

**LAPORAN TUGAS KECIL I**

**PENYELESAIAN *WORD SEARCH PUZZLE***

**DENGAN ALGORITMA *BRUTE FORCE***

Laporan dibuat untuk memenuhi salah satu tugas mata kuliah

IF2211 Strategi Algoritma



Disusun oleh:

**Hilda Carissa Widelia      13520164**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2022**

# **DAFTAR ISI**

<b>DAFTAR ISI</b>	<b>1</b>
<b>Algoritma Brute Force</b>	<b>2</b>
<b>Source Program</b>	<b>3</b>
<b>Screenshot Input dan Output</b>	<b>10</b>
<b>Link Drive Kode Program</b>	<b>69</b>
<b>Checklist</b>	<b>69</b>
<b>Daftar Referensi</b>	<b>69</b>

## Algoritma Brute Force

Dalam pengerjaan tugas kecil ini, algoritma *brute force* sangat diutamakan. Dalam program yang dibuat penulis, setelah menerima input kata yang harus dicari, penulis langsung mencari indeks dari huruf pertama kata yang dicari. Setelah menemukan indeks tersebut, dipanggil fungsi yang berfungsi untuk mencari arah, dapat dibilang fungsi ini juga yang menjadi fungsi utama untuk mencocokkan kata yang dicari dengan huruf dalam puzzle.

Memasuki fungsi ini, atau yang dalam program disebut fungsi `findArah`, pertama ditentukan dulu *value* dari variabel arah. Disini arah direpresentasikan dengan angka 1 hingga 8, dimana angka 1 adalah arah atas, 2 arah kanan atas, 3 arah kanan, dan seterusnya mengikuti arah jarum jam. Jika didapat bahwa ternyata huruf kedua yang ada di puzzle tidak sesuai dengan huruf kedua dari kata yang dicari, maka nilai arah ditambah 1. Sedangkan jika huruf kedua cocok, maka variabel penghitung panjang huruf akan ditambah 1. Saat nilai variabel penghitung panjang huruf tersebut sudah sama dengan panjang kata yang dicari, maka kata sudah ditemukan, dan langsung masuk ke fungsi untuk mencetak jawaban. Namun, jika semua arah sudah dicek dan masih tidak dapat ditemukan katanya, maka akan kembali ke fungsi utama dan mencari indeks lain untuk huruf pertama kata yang dicari.

Setelah kata ditemukan, akan masuk ke fungsi yang bernama `printMatrixJawab`, disini jawaban akan dicetak sesuai dengan arah yang sudah didapatkan sebelumnya. Untuk setiap arah, terdapat batasan yang berbeda agar kata yang dimaksud yang di print. Jika indeks yang sedang diproses tidak sesuai dengan batasan, akan diprint karakter “-”. Selama proses ini, setiap perbandingan kata dihitung dan setelah print matriks solusi, diprint juga jumlah perbandingan kata serta waktu yang dibutuhkan untuk menjalankan semua proses dari awal ini.

Setelah semua di print, program akan melanjutkan ke kata selanjutnya dan mengulang proses dari awal. Apabila sudah tidak ada kata lagi, maka akan dicetak total waktu yang dibutuhkan dari awal setelah input file hingga semua kata ditemukan. Setelah itu program selesai.

## Source Program

```
1 package src;
2 import java.io.*;
3 import java.util.*;
4 import java.nio.file.Path;
5 import java.nio.file.WatchEvent;
6
7 public class App{
8     public static void main(String[] args){
9         Scanner sc = new Scanner(System.in);
10         int row=-1, col=-1; //word length counter and counter for word comparison
11         int[] find = {0,0}; // 1 0 for found, counter per kata
12         char[][] matrix;
13         String search;
14         String namaf;
15         System.out.println("Welcome to Word Search Solver!\nPlease insert the puzzle file name : ");
16         namaf = sc.nextLine();
17         try{
18             namaf = (Path.of("../test", namaf)).toString();
19             File myFile = new File(namaf);
20             Scanner scnn = new Scanner(myFile);
21             int m = 0; //total row
22             int n = 0; //total col
23             String data=scnn.nextLine();
24             while(data.length() != 0){
25                 String array[] = data.split(" ");
26                 n = 0;
27                 for (String i : array){
28                     n++;
29                 }
30                 m++;
31                 data = scnn.nextLine();
32             }
33             scnn.close();
34 }
```

Gambar 1.1 Main program dan menghitung baris dan kolom puzzle

```
34
35         File myFile2 = new File(namaf);
36         Scanner scn = new Scanner(myFile2);
37         matrix = new char[m][n];
38         int i = 0;
39         String data1=scn.nextLine();
40         while(data1.length()!=0){
41             String rows = data1.replaceAll("\\s","");
42             char[] array2 = rows.toCharArray();
43             for (int j = 0; j < n; j++){
44                 matrix[i][j] = array2[j];
45             }
46             i++;
47             data1 = scn.nextLine();
48         }
```

Gambar 1.2 Memasukkan puzzle dari file ke matrix

```

49     long startAllTime = System.nanoTime();
50     while(scن.hasNextLine()){
51         search = scن.nextLine();
52         long startTime = System.nanoTime();
53         System.out.println("Sedang mencari " + search);
54         find[0] = 0;
55         find[1] = 0;
56         // cari huruf pertama
57         for(int k = 0; k < m; k++){
58             for(int l = 0; l < n; l++){
59                 if (matrix[k][l] == search.charAt(0)){
60                     row = k;
61                     col = l;
62                     find = findArah(matrix, m, n, row, col, search, find);
63                     if(find[0] == 1){
64                         break;
65                     }else{
66                         row = -1;
67                         col = -1;
68                     }
69                 }else{
70                     find[1]++;
71                 }
72             }
73             if(find[0] == 1){
74                 break;
75             }
76         }
77         long elapsedTime = System.nanoTime()-startTime;
78         System.out.println("Total execution time in milis: " + elapsedTime/1000000);
79     }
80     scن.close();
81     long finalTime = System.nanoTime()-startAllTime;
82     System.out.println("Total all execution time in milis: " + finalTime/1000000);
83 } catch (FileNotFoundException e){
84     System.out.println("File not found");
85 }
86 sc.close();

```

Gambar 1.3 Mencari kata di puzzle dan setelah menemukan menuliskan total waktu eksekusi

```

90 public static void printMatrixJawab(char[][] matrix, int row, int col, int arah, int n, int m, int wlength, int attCounter){
91     if(arah == 1){ // atas
92         for(int i = 0; i < m; i++){
93             for(int j = 0; j < n; j++){
94                 if(j == col && (i >= row-wlength+1 && i <= row)){
95                     System.out.print(matrix[i][j] + " ");
96                 }else{
97                     System.out.print("- ");
98                 }
99             }
100             System.out.println();
101         }
102     }else if(arah == 2){ // kanan atas
103         int wc = wlength;
104         for(int i = 0; i < m; i++){
105             for(int j = 0; j < n; j++){
106                 if((i <= row && i >= row - wlength+1) && (j >= col && j <= col+wlength) && i+j == row+col && wc>0){
107                     System.out.print(matrix[i][j] + " ");
108                     wc--;
109                 }else{
110                     System.out.print("- ");
111                 }
112             }
113             System.out.println();
114         }
115     }else if(arah == 3){ // kanan
116         for(int i = 0; i < m; i++){
117             for(int j = 0; j < n; j++){
118                 if(i == row && (j >= col && j <= col+wlength-1)){
119                     System.out.print(matrix[i][j] + " ");
120                 }else{
121                     System.out.print("- ");
122                 }
123             }
124             System.out.println();
125         }

```

Gambar 1.4 Print matrix jawaban

```

126         }else if(arah == 4){ // kanan bawah
127             int wc = 0;
128             for(int i = 0; i < m; i++){
129                 for (int j = 0; j < n; j++){
130                     if(i >= row && j >= col && i-j == row-col && wc<=wlength-1){
131                         System.out.print(matrix[i][j] + " ");
132                         wlength--;
133                     }else{
134                         System.out.print("- ");
135                     }
136                 }
137                 System.out.println();
138             }
139
140         }else if(arah == 5){ // bawah
141             for(int i = 0; i < m; i++){
142                 for (int j = 0; j < n; j++){
143                     if(j == col && (i >= row && i <= row+wlength-1)){
144                         System.out.print(matrix[i][j] + " ");
145                     }else{
146                         System.out.print("- ");
147                     }
148                 }
149                 System.out.println();
150             }
151
152         }else if(arah == 6){ // kiri bawah
153             for(int i = 0; i < m; i++){
154                 for(int j = 0; j < n; j++){
155                     if(i + j == row+col && i >= row && wlength > 0 ){
156                         System.out.print(matrix[i][j] + " ");
157                         wlength--;
158                     }else{
159                         System.out.print("- ");
160                     }
161                 }
162                 System.out.println();
163             }

```

Gambar 1.5 Print matrix jawaban

```

163         }else if(arah == 7){ // kiri
164             for(int i = 0; i < m; i++){
165                 for(int j = 0; j < n; j++){
166                     if(i == row && (j >= col-wlength+1 && j <= col)){
167                         System.out.print(matrix[i][j] + " ");
168                     }else{
169                         System.out.print("- ");
170                     }
171                 }
172                 System.out.println();
173             }
174
175         }else if(arah == 8){ // kiri atas
176             int wc = wlength;
177             for(int i = 0; i < m; i++){
178                 for(int j = 0; j < n; j++){
179                     if(i >= row-wlength+1 && j >= col-wlength+1 && i-j == row-col && wc>0){
180                         System.out.print(matrix[i][j] + " ");
181                         wc--;
182                     }else{
183                         System.out.print("- ");
184                     }
185                 }
186                 System.out.println();
187             }
188             System.out.println("Banyak percobaan : " + attCounter);
189         }

```

Gambar 1.6 Print matrix jawaban

```

191 public static int[] findArah(char[][] matrix, int m, int n, int row, int col, String search, int[] find){
192     int wordlength = search.length()-1;
193     if(row>-1 && col>-1){
194         int wlc = 1, arah = 1, o = row, p = col;
195         // o p counter buat index satu satu
196         while(wlc < wordlength+1 && arah < 9){
197             if(arah == 1){ //keatas
198                 if(row < wordlength){
199                     arah++;
200                 }else{
201                     o--;
202                     if(matrix[o][p]==search.charAt(wlc)){
203                         wlc++;
204                     }else{
205                         wlc = 1;
206                         arah++;
207                         o = row;
208                         p = col;
209                     }
210                 }
211                 find[1]++;
212             }else if(arah == 2){ //ke kanan atas
213                 if(n-col < wordlength | row < wordlength){
214                     arah++;
215                 }else{
216                     o--;
217                     p++;
218                     if(matrix[o][p]==search.charAt(wlc)){
219                         wlc++;
220                     }else{
221                         wlc = 1;
222                         arah++;
223                         o = row;
224                         p = col;
225                     }
226                 }
227                 find[1]++;

```

Gambar 1.7 Fungsi mencocokkan kata

```

228             }else if(arah == 3){ //ke kanan
229                 if(n-col < wordlength){
230                     arah++;
231                 }else{
232                     p++;
233                     if(matrix[o][p]==search.charAt(wlc)){
234                         wlc++;
235                     }else{
236                         wlc = 1;
237                         arah++;
238                         o = row;
239                         p = col;
240                     }
241                 }
242                 find[1]++;
243             }else if(arah == 4){ // kanan bawah
244                 if(n-col < wordlength | m-row < wordlength){
245                     arah++;
246                 }else{
247                     o++;
248                     p++;
249                     if(matrix[o][p]==search.charAt(wlc)){
250                         wlc++;
251                     }else{
252                         wlc = 1;
253                         arah++;
254                         o = row;
255                         p = col;
256                     }
257                 }
258                 find[1]++;

```

Gambar 1.8 Fungsi mencocokkan kata



```

259         }else if(arah == 5){ // bawah
260             if(m-row < wordlength){
261                 arah++;
262             }else{
263                 o++;
264                 if(matrix[o][p]==search.charAt(wlc)){
265                     wlc++;
266                 }else{
267                     wlc = 1;
268                     arah++;
269                     o = row;
270                     p = col;
271                 }
272             }
273             find[1]++;
274         }else if(arah == 6){ // bawah kiri
275             if(m-row < wordlength | col < wordlength){
276                 arah++;
277             }else{
278                 o++;
279                 p--;
280                 if(matrix[o][p]==search.charAt(wlc)){
281                     wlc++;
282                 }else{
283                     wlc = 1;
284                     arah++;
285                     o = row;
286                     p = col;
287                 }
288             }
289             find[1]++;

```

Gambar 1.9 Fungsi mencocokkan kata

```

290         }else if(arah == 7){ // kiri
291             if(col < wordlength){
292                 arah++;
293             }else{
294                 p--;
295                 if(matrix[o][p]==search.charAt(wlc)){
296                     wlc++;
297                 }else{
298                     wlc = 1;
299                     arah++;
300                     o = row;
301                     p = col;
302                 }
303             }
304             find[1]++;
305         }else if(arah == 8){ // kiri atas
306             if(col < wordlength | row < wordlength){
307                 arah++;
308             }else{
309                 p--;
310                 o--;
311                 if(matrix[o][p]==search.charAt(wlc)){
312                     wlc++;
313                 }else{
314                     wlc = 1;
315                     arah++;
316                     o = row;
317                     p = col;
318                 }
319             }
320             find[1]++;
321         }

```

Gambar 1.10 Fungsi mencocokkan kata

```

322     }
323     if(arah > 8){
324         find[0] = 0;
325     }else{
326         printMatrixJawab(matrix, row, col, arah, n, m, search.length(), find[1]);
327         find[0] = 1;
328     }
329 }
330 return find;
331 }
332 }

```

Gambar 1.11 Mencetak matrix jika kata ditemukan

[illegible]

```
Total execution time in milis: 156
```

```
Total execution time in milis: 110
```

```
Total execution time in millis: 151
```

MUCH

```
Total execution time in milis: 125
```

A 20x20 grid of dots. The letters M, Y, S, E, L, and F are placed at the following coordinates (row, column): M(10, 10), Y(12, 12), S(14, 14), E(16, 16), L(18, 18), and F(19, 19).

Sedang mencari PICK

P I C K

A 10x10 grid of dots. The letters are placed at the following intersections (row, column) starting from the top-left:

- N: (9, 3)
- E: (9, 4)
- V: (8, 5)
- E: (7, 5)
- S: (7, 6)

Sedang mencari SIX

X I S

Y  
A  
D  
O  
T

A 20x20 grid of dots. The letters are placed at the following intersections (row, column):

- Y: (4, 10), (5, 10), (6, 10), (7, 10), (8, 10), (9, 10), (10, 10), (11, 10), (12, 10), (13, 10), (14, 10), (15, 10), (16, 10), (17, 10), (18, 10), (19, 10)
- R: (4, 12), (5, 12), (6, 12), (7, 12), (8, 12), (9, 12), (10, 12), (11, 12), (12, 12), (13, 12), (14, 12), (15, 12), (16, 12), (17, 12), (18, 12), (19, 12)
- T: (4, 14), (5, 14), (6, 14), (7, 14), (8, 14), (9, 14), (10, 14), (11, 14), (12, 14), (13, 14), (14, 14), (15, 14), (16, 14), (17, 14), (18, 14), (19, 14)

13 - IF2211 - Strategi Algoritma

Sedang mencari ALBANY

[illegible]

A 20x20 grid of dots on a black background. The letters 'A', 'N', 'A', 'P', 'O', 'L', 'I', and 'S' are formed by removing dots from specific rows and columns. The letters are arranged to read 'ANAPOLIS' from left to right, top to bottom.

Sedang mencari AUSTIN

AUSTIN

[illegible]

14 - IF2211 - Strategi Algoritma

[illegible]

A 10x10 grid of dots, intended for drawing a picture related to the text.

---

[illegible]

```
Banyak percobaan : 315
Total execution time in milis:
```

---



- - Y T I C N O S R E F F E J -

```
Total execution time in millis: 64
```

A 20x20 grid of dots. The letters G, N, I, S, N, A, L are arranged in a diagonal pattern from the 10th row to the 16th row, starting from the 2nd column and ending at the 8th column.

```
Total execution time in millis: 40
```

MONTGOMERY

```
Total execution time in millis: 47
```

NASHVILLE

```
Total execution time in millis: 61
```

```
Sedang mencari RALEIGH  
- - - - - R - - - - -  
- - - - - A - - - - -  
- - - - - L - - - - -  
- - - - - E - - - - -  
- - - - - I - - - - -  
- - - - - G - - - - -  
- - - - - H - - - - -  
  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
Banyak percobaan : 22  
Total execution time in milis: 75  
Total all execution time in milis: 694  
Press any key to continue . . .
```

### Gambar 2.2 Input Output Test Case Small 2



[illegible]

Sedang mencari CHESS

CHES

```
Total execution time in milis: 137
```

[illegible]

# CONK

Sedang mencari COVER

R  
E  
V  
O  
C

```
Total execution time in millis: 153
```

[illegible]

Sedang mencari DISTINCT

```
Total execution time in milis: 170
```

A 10x10 grid of dots. The word "DOLAR" is written vertically in the center of the grid, with each letter occupying one row and one column. The letters are: D (row 4, column 4), O (row 5, column 4), L (row 6, column 4), A (row 7, column 4), R (row 8, column 4). The grid is composed of 10 rows and 10 columns of dots.

Sedang mencari DOSES

```
Total execution time in millis: 125
```

- - - - - D U M F O U N D - - - - -

N  
 O  
 I  
 T  
 R  
 O  
 T  
 X  
 E

FACTUALITY



```
- - - - - E V I T A M R I F F A -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
Banyak percobaan : 79  
Total execution time in milis: 78
```

```

Banyak percobaan : 559
Total execution time in milis: 90

```

```
- - - - - Y - - - -  
- - - - - R - - - -  
- - - - - A - - - -  
- - - - - L - - - -  
- - - - - I - - - -  
- - - - - P - - - -  
- - - - - A - - - -  
- - - - - C - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
- - - - -  
Banyak percobaan : 264  
Total execution time in milis: 84
```

```
C O U R S E   O F
```

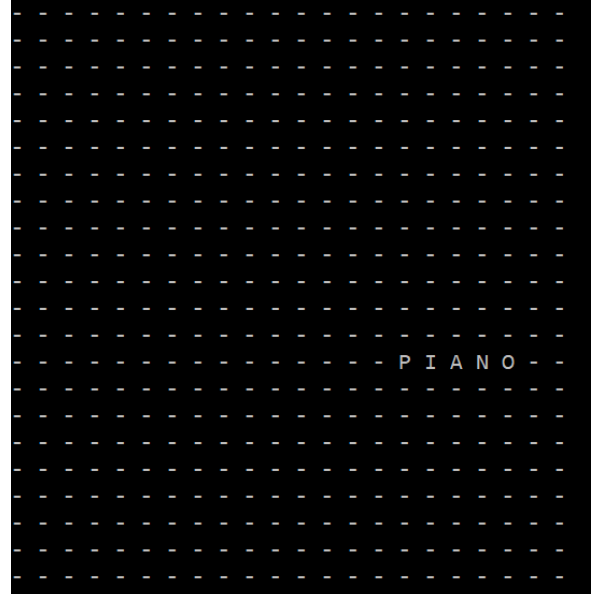
Banyak percobaan : 111  
Total execution time in milis: 96





A 20x20 grid of dots on a black background. The letters 'REVEL' are formed by the absence of dots in the center. The 'R' is at column 10, row 10. The 'E' is at column 11, row 10. The 'V' is at column 12, row 10. The 'E' is at column 13, row 10. The 'L' is at column 14, row 10.

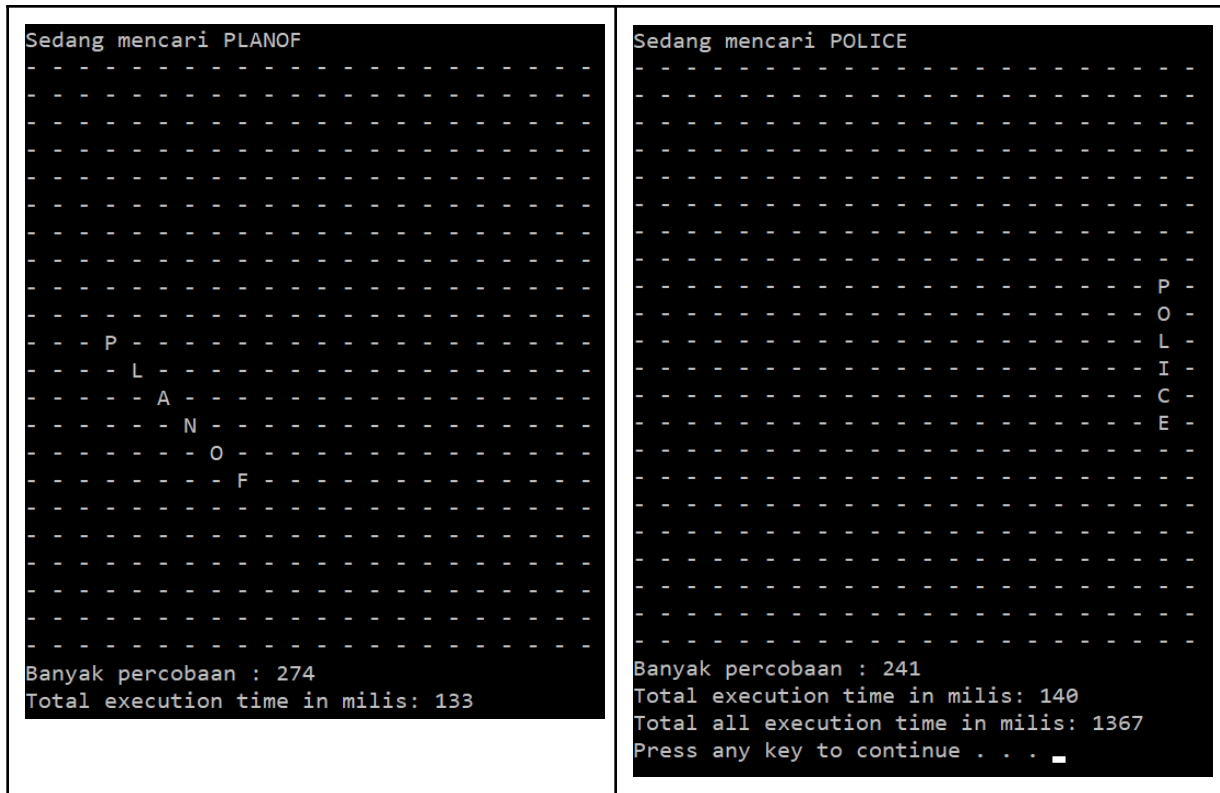
F  
O  
N  
A  
M



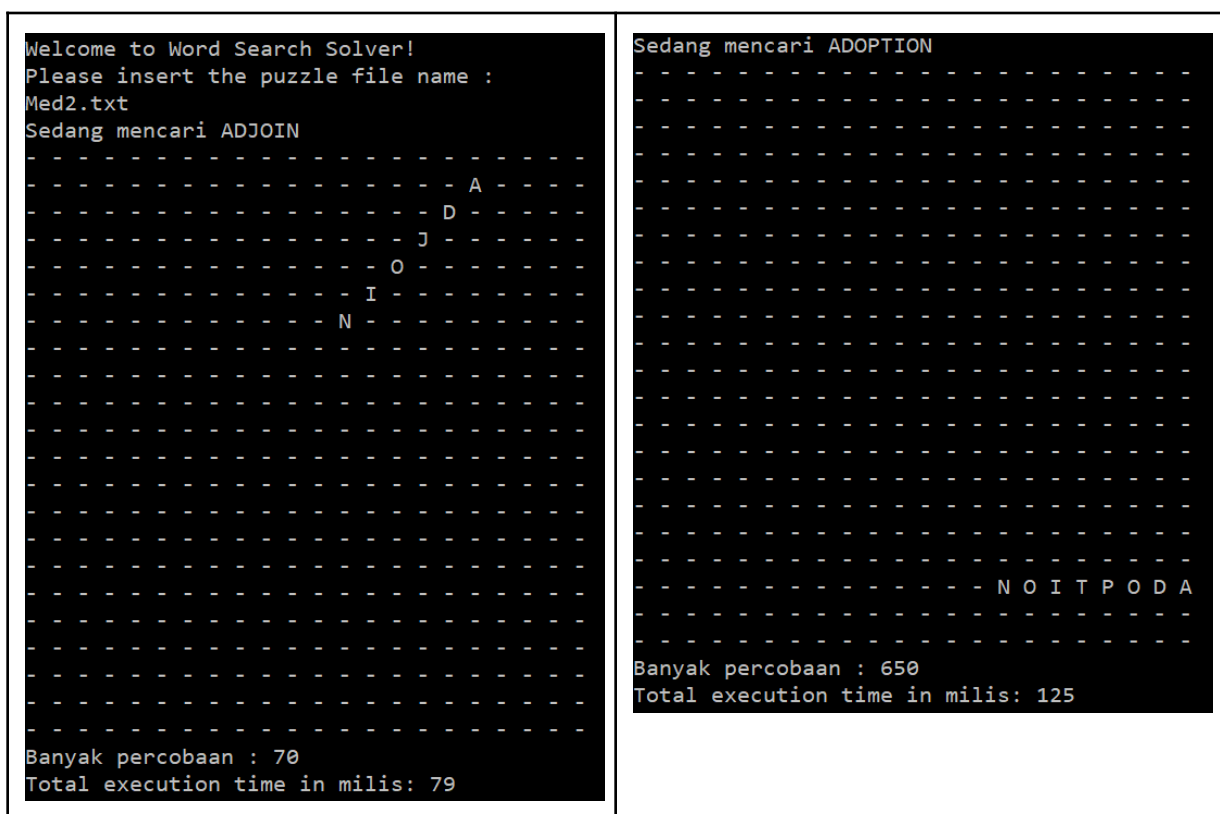
PIANO

- P I A N O -

25 - IF2211 - Strategi Algoritma



Gambar 2.4 Input dan Output Test Case Medium 1





CERTITUDE

A 20x20 grid of dots with a solid black vertical bar on the left side. The letters are placed at the following grid intersections (row, column):

- H: (18, 1)
- C: (17, 2)
- A: (16, 3)
- O: (15, 4)
- R: (14, 5)
- K: (13, 6)
- C: (12, 7)
- O: (11, 8)
- U: (10, 9)

RO TUB I R T N O C

A 20x20 grid of dots on a black background. The letters 'S', 'L', 'O', and 'C' are formed by removing dots in specific patterns. 'S' is in the top right, 'L' is below it, 'O' is to the right of 'L', and 'C' is to the right of 'O'.

28 - IF2211 - Strategi Algoritma

REH SURC

U  
C  
R  
A  
T  
O  
R

DEAL

D  
 I  
 S  
 O  
 B  
 E  
 D  
 I  
 E  
 N  
 T

29 - IF2211 - Strategi Algoritma

```
- - - - -
```

```
E
```

```
N
```

```
J
```

```
O
```

```
Y
```

```
A
```

```
B
```

```
L
```

```
Y
```

```
- - - - -
```

Banyak percobaan : 449  
Total execution time in milis: 102

[illegible]

```
Banyak percobaan : 463  
Total execution time in milis: 130
```

[illegible]

H  
O  
L  
O  
G  
R  
A  
M

A 20x20 grid of dots. The letters E, L, A, and M are placed at the following coordinates (row, column): E at (10, 10), L at (10, 11), A at (11, 11), and M at (12, 11).

31 - IF2211 - Strategi Algoritma



[illegible][illegible]

32 - IF2211 - Strategi Algoritma



[illegible]

```

A
N
N
E
X

```

```
B A C K P A C K I N G
```

Banyak percobaan : 206  
Total execution time in milis: 169

```
- - - - - M O O R L L A B - - - - -
```

Banyak percobaan : 48  
Total execution time in milis: 114

LANRAC

C  
 H  
 E  
 R  
 F  
 U  
 L

EDULCIN

ENOSITROC

35 - IF2211 - Strategi Algoritma

Sedang mencari DANGLING

- - - - - D A N G L I N G - - - - -

Banyak percobaan : 273

Total execution time in milis: 113

Sedang mencari DOUBLED

- - - - - D O U B L E D - - - - -

Banyak percobaan : 595

Total execution time in milis: 134

Sedang mencari DUPLICITY

- Y T I C I L P U D - - - - -

Banyak percobaan : 454

Total execution time in milis: 131

Sedang mencari ECTOPLASM

M  
S  
A  
L  
P  
O  
T  
C  
E

Banyak percobaan : 732

Total execution time in milis: 127

S R E H T A E F

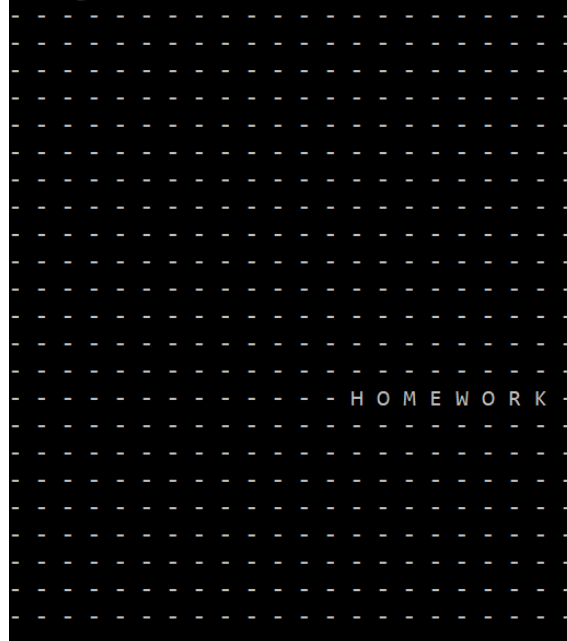
FLEX

S U O I R O L G

37 - IF2211 - Strategi Algoritma

G  
 N  
 I  
 T  
 S  
 E  
 V  
 R  
 A  
 H  
 -

H  
O  
M  
E  
S



HOMEWORK

IMMIGRATION

38 - IF2211 - Strategi Algoritma





S U O T N E M O M

S E S U M



NASTINESS

40 - IF2211 - Strategi Algoritma

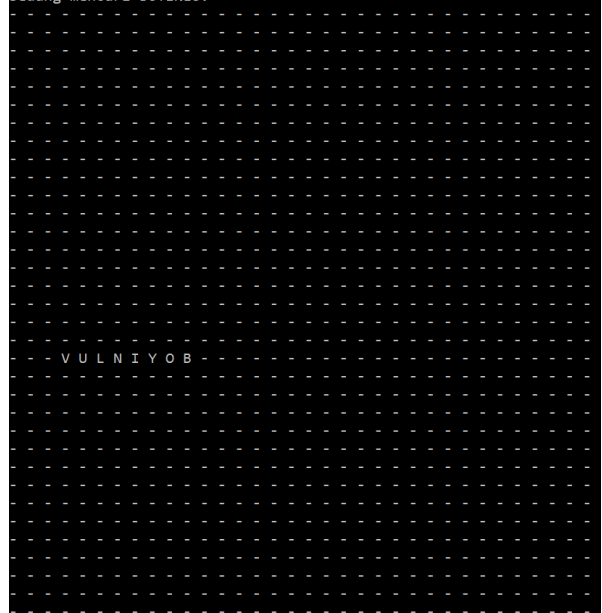


EMFOTSEB

```
Banyak percobaan : 299
Total execution time in milis: 132
```

SRAETDNATAEWSDOOLB

```
Banyak percobaan : 437
Total execution time in milis: 128
```



VULNIY OB

```
Banyak percobaan : 758
Total execution time in milis: 182
```

B  
U  
T  
T  
E  
R  
F  
L  
Y

```
Banyak percobaan : 773
Total execution time in milis: 179
```



[illegible]

```
Sedang mencari GOGO
```

```
G  
O  
G  
O
```

```
Banyak percobaan : 1067  
Total execution time in millis: 190
```

```

- - - - - HOUSE OF CARDS - - - - -

```

Banyak percobaan : 667  
Total execution time in millis: 181

```

      U
      D
      E
      E
      N
      I

```

Sedang mencari JUSTONEDAY

J  
U  
S  
T  
O  
N  
E  
D  
A  
Y

Banyak percobaan : 64  
Total execution time in milis: 169

Sedang mencari LETMEKNOW

W  
O  
N  
K  
E  
M  
T  
E  
L

Banyak percobaan : 884  
Total execution time in milis: 189

Sedang mencari LIE

L  
I  
E

Banyak percobaan : 686  
Total execution time in milis: 196

Sedang mencari LOST

L  
O  
S  
T

Banyak percobaan : 320  
Total execution time in milis: 187

```

      A
      M
      A
      M

```

Banyak percobaan : 362

```

N
0

```

```
M
A
E
R
D
E
R
O
M
O
N
```

Banyak percobaan : 1381

46 - IF2211 - Strategi Algoritma

Sedang mencari NOTTODAY

Y A D O T T O N

Banyak percobaan : 610  
Total execution time in milis: 204

Sedang mencari REFLECTION

N  
O  
I  
T  
C  
E  
L  
F  
E  
R

Banyak percobaan : 888  
Total execution time in milis: 206

Sedang mencari RUN

R  
U  
N

Banyak percobaan : 36  
Total execution time in milis: 202

Sedang mencari SAVEME

E  
M  
E  
V  
A  
S

Banyak percobaan : 1217  
Total execution time in milis: 208



Sedang mencari SERENDIPITY

- - - S E R E N D I P I T Y - - -

Banyak percobaan : 1062  
Total execution time in milis: 187

Sedang mencari SILVERSPoon

S  
I  
L  
V  
E  
R  
S  
P  
O  
O  
N

Banyak percobaan : 150  
Total execution time in milis: 160

Sedang mencari SPINEBREAKER

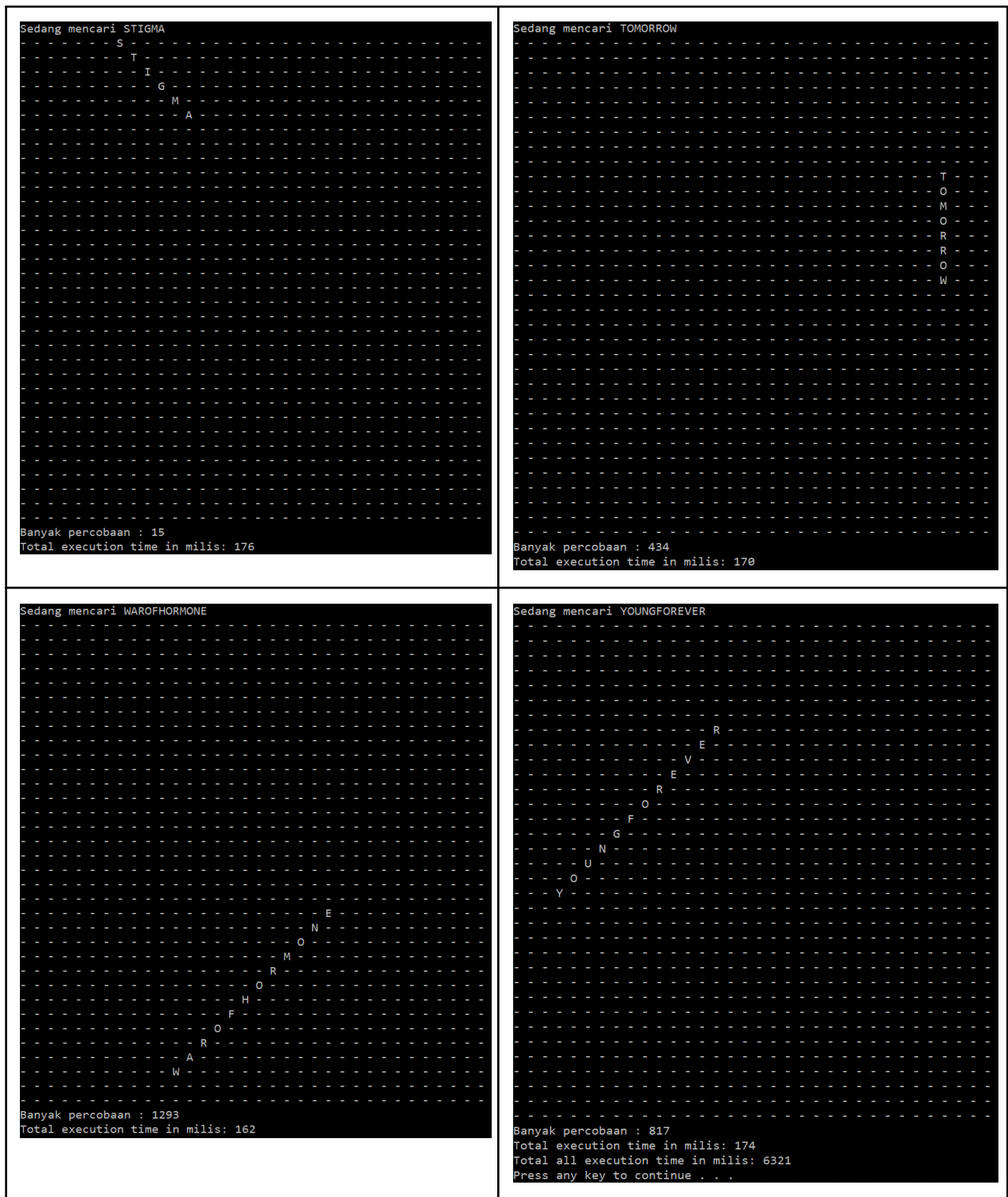
S  
P  
I  
N  
E  
B  
R  
E  
A  
K  
E  
R

Banyak percobaan : 966  
Total execution time in milis: 176

Sedang mencari SPRINGDAY

S  
P  
R  
I  
N  
G  
D  
A  
Y

Banyak percobaan : 29  
Total execution time in milis: 187



Gambar 2.7 Input dan Output Test Case Big 1

```
Welcome to Word Search Solver!
Please insert the puzzle file name :
Big2.txt
Sedang mencari AFRESH
```

```
H S E R F A
```

```
Banyak percobaan : 177
Total execution time in milis: 132
```

```
Sedang mencari AVALANCHING
```

```
G
N
I
H
C
N
A
L
A
V
A
```

```
Banyak percobaan : 959
Total execution time in milis: 193
```

```
Sedang mencari BACKREST
```

```
T
S
E
R
K
C
A
B
```

```
Banyak percobaan : 298
Total execution time in milis: 199
```

```
Sedang mencari BAGHDAD
```

```
D
A
D
H
G
A
B
```

```
Banyak percobaan : 981
Total execution time in milis: 251
```

Sedang mencari CHILLED

- - - - C H I L L E D - - - -

Banyak percobaan : 900  
Total execution time in milis: 203

Sedang mencari COCKY

- C -  
- O -  
- C -  
- K -  
- Y -

Banyak percobaan : 1105  
Total execution time in milis: 154

Sedang mencari CONQUERABLE

C  
O  
N  
Q  
U  
E  
R  
A  
B  
L  
E

Banyak percobaan : 645  
Total execution time in milis: 202

Sedang mencari CORDS

- C  
- O  
- R  
- D  
- S

Banyak percobaan : 1187  
Total execution time in milis: 199

```
- - Y T I E D - -
```

Banyak percobaan : 1236  
Total execution time in milis: 171

```
Banyak percobaan : 1533
Total execution time in milis: 127
```

```

- - - - - D
- - - - - N
- - - - - I
- - - W -
- N -
W -
O -
D O
D -

```

Banyak percobaan : 1099  
Total execution time in millis: 148

```

- - - - - E M P T I N E S S - - - - -
Banyak percobaan : 898
Total execution time in millis: 147

```

Sedang mencari FINDINGS

F  
I  
N  
D  
I  
N  
G  
S

Banyak percobaan : 495  
Total execution time in milis: 167

Sedang mencari FOUNDER

F  
O  
U  
N  
D  
E  
R

Banyak percobaan : 249  
Total execution time in milis: 170

Sedang mencari GREEDILY

G  
R  
E  
E  
D  
I  
L  
Y

Banyak percobaan : 501  
Total execution time in milis: 180

Sedang mencari GRUFF

F F U R G

Banyak percobaan : 669  
Total execution time in milis: 168

Sedang mencari GUARANTEE

E  
E  
T  
N  
A  
R  
A  
U  
G

Banyak percobaan : 1347  
Total execution time in milis: 189

Sedang mencari HYPERBOLE

E  
L  
O  
B  
R  
E  
P  
Y  
H

Banyak percobaan : 705  
Total execution time in milis: 172

Sedang mencari INTERVIEWED

D  
E  
W  
E  
I  
V  
R  
E  
T  
N  
I

Banyak percobaan : 742  
Total execution time in milis: 178

Sedang mencari LAUGHED

LAUGHED

Banyak percobaan : 611  
Total execution time in milis: 175

```

M
I
S
E
R
Y

```

```
Banyak percobaan : 1102
Total execution time in milis: 167
```

[illegible]

```

- - - - - L A T N E M A N R O - - - - -

```

Banyak percobaan : 243  
Total execution time in milis: 166



```
Banyak percobaan : 409  
Total execution time in milis: 190
```

```

- - - - - S - - - - -
- - - - - Y - - - - -
- - - O - - - - -
- - B - - - - -
- - Y - - - - -
- A - - - - -
- L - - - - -
P - - - - -

Banyak percobaan : 310
Total execution time in millis: 179

```

```
- PREYED -
```

Banyak percobaan : 662  
Total execution time in milis: 167

```

Banyak percobaan : 835
Total execution time in milis: 176

```

Sedang mencari REAPER

R  
E  
A  
P  
E  
R

Banyak percobaan : 144  
Total execution time in milis: 178

Sedang mencari RENAL

R  
E  
N  
A  
L

Banyak percobaan : 35  
Total execution time in milis: 178

Sedang mencari REPENTANT

T N A T N E P E R

Banyak percobaan : 1126  
Total execution time in milis: 179

Sedang mencari SCRIPTURAL

L  
A  
R  
U  
T  
P  
I  
R  
C  
S

Banyak percobaan : 618  
Total execution time in milis: 157

Sedang mencari SHRANK

K  
N  
A  
R  
H  
S

Banyak percobaan : 271  
Total execution time in milis: 182

Sedang mencari SMEARED

D  
E  
R  
A  
E  
M  
S

Banyak percobaan : 1032  
Total execution time in milis: 172

Sedang mencari SPECIALTY

SPECIALTY

Banyak percobaan : 918  
Total execution time in milis: 164

Sedang mencari SPINSTER

S  
P  
I  
N  
S  
T  
E  
R

Banyak percobaan : 1154  
Total execution time in milis: 188

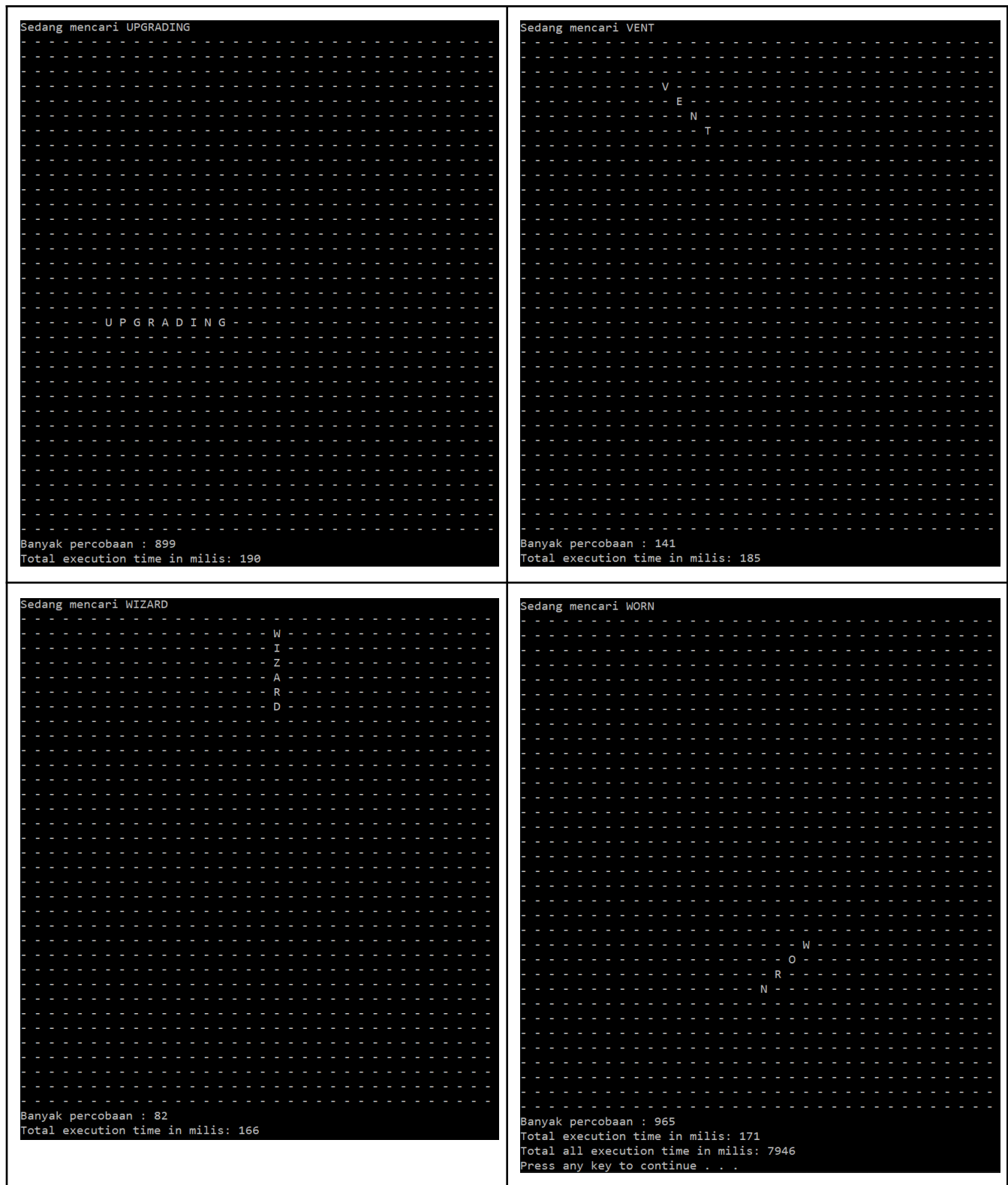
A 20x20 grid of dots. The word "DONATIONS" is spelled out by removing dots in a specific pattern. The letters are formed by the following dot coordinates (row, column) where the dot is missing:

- D**: (10, 12), (10, 13), (10, 14), (10, 15), (10, 16), (10, 17), (10, 18), (10, 19), (10, 20), (10, 21), (10, 22), (10, 23), (10, 24), (10, 25), (10, 26), (10, 27), (10, 28), (10, 29), (10, 30), (10, 31), (10, 32), (10, 33), (10, 34), (10, 35), (10, 36), (10, 37), (10, 38), (10, 39), (10, 40), (10, 41), (10, 42), (10, 43), (10, 44), (10, 45), (10, 46), (10, 47), (10, 48), (10, 49), (10, 50), (10, 51), (10, 52), (10, 53), (10, 54), (10, 55), (10, 56), (10, 57), (10, 58), (10, 59), (10, 60), (10, 61), (10, 62), (10, 63), (10, 64), (10, 65), (10, 66), (10, 67), (10, 68), (10, 69), (10, 70), (10, 71), (10, 72), (10, 73), (10, 74), (10, 75), (10, 76), (10, 77), (10, 78), (10, 79), (10, 80), (10, 81), (10, 82), (10, 83), (10, 84), (10, 85), (10, 86), (10, 87), (10, 88), (10, 89), (10, 90), (10, 91), (10, 92), (10, 93), (10, 94), (10, 95), (10, 96), (10, 97), (10, 98), (10, 99), (10, 100), (10, 101), (10, 102), (10, 103), (10, 104), (10, 105), (10, 106), (10, 107), (10, 108), (10, 109), (10, 110), (10, 111), (10, 112), (10, 113), (10, 114), (10, 115), (10, 116), (10, 117), (10, 118), (10, 119), (10, 120), (10, 121), (10, 122), (10, 123), (10, 124), (10, 125), (10, 126), (10, 127), (10, 128), (10, 129), (10, 130), (10, 131), (10, 132), (10, 133), (10, 134), (10, 135), (10, 136), (10, 137), (10, 138), (10, 139), (10, 140), (10, 141), (10, 142), (10, 143), (10, 144), (10, 145), (10, 146), (10, 147), (10, 148), (10, 149), (10, 150), (10, 151), (10, 152), (10, 153), (10, 154), (10, 155), (10, 156), (10, 157), (10, 158), (10, 159), (10, 160), (10, 161), (10, 162), (10, 163), (10, 164), (10, 165), (10, 166), (10, 167), (10, 168), (10, 169), (10, 170), (10, 171), (10, 172), (10, 173), (10, 174), (10, 175), (10, 176), (10, 177), (10, 178), (10, 179), (10, 180), (10, 181), (10, 182), (10, 183), (10, 184), (10, 185), (10, 186), (10, 187), (10, 188), (10, 189), (10, 190), (10, 191), (10, 192), (10, 193), (10, 194), (10, 195), (10, 196), (10, 197), (10, 198), (10, 199), (10, 200), (10, 201), (10, 202), (10, 203), (10, 204), (10, 205), (10, 206), (10, 207), (10, 208), (10, 209), (10, 210), (10, 211), (10, 212), (10, 213), (10, 214), (10, 215), (10, 216), (10, 217), (10, 218), (10, 219), (10, 220), (10, 221), (10, 222), (10, 223), (10, 224), (10, 225), (10, 226), (10, 227), (10, 228), (10, 229), (10, 230), (10, 231), (10, 232), (10, 233), (10, 234), (10, 235), (10, 236), (10, 237), (10, 238), (10, 239), (10, 240), (10, 241), (10, 242), (10, 243), (10, 244), (10, 245), (10, 246), (10, 247), (10, 248), (10, 249), (10, 250), (10, 251), (10, 252), (10, 253), (10, 254), (10, 255), (10, 256), (10, 257), (10, 258), (10, 259), (10, 260), (10, 261), (10, 262), (10, 263), (10, 264), (10, 265), (10, 266), (10, 267), (10, 268), (10, 269), (10, 270), (10, 271), (10, 272), (10, 273), (10, 274), (10, 275), (10, 276), (10, 277), (10, 278), (10, 279), (10, 280), (10, 281), (10, 282), (10, 283), (10, 284), (10, 285), (10, 286), (10, 287), (10, 288), (10, 289), (10, 290), (10, 291), (10, 292), (10, 293), (10, 294), (10, 295), (10, 296), (10, 297), (10, 298), (10, 299), (10, 300), (10, 301), (10, 302), (10, 303), (10, 304), (10, 305), (10, 306), (10, 307), (10, 308), (10, 309), (10, 310), (10, 311), (10, 312), (10, 313), (10, 314), (10, 315), (10, 316), (10, 317), (10, 318), (10, 319), (10, 320), (10, 321), (10, 322), (10, 323), (10, 324), (10, 325), (10, 326), (10, 327), (10, 328), (10, 329), (10, 330), (10, 331), (10, 332), (10, 333), (10, 334), (10, 335), (10, 336), (10, 337), (10, 338), (10, 339), (10, 340), (10, 341), (10, 342), (10, 343), (10, 344), (10, 345), (10, 346), (10, 347), (10, 348), (10, 349), (10, 350), (10, 351), (10, 352), (10, 353), (10, 354), (10, 355), (10, 356), (10, 357), (10, 358), (10, 359), (10, 360), (10, 361), (10, 362), (10, 363), (10, 364), (10, 365), (10, 366), (10, 367), (10, 368), (10, 369), (10, 370), (10, 371), (10, 372), (10, 373), (10, 374), (10, 375), (10, 376), (10, 377), (10, 378), (10, 379), (10, 380), (10, 381), (10, 382), (10, 383), (10, 384), (10, 385), (10, 386), (10, 387), (10, 388), (10, 389), (10, 390), (10, 391), (10, 392), (10, 393), (10, 394), (10, 395), (10, 396), (10, 397), (10, 398), (10, 399), (10, 400), (10, 401), (10, 402), (10, 403), (10, 404), (10, 405), (10, 406), (10, 407), (10, 408), (10, 409), (10, 410), (10, 411), (10, 412), (10, 413), (10, 414), (10, 415), (10, 416), (10, 417), (10, 418), (10, 419), (10, 420), (10, 421), (10, 422), (10, 423), (10, 424), (10, 425), (10, 426), (10, 427), (10, 428), (10, 429), (10, 430), (10, 431), (10, 432), (10, 433), (10, 434), (10, 435), (10, 436), (10, 437), (10, 438), (10, 439), (10, 440), (10, 441), (10, 442), (10, 443), (10, 444), (10, 445), (10, 446), (10, 447), (10, 448), (10, 449), (10, 450), (10, 451), (10, 452), (10, 453), (10, 454), (10, 455), (10, 456), (10, 457), (10, 458), (10, 459), (10, 460), (10, 461), (10, 462), (10, 463), (10, 464), (10, 465), (10,

SDUTS

S  
S  
E  
R  
G  
I  
T

59 - IF2211 - Strategi Algoritma



Gambar 2.8 Input dan Output Test Case Big 2

---

ORIDUA

---

F  
O  
O  
R  
P  
T  
E  
L  
L  
U  
B

---

P  
M  
U  
B

---

A 20x20 grid of dots. The letters are placed at the following intersections (row, column):

- S: (10, 10)
- T: (11, 10)
- T: (12, 10)
- U: (13, 10)
- B: (14, 5)

COLLECTIVELY

DEHYDRATING

62 - IF2211 - Strategi Algoritma

Sedang mencari DITTO

- - - D - - -  
- - - I - - -  
- - - T - - -  
- - - T - - -  
O - - -

Banyak percobaan : 14  
Total execution time in milis: 239

Sedang mencari DONATING

- - - G N I T A N O D - - -

Banyak percobaan : 158  
Total execution time in milis: 209

Sedang mencari ETUI

I - - -  
- U - - -  
- T - - -  
- E - - -

Banyak percobaan : 473  
Total execution time in milis: 219

Sedang mencari FAIREST

T -  
S -  
E -  
R -  
I -  
A -  
F -

Banyak percobaan : 1057  
Total execution time in milis: 255



Sedang mencari FATE

F  
A  
T  
E

Banyak percobaan : 339  
Total execution time in milis: 216

Sedang mencari HEADGEAR

H  
E  
A  
D  
G  
E  
A  
R

Banyak percobaan : 497  
Total execution time in milis: 234

Sedang mencari HEAVILY

H  
E  
A  
V  
I  
L  
Y

Banyak percobaan : 25  
Total execution time in milis: 190

Sedang mencari HYDRAULIC

C  
I  
L  
U  
A  
R  
D  
Y  
H

Banyak percobaan : 219  
Total execution time in milis: 181

A 20x20 grid of dots. The word "ENIGMA" is spelled out in larger dots, centered in the grid. The letters are arranged as follows: E (row 10, col 14), N (row 11, col 14), I (row 12, col 14), G (row 13, col 14), M (row 14, col 14), A (row 15, col 14). The word is oriented vertically.

N  
A  
M  
H  
S  
I  
R  
I

A 20x20 grid of dots on a black background. The letters 'I', 'L', 'A', 'T', 'I', 'C', 'Z', 'E', 'D' are placed at specific grid intersections to form a stylized representation of the word 'LATICIDE'. The letters are arranged as follows: 'I' at (10, 10), 'L' at (10, 12), 'A' at (10, 14), 'T' at (10, 16), 'I' at (10, 18), 'C' at (10, 20), 'Z' at (10, 22), 'E' at (10, 24), and 'D' at (10, 26). The grid is 20 columns wide and 20 rows high.



LIMITATION -

65 - IF2211 - Strategi Algoritma

A 20x20 grid of dots. The letters M, O, T, and H are placed at the end of the 10th, 11th, 12th, and 13th rows respectively.

NOITALITUM

66 - IF2211 - Strategi Algoritma

Sedang mencari OBTUSE

E  
S  
U  
T  
B  
O

Banyak percobaan : 1226  
Total execution time in milis: 182

Sedang mencari PEOPLE

D  
E  
L  
P  
O  
E  
P

Banyak percobaan : 877  
Total execution time in milis: 207

Sedang mencari PERFORATION

N  
O  
I  
T  
A  
R  
O  
F  
R  
E  
P

Banyak percobaan : 754  
Total execution time in milis: 189

Sedang mencari PIVOTAL

P I V O T A L

Banyak percobaan : 712  
Total execution time in milis: 171



## Link Repository Kode Program

[https://github.com/hcarissa/Tucil1\\_13520164.git](https://github.com/hcarissa/Tucil1_13520164.git)

## Checklist

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

## Daftar Referensi

Slide Kuliah IF2211 Strategi Algoritma - Algoritma Brute Force Bagian 1 -

[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)

Perbandingan Algoritma Brute Force dan Backtracking dalam Permainan Word Search Puzzle

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2016-2017/Makalah2017/Makalah-IF2211-2017-077.pdf>