Z	L	S	D	N	U	O	H	H	T	F	WHITE
Q	H	Z	B	U	M	E	T	F	N	A	B	F	G	C	ZEBRA

BLIND	BLUE	BULL	CAT	FOSSIL
HOUND	LEMON	MILK	PIGEY	SILKY
SLITEYE	TIGER	WEASEL	WHITE	ZEBRA

Waktu eksekusi program: 0.348416s
Jumlah total perbandingan huruf: 2063

small3.txt

WNUHHYTLAMJAYMW	ANTIC
QWEIFNUBPRANKMA	COMICAL
JSRMCNIHSTGERL	DECEPTION
KLTHXUYIHHWCDBG	DUPE
WJPTGFCPOYIGIDE	FUNNY
SOUFRZMOCTEDCFZ	GULLIBLE
UETXIIDGNOERUSE	HOAX
OFNVAWCAUCMOYRO	HOODWINK
ROCEIOEKELPIHNA	HUMOROUS
OONQTHPERLTCXM	JOKER
MPKOOKTWURRIAMAF	PRANK
USZRUIOQBDEBWL	RUSE
HSWDODSALPJLKL	SPOOF
HBBNYIETWMEEOE	STUNT
QZSDVCCRTNUTSSJ	TRICKERY

ANTIC	COMICAL	DECEPTION	DUPE	FUNNY
A	C	D		Y
N	O	E		N
T	M	C		N
I	I	P	E	U
C	A	E		F

[illegible]

```
Waktu eksekusi program: 0.608647s
Jumlah total perbandingan huruf: 2480
```

medium1.txt

N P V N A N S E S U E A V F S Y D U C Y
 E F O O O C I U S M E U N N O I T C A F
 V M V I O P C C U P D U O O P Y F M P Y
 C M T T D T O T N O I T C I F U I S T U
 P O V P I D A C A Y T T P T N E C T I A
 P Y M O C I A N I N I C T I P R F U O R
 I V N R N T O Y L P O F D U S R Y Y N R
 A P E A I L U L R U N O I T O L Y V V O
 E U T I U P P E E D M P V V U V N N N A
 L O C U A P F V N M U D N M U M N V O N
 R R D T N F U P O R T I O N R A C A I O
 A R C D I S A M I Y L A R F T U A P L I
 T S N U I O S M T U P V N I Y D Y E S T
 I U O O U U N C A E A U O P D N P I F C
 O D I N I S V N T C F N E F O F R N D A
 N F T F E T U C S L Y E R I M R V E D O
 A M C L M U O A S C M O T I O N U U C Y
 D M E N T I O N U C S O M E O N N M F E
 P V S R U F M R Y V M F D D L S R I S O
 T L D E C S U P D E F I D F N U M T Y M

ACTION
AUCTION
CAPTION
EDITION
EMOTION
FACTION
FICTION
LOTION
MENTION
MOTION
NATION
NOTION
OPTION
PORTION
POTION

ACTION

NOITCA

AUCTION

A 20x20 grid of dots. The letters A, U, C, T, I, O, N are placed on the grid such that they form a diagonal line from the top-left to the bottom-right. The letters are: A (row 10, col 1), U (row 11, col 2), C (row 12, col 3), T (row 13, col 4), I (row 14, col 5), O (row 15, col 6), and N (row 16, col 7). All other positions in the grid are empty dots.

CAPTION

C
A
P
T
I
O
N

EDITION

EDITION

EMOTION

A 20x20 grid of dots. The word "MONITOR" is spelled out in a larger font in the bottom right corner, with each letter formed by a cluster of dots.

FACTION

[illegible]

[illegible][illegible]

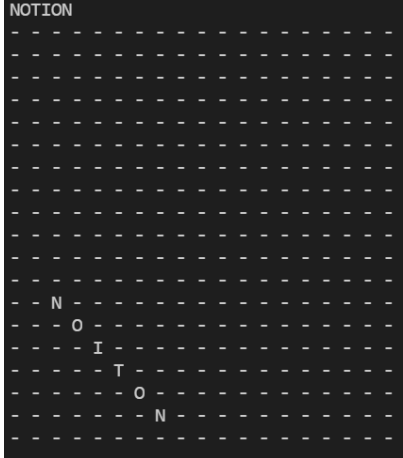
MENTION

MOTION

MOTION

NATION

NOTION



The image shows a black background with a grid of white dots. The word "NOTION" is written in white capital letters at the top left. The letters are positioned such that they align with specific dots in the grid, illustrating the concept of a 'notion' as a point or idea within a structured space.

[illegible]

PORTION

PORTION

[illegible]

```
Waktu eksekusi program: 0.885438s
Jumlah total perbandingan huruf: 4294
```

medium2.txt

Q	P	I	Q	H	H	E	J	A	Y	W	K	S	S	N	A	M	R	E	G		ARABIC
K	Q	J	S	G	S	F	Z	I	D	E	A	F	X	A	T	H	Q	W	R		CHINESE
Z	C	S	A	O	R	I	R	I	N	A	I	L	A	T	I	E	O	E			CZECH
N	L	H	V	P	Y	H	N	M	B	S	D	P	Y	O	S	O	I	H	E		DANISH
Q	K	S	K	F	A	L	T	A	F	F	K	O	C	N	Y	W	H	S	K		DUTCH
F	Z	G	Z	W	R	N	P	D	D	R	I	H	O	T	E	X	N	I	U		ENGLISH
L	X	V	Q	G	U	T	E	H	L	S	I	R	Q	N	T	V	U	N	O		FRENCH
I	C	H	H	Z	Y	F	E	S	E	N	X	V	I	G	V	Q	Y	A	J		GERMAN
T	H	A	I	F	M	Q	N	T	E	W	X	T	E	V	O	N	R	P	F		GREEK
H	K	V	N	J	D	S	M	S	U	J	A	E	W	M	U	V	S	S	H		HEBREW
U	E	B	D	F	A	X	E	Y	F	L	L	L	T	Y	C	X	W	A	L		HINDI
A	U	E	I	G	Z	B	D	F	Q	Z	H	H	O	Q	B	E	E	U	Z		ITALIAN
N	N	R	I	W	S	L	A	C	M	I	U	H	I	J	D	I	V	P	A		JAPANESE
I	J	Q	L	P	S	W	E	R	B	E	H	S	E	I	F	B	Z	A	A		LAO
A	I	H	J	H	G	O	W	O	C	A	Q	K	S	W	Y	Y	T	C	R		LATIN
N	S	C	J	W	S	C	R	L	Z	Z	O	H	N	H	T	O	K	Y	A		
J	E	N	G	K	X	I	R	K	G	L	R	D	J	W	V	A	A	A	B		
C	Z	E	C	H	N	W	L	F	F	P	R	F	S	Y	T	L	H	H	I		
Y	F	R	Z	C	C	Z	I	O	D	H	C	T	U	D	Y	O	F	L	C		
B	Z	F	K	E	G	W	L	V	P	P	E	N	G	L	I	S	H	K	Q		

[illegible]

CHINESE

CZECH

DANISH

DUTCH

HCTUD

The image features a dark gray background with a fine, light gray grid. The word "ENGLISH" is printed in a small, white, sans-serif font at the top left corner. At the bottom center, the words "ENGLISH" are repeated, flanked by hyphens, also in a small, white, sans-serif font.

medium3.txt

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

[illegible]

BREATHE

B
R
E
A
T
H
E

BURN

BURN

CENTERFOLD

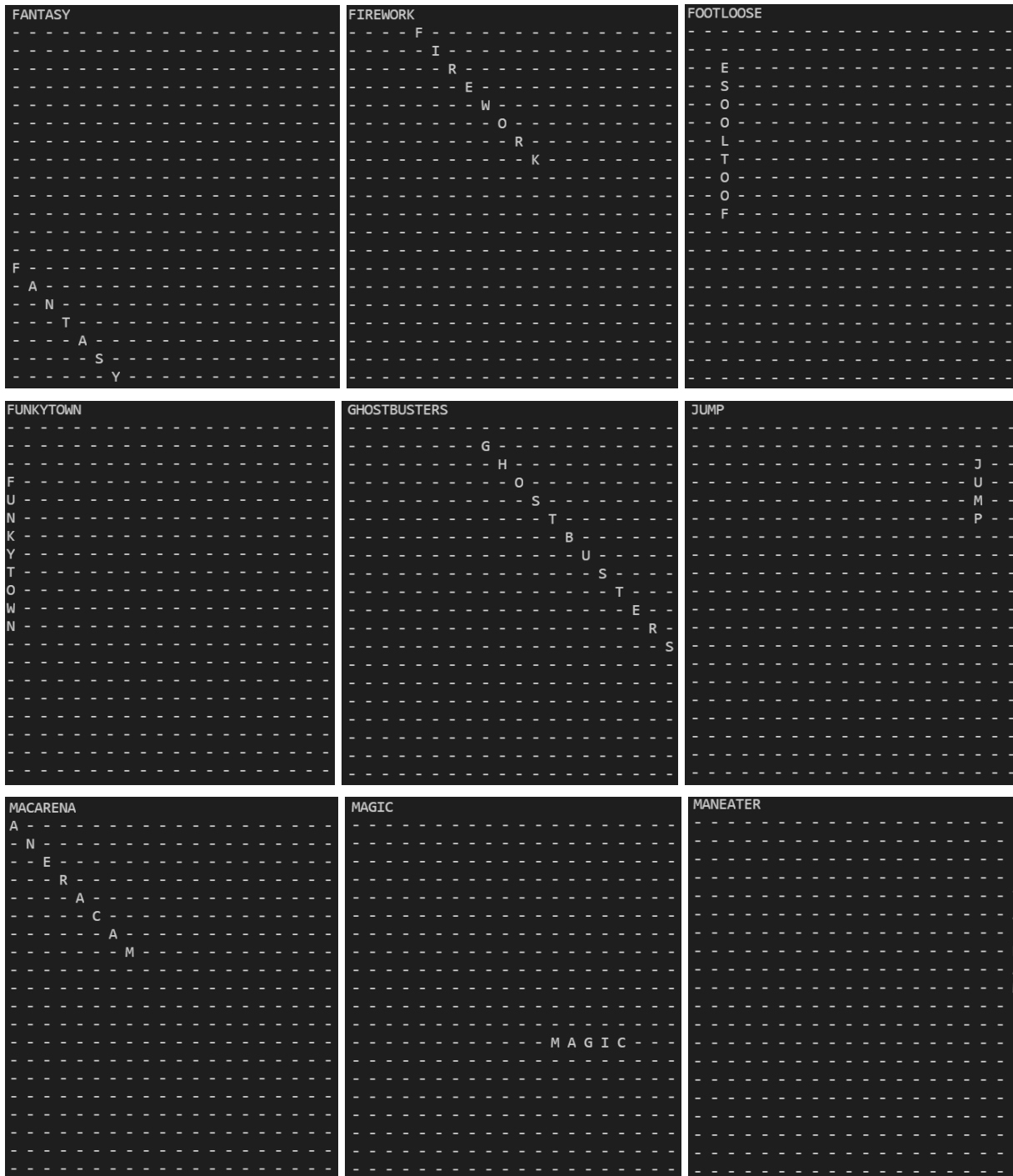


D L O F R E T N E C

[illegible]

FAITH

F
A
I
T
H



Waktu eksekusi program: 1.05923s
Jumlah total perbandingan huruf: 4165

large1.txt

QPEBMWZJJCSAZAJECSZSAOMGOIRITWYDS
GJKNROBHIITVREFNLMEDTHAXNHGUHQFC
NSRPMKZTWACFPXUFFMEBGIYOIRBNEFPQ
KXCDVSTLDLIONYSUXSIEDIXOVRLFIRFNNL
QOOPPNJMVXZZUAHAZORDXDLBHPKBGKVP
XXVZMNSULYPPPTPTTRWAOAHLMSRMJYQRO
SYRXNRECLXWQJHKMSKGKSWJPODOLKOPD
MGZWSYDACMTWRTDVCHEMQSBBZMQPSXZOQ
RILPOHAPWCVOJKEBEYSQKCGAUOAWSDJB
ZXHLMKHYZRDYJBHRJOOGBYLFANHEZQBE
IBLAKJAVIENNZAHEHMLLOVAHFPIFVTZT
ZAPELEQOTMTYQLWTJVPZPVCHEMDFIBOIR
MWLVKHREGRCBZUCOTHOPXGAWQOVVKNNAVD
NWSBOIAEZGRBIFTEIDRSCTINZPNXBZMC
SRVXNYEAJXLIKAVDOMICWXXWCVVHZFR
IKIQRYZIAZOZSTBMRCXCPCBCXZVUKC
BHOWSYDVOFBWQJEXSBWUTYMLVFLRCARB
REUJZSVFKVEVPTRSCMSYIGBWXCOAROLO
NQRWETOJPELKHNEMUKXBSQMQVZVMHMT
QSZTBDJCSUXEGRJOLWEAMZKLNVLIFXS
YPCOHHLLJUTAXIAGTMOOVSWKZDCABPW
BYQQBBTVASZHYMYMKIRGRZUSRNKEULCF
DJHCKLKKPVGTMXNPELNMCKJGHMURITAF
TKJMFBSBNIODEMITHPHDPVBDQUIKIDSFW
VGCULNLLDVQGVPAEQSBEHKDSIRIADZO
NQGGQZKZCROPBZUWRYJYKNCBZDTWTQZ
WSSCVZTNOCEEQYKSNQOAGHBZFTAJZDQR
VGATSYLAFHGDDZZXDPSQFDHYJYHHGKFKX
WMHLXZVCNMHWVJWJSOGZZQAXRZXGVV
ZQPDLDLGLJCKLAMHYCDWWQWLDFIWNNAU
HABDNAOYFVSRZARZACAMHNFZPSLFWJUD
YUOXSNXEHPDMLNAVNAQAQDXXCPCMXXVAQ

APHRODITE
APOLLO
ARES
ATLAS
CHAOS
DIONYSUS
EROS
HADES
HERA
HERMES
IRIS
MORPHEUS
POSEIDON
PROMETHEUS
ZEUS

APHRODITE

```

graph TD
    A --> P
    A --> H
    P --> R
    R --> O
    O --> D
    D --> I
    I --> T
    T --> E
  
```

APOLLO

A diagram showing a sequence of points O , L , O , P , A connected by line segments, illustrating a path in a geometric space.

ARES

A diagram showing a vertical stack of four rectangular blocks labeled S, E, R, and A from top to bottom. The blocks are connected by dashed lines, indicating a sequential process.

ATLAS

S
A
L
T
A

CHAOS

S
O
A
H
C

DIONYSUS

D I O N Y S U S

EROS

E
R
O
S

HADES

S
E
D
A
H

HERA

H
E
R
A

HERMES

S
E
M
R
E
H

IRIS

S I R I

MORPHEUS

M
O
R
P
H
E
U
S

POSEIDON

P
O
S
E
I
D
O
N

PROMETHEUS

P
R
O
M
E
T
H
E
U
S

ZEUS

Z
E
U
S

Waktu eksekusi program: 1.88457s
Jumlah total perbandingan huruf: 9402

large2.txt

YBRLGBXFLWQNLVYQRDCNZJYNEAQJXXDR
ONCNHWETYHHNKSGBNAKALPHALEWHQDNM
SKDZQDVGNGCZULBQCVBSQSUENJPBICTY
PJNTBBGBRZXXQMCAYWCVNCPIBKJZDBBP
COUKBBOZZYPPDUNKACQOWPZVQTURQHPI
CDEEPLYHBLALFDHVFULANKGWFVYKPSVS
MNUJYEFKZVRFFEERWAXPTVWMPKRYSEIU
TGUONYJZOKAVHWHQXXCORLMOTLKACALN
WATHMNNWTOAMUYPZVPKORFAKEIMYLXOG
LYHQGFVNITYXBXPQLBQRVNICQISSIFNH
FZQSPGBZUTELLOLLAKQKWDJGPNMEFLFZ
ATUFSJPIYJPOANYIPMMSVINNQDDYKFF
LOTAAPBNFYFITPAGTLXBKIIIEVUTFQECA
VRCTEPTSLONSEPLVQWNFDBWEYNWAKKZB
FKEJGNIVUTYERACUSTXOTAPTOBZIAIET
EBPUMGJZGRUBKAKSSKUTUMTOSLMHPFND
FJEPNRNGJQXFWMBAKWXCSUTBEAKWPCYY
VLJJOKVLFLEZRTYQYMLYNFBHALWNRRK
EYMWUEUJKJNEVIKLTDFGQCCILLTSZBZMU
AQVZATFVTCSBHORECOYASNCBBYECKJCYE
TOSRFCSKGBDIUDHUEMMPNFFLLZNEDBDO
QQAQXZYMQCFPXWFSXGYMYPNBZCOLECN
GIFZSSWBAQQCCGGMMMBJIWAMHULTTRFQA
YLFMQELUTFQSAJPWONLUOZNTOFXOCTN
OVSOKQEQAATERABWUTMGKTZGSBHKKQS
PMKZUPJNHVMUQSRHAFOZFEASIXQAOHOY
MIWNBRBNEQECYQWLAJXHMZNGNJYGYAT
AXPVNORCIEMOTLRSDVBUCBPTBNZKMTWKCGJR
JFJYVRSWOGORCVRUBUUMBYCGKAVHYERRA
BZNWCJSGAGFXFNHLLTDBWXMFLRKPMCNGL
SHPMFHCFCMDPBRFHRCQIOEJNLLOZQVGL
KBOLMTAWSYFSXBZTIWPCZTQGBKJCTBZ

ALPHA
BETA
DELTA
EPSILON
GAMMA
IOTA
KAPPA
LAMBDA
OMEGA
OMICRON
SIGMA
TAU
THETA
UPSILON
ZETA

ALPHA

ALPHA

BETA

DELTA

- A
- T
- L
- E
- D

EPSILON

- E P S I L O N

GAMMA

G
A
M
A

IOTA

I
O
T
A

KAPPA

A
P
P
A
K

LAMBDA

L
A
M
B
D
A

OMEGA

A
G
E
M
O

OMICRON

N O R C I M O

THETA

A
T
E
H
T

SIGMA

S
I
G
M
A

UPSILON

U
P
S
I
L
O
N

TAU

T
A
U

ZETA

A
T
E
Z

Waktu eksekusi program: 1.90698s
Jumlah total perbandingan huruf: 13273

Link Source Code

Drive: <https://drive.google.com/drive/folders/13KtQWf-Wczzl9OXMUZIDuq8EjiOjgUJg?usp=sharing>

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