

Laporan
Tugas Kecil Strategi Algoritma
Penggunaan Algoritma Brute Force pada Permainan Word Search



Disusun Oleh:
Fransiskus Davin Anwari/13520025
**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2021**

ALGORITMA BRUTE FORCE

Algoritma Brute Force adalah pendekatan yang lempang untuk memecahkan suatu masalah. Algoritma ini biasanya dilakukan dengan melakukan semua tindakan yang dapat dilakukan untuk memecahkan suatu masalah yang diberikan.

Dalam kasus ini kita menggunakan Algoritma Brute Force untuk menyelesaikan suatu puzzle word search. Kita dapat menggunakan perbandingan huruf yang ada pada kata yang dicari dengan karakter pada kumpulan huruf-huruf yang terdapat pada puzzle word search. Hal ini dilakukan berkali-kali sehingga mendapat huruf yang sama pada kata yang dicari dan di puzzle word search, kemudian dilanjutkan dengan melihat huruf selanjutnya pada kata dan pada puzzle. Jika huruf selanjutnya tidak sama maka kita akan membandingkan ulang kembali mulai dari huruf pertama pada kata yang dicari, sehingga kita perlu mendapat perbandingan yang mendapat huruf sama berturut-turut hingga kata yang dicaripun terbentuk.

Program dimulai dengan meminta nama file wordsearch yang ingin diselesaikan, kemudian menampilkan puzzlenya terlebih dahulu secara penuh sebelum menyelesaikannya. Karena pada puzzle word search dapat menemukan kata dari segala arah, maka dibuat kode sendiri untuk masing-masing arah. Dari horizontal kiri ke kanan dan kanan ke kiri, vertical ke atas, vertical ke bawah, diagonal kiri atas ke kanan bawah, diagonal kanan atas ke kiri bawah, diagonal kanan bawah ke kiri atas, dan yang terakhir diagonal kiri bawah ke kanan atas. Program akan melakukan masing-masing tersebut dan hanya berhenti juga semua fungsi sudah dicoba sekali atau kata yang dicari sudah ditemukan.

Program jika telah menemukan kata pada puzzle akan menulis ulang puzzle tersebut menjadi semua character '-' kecuali kata yang telah ditemukan untuk menunjukkan letak kata yang dicari pada puzzle tersebut. Selama pencarian kata juga dilakukan perhitungan banyak perbandingan yang dilakukan sehingga mendapatkan kata yang diinginkan. Program kemudian juga menuliskan banyak waktu yang terlewat selama algoritma word search berjalan.

SOURCE CODE WORDSEARCH PADA JAVA

```
package src;
import java.util.Scanner;
import java.io.*;

public class wordsearch {
    public static boolean findwordH(int banyak, String W, char[][] S){
        boolean ketemu = false;
        int lenw = W.length();
        int lens = S[0].length;
        for (int h=0; h<S.length;h++){
            for (int i=0; i<= lens-lenw;i++){
                int k = 0;
                for (int j=i; j<lens ; j++){
                    if (k < lenw && W.charAt(k)==S[h][j]) {
```

```

        k++;
        banyak++;
    } else if (k==lenw){
        banyak++;
        break;
    } else {
        banyak++;
        break;
    }
}
if (k==lenw){
    System.out.println(W+" Ditemukan");
    ketemu = true;
    break;
}
}
if(ketemu){
    break;
}
}
return ketemu;
}

public static int countwordH(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=0; h<S.length;h++){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            for (int j=i; j<lens ; j++){
                if (k < lenw && W.charAt(k)==S[h][j]) {
                    k++;
                    banyak++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                ketemu = true;
                break;
            }
        }
    }
    if(ketemu){

```

```

        break;
    }
}
return banyak;
}

public static char[][] RewritefindwordH(String W, char[][] S){
    boolean ketemu = false;
    char[][] finalm = new char[S.length][S[0].length];
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=0; h<S.length;h++){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            for (int j=i; j<lens ; j++){
                if (k < lenw && W.charAt(k)==S[h][j] && !ketemu) {
                    k++;
                    finalm[h][j] = S[h][j];
                } else if (k==lenw){
                    finalm[h][j] = '-';
                } else {
                    finalm[h][j] = '-';
                    break;
                }
            }
            if (k==lenw){
                ketemu = true;
                break;
            } else {
                for(int l=0;l<lens;l++){
                    finalm[h][l] = '-';
                }
            }
        }
        if (!ketemu){
            for(int l=0;l<lens;l++){
                finalm[h][l] = '-';
            }
        }
    }
    return finalm;
}

public static boolean findwordDTopLeft(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=0; h<S.length;h++){

```

```

        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            int tambah = 0;
            for (int j=i; j<lens ; j++){
                if (h+tambah < S.length && k < lenw &&
W.charAt(k)==S[h+tambah][j]) {
                    k++;
                    banyak++;
                    tambah++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                System.out.println(W+" Ditemukan");
                ketemu = true;
                break;
            }
        }
        if(ketemu){
            break;
        }
    }
    return ketemu;
}

```

```

public static int countwordDTopLeft(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=0; h<S.length;h++){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            int tambah = 0;
            for (int j=i; j<lens ; j++){
                if (h+tambah < S.length && k < lenw &&
W.charAt(k)==S[h+tambah][j]) {
                    k++;
                    banyak++;
                    tambah++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {

```

```

        banyak++;
        break;
    }
}
if (k==lenw){
    ketemu = true;
    break;
}
}
if(ketemu){
    break;
}
}
return banyak;
}

public static char[][] RewritefindwordDTopLeft(String W, char[][] S){
    boolean ketemu = false;
    char[][] finalm = new char[S.length][S[0].length];
    int lenw = W.length();
    int lens = S[0].length;
    for (int i1=0;i1<S.length;i1++){
        for(int j1=0;j1<S[0].length;j1++){
            finalm[i1][j1] = '-';
        }
    }
    for (int h=0; h<S.length;h++){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            int tambah = 0;
            for (int j=i; j<lens ; j++){
                if (h+tambah < S.length && k < lenw &&
W.charAt(k)==S[h+tambah][j] && !ketemu) {
                    k++;
                    finalm[h+tambah][j] = S[h+tambah][j];
                    tambah++;
                } else{
                    break;
                }
            }
            if (k==lenw){
                ketemu = true;
                break;
            } else {
                for (int i1=0;i1<S.length;i1++){
                    for(int j1=0;j1<S[0].length;j1++){
                        finalm[i1][j1] = '-';
                    }
                }
            }
        }
    }
}

```

```

        }
    }
}
if (ketemu){
    break;
}
}
return finalm;
}

public static boolean findwordDTopRight(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=0; h<S.length;h++){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            int tambah = 0;
            for (int j=i; j>=0 ; j--){
                if (h+tambah < S.length && k < lenw &&
W.charAt(k)==S[h+tambah][j]) {
                    k++;
                    tambah++;
                    banyak++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                System.out.println(W+" Ditemukan");
                ketemu = true;
                break;
            }
        }
        if(ketemu){
            break;
        }
    }
    return ketemu;
}

public static int countwordDTopRight(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();

```

```

        int lens = S[0].length;
        for (int h=0; h<S.length;h++){
            for (int i=lens-1; i>=lenw-1;i--){
                int k = 0;
                int tambah = 0;
                for (int j=i; j>=0 ; j--){
                    if (h+tambah < S.length && k < lenw &&
W.charAt(k)==S[h+tambah][j]) {
                        k++;
                        tambah++;
                        banyak++;
                    } else if (k==lenw){
                        banyak++;
                        break;
                    } else {
                        banyak++;
                        break;
                    }
                }
                if (k==lenw){
                    ketemu = true;
                    break;
                }
            }
            if(ketemu){
                break;
            }
        }
        return banyak;
    }

    public static char[][] RewritefindwordDTopRight(String W, char[][] S){
        boolean ketemu = false;
        char[][] finalm = new char[S.length][S[0].length];
        int lenw = W.length();
        int lens = S[0].length;
        for (int i1=0;i1<S.length;i1++){
            for(int j1=0;j1<S[0].length;j1++){
                finalm[i1][j1] = '-';
            }
        }
        for (int h=0; h<S.length;h++){
            for (int i=lens-1; i>=lenw-1;i--){
                int k = 0;
                int tambah = 0;
                for (int j=i; j>=0 ; j--){
                    if (h+tambah < S.length && k < lenw &&
W.charAt(k)==S[h+tambah][j] && !ketemu) {

```



```

        k++;
        finalm[h+tambah][j] = S[h+tambah][j];
        tambah++;
    } else {
        break;
    }
}
if (k==lenw){
    ketemu = true;
    break;
} else {
    for (int i1=0;i1<S.length;i1++){
        for(int j1=0;j1<S[0].length;j1++){
            finalm[i1][j1] = '-';
        }
    }
}
if (ketemu){
    break;
}
}
return finalm;
}

public static boolean findwordDBotLeft(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=S.length-1; h>=0;h--){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            int tambah = 0;
            for (int j=i; j<lens ; j++){
                if (h-tambah >= 0 && k < lenw && W.charAt(k)==S[h-tambah][j])
                {
                    k++;
                    tambah++;
                    banyak++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                System.out.println(W+" Ditemukan");
            }
        }
    }
}

```

```

        ketemu = true;
        break;
    }
}
if(ketemu){
    break;
}
}
return ketemu;
}

public static int countwordDBotLeft(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=S.length-1; h>=0;h--){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            int tambah = 0;
            for (int j=i; j<lens ; j++){
                if (h-tambah >= 0 && k < lenw && W.charAt(k)==S[h-tambah][j])
{
                    k++;
                    tambah++;
                    banyak++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                ketemu = true;
                break;
            }
        }
        if(ketemu){
            break;
        }
    }
    return banyak;
}

public static char[][] RewritefindwordDBotLeft(String W, char[][] S){
    boolean ketemu = false;
    char[][] finalm = new char[S.length][S[0].length];

```

```

        int lenw = W.length();
        int lens = S[0].length;
        for (int i1=0;i1<S.length;i1++){
            for(int j1=0;j1<S[0].length;j1++){
                finalm[i1][j1] = '-';
            }
        }
        for (int h=S.length-1; h>=0;h--){
            for (int i=0; i<= lens-lenw;i++){
                int k = 0;
                int tambah = 0;
                for (int j=i; j<lens ; j++){
                    if (h-tambah >= 0 && k < lenw && W.charAt(k)==S[h-tambah][j]
&& !ketemu) {
                        k++;
                        finalm[h-tambah][j] = S[h-tambah][j];
                        tambah++;
                    } else{
                        break;
                    }
                }
                if (k==lenw){
                    ketemu = true;
                    break;
                } else {
                    for (int i1=0;i1<S.length;i1++){
                        for(int j1=0;j1<S[0].length;j1++){
                            finalm[i1][j1] = '-';
                        }
                    }
                }
            }
            if (ketemu){
                break;
            }
        }
        return finalm;
    }
}

```

```

public static boolean findwordDBotRight(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=S.length-1; h>=0;h--){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            int tambah = 0;
            for (int j=i; j>=0 ; j--){

```

```

        if (h-tambah >= 0 && k < lenw && W.charAt(k)==S[h-tambah][j])
    {
        k++;
        banyak++;
        tambah++;
    } else if (k==lenw){
        banyak++;
        break;
    } else {
        banyak++;
        break;
    }
    }
    if (k==lenw){
        System.out.println(W+" Ditemukan");
        ketemu = true;
        break;
    }
    }
    if(ketemu){
        break;
    }
    }
    return ketemu;
}

public static int countwordDBotRight(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=S.length-1; h>=0;h--){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            int tambah = 0;
            for (int j=i; j>=0 ; j--){
                if (h-tambah >= 0 && k < lenw && W.charAt(k)==S[h-tambah][j])
    {
        k++;
        banyak++;
        tambah++;
    } else if (k==lenw){
        banyak++;
        break;
    } else {
        banyak++;
        break;
    }
    }
    }
    }
}

```

```

        if (k==lenw){
            ketemu = true;
            break;
        }
    }
    if(ketemu){
        break;
    }
}
return banyak;
}

public static char[][] RewritefindwordDBotRight(String W, char[][] S){
    boolean ketemu = false;
    char[][] finalm = new char[S.length][S[0].length];
    int lenw = W.length();
    int lens = S[0].length;
    for (int i1=0;i1<S.length;i1++){
        for(int j1=0;j1<S[0].length;j1++){
            finalm[i1][j1] = '-';
        }
    }
    for (int h=S.length-1; h>=0;h--){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            int tambah = 0;
            for (int j=i; j>=0 ; j--){
                if (h-tambah >= 0 && k < lenw && W.charAt(k)==S[h-tambah][j]
&& !ketemu) {
                    k++;
                    finalm[h-tambah][j] = S[h-tambah][j];
                    tambah++;
                } else{
                    break;
                }
            }
            if (k==lenw){
                ketemu = true;
                break;
            } else {
                for (int i1=0;i1<S.length;i1++){
                    for(int j1=0;j1<S[0].length;j1++){
                        finalm[i1][j1] = '-';
                    }
                }
            }
        }
    }
    if (ketemu){

```

```

        break;
    }
}
return finalm;
}

public static boolean findwordV(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S.length;
    for (int h=0; h<S[0].length;h++){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            for (int j=i; j<lens ; j++){
                if (k < lenw && W.charAt(k)==S[j][h]) {
                    k++;
                    banyak++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                System.out.println(W+" Ditemukan");
                ketemu = true;
                break;
            }
        }
        if(ketemu){
            break;
        }
    }
    return ketemu;
}
}

```

```

public static int countwordV(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S.length;
    for (int h=0; h<S[0].length;h++){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            for (int j=i; j<lens ; j++){
                if (k < lenw && W.charAt(k)==S[j][h]) {
                    k++;
                    banyak++;
                }
            }
        }
    }
}

```

```

        } else if (k==lenw){
            banyak++;
            break;
        } else {
            banyak++;
            break;
        }
    }
    if (k==lenw){
        ketemu = true;
        break;
    }
}
if(ketemu){
    break;
}
}
return banyak;
}

public static char[][] RewritefindwordV(String W, char[][] S){
    boolean ketemu = false;
    char[][] finalm = new char[S.length][S[0].length];
    int lenw = W.length();
    int lens = S.length;
    for (int h=0; h<S[0].length;h++){
        for (int i=0; i<= lens-lenw;i++){
            int k = 0;
            for (int j=i; j<lens ; j++){
                if (k < lenw && W.charAt(k)==S[j][h] && !ketemu) {
                    k++;
                    finalm[j][h] = S[j][h];
                } else if (k==lenw){
                    finalm[j][h] = '-';
                } else {
                    finalm[j][h] = '-';
                    break;
                }
            }
            if (k==lenw){
                ketemu = true;
                break;
            } else {
                for(int l=0;l<lens;l++){
                    finalm[l][h] = '-';
                }
            }
        }
    }
}

```

```

        if (!ketemu){
            for(int l=0;l<lens;l++){
                finalm[l][h] = '-';
            }
        }
    }
    return finalm;
}

```

```

public static boolean findwordVReverse(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S.length;
    for (int h=0; h<S[0].length;h++){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            for (int j=i; j>=0 ; j--){
                if (k < lenw && W.charAt(k)==S[j][h]) {
                    k++;
                    banyak++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                System.out.println(W+" Ditemukan");
                ketemu = true;
                break;
            }
        }
        if(ketemu){
            break;
        }
    }
    return ketemu;
}

```

```

public static int countwordVReverse(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S.length;
    for (int h=0; h<S[0].length;h++){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;

```



```

        for (int j=i; j>=0 ; j--){
            if (k < lenw && W.charAt(k)==S[j][h]) {
                k++;
                banyak++;
            } else if (k==lenw){
                banyak++;
                break;
            } else {
                banyak++;
                break;
            }
        }
        if (k==lenw){
            ketemu = true;
            break;
        }
    }
    if(ketemu){
        break;
    }
}
return banyak;
}

public static char[][] RewritefindwordVReverse(String W, char[][] S){
    boolean ketemu = false;
    char[][] finalm = new char[S.length][S[0].length];
    int lenw = W.length();
    int lens = S.length;
    for (int h=0; h<S[0].length;h++){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            for (int j=i; j>=0 ; j--){
                if (k < lenw && W.charAt(k)==S[j][h] && !ketemu) {
                    k++;
                    finalm[j][h] = S[j][h];
                } else if (k==lenw){
                    finalm[j][h] = '-';
                } else {
                    finalm[j][h] = '-';
                    break;
                }
            }
        }
        if (k==lenw){
            ketemu = true;
            break;
        } else {
            for(int l=0;l<lens;l++){

```

```

        finalm[l][h] = '-';
    }
}
}
if (!ketemu){
    for(int l=0;l<lens;l++){
        finalm[l][h] = '-';
    }
}
}
return finalm;
}

public static boolean findwordHReverse(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=0; h<S.length;h++){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            for (int j=i; j>=0 ; j--){
                if (k < lenw && W.charAt(k)==S[h][j]) {
                    banyak++;
                    k++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                System.out.println(W+" Ditemukan");
                ketemu = true;
                break;
            }
        }
        if(ketemu){
            break;
        }
    }
    return ketemu;
}

public static int countwordHReverse(int banyak, String W, char[][] S){
    boolean ketemu = false;
    int lenw = W.length();

```

```

    int lens = S[0].length;
    for (int h=0; h<S.length;h++){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            for (int j=i; j>=0 ; j--){
                if (k < lenw && W.charAt(k)==S[h][j]) {
                    banyak++;
                    k++;
                } else if (k==lenw){
                    banyak++;
                    break;
                } else {
                    banyak++;
                    break;
                }
            }
            if (k==lenw){
                ketemu = true;
                break;
            }
        }
        if(ketemu){
            break;
        }
    }
    return banyak;
}

public static char[][] RewritefindwordHReverse(String W, char[][] S){
    boolean ketemu = false;
    char[][] finalm = new char[S.length][S[0].length];
    int lenw = W.length();
    int lens = S[0].length;
    for (int h=0; h<S.length;h++){
        for (int i=lens-1; i>=lenw-1;i--){
            int k = 0;
            for (int j=i; j>=0 ; j--){
                if (k < lenw && W.charAt(k)==S[h][j] && !ketemu) {
                    k++;
                    finalm[h][j] = S[h][j];
                } else if (k==lenw){
                    finalm[h][j] = '-';
                } else {
                    finalm[h][j] = '-';
                    break;
                }
            }
        }
        if (k==lenw){

```

```

        ketemu = true;
        break;
    } else {
        for(int l=0;l<lens;l++){
            finalm[h][l] = '-';
        }
    }
}
if (!ketemu){
    for(int l=0;l<lens;l++){
        finalm[h][l] = '-';
    }
}
return finalm;
}

public static void tulisMatriks(char[][] m) {
    int i,j;
    int brs = m.length;
    int kol = m[0].length;
    for(i=0;i<brs;++i)
    {
        for(j=0;j<kol;++j)
        {
            System.out.print(m[i][j]+" ");
        }
        System.out.println();
    }
}

public static char[][] ReadFile(String filePath) throws IOException {
    Scanner scnr = new Scanner(new BufferedReader(new
FileReader("../test/"+filePath+".txt")));
    char[][] mat = new char[99][99];
    int j=0;
    int k=0;
    while(scnr.hasNextLine()){
        String line = scnr.nextLine();
        if(line==""){
            break;
        }
        String noSpaceStr = line.replaceAll("\\s", "");
        char[] line1 = noSpaceStr.toCharArray();
        mat[j] = line1;
        j++;
        k = line1.length;
    }
}

```

```

        char[][] finmat = new char[j][k];
        for (int i=0;i<j;i++){
            for (int i1=0;i1<k;i1++){
                finmat[i][i1]=mat[i][i1];
            }
        }
        scnr.close();
        return finmat;
    }

    public static String[] ReadFileString(String filePath) throws IOException {
        Scanner scnr = new Scanner(new BufferedReader(new
        FileReader("../test/"+filePath+".txt")));
        while(scnr.hasNextLine()){
            String line = scnr.nextLine();
            if(line==""){
                break;
            }
        }
        String[] Words = new String[99];
        int i = 0;
        while(scnr.hasNextLine()){
            String line = scnr.nextLine();
            String noSpaceStr = line.replaceAll("\\s", "");
            Words[i] = noSpaceStr;
            i++;
        }
        scnr.close();
        String[] finals = new String[i];
        for(int j=0;j<i;j++){
            finals[j]=Words[j];
        }
        return finals;
    }

    public static void main(String[] args) throws Exception {
        Scanner input = new Scanner(System.in);
        System.out.print("Nama file:");
        String filename = input.next();
        String[] Words = ReadFileString(filename);
        char[][] puzzle = ReadFile(filename);
        tulisMatriks(puzzle);
        long Start = System.nanoTime();
        for(int i=0;i<Words.length;i++){
            int banyak = 0;
            boolean found = findwordH(banyak, Words[i],puzzle);
            banyak += countwordH(banyak, Words[i],puzzle);
            if (!found){

```

```

        boolean foundR = findwordHReverse(banyak, Words[i], puzzle);
        banyak += countwordHReverse(banyak, Words[i], puzzle);
        if (foundR){
            tulisMatriks(RewritefindwordHReverse(Words[i], puzzle));
            System.out.println("Telah dilakukan "+banyak+" perbandingan
karakter.");
        } else{
            boolean foundV = findwordV(banyak, Words[i], puzzle);
            banyak += countwordV(banyak, Words[i], puzzle);
            if (foundV){
                tulisMatriks(RewritefindwordV(Words[i], puzzle));
                System.out.println("Telah dilakukan "+banyak+"
perbandingan karakter.");
            } else {
                boolean foundvR = findwordVReverse(banyak, Words[i],
puzzle);
                banyak += countwordVReverse(banyak, Words[i], puzzle);
                if (foundvR){
                    tulisMatriks(RewritefindwordVReverse(Words[i],
puzzle));
                    System.out.println("Telah dilakukan "+banyak+"
perbandingan karakter.");
                } else {
                    boolean foundDTL = findwordDTopLeft(banyak, Words[i],
puzzle);
                    banyak += countwordDTopLeft(banyak, Words[i], puzzle);
                    if (foundDTL){
                        tulisMatriks(RewritefindwordDTopLeft(Words[i],
puzzle));
                        System.out.println("Telah dilakukan "+banyak+"
perbandingan karakter.");
                    } else {
                        boolean foundDTR = findwordDTopRight(banyak,
Words[i], puzzle);
                        banyak += countwordDTopLeft(banyak, Words[i],
puzzle);
                        if (foundDTR){
                            tulisMatriks(RewritefindwordDTopRight(Words[i]
, puzzle));
                            System.out.println("Telah dilakukan "+banyak+"
perbandingan karakter.");
                        } else {
                            boolean foundDBL = findwordDBotLeft(banyak,
Words[i], puzzle);
                            banyak += countwordDBotLeft(banyak, Words[i],
puzzle);
                            if (foundDBL){

```

```

        tulisMatriks(RewritefindwordDBotLeft(Words
[i], puzzle));

        System.out.println("Telah dilakukan
"+banyak+" perbandingan karakter.");
    } else {
        boolean foundDBR =
findwordDBotRight(banyak, Words[i], puzzle);
        banyak += countwordDBotRight(banyak,
Words[i], puzzle);

        if (foundDBR){
            tulisMatriks(RewritefindwordDBotRight(
Words[i], puzzle));

            System.out.println("Telah dilakukan
"+banyak+" perbandingan karakter.");
        }
    }
}
}
}
}
}
if (found){
    tulisMatriks(RewritefindwordH(Words[i], puzzle));
    System.out.println("Telah dilakukan "+banyak+" perbandingan
karakter.");
}
}

long end = System.nanoTime();
long Total = end-Start;
double TotalTime = (double) Total / 1_000_000_000;
input.close();
System.out.println("Kode berjalan selama "+TotalTime+" detik");
}
}

```

SCREENSHOT INPUT DAN OUTPUT

```
Nama file:smallgenshin
I C H I L D E U N Y Z Y
R E B M A J E O N N H R
B X E B A E K D X N O O
E Q I N U A L I I U N Z
C G I A I N E E A H G A
H A T B N N E B O Z L R
O N N U E G G A B I I A
N Y E E D N L G E A U B
G U V A U I N I U B G R
Y U Q I Q I L E N A N A
U L I S A A N U T G N B
N A N O E L L E C T R G
F I S C H L K A E Y A U
T B X I N G Q U I L A A
AMBER Ditemukan
- - - - -
R E B M A - - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
Telah dilakukan 266 perbandingan karakter.

Telah dilakukan 91 perbandingan karakter.
LISA Ditemukan
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- L I S A - - - - -
- - - - -
- - - - -
Telah dilakukan 101 perbandingan karakter.
RAZOR Ditemukan
- - - - -
- - - - - R
- - - - - O
- - - - - Z
- - - - - A
- - - - - R
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
- - - - -
Telah dilakukan 1746 perbandingan karakter.
Kode berjalan selama 0.0701193 detik
```

Input Output untuk smallgenshin.txt

```

Nama file:largebooks
SUOSMYENONASCILOHOCCLAMXUIPLTQGGFTWX
HZCSOMYEOPSOAMNAMRIAHCVVRAGIFYYYHIFYI
TIBBBARRETEPEFEOELATEHTEBTTBQRRREDTTN
TANNEOFGREENENGABLESMVBHBROAAATUTAH
OHTRAETENALPTAERGAOKEOYTNLHAHZAU
KMNHCGCFZNYZLDYZYHTHLHESOCIEEEECNA
IQSCSOUTINGFORBOYSPIEAEIHNBCFEJGD
LTHELITTTLEPRINCEZTHMVTTEKKKEWPCGEI
LNPBMBMHPNBJNILISTTEOCMAACLAUDZNXLCL
ANKKORANEBEEEGMCLLESIIINAEUDDOJHST
MIORAGSWDITYVAOEYSIDSDLSAGVDPZQAI
ONOXRIEMSQHMIRNAHSTGBTGRIEARBRNO
CEBRQEXTKLFQERGERCRIIZEZXZNJGEADN
KITKIEITCSOLDREDEWOHNCBUOLTKVWADA
IEOZMHMTTTTBTNTTEWIEQMOAPBUEZSKER
NEOYNTNTLOOHODSBRSPANTNMWQORESELMY
GNCTUBOAOOPGEBEIIIROHHFVUOTOEHDTXOB
BERSXVGVKFAYESCHORCPOTKDFFSVLTXNJ
ITIEIEEHSRDWRHAPTTDSROHUUKOWMORST
RGKRCTXCXQOERHUEFESFUOEUSFECLZLH
DHCHLNICIWFGEARRLOMXYPCWRPGLRADE
KTOCWAELBIBCEEHCUOYEMMEMIIAKALED
AYROYSGUPCYBHHILRSVEYCCHSNKGHTCA
NFCTFQAQKDFFTHTTMTVOAALOLATODCCSAV
EOYSLRNEGOFHRWOMLENEBLRRZCCNOEEI
AUTPDIAARYOFANNEFRANKJKAWQCJHRXPN
NRTECEWHOMOVEDMYCHEESEXEVJHTPCDC
DRETYPLAHHHLGTSWOLLIWEHTNIDNIWN I
AZBSTHELORDOFTHERINGSALNGOEHVNAC
BBNLLNRWEFILRUOYLAENACUOYWXDUVRO
EAXMDFJBAJJTHECATCHERINTHERYERAD
LGVNETLNOENONEREWEREHTNEHTDNAJWE
THINKANDGROWRICH Ditemukan
Telah dilakukan 36963 perbandingan karakter.
Kode berjalan selama 0.3579624 detik

```

Input Output untuk largebooks.txt


```

- - - - -
- - - - -
- - - - -
- - - A -
- - F -
- - F -
- - I -
- - R -
- - M -
- - A -
- T -
- I -
- V -
- E -
Telah dilakukan 5089 perbandi

```

F
 O
 N
 A
 L
 P

POLICE

STORY

KARUN KARUN

Input Output untuk smallaction.txt

Kode Program

Google Drive: <https://drive.google.com/drive/folders/1PcsefwNA0VhClIplHy-mwx3c0Nm7zuiU?usp=sharing>

Cara menjalankan Program dapat dibaca pada README.md

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan (nosyntax error)	✓	
2. Program berhasil running	✓	
3. Program dapat membaca file masukan dan menuliskan keluaran	✓	
4. Program berhasil menemukan semua kata di dalam puzzle	✓	