

LAPORAN TUGAS KECIL I

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

Laporan Ini Dibuat Untuk Memenuhi Tugas Perkuliahan

Mata Kuliah Strategi Algoritma (IF2211)

KELAS 02

Dosen : Dr. Nur Ulfa Maulidevi, S.T., M.Sc.



DISUSUN OLEH:

Yakobus Iryanto Prasethio (13520104)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

SEMESTER II TAHUN 2021-2022

Daftar Isi

Daftar Isi	1
BAB I	2
BAB II	3
BAB III	14
BAB IV	23
BAB V	23

BAB I

Langkah Algoritma *Brute Force*

Deskripsi Langkah Algoritma

Algoritma *Brute Force* adalah algoritma yang menggunakan pendekatan *straightforward* untuk menyelesaikan sebuah permasalahan. Algoritma *brute force* umumnya tidak “cerdas” dan tidak mangkus, karena program membutuhkan volume komputasi yang besar dan waktu yang lama dalam penyelesaiannya.

Langkah – Langkah algoritma untuk program Word Search Puzzle Solver:

1. Program membaca setiap elemen matriks puzzle untuk mencari huruf pertama dari kata yang ingin dicari. Misalnya EARTH, maka program akan mencari huruf “E” di dalam matriks puzzle. Apabila program tidak menemukan huruf tersebut di dalam matriks, maka kata tersebut tidak terdapat dalam puzzle.
2. Setelah huruf ditemukan, maka program akan mencari ke setiap arah (seperti arah mata angin) untuk mencari huruf kedua. Misalnya “E” dalam EARTH sudah ditemukan, maka program akan mencari huruf “A”. Jika program menemukan huruf kedua di kanan dari huruf pertama, maka *flag* kanan akan menjadi *true*. Apabila program tidak menemukan huruf kedua di arah manapun, maka kata tidak terdapat dalam matriks itu, sehingga program akan melanjutkan pencarian ke huruf pertama selanjutnya.
3. Program akan mengecek *flag* yang berada dalam status *true* dan mengecek kecocokan huruf selanjutnya dengan kata yang dicari. Misalnya *flag* kiri dari kata EARTH menjadi *true*, maka program akan mengecek apabila setiap huruf selanjutnya sama. Apabila semua huruf sama dan panjang dari kata sama, maka program akan selesai dan mencetak matriks solusi.

BAB II

Source Program dalam Bahasa Java

```
import java.io.*;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.*;

public class wordPuzzle {
    // DECLARING COLOR SPACES
    public static final String ANSI_RESET = "\033[0m";
    public static final String ANSI_MAGENTA = "\033[95m";
    public static final String ANSI_RED = "\033[31m";
    public static final String ANSI_GREEN = "\033[32m";
    public static final String ANSI_YELLOW = "\033[33m";
    public static final String ANSI_BLUE = "\033[34m";
    public static final String ANSI_PURPLE = "\033[35m";
    public static final String ANSI_CYAN = "\033[36m";
    public static final String ANSI_BRIGHT_RED = "\033[91m";

    public static boolean isWithinRange(int index, int lowerBound, int
upperBound) {
        if ((index >= lowerBound) && (index < upperBound)) {
            return true;
        } else {
            return false;
        }
    }

    public static void main (String[] args) {
        int i, j, k, x, y;
        int letterCounter, charIndexX, charIndexY, stepsTaken;
        int ansX, ansY;
        String testCase, chars, path1, path2, fullPath;
        Character currentChar;
        boolean markFound = false;
        boolean checkFlag;
        boolean up, rightup, right, rightdown, down, leftdown, left, leftup;
        boolean finalup, finalrightup, finalright, finalrightdown, finaldown,
finalleftdown, finalleft, finalleftup;
        long startComp, endComp, compDur, startProg, endProg, progDur;
        float elapsedTime, fileDuration;

        // INITIATE BOOLEANS
        up = false;
        rightup = false;
        right = false;
        rightdown = false;
        down = false;
        leftdown = false;
        left = false;
        leftup = false;
        finalup = false;
        finalrightup = false;
        finalright = false;
        finalrightdown = false;
        finaldown = false;
        finalleftdown = false;
        finalleft = false;
        finalleftup = false;

        System.out.println("Please input your file name in the test folder:");
        path1 = "../test/";
        Scanner input = new Scanner(System.in);
        path2 = input.nextLine();
        fullPath = path1 + path2;
        startProg = System.nanoTime();
```

```

        input.close();

        try {
            .....

```

Blok kode dipisah agar dapat terbaca dengan baik

```

try {
    List<String> fullFile = Files.readAllLines(Paths.get(fullPath));
    ArrayList<ArrayList<Character>> game = new
    ArrayList<ArrayList<Character>>();

    // READING MATRIX
    for (String line : fullFile) {
        if (line.trim().isEmpty()) {
            break;
        } else {
            Scanner scanElmt = new Scanner (line);
            ArrayList<Character> col = new ArrayList<Character>();
            while (scanElmt.hasNext()) {
                chars = scanElmt.next().trim();
                col.add(chars.charAt(0));
            }
            game.add(col);
            scanElmt.close();
        }
    }

    // CREATE COLOR MATRIX
    String[][] color_codes = new String[game.size()][game.get(0).size()];
    for (i=0;i<game.size();i++) {
        for(j=0;j<game.get(i).size();j++) {
            color_codes[i][j] = ANSI_RESET;
        }
    }

    // CREATING ANSWER MATRIX
    String[][] answer = new String[game.size()][game.get(0).size()];
    for (i=0;i<game.size();i++) {
        for (j=0;j<game.get(i).size();j++) {
            answer[i][j] = game.get(i).get(j);
        }
    }

    // CREATE COLOR MATRIX
    String[][] color_codes = new String[game.size()][game.get(0).size()];
    for (i=0;i<game.size();i++) {
        for(j=0;j<game.get(i).size();j++) {
            color_codes[i][j] = ANSI_RESET;
        }
    }

    // CREATING ANSWER MATRIX
    String[][] answer = new String[game.size()][game.get(0).size()];
    for (i=0;i<game.size();i++) {
        for (j=0;j<game.get(i).size();j++) {
            answer[i][j] = "-";
        }
    }

    // READ TEST CASES AND CALCULATING
    for (String line : fullFile) {
        stepsTaken = 0;
        if (line.trim().isEmpty()) {
            markFound = true;
            continue;
        }
        if (markFound) {

```

```

startComp = System.nanoTime();
Scanner readCase = new Scanner(line);
testCase = readCase.next();
for (i=0;i<game.size();i++) {
    .....

```

Blok kode dipisah agar dapat terbaca dengan baik

```

for (i=0;i<game.size();i++) {
    for (j=0;j<game.get(0).size();j++) {
        stepsTaken++;
        if (game.get(i).get(j) == testCase.charAt(0)) {
            k = 0;
            x = j; // Letak baris
            y = i; // Letak kolom
            // CHECK EVERY DIRECTION
            // IF CURRENTLY SCANNING TOP LEFT CORNER
            if ((i == 0) && (j == 0)) {
                // Arah kanan
                if (testCase.charAt(k+1) == game.get(y).get(x+1)) {
                    stepsTaken++;
                    right = true;
                }
                // Arah kanan bawah
                if (testCase.charAt(k+1) == game.get(y+1).get(x+1)) {
                    stepsTaken++;
                    rightdown = true;
                }
                // Arah bawah
                if (testCase.charAt(k+1) == game.get(y+1).get(x)) {
                    stepsTaken++;
                    down = true;
                }
                // IF CURRENTLY SCANNING TOP RIGHT CORNER
            } else if ((i == 0) && (j == game.get(i).size()-1)) {
                // Arah bawah
                if (testCase.charAt(k+1) == game.get(y+1).get(x)) {
                    stepsTaken++;
                    down = true;
                }
                // Arah kiri bawah
                if (testCase.charAt(k+1) == game.get(y+1).get(x-1)) {
                    stepsTaken++;
                    leftdown = true;
                }
                // Arah kiri
                if (testCase.charAt(k+1) == game.get(y).get(x-1)) {
                    stepsTaken++;
                    left = true;
                }
                // IF CURRENTLY SCANNING BOTTOM LEFT CORNER
            } else if ((i == game.size()-1) && (j == 0)) {
                // Arah atas
                if (testCase.charAt(k+1) == game.get(y-1).get(x)) {
                    stepsTaken++;
                    up = true;
                }
                // Arah kanan atas
                if (testCase.charAt(k+1) == game.get(y-1).get(x+1)) {
                    stepsTaken++;
                    rightup = true;
                }
                // Arah kanan
                if (testCase.charAt(k+1) == game.get(y).get(x+1)) {
                    stepsTaken++;
                    right = true;
                }
                // IF CURRENTLY SCANNING BOTTOM RIGHT CORNER
            } else if ((i == game.size()-1) && (j == game.get(i).size()-1)) {
                // Arah atas
                if (testCase.charAt(k+1) == game.get(y-1).get(x)) {
                    stepsTaken++;
                    up = true;
                }
                // Arah kiri atas
                if (testCase.charAt(k+1) == game.get(y-1).get(x-1)) {
                    stepsTaken++;
                    leftup = true;
                }
                // Arah kiri
                if (testCase.charAt(k+1) == game.get(y).get(x-1)) {
                    stepsTaken++;
                    left = true;
                }
                // Arah bawah
                if (testCase.charAt(k+1) == game.get(y+1).get(x)) {
                    stepsTaken++;
                    down = true;
                }
                // Arah kanan bawah
                if (testCase.charAt(k+1) == game.get(y+1).get(x+1)) {
                    stepsTaken++;
                    rightdown = true;
                }
            }
        }
    }
}

```

```

} else if ((i == game.size()-1) && (j == game.get(i).size()-1)) {
    // Arah atas
    if (testCase.charAt(k+1) == game.get(y-1).get(x)) {
        stepsTaken++;
        up = true;
    }
    // Arah kiri atas
    if (testCase.charAt(k+1) == game.get(y-1).get(x-1)) {
        stepsTaken++;
        leftup = true;
    }
    // Arah kiri
    if (testCase.charAt(k+1) == game.get(y).get(x-1)) {
        stepsTaken++;
        left = true;
    }
}
// IF CURRENTLY SCANNING TOP ROW
} else if ((i == 0) && (j != 0)) {
    // Arah kanan
    if (testCase.charAt(k+1) == game.get(y).get(x+1)) {
        stepsTaken++;
        right = true;
    }
    // Arah kanan bawah
    if (testCase.charAt(k+1) == game.get(y+1).get(x+1)) {
        stepsTaken++;
        rightdown = true;
    }
    // Arah bawah
    if (testCase.charAt(k+1) == game.get(y+1).get(x)) {
        stepsTaken++;
        down = true;
    }
    // Arah kiri bawah
    if (testCase.charAt(k+1) == game.get(y+1).get(x-1)) {
        stepsTaken++;
        leftdown = true;
    }
    // Arah kiri
    if (testCase.charAt(k+1) == game.get(y).get(x-1)) {
        stepsTaken++;
        left = true;
    }
}
// IF CURRENTLY SCANNING LEFT COLUMN
} else if ((i != 0) && (j == 0)) {
    // Arah atas
    if (testCase.charAt(k+1) == game.get(y-1).get(x)) {
        stepsTaken++;
        up = true;
    }
    // Arah kanan atas
    if (testCase.charAt(k+1) == game.get(y-1).get(x+1)) {
        stepsTaken++;
        rightup = true;
    }
    // Arah kanan
    if (testCase.charAt(k+1) == game.get(y).get(x+1)) {
        stepsTaken++;
        right = true;
    }
    // Arah kanan bawah
    if (testCase.charAt(k+1) == game.get(y+1).get(x+1)) {
        stepsTaken++;
        rightdown = true;
    }
    // Arah bawah
    if (testCase.charAt(k+1) == game.get(y+1).get(x)) {
        stepsTaken++;
    }
}

```

```

        down = true;
    }
    // IF CURRENTLY SCANNING BOTTOM ROW
    } else if ((i == game.size()-1) && (j != 0)) {
        // Arah kiri
        if (testCase.charAt(k+1) == game.get(y).get(x-1)) {
            stepsTaken++;
            left = true;
        }
        // Arah kiri atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x-1)) {
            stepsTaken++;
            leftup = true;
        }
        // Arah atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x)) {
            stepsTaken++;
            up = true;
        }
        // Arah kanan atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x+1)) {
            stepsTaken++;
            rightup = true;
        }
        // Arah kanan
        if (testCase.charAt(k+1) == game.get(y).get(x+1)) {
            stepsTaken++;
            right = true;
        }
    }
    // IF CURRENTLY SCANNING RIGHT COLUMN
    } else if ((i != 0) && (j == game.get(i).size()-1)) {
        // Arah atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x)) {
            stepsTaken++;
            up = true;
        }
        // Arah kiri atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x-1)) {
            stepsTaken++;
            leftup = true;
        }
        // Arah kiri
        if (testCase.charAt(k+1) == game.get(y).get(x-1)) {
            stepsTaken++;
            left = true;
        }
        // Arah kiri bawah
        if (testCase.charAt(k+1) == game.get(y+1).get(x-1)) {
            stepsTaken++;
            leftdown = true;
        }
        // Arah bawah
        if (testCase.charAt(k+1) == game.get(y+1).get(x)) {
            stepsTaken++;
            down = true;
        }
    }
    // IF CURRENTLY SCANNING THE MIDDLE OF THE MATRIX
    } else {
        // Arah atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x)) {
            stepsTaken++;
            up = true;
        }
        // Arah kanan atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x+1)) {
            stepsTaken++;
            rightup = true;
        }
    }
}

```



```

        // Arah kanan
        if (testCase.charAt(k+1) == game.get(y).get(x+1)) {
            stepsTaken++;
            right = true;
        }
        // Arah kanan bawah
        if (testCase.charAt(k+1) == game.get(y+1).get(x+1)) {
            stepsTaken++;
            rightdown = true;
        }
        // Arah bawah
        if (testCase.charAt(k+1) == game.get(y+1).get(x)) {
            stepsTaken++;
            down = true;
        }
        // Arah kiri bawah
        if (testCase.charAt(k+1) == game.get(y+1).get(x-1)) {
            stepsTaken++;
            leftdown = true;
        }
        // Arah kiri
        if (testCase.charAt(k+1) == game.get(y).get(x-1)) {
            stepsTaken++;
            left = true;
        }
        // Arah kiri atas
        if (testCase.charAt(k+1) == game.get(y-1).get(x-1)) {
            stepsTaken++;
            leftup = true;
        }
    }

    // ITERATE THE REST OF CHARACTERS BASED ON FLAG
    checkFlag = true;
    .....

```

Blok kode dipisah agar dapat terbaca dengan baik

```

// ITERATE THE REST OF CHARACTERS BASED ON FLAG
checkFlag = true;
k = 1;
letterCounter = 1;
charIndexX = 0;
charIndexY = 0;
while ((k < testCase.length()) && (checkFlag)) {
    if (up) {
        charIndexX = x;
        charIndexY = y-1;
        if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
            (isWithinRange(charIndexY, 0, game.size()))) {
            if (game.get(y-1).get(x) == testCase.charAt(k)) &&
                (letterCounter != testCase.length()) {
                stepsTaken++;
                letterCounter++;
                y--;
                if (letterCounter == testCase.length()) {
                    finalup = true;
                    up = false;
                }
            }
        } else {
            x = j;
            y = i;
            k = 0;
            letterCounter = 1;
            up = false;
        }
    } else {
        x = j;

```

```

        y = i;
        k = 0;
        letterCounter = 1;
        up = false;
    }
} else if (rightup) {
    charIndexX = x+1;
    charIndexY = y-1;
    if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
        (isWithinRange(charIndexY, 0, game.size())) {
        if ((game.get(y-1).get(x+1) == testCase.charAt(k)) &&
            (letterCounter != testCase.length())) {
            stepsTaken++;
            letterCounter++;
            x++;
            y--;
            if (letterCounter == testCase.length()) {
                finalrightup = true;
                rightup = false;
            }
        } else {
            x = j;
            y = i;
            k = 0;
            letterCounter = 1;
            rightup = false;
        }
    } else {
        x = j;
        y = i;
        k = 0;
        letterCounter = 1;
        rightup = false;
    }
} else if (right) {
    charIndexX = x+1;
    charIndexY = y;
    if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
        (isWithinRange(charIndexY, 0, game.size())) {
        if ((game.get(y).get(x+1) == testCase.charAt(k)) &&
            (letterCounter != testCase.length())) {
            stepsTaken++;
            letterCounter++;
            x++;
            if (letterCounter == testCase.length()) {
                finalright = true;
                right = false;
            }
        } else {
            x = j;
            y = i;
            k = 0;
            letterCounter = 1;
            right = false;
        }
    } else {
        x = j;
        y = i;
        k = 0;
        letterCounter = 1;
        right = false;
    }
} else if (rightdown) {
    charIndexX = x+1;
    charIndexY = y+1;
    if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
        (isWithinRange(charIndexY, 0, game.size())) {

```

```

        if ((game.get(y+1).get(x+1) == testCase.charAt(k)) &&
            (letterCounter != testCase.length())) {
            stepsTaken++;
            letterCounter++;
            x++;
            y++;
            if (letterCounter == testCase.length()) {
                finalrightdown = true;
                rightdown = false;
            }
        } else {
            x = j;
            y = i;
            k = 0;
            letterCounter = 1;
            rightdown = false;
        }
    } else {
        x = j;
        y = i;
        k = 0;
        letterCounter = 1;
        rightdown = false;
    }
} else if (down) {
    charIndexX = x;
    charIndexY = y+1;
    if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
        (isWithinRange(charIndexY, 0, game.size()))) {
        if ((game.get(y+1).get(x) == testCase.charAt(k)) &&
            (letterCounter != testCase.length())) {
            stepsTaken++;
            letterCounter++;
            y++;
            if (letterCounter == testCase.length()) {
                finaldown = true;
                down = false;
            }
        } else {
            x = j;
            y = i;
            k = 0;
            letterCounter = 1;
            down = false;
        }
    } else {
        x = j;
        y = i;
        k = 0;
        letterCounter = 1;
        down = false;
    }
} else if (leftdown) {
    charIndexX = x-1;
    charIndexY = y+1;
    if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
        (isWithinRange(charIndexY, 0, game.size()))) {
        if ((game.get(y+1).get(x-1) == testCase.charAt(k)) &&
            (letterCounter != testCase.length())) {
            stepsTaken++;
            letterCounter++;
            x--;
            y++;
            if (letterCounter == testCase.length()) {
                finalleftdown = true;
                leftdown = false;
            }
        } else {

```

```

        x = j;
        y = i;
        k = 0;
        letterCounter = 1;
        leftdown = false;
    }
} else {
    x = j;
    y = i;
    k = 0;
    letterCounter = 1;
    leftdown = false;
}
} else if (left) {
    charIndexX = x-1;
    charIndexY = y;
    if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
        (isWithinRange(charIndexY, 0, game.size()))) {
        if ((game.get(y).get(x-1) == testCase.charAt(k)) &&
            (letterCounter != testCase.length())) {
            stepsTaken++;
            letterCounter++;
            x--;
            if (letterCounter == testCase.length()) {
                finalleft = true;
                left = false;
            }
        }
    } else {
        x = j;
        y = i;
        k = 0;
        letterCounter = 1;
        left = false;
    }
} else {
    x = j;
    y = i;
    k = 0;
    letterCounter = 1;
    left = false;
}
} else if (leftup) {
    charIndexX = x-1;
    charIndexY = y-1;
    if ((isWithinRange(charIndexX, 0, game.get(i).size())) &&
        (isWithinRange(charIndexY, 0, game.size()))) {
        if ((game.get(y-1).get(x-1) == testCase.charAt(k)) &&
            (letterCounter != testCase.length())) {
            stepsTaken++;
            letterCounter++;
            x--;
            y--;
            if (letterCounter == testCase.length()) {
                finalleftup = true;
                leftup = false;
            }
        }
    } else {
        x = j;
        y = i;
        k = 0;
        letterCounter = 1;
        leftup = false;
    }
} else {
    x = j;
    y = i;
    k = 0;
    letterCounter = 1;

```

```

        leftup = false;
    }
}
if (letterCounter == testCase.length()) {
    endComp = System.nanoTime();
    compDur = endComp - startComp;
    elapsedTime = (float)compDur / (float)1000000;

    System.out.println("===== Word Found! =====");
    System.out.println("Program took " + stepsTaken + " step(s) to find
the word");
    System.out.print("Exited in ");
    System.out.printf("%.5f", elapsedTime);
    System.out.println(" ms");

    ansX = j;
    ansY = i;
    for (i=0;i<testCase.length();i++) {
        if (finalup) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_BLUE;
            ansY--;
        } else if (finalrightup) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_CYAN;
            ansX++;
            ansY--;
        } else if (finalright) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_GREEN;
            ansX++;
        } else if (finalrightdown) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_MAGENTA;
            ansX++;
            ansY++;
        } else if (finaldown) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_PURPLE;
            ansY++;
        } else if (finalleftdown) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_RED;
            ansX--;
            ansY++;
        } else if (finalleft) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_BRIGHT_RED;
            ansX--;
        } else if (finalleftup) {
            currentChar = testCase.charAt(i);
            answer[ansY][ansX] = String.valueOf(currentChar);
            color_codes[ansY][ansX] = ANSI_YELLOW;
            ansX--;
            ansY--;
        }
    }
}

for (i=0;i<game.size();i++) {
    for (j=0;j<game.get(i).size();j++) {

```

```

        System.out.print(color_codes[i][j] + answer[i][j] +
        ANSI_RESET);
        System.out.print(" ");
    }
    System.out.println();

    finalup = false;
    finalrightup = false;
    finalright = false;
    finalrightdown = false;
    finaldown = false;
    finalleftdown = false;
    finalleft = false;
    finalleftup = false;
    checkFlag = false;

```

Closing bracket tidak teratur akibat kode yang terpisah – pisah

```

    }
    k++;
}
}
}
}
}
}

} catch (IOException e) {
    System.out.println("File not found!");
}
endProg = System.nanoTime();
progDur = endProg - startProg;
fileDuration = (float)progDur/(float)1000000;
System.out.println("Program fully exited in " + fileDuration + " ms");
}
}

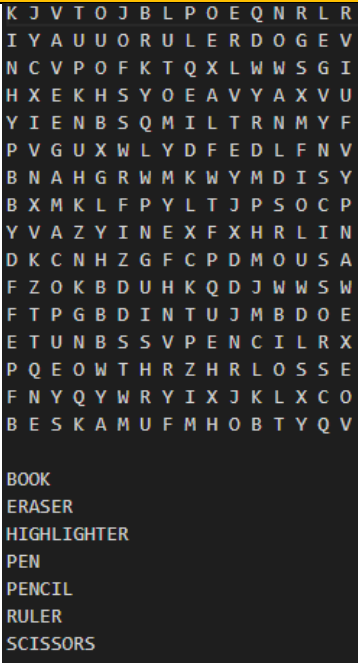
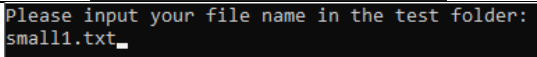
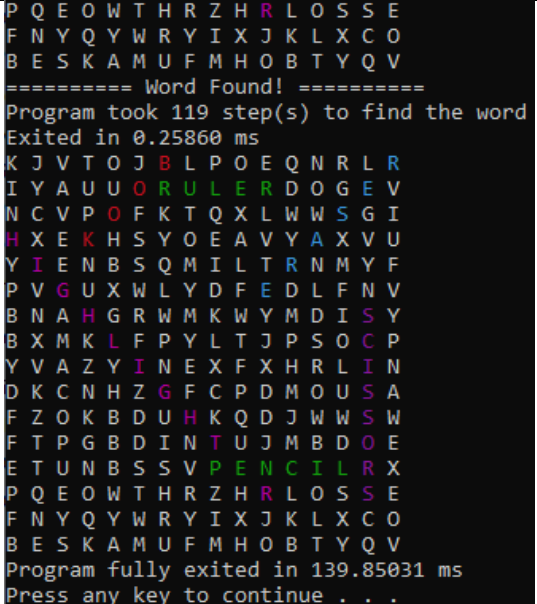
```

- Kode dapat terlihat lebih jelas dalam file wordPuzzle.java

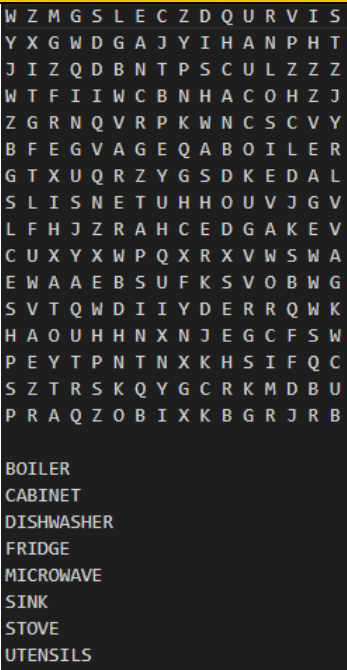
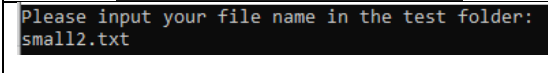
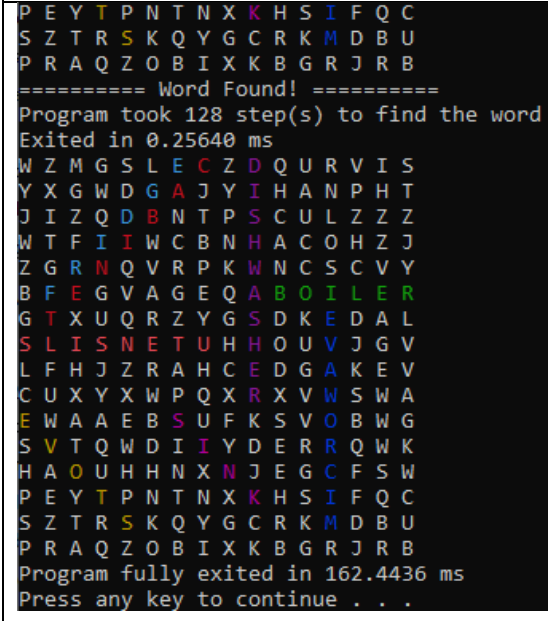
BAB III

Screenshot Input dan Output

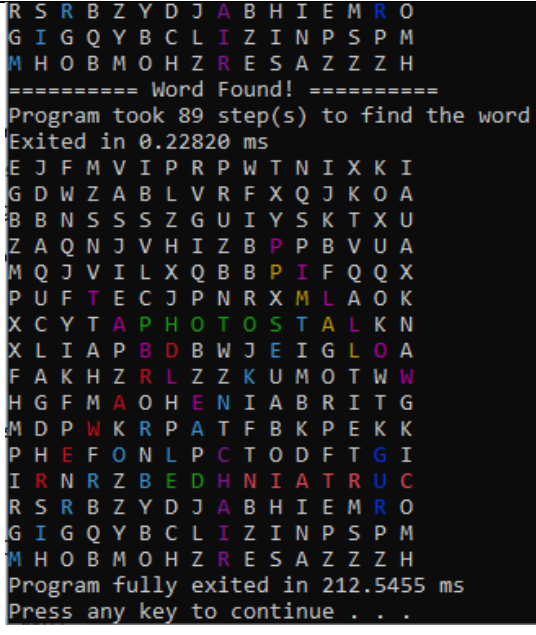
3.1.Ukuran kecil 1

Gambar	Keterangan
	<p>Isi file small1.txt.</p> <p>Tema dari word puzzle adalah nama beberapa alat tulis kantor.</p>
	<p>Input nama file ke dalam program</p>
	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>

3.2.Ukuran kecil 2

Gambar	Keterangan
 <p>The image shows a 15x15 grid of letters. Below the grid, the following words are listed: BOILER, CABINET, DISHWASHER, FRIDGE, MICROWAVE, SINK, STOVE, and UTENSILS.</p>	<p>Isi file small2.txt.</p> <p>Tema dari word puzzle adalah nama objek yang berada di dapur.</p>
 <p>Please input your file name in the test folder: small2.txt</p>	<p>Input nama file ke dalam program</p>
 <p>The image shows the final output of the program, including the word puzzle grid and the found words. The output is as follows:</p> <pre> P E Y T P N T N X K H S I F Q C S Z T R S K Q Y G C R K M D B U P R A Q Z O B I X K B G R J R B ===== Word Found! ===== Program took 128 step(s) to find the word Exited in 0.25640 ms W Z M G S L E C Z D Q U R V I S Y X G W D G A J Y I H A N P H T J I Z Q D B N T P S C U L Z Z Z W T F I I W C B N H A C O H Z J Z G R N Q V R P K W N C S C V Y B F E G V A G E Q A B O I L E R G T X U Q R Z Y G S D K E D A L S L I S N E T U H H O U V J G V L F H J Z R A H C E D G A K E V C U X Y X W P Q X R X V W S W A E W A A E B S U F K S V O B W G S V T Q W D I I Y D E R R Q W K H A O U H H N X N J E G C F S W P E Y T P N T N X K H S I F Q C S Z T R S K Q Y G C R K M D B U P R A Q Z O B I X K B G R J R B Program fully exited in 162.4436 ms Press any key to continue . . . </pre>	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>

3.3.Ukuran kecil 3

Gambar	Keterangan
 <pre> E J F M V I P R P W T N I X K I G D W Z A B L V R F X Q J K O A B B N S S S Z G U I Y S K T X U Z A Q N J V H I Z B P P B V U A M Q J V I L X Q B B P I F Q Q X P U F T E C J P N R X M L A O K X C Y T A P H O T O S T A L K N X L I A P B D B W J E I G L O A F A K H Z R L Z Z K U M O T W W H G F M A O H E N I A B R I T G M D P W K R P A T F B K P E K K P H E F O N L P C T O D F T G I I R N R Z B E D H N I A T R U C R S R B Z Y D J A B H I E M R O G I G Q Y B C L I Z I N P S P M M H O B M O H Z R E S A Z Z Z H BED BLANKET CHAIR CURTAIN DRAWER LAMP MIRROR PHOTOS PILLOW RUG TABLE </pre>	<p>Isi file small3.txt.</p> <p>Tema dari word puzzle adalah nama nama objek di kamar tidur.</p>
 <pre> Please input your file name in the test folder: small3.txt </pre>	<p>Input nama file ke dalam program</p>
 <pre> R S R B Z Y D J A B H I E M R O G I G Q Y B C L I Z I N P S P M M H O B M O H Z R E S A Z Z Z H ===== Word Found! ===== Program took 89 step(s) to find the word Exited in 0.22820 ms E J F M V I P R P W T N I X K I G D W Z A B L V R F X Q J K O A B B N S S S Z G U I Y S K T X U Z A Q N J V H I Z B P P B V U A M Q J V I L X Q B B P I F Q Q X P U F T E C J P N R X M L A O K X C Y T A P H O T O S T A L K N X L I A P B D B W J E I G L O A F A K H Z R L Z Z K U M O T W W H G F M A O H E N I A B R I T G M D P W K R P A T F B K P E K K P H E F O N L P C T O D F T G I I R N R Z B E D H N I A T R U C R S R B Z Y D J A B H I E M R O G I G Q Y B C L I Z I N P S P M M H O B M O H Z R E S A Z Z Z H Program fully exited in 212.5455 ms Press any key to continue . . . </pre>	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>


3.4.Ukuran sedang 1

Gambar	Keterangan
	<p>Isi file medium1.txt.</p> <p>Tema dari word puzzle adalah nama kota – kota di Indonesia.</p>
	<p>Input nama file ke dalam program</p>
	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>

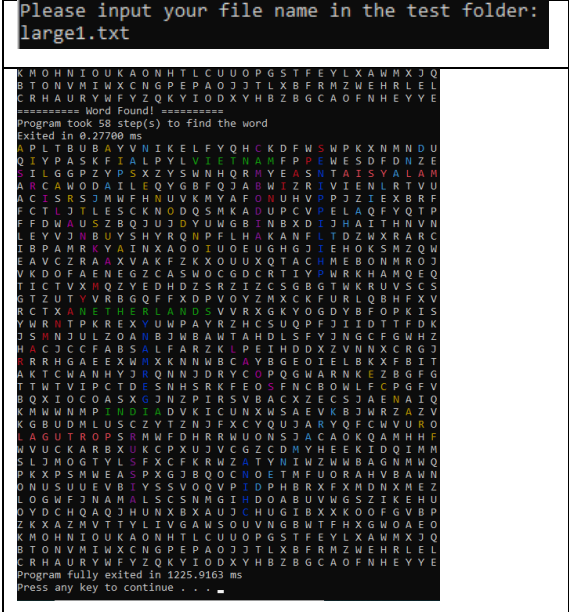
3.5.Ukuran sedang 2

Gambar	Keterangan
 <p>The image shows a 20x20 grid of letters. Below the grid is a list of chemical elements: CALSIUM, CARBON, COBALT, COPPER, HELIUM, HYDROGEN, MAGNESIUM, NICKEL, NITROGEN, OXYGEN, SILICON, and SODIUM.</p>	<p>Isi file medium2.txt.</p> <p>Tema dari word puzzle adalah nama elemen di tabel periodik.</p>
 <p>Please input your file name in the test folder: medium2.txt_</p>	<p>Input nama file ke dalam program</p>
 <p>The image shows the program's output. It displays the word found in the puzzle, the time taken to find it (461 steps, 0.35040 ms), and the full execution time (363.8663 ms). It also shows the word puzzle grid again with the found word highlighted.</p>	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>


3.6.Ukuran sedang 3

Gambar	Keterangan
 <p>The image shows a 20x20 grid of letters. Below the grid is a list of common food items: CANDY, COOKIE, CRISPS, DUMPLING, FRIES, HAMBURGER, HOTDOG, MEATBALL, NOODLE, PASTA, PIZZA.</p>	<p>Isi file medium3.txt.</p> <p>Tema dari word puzzle adalah nama makanan umum.</p>
 <p>Please input your file name in the test folder: medium3.txt_</p>	<p>Input nama file ke dalam program</p>
 <p>The image shows the final output of the program. It displays the word puzzle grid with the found words highlighted in different colors. The output includes the message "Word Found!" and the time taken to find the word (0.23100 ms). The program fully exited in 337.95322 ms.</p>	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>

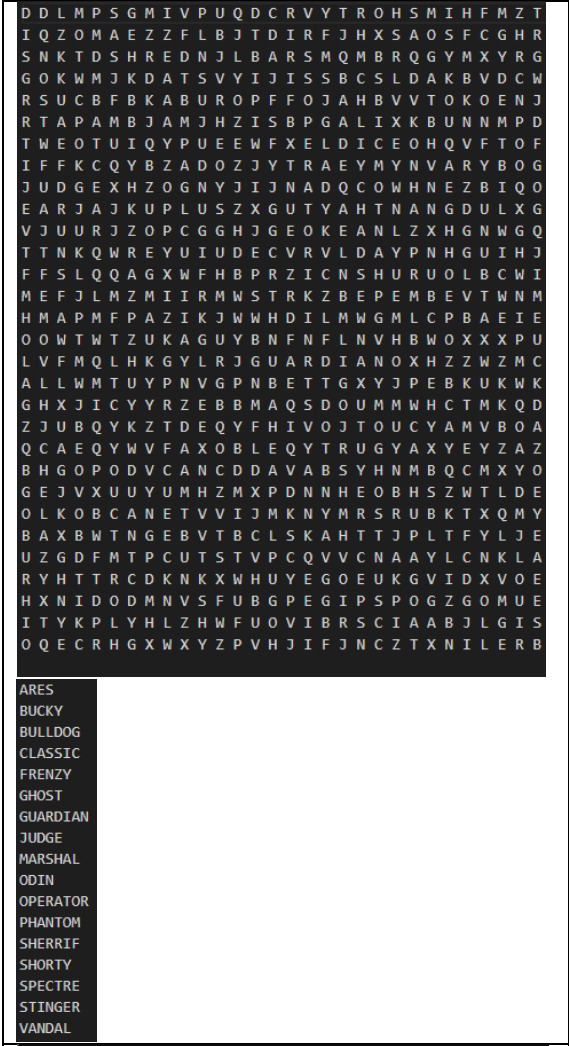

3.7. Ukuran besar 1

Gambar	Keterangan
	<p>Isi file large1.txt.</p> <p>Tema dari word puzzle adalah nama negara di seluruh dunia.</p>
	<p>Input nama file ke dalam program</p> <p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>

3.8. Ukuran besar 2

Gambar	Keterangan
	<p>Isi file large2.txt.</p> <p>Tema dari word puzzle adalah nama brand mobil di seluruh dunia.</p>
	<p>Input nama file ke dalam program</p>
	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>

3.9. Ukuran besar 3

Gambar	Keterangan
	<p>Isi file large3.txt.</p> <p>Tema dari word puzzle adalah nama senjata di game Valorant.</p>
	<p>Input nama file ke dalam program</p>
	<p>Bagian akhir dari eksekusi program.</p> <p>Bagian atas tidak terlihat karena output program akan mencetak array setiap kali sebuah kata ditemukan.</p>

BAB IV

Link Kode Program

Kode program dapat dilihat pada halaman github berikut

<https://github.com/YakobusIP/Tugas-Kecil-Stima-1-Word-Puzzle.git>

BAB V

Daftar Referensi

Munir, Rinaldi. 2022. Algoritma *Brute Force*. Diakses pada 24 Januari 2022, dari

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-\(2022\)-Bag1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf).