

TUGAS KECIL 1

IF2211 Strategi Algoritma

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



Dipersiapkan oleh:

Taufan Fajarama Putrawansyah Ruslanali (13520031)

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2022**

ALGORITMA

Pemrosesan Input

Input file akan diproses menjadi dua data, yaitu matriks untuk puzzle dan list untuk daftar kata yang dicari. Program akan menerima input file menjadi satu String yang kemudian dipisahkan menjadi beberapa String berdasarkan *newline* (\n). Kemudian program mencari String yang kosong dalam hal ini adalah baris pembatas antara puzzle dan kata untuk kemudian memasukkan puzzle ke matriks dan kata ke list.

Metode Pencarian dengan Bruteforce

Metode pencarian yang saya lakukan adalah dengan melakukan loop searching pada matriks untuk setiap kata dari list. Misalnya terdapat list kata {AKU, KAMU}, maka program saya akan mencari terlebih dahulu kata "AKU" di dalam puzzle, setelah kata ditemukan dan ditampilkan ke layar, maka program akan kemudian mencari kata "KAMU" di dalam puzzle.

Untuk algoritma bruteforce, saya menggunakan metode pencarian 'mata angin'. Jika huruf pertama dari kata cocok dengan pivot dari searching puzzle, maka akan dilakukan bruteforce untuk pattern matching ke arah timur, jika tidak ditemukan, maka akan mencari ke arah tenggara, hingga terakhir ke arah timur laut, jika dari semua arah tidak cocok, maka program akan melanjutkan pivot searchingnya. Begitu seterusnya hingga ditemukan pattern yang cocok.

Sebagai contoh, jika matriks puzzle adalah

```
A A B C D
M K A M U
A U B C D
```

Pivot searching akan dimulai dari posisi [0][0], di mana karakternya cocok dengan karakter dari kata pertama yang akan dicari yaitu "AKU". Kemudian dilakukan bruteforce ke arah timur, karena karakter selanjutnya tidak cocok, maka akan dilanjutkan ke arah tenggara, lalu karena karakter kedua cocok maka akan diteruskan pada arah tenggara, tetapi karakter ketiga yaitu "B" tidak cocok dengan yang seharusnya yaitu "U", sehingga bruteforce akan dilanjutkan ke arah selatan, karena karakter selanjutnya tidak cocok lagi, maka akan dilanjutkan ke arah barat hingga timur laut (tetapi tidak dijalankan karena tidak ada karakter yang dapat dicocokkan).

Dengan demikian pivot searching akan berlanjut ke posisi [0][1], karena karakter pertama cocok, akan dilakukan bruteforce yang mana dalam hal ini akhirnya pattern ditemukan ke arah selatan, sehingga program akan menampilkan luaran tersebut dan melanjutkan pencarian kata berikutnya yaitu "KAMU" yang dimulai kembali dari posisi pivot searching[0][0]. Begitu seterusnya hingga seluruh kata dapat ditampilkan.

SOURCE CODE

Class Matrix

Class Matrix digunakan untuk menyimpan character word puzzle agar lebih mudah mengaksesnya. Berikut source codenya dalam bahasa Java:

```
public class Matrix {
    char[][] M;
    int rows=0, cols=0;

    public Matrix(int rows, int cols) {
        M = new char[rows][cols];
        this.rows = rows;
        this.cols =cols;
    }

    public Matrix(char[][] M){
        rows=M.length;
        cols=M[0].length;
        this.M =new char[rows][cols];
        for(int i=0;i<rows;i++) {
            for(int j=0;j<cols;j++) {
                this.M[i][j]= M[i][j];
            }
        }
    }
}
```

Class Operations

Class Operations digunakan untuk menerapkan algoritma brute force dalam pattern matching dan menampilkan keluaran dari kata yang ditemukan. Berikut source codenya

```
public class Operations {
    boolean checkEast(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
        int idx=0;
        while (colIdx < puzz.cols) {
            comp++;
            if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
                idx++;
                if (idx == word.length())
                    return true;
                else
                    colIdx++;
            }
            else
                return false;
        }
        return false;
    }

    void printEast(Matrix puzz, String word, int rowIdx, int colIdx) {
        int m=puzz.rows, n=puzz.cols, l=word.length()-1;
        for (int i=0; i<m; i++) {
            for (int j=0; j<n; j++) {
                if ((i == rowIdx) && (j >= colIdx) && (j <= colIdx+l))
                    System.out.print(puzz.M[i][j]);
                else
                    System.out.print("-");

                if (j == n-1)
                    System.out.print("\n");
                else
                    System.out.print(" ");
            }
        }
    }
}
```

```

boolean checkSouthEast(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
    int idx=0;
    while ((rowIdx < puzz.rows) && (colIdx < puzz.cols)) {
        comp++;
        if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
            idx++;
            if (idx == word.length())
                return true;
            else {
                rowIdx++;
                colIdx++;
            }
        }
        else
            return false;
    }
    return false;
}

void printSouthEast(Matrix puzz, String word, int rowIdx, int colIdx) {
    int m=puzz.rows, n=puzz.cols, l=word.length()-1;
    for (int i=0; i<m; i++) {
        for (int j=0; j<n; j++) {
            if ((i-j == rowIdx-colIdx) && (i >= rowIdx) && (i <= rowIdx+1) && (j >= colIdx) && (j <= colIdx+1))
                System.out.print(puzz.M[i][j]);
            else
                System.out.print("-");

            if (j == n-1)
                System.out.print("\n");
            else
                System.out.print(" ");
        }
    }
}

```

```

boolean checkSouth(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
    int idx=0;
    while (rowIdx < puzz.rows) {
        comp++;
        if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
            idx++;
            if (idx == word.length())
                return true;
            else
                rowIdx++;
        }
        else
            return false;
    }
    return false;
}

void printSouth(Matrix puzz, String word, int rowIdx, int colIdx) {
    int m=puzz.rows, n=puzz.cols, l=word.length()-1;
    for (int i=0; i<m; i++) {
        for (int j=0; j<n; j++) {
            if ((j == colIdx) && (i >= rowIdx) && (i <= rowIdx+1))
                System.out.print(puzz.M[i][j]);
            else
                System.out.print("-");

            if (j == n-1)
                System.out.print("\n");
            else
                System.out.print(" ");
        }
    }
}

```

```

boolean checkSouthWest(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
    int idx=0;
    while ((rowIdx < puzz.rows) && (colIdx >= 0)) {
        comp++;
        if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
            idx++;
            if (idx == word.length())
                return true;
            else {
                rowIdx++;
                colIdx--;
            }
        }
        else
            return false;
    }
    return false;
}

void printSouthWest(Matrix puzz, String word, int rowIdx, int colIdx) {
    int m=puzz.rows, n=puzz.cols, l=word.length()-1;
    for (int i=0; i<m; i++) {
        for (int j=0; j<n; j++) {
            if ((i+j == rowIdx+colIdx) && (i >= rowIdx) && (i <= rowIdx+1) && (j >= colIdx-1) && (j <= colIdx))
                System.out.print(puzz.M[i][j]);
            else
                System.out.print("-");

            if (j == n-1)
                System.out.print("\n");
            else
                System.out.print(" ");
        }
    }
}

```

```

boolean checkWest(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
    int idx=0;
    while (colIdx >= 0) {
        comp++;
        if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
            idx++;
            if (idx == word.length())
                return true;
            else
                colIdx--;
        }
        else
            return false;
    }
    return false;
}

void printWest(Matrix puzz, String word, int rowIdx, int colIdx) {
    int m=puzz.rows, n=puzz.cols, l=word.length()-1;
    for (int i=0; i<m; i++) {
        for (int j=0; j<n; j++) {
            if ((i == rowIdx) && (j >= colIdx-1) && (j <= colIdx))
                System.out.print(puzz.M[i][j]);
            else
                System.out.print("-");

            if (j == n-1)
                System.out.print("\n");
            else
                System.out.print(" ");
        }
    }
}

```

```

boolean checkNorthWest(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
    int idx=0;
    while ((rowIdx >= 0) && (colIdx >= 0)) {
        comp++;
        if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
            idx++;
            if (idx == word.length())
                return true;
            else {
                rowIdx--;
                colIdx--;
            }
        }
        else
            return false;
    }
    return false;
}

void printNorthWest(Matrix puzz, String word, int rowIdx, int colIdx) {
    int m=puzz.rows, n=puzz.cols, l=word.length()-1;
    for (int i=0; i<m; i++) {
        for (int j=0; j<n; j++) {
            if ((i-j == rowIdx-colIdx) && (i >= rowIdx-1) && (i <= rowIdx) && (j >= colIdx-1) && (j <= colIdx))
                System.out.print(puzz.M[i][j]);
            else
                System.out.print("-");

            if (j == n-1)
                System.out.print("\n");
            else
                System.out.print(" ");
        }
    }
}

```

```

boolean checkNorth(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
    int idx=0;
    while (rowIdx >= 0) {
        comp++;
        if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
            idx++;
            if (idx == word.length())
                return true;
            else
                rowIdx--;
        }
        else
            return false;
    }
    return false;
}

void printNorth(Matrix puzz, String word, int rowIdx, int colIdx) {
    int m=puzz.rows, n=puzz.cols, l=word.length()-1;
    for (int i=0; i<m; i++) {
        for (int j=0; j<n; j++) {
            if ((j == colIdx) && (i >= rowIdx-1) && (i <= rowIdx))
                System.out.print(puzz.M[i][j]);
            else
                System.out.print("-");

            if (j == n-1)
                System.out.print("\n");
            else
                System.out.print(" ");
        }
    }
}

```

```

boolean checkNorthEast(Matrix puzz, String word, int rowIdx, int colIdx, int comp) {
    int idx=0;
    while ((rowIdx >= 0) && (colIdx < puzz.cols)) {
        comp++;
        if (puzz.M[rowIdx][colIdx] == word.charAt(idx)) {
            idx++;
            if (idx == word.length())
                return true;
            else {
                rowIdx--;
                colIdx++;
            }
        }
        else
            return false;
    }
    return false;
}

void printNorthEast(Matrix puzz, String word, int rowIdx, int colIdx) {
    int m=puzz.rows, n=puzz.cols, l=word.length()-1;
    for (int i=0; i<m; i++) {
        for (int j=0; j<n; j++) {
            if ((i+j == rowIdx+colIdx) && (i >= rowIdx-1) && (i <= rowIdx) && (j >= colIdx) && (j <= colIdx+1))
                System.out.print(puzz.M[i][j]);
            else
                System.out.print("-");

            if (j == n-1)
                System.out.print("\n");
            else
                System.out.print(" ");
        }
    }
}

```

Class Main

Class main adalah tempat untuk menjalankan program, di dalamnya terdapat pemrosesan file input, berikut kodenya

```

import java.io.*;
import java.util.*;

You, 5 hours ago | 1 author (You)
public class main {
    Run | Debug
    public static void main(String[] args) throws IOException {
        Operations ops = new Operations();
        Scanner input = new Scanner(System.in);
        System.out.println("Masukkan nama file ");
        String filename = input.nextLine();
        String text = "";

        try {
            FileReader reader = new FileReader(filename);
            int data = reader.read();
            while(data != -1){
                text += (char) data;
                data = reader.read();
            }
            reader.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        String[] def = text.split("\\r?\\n"); // array default yang menyimpan tiap baris file input
        Matrix puzzle;
        String[] keyword;
        int rows=0, cols=0;
    }
}

```

Berikut kode pengisian matriks puzzle dan list keyword

```
// mencari index baris pembatas matriks puzzle dan list word
int idx = 0;
while (def[idx] != "") {
    idx++;
}

// mengisi matriks puzzle
rows = idx;
cols = def[0].split(" ").length;
puzzle = new Matrix(rows, cols);
for(int i = 0; i < rows; i++){
    String[] temp = def[i].split(" ");
    for(int j = 0; j < cols; j++){
        puzzle.M[i][j] = temp[j].charAt(0);
    }
}

// mengisi list word
int n = def.length - 1 - idx;
keyword = new String[n];
int k = 0;
for (int i = idx+1; i < def.length; i++) {
    keyword[k] = def[i];
    k++;
}
```

Berikut kode pencarian kata serta perhitungan waktu eksekusi program dan jumlah perbandingan

```
// menjalankan word search tiap keyword
long startTime = System.nanoTime();
int compTotal = 0;
for (int i = 0; i < n; i++) {
    long tempTimeIn = System.nanoTime();
    int rPivot=0, cPivot=0;
    int comp = 0;
    String findNow = keyword[i];
    boolean wordFound = false;

    while ((rPivot < puzzle.rows) && (cPivot < puzzle.cols) && (!wordFound)) {
        comp++;
        if (puzzle.M[rPivot][cPivot] == findNow.charAt(0)) {
            if (ops.checkEast(puzzle, findNow, rPivot, cPivot, comp)) {
                ops.printEast(puzzle, findNow, rPivot, cPivot);
                long tempTimeOut = System.nanoTime();
                long totalTempTime = tempTimeOut - tempTimeIn;
                System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
                compTotal += comp;
                System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
                System.out.println();
                wordFound = true;
            } else if (ops.checkSouthEast(puzzle, findNow, rPivot, cPivot, comp)) {
                ops.printSouthEast(puzzle, findNow, rPivot, cPivot);
                long tempTimeOut = System.nanoTime();
                long totalTempTime = tempTimeOut - tempTimeIn;
                System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
                compTotal += comp;
                System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
                System.out.println();
                wordFound = true;
            } else if (ops.checkSouth(puzzle, findNow, rPivot, cPivot, comp)) {
                ops.printSouth(puzzle, findNow, rPivot, cPivot);
                long tempTimeOut = System.nanoTime();
                long totalTempTime = tempTimeOut - tempTimeIn;
                System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
                compTotal += comp;
                System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
                System.out.println();
                wordFound = true;
            }
        }
        rPivot++;
        cPivot++;
    }
}
```



```

    } else if (ops.checkSouthWest(puzzle, findNow, rPivot, cPivot, comp)) {
        ops.printSouthWest(puzzle, findNow, rPivot, cPivot);
        long tempTimeOut = System.nanoTime();
        long totalTempTime = tempTimeOut - tempTimeIn;
        System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
        compTotal += comp;
        System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
        System.out.println();
        wordFound = true;
    } else if (ops.checkWest(puzzle, findNow, rPivot, cPivot, comp)) {
        ops.printWest(puzzle, findNow, rPivot, cPivot);
        long tempTimeOut = System.nanoTime();
        long totalTempTime = tempTimeOut - tempTimeIn;
        System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
        compTotal += comp;
        System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
        System.out.println();
        wordFound = true;
    } else if (ops.checkNorthWest(puzzle, findNow, rPivot, cPivot, comp)) {
        ops.printNorthWest(puzzle, findNow, rPivot, cPivot);
        long tempTimeOut = System.nanoTime();
        long totalTempTime = tempTimeOut - tempTimeIn;
        System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
        compTotal += comp;
        System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
        System.out.println();
        wordFound = true;
    } else if (ops.checkNorth(puzzle, findNow, rPivot, cPivot, comp)) {
        ops.printNorth(puzzle, findNow, rPivot, cPivot);
        long tempTimeOut = System.nanoTime();
        long totalTempTime = tempTimeOut - tempTimeIn;
        System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
        compTotal += comp;
        System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
        System.out.println();
        wordFound = true;
    }

```

```

    } else if (ops.checkNorthEast(puzzle, findNow, rPivot, cPivot, comp)) {
        ops.printNorthEast(puzzle, findNow, rPivot, cPivot);
        long tempTimeOut = System.nanoTime();
        long totalTempTime = tempTimeOut - tempTimeIn;
        System.out.println("Waktu untuk menemukan kata (dalam nanosecond): " + totalTempTime);
        compTotal += comp;
        System.out.println("Jumlah perbandingan huruf untuk menemukan kata: " + comp);
        System.out.println();
        wordFound = true;
    } else {
        if (cPivot == cols-1) {
            cPivot = 0;
            rPivot++;
        } else {
            cPivot++;
        }
    }
}
else {
    if (cPivot == cols-1) {
        cPivot = 0;
        rPivot++;
    } else {
        cPivot++;
    }
}
}
}
long endTime = System.nanoTime();
long totalTime = endTime - startTime;
System.out.println("Waktu eksekusi program (dalam nanosecond): " + totalTime);
System.out.println("Jumlah total perbandingan huruf untuk menemukan seluruh kata: " + compTotal);
input.close();
}
}

```

CONTOH INPUT/OUTPUT

Ukuran Small (15x15)

1) small1.txt

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

```
ALCOHOL
BAVARIA
BEER
BREW
DANCING
FESTIVAL
GERMAN
LAGER
PARADE
PRETZELS
PROSIT
SAUSAGE
SPALTER
STEIN
STRUDEL
```

```
Masukkan nama file
../test/small1.txt
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- L - - - - -
- O - - - - -
- H - - - - -
- O - - - - -
- C - - - - -
- L - - - - -
- A - - - - -
- - - - -
- - - - -
Waktu untuk menemukan kata (dalam nanosecond): 32387200
Jumlah perbandingan huruf untuk menemukan kata: 183
```

```
N I E T S - - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
Waktu untuk menemukan kata (dalam nanosecond): 81594300
Jumlah perbandingan huruf untuk menemukan kata: 5
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- L - - - - -
- E - - - - -
- D - - - - -
- U - - - - -
- R - - - - -
- T - - - - -
- S - - - - -
- - - - -
- - - - -
- - - - -
- - - - -
Waktu untuk menemukan kata (dalam nanosecond): 76445500
Jumlah perbandingan huruf untuk menemukan kata: 150
Waktu eksekusi program (dalam nanosecond): 1249806500
Jumlah total perbandingan huruf untuk menemukan seluruh kata: 1456
```

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

```
Masukkan nama file
../test/small12.txt
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - A - - -
- - - V - - -
- - - A - - -
- - - L - - -
- - - A - - -
- - - N - - -
- - - C - - -
- - - H - - -
- - - E - - -
Waktu untuk menemukan kata (dalam nanosecond): 30429000
Jumlah perbandingan huruf untuk menemukan kata: 101
```

IF2211 Strategi Algoritma | 10

3) small3.txt

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

ASPARAGUS
AVOCADO
DARK MOSS
EMERALD
FOREST
HARLEQUIN
JADE
LIME
MINT
OLIVE
PARIS
PERSIAN
PINE
SAGE
SHAMROCK

```
Masukkan nama file
../test/small3.txt
```

```
Waktu untuk menemukan kata (dalam nanosecond): 44297900
Jumlah perbandingan huruf untuk menemukan kata: 58
```

```
Waktu untuk menemukan kata (dalam nanosecond): 78858700
Jumlah perbandingan huruf untuk menemukan kata: 89
```

```
Waktu untuk menemukan kata (dalam nanosecond): 65183900
Jumlah perbandingan huruf untuk menemukan kata: 33
```

```
Waktu eksekusi program (dalam nanosecond): 1139477800
Jumlah total perbandingan huruf untuk menemukan seluruh kata: 1469
```


Ukuran Medium (20x22)

1) medium1.txt

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

```
ADLIB
AMORTIZE
CAGEY
CONGESTION
DARTS
DECLARING
EMPLOY
FINISH
GIZZARD
GREW
NETWORK
NOTEPAD
OUTFACE
PRESERVING
PUNTED
SORTING
SURVEY
THINKER
UNBEKNOWN
WHACKO
```

```
../test/medium1.txt
```

```
Waktu untuk menemukan kata (dalam nanosecond): 55834500
Jumlah perbandingan huruf untuk menemukan kata: 241
```

```
Waktu untuk menemukan kata (dalam nanosecond): 106360700
Jumlah perbandingan huruf untuk menemukan kata: 436

Waktu eksekusi program (dalam nanosecond): 2434193200
Jumlah total perbandingan huruf untuk menemukan seluruh kata: 4826
```

2) medium2.txt

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

ADULTS
ANGLED
CENSUS
CHANTING
CHARLES
DUNGEON
EMIGRATING
GALE
IMPANELED
JAGUAR
MEDICATED
PEGGING
POINT
PUDDING
SEQUENCING
SPOUTING
TOWELED
TWITCH
VICIOUS
YOUNGSTER

```
../test/medium2.txt
```

```
Waktu untuk menemukan kata (dalam nanosecond): 120778400
Jumlah perbandingan huruf untuk menemukan kata: 371
```

```
- - - - - Y -  
- - - - - O -  
- - - - - U -  
- - - - - N -  
- - - - - G -  
- - - - - S -  
- - - - - T -  
- - - - - E -  
- - - - - R -  
  
Waktu untuk menemukan kata (dalam nanosecond): 117566000  
Jumlah perbandingan huruf untuk menemukan kata: 16  
  
Waktu eksekusi program (dalam nanosecond): 2681738800  
Jumlah total perbandingan huruf untuk menemukan seluruh kata: 4218
```


3) medium3.txt

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

ABRASION
BETA
CARNAGE
COPSE
CORD
DIZZIEST
EMPHATIC
GOOSE
IMMERSING
INCLINED
MIMIC
NATURALISM
NIGHTMARE
RECOUNT
SEETHE
SHINING
SLICING
SOAKER
TURKISH
USEFULNESS

[illegible][illegible]

Ukuran Large (32x34)

1) large1.txt

ACROBATIC
ADVICE
ARRIVE
BENEFIT
BILLIONAIRE
BLUE
BRAVO
BUCCANEER
CHARMED
CONVERSANT
COOS
COUNTLESS
CRIPPLING
DECADENCE
DEGENERATE
EVAPORATING
EVENLY
EXCELLENT
EXPEDITED
GESTALT
HIVE
HUNK
INFIELD
LAMBDA
LORDS
LYNCHPIN
MIGRAINE
MISSED
OVERDUBBING
PETUNIA
PINGS
PRANK
REPAINTED
RESOLVABLE
SAVED
SECRETIVE
SENSITIZE
SOFTIE
SPUTNIK
SQUEAMISH
STEWARD
STOKED
SUNBLOCK
UPBRAID
URBANIZED
WAFER
WEED
WHITENED

DVOHIVEDONPFBUCCANEERABBQGGJNGDWRMX
EAINUTEPUSSGESTALTFNOPLJLFKWYRMBJKB
SHAKDLRDQJTXRDSNUKGLCGDEKOTSICMNL
SVSOLZRYROORTEJOKQRCITABORCAXDAMKV
ITXHXOKBZMXHVNSMTBDPESQIJOLYRDTUE
MPMKZSECRETIVECHSENAEKXGSKCSPAQECN
CADQCQCVUOOCKPYPAZTZNCENSNUNBLOCKL
DIDKTSTJDOBLKSKQDHIATIAIEIULFFAOMY
DLQBNFJMWVLLFTBPNITVRSZDFLPYGHVGGK
QVYF MUOWOEUYKQECINOXEBKEESLIFAQEQB
XHVMMVAHCPRENDPJXUYJRPIYMDNUERBDABP
EWZNNWLLKYDKCMOBECSVWAWZGQICBNICIWU
LEKIVUFNVUEHSRJNBILLIONAIREEEETNAAW
BNHFARTEVBWPSXPIZIXXNWUZABWYSFFKHU
AETBTSFGJBMIEFHAAIFATQEHBTISIESFOB
VQYIEBEFZIIYNLJQREUUEADVICEERVPWF
LBBAFCCNINQLTZJGARYSDKEIMULQLTFFCJ
OFWRHEGQSGHDNTSIIHAGITLKYXDETIDEPE
SHDJUENSBI RHUYGMLDQYLVGFNKRYOXHGQR
EDVELJVEAFTA OOSSDROLPSWWWHTDJNQTC
RPJRNIRBBODICMUOBTMR AUQKEUHDAIEEEP
QLIPDEPZGAZZZVTGFCOQZXOUWEECTKEANW
KPZRVD TNRAQHFEPEGTTIEMMFEVDATFTQSW
SLESQMIITKWDODGZONI HQH KYAAREEMALYD
IOJVI LMRHUVSEPCCBK EUIHSJOMFDNR IXG
ESRXPMYDAWYMATNASREVNOCFPMZIJKEASU
LUJPVHELIRREEFJJXYKTMOGAFTFBSQNTTL
XYIHABOHSARHNL PWMUUSOUVGWAXEQHEAKO
ERDI ZJJXHQRI BLOOEPZXREZJTQSMJWGSMM
CDQAGFJCCDABVUSFSVYTXEMIQQKBAHEUOP
BFWWIACGQYTTPEXPYKYJLXGGQRURD TDVPS
FUGQYODK MVRCHUNOPXJANIWQMHDVACWZTO

Waktu untuk menemukan kata (dalam nanosecond): 241824300
Jumlah perbandingan huruf untuk menemukan kata: 860
Waktu eksekusi program (dalam nanosecond): 11722661600
Jumlah total perbandingan huruf untuk menemukan seluruh kata: 22907

2) large2.txt

ADDENDA
 ADVERSE
 ASIATIC
 BIOCHEMIST
 BLACKHEAD
 BROTHERHOOD
 COMPLICIT
 CONFIGURED
 DIANA
 DOLPHIN
 EDEMA
 ENACTED
 FASTENER
 FOLLOWED
 FOOT
 HACKNEYED
 HILL
 ICICLE
 IMPORTANTLY
 INVECTIVE
 LACK
 LEOPARD
 MELT
 MONETARISM
 MUGGER
 NOMINATE
 NOVICE
 OBJECTIVITY
 ORIENTATE
 OXEN
 PALPITATED
 PEDESTAL
 POIGNANT
 PURSUING
 QUADRATIC
 RECAPTURE
 REGRESSED
 REPLACING
 SKID
 SOAPY
 SPINAL
 SQUID
 SUSPEND
 TAXIS
 TELESCOPIC
 UNCOUTH
 VIBRATOR
 XRAYs

HWUFI BWTXBWM LCOMLLHMB OIUUVW KXCX RAGN
 ENACTEDSKWO OAYEOLCEUSQH NYIQSVQJGET
 SUZNZYFMQE PNVLMADVERSE O KNSAOQISWGE
 CNWIGLLAGU UETQITDEFULVXTAQUADRATIC
 LLUHWXIPICRTYQAOLRB JIJNPD TLELOEYPM
 DAAPORNANCSAVCAVONDCGMEPELVBFSC TU
 VTTLEADKNAUREGRE SSEDHNGIXURNPBJIKG
 ISYOXYKQBNI IYLTNATROP MIXSLDULPJJGG
 OEPDHSLEBAN SFCCFPVFASTENERZJGOGQDE
 XDUIDOCIEIGMXELCIC IJTD SGWMENLIUOYR
 YEZFDMBAZDONKOZATEWXPBWOTL FCTGFDVQ
 OPO NB DNGIBVCNZISQLWHLCDRJKXTENUNOB
 GROZATLYUMKBHDSINN QKT VZZHEQXFAOOOH
 YKVCAESPINALNELACMXBORIENTATENDVZC
 TRCHOWUQAVGSZKMT PVDFFKKVLYGMDETOIRL
 COMPLICITWVLJCBIXGFFTIAHRNGLNECUKC
 FTAMEPRRSAYZEBICSJEL YCDEVUHM DTGBRS
 NROYUFOQKGGY LXATNTZTIJC DUDYSQUIDSS
 DSZOZOHKMI VTFHMAUWFATAERUBLYPAOSWI
 YHUNFLEHOCYIESIXATNZPTGDAEHKCALBKH
 UHMONLQINIEVOURIGNVTAROTARBIVBVIXM
 LUUQLORLXPYIYWIZDLUTIBYPFDGSI ALECU
 DWOISWOLCOTTCLSFDRINFL LOOBLJKQXVO
 EOTWSEPPQUICHOOC E PVLPOUOEPSREDANIA
 YDDHUDPLG SVEQHCVLEODPEH ZTXWEDNBBD
 EIMKYE EEWQEFJWITACCCIKRRAAPHPEAZDFD
 NTGNE XOI V LQBGRPTWFKXEZEPNZKLMSDJNE
 KUNCOUTHKEW OVUIOPTNH CXUMIMRAAMQSXN
 CUIKBFLWP THJIVUJWFTNCE SKMIDCXKVVM D
 AHS MJYQXSUSPENDFXOPHOPCZOIEIMUKNLA
 HPZGALJXMBPLBYPMRZVDIASJNGDNHMHMML
 VUMIZUMLCQNRDKNB JD B WLOJMSALGOEOAXN

Waktu untuk menemukan kata (dalam nanosecond): 239417700
 Jumlah perbandingan huruf untuk menemukan kata: 142

Waktu eksekusi program (dalam nanosecond): 11863222600
 Jumlah total perbandingan huruf untuk menemukan seluruh kata: 24923

3) large3.txt

ADDICT
ALGEBRAIC
ANNOUNCE
ARKS
AUGUR
BASTED
BOILER
CERTAINLY
CLERICAL
COLLECTIVE
CONSTITUTE
CONVINCING
CUING
DECIPHER
DEIST
EARMARK
EFFACE
FEST
FINGERED
FLEECE
FURBISHING
GARGLING
GENEROSITY
GLUTAMATE
HYPERBOLA
INFORMER
INHIBITOR
INTRICATE
LOOT
MELANCHOLY
MONOTONOUS
MOOR
ORGANIST
PLAY
POUNDING
PRETENSION
RECOGNIZING
REPTILES
SEAMSTRESS
SEATED
SLOWLY
STAMMER
STEPPED
SUBALTERN
TRICK
WADDLE
WEAKNESSES
WILDBEEST

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

```
Waktu untuk menemukan kata (dalam nanosecond): 227878100
Jumlah perbandingan huruf untuk menemukan kata: 186

Waktu eksekusi program (dalam nanosecond): 11971959200
Jumlah total perbandingan huruf untuk menemukan seluruh kata: 27558
```

LAMPIRAN

Link Github/Drive

https://github.com/roastland/Tucil1_13520031

https://drive.google.com/drive/folders/1_jly_Ah-hXBabA8vCLKeeHmfbWvJe-t8?usp=sharing

Check List Asisten

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

Catatan

Contoh output yang dimasukkan dalam laporan hanya 1-3 kata karena outputnya akan terlalu memenuhi laporan jika dimasukkan semua. Program dapat memberikan semua kata yang ditemukan, disarankan untuk menggunakan *command prompt* saja agar dapat melihat seluruh output.