

**LAPORAN TUGAS KECIL 1 IF2211 STRATEGI ALGORITMA
SEMESTER II TAHUN 2021/2022**

**PENYELESAIAN WORD SEARCH PUZZLE DENGAN ALGORITMA
BRUTE FORCE**



Disusun oleh :

STEVEN

13520131

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG**

2022

DAFTAR ISI

BAB I DESKRIPSI MASALAH	2
BAB II PENJELASAN ALGORITMA <i>BRUTEFORCE</i> YANG DIGUNAKAN.....	3
BAB III KODE PROGRAM DALAM BAHASA C++	4
BAB IV INPUT/OUTPUT PROGRAM	8
BAB V TABEL PENILAIAN.....	77

BAB I

DESKRIPSI MASALAH

Merancang algoritma *brute force* untuk menemukan semua kata di dalam *word search puzzle*.

1. Tulislah program kecil (sederhana) dalam Bahasa C/C++/Java yang mengimplementasikan algoritma Brute Force untuk mencari solusi word search puzzle.
2. Input: file teks yang berisi matriks huruf di dalam puzzle (antar huruf dipisahkan oleh spasi), diikuti satu baris kosong, dan daftar kata-kata yang akan dicari di dalam puzzle. Satu baris satu kata. Puzzle boleh dalam Bahasa Inggris atau Bahasa Indonesia.
3. Output: Tampilan di layar yang memuat:
 - a. Semua kata yang ditemukan
 - b. Waktu eksekusi program (tidak termasuk waktu pembacaan file input).
 - c. Jumlah total perbandingan huruf yang dilakukan untuk menemukan kata di dalam puzzle.

BAB II

PENJELASAN ALGORITMA *BRUTEFORCE* YANG DIGUNAKAN

Pertama-tama, akan dilakukan pembacaan file yang diinput menjadi dua buah vector of string, yaitu satu untuk menampung huruf di dalam puzzle (*word_matrix*) dan satu lagi untuk menampung daftar kata-kata yang akan dicari di dalam *puzzle* (*word_list*). Selanjutnya, akan dijelaskan algoritma untuk menyelesaikan permasalahan ini. Kita ambil contoh sebagai berikut.

J	S	O	L	U	T	I	S
S	U	N	A	R	U	U	A
N	E	P	T	U	N	E	T
S	O	N	I	E	I	S	U
R	C	E	V	T	R	E	R
A	H	T	R	A	E	S	N
M	M	E	R	C	U	R	Y

EARTH
JUPITER
MARS
MERCURY
NEPTUNE
SATURN
URANUS
VENUS

Pertama-tama, dilakukan pencarian lokasi dari seluruh abjad pertama dari kata-kata yang akan dicari, lalu catat kordinatnya dan simpan dalam array of array of tuple. Setelah diperoleh koordinat-koordinat dari karakter tersebut, barulah kita memulai pencarian. Pencarian akan dimulai dari koordinat yang telah diperoleh sebelumnya dengan cara mencari pada 8 arah. Pencarian 8 arah ini merupakan algoritma *brute force*.

Misalnya kita ingin mencari EARTH, maka kita perlu mendata seluruh koordinat dari abjad pertamanya. Selanjutnya, akan dilakukan pencarian pada 8 arah dengan melakukan pencocokan string dengan algoritma *brute force*. Untuk karakter-karakter yang cocok, akan disimpan koordinatnya lalu selanjutnya akan dioutputkan hasilnya.

BAB III

KODE PROGRAM DALAM BAHASA C++

```
main.cpp x
1 #include <fstream>
2 #include <string>
3 #include <vector>
4 #include <iostream>
5 #include <stdint>
6 #include <chrono>
7 #include <map>
8 #include <tuple>
9 #include <algorithm>
10 using std::int64_t;
11
12 // Debugging Helper
13 void print(std::vector<std::tuple<char, int64_t, int64_t>>& arg) {
14     for(int i = 0; i < arg.size(); ++i) {
15         std::cout << std::get<0>(arg[i]) << ' ';
16     }
17     std::cout << '\n';
18 }
19 // End of Debugging Helper
20
21 int main()
22 {
23     std::ifstream file_input;
24     std::string file_name;
25     std::cout << "Masukkan nama file \n";
26     std::cout << "Contoh: ../test/tgl.txt \n\n";
27     std::cout << ">>> ";
28     std::cin >> file_name;
29     std::cout << "\n";
30     file_input.open(file_name);
31     std::vector<std::string> word_matrix;
32     std::vector<std::string> word_list;
33     std::map<char, std::vector<std::tuple<int64_t, int64_t>>> word_pos;
34
35     int64_t compare_count = 0;
36
37     for(std::string line_in; std::getline(file_input, line_in);)
38     {
39         if(line_in == "") continue;
40         if(line_in[0] == ' ')
41         {
42             std::string temp;
43             for(int64_t i = 0; i < line_in.size(); i += 2)
44             {
45                 temp.push_back(line_in[i]);
46             }
47             word_matrix.emplace_back(std::move(temp));
48         }
49         else
50         {
51             word_list.emplace_back(std::move(line_in));
52             auto iter = word_pos.find(word_list.back()[0]);
53             if(iter == word_pos.end())
54             {
55                 word_pos[word_list.back()[0]] = std::vector<std::tuple<int64_t, int64_t>>();
56             }
57         }
58     }
59
60     std::vector<std::vector<std::tuple<char, int64_t, int64_t>>> output;
61
62     auto time_start = std::chrono::steady_clock::now();
63
64     for(int64_t i = 0; i < word_matrix.size(); ++i)
65     {
66         for(int64_t j = 0; j < word_matrix[i].size(); ++j)
67         {
68             compare_count++;
69             auto iter = word_pos.find(word_matrix[i][j]);
70             if(iter != word_pos.end())
71             {
72                 word_pos[word_matrix[i][j]].emplace_back(std::make_tuple(i, j));
73             }
74         }
75     }
76
77     for(int i = 0; i < word_list.size(); ++i)
78     {
79         std::string word = word_list[i];
80         for(int j = 0; j < word_pos[word[0]].size(); ++j)
81         {
82             int64_t row_num;
83             int64_t col_num;
84             std::tie(row_num, col_num) = word_pos[word[0]][j];
85
86             bool found = true;
87             // Check to the top
88             std::vector<std::tuple<char, int64_t, int64_t>> output_temp;
89             for(int64_t index = 0; index < word.size(); ++index)
90             {
91                 if(row_num - index < 0)
92                 {
93                     found = false;
94                     break;
95                 }
96                 if(word[index] == word_matrix[row_num - index][col_num])
97                 {
98                     output_temp.emplace_back(std::make_tuple(word[index], row_num - index, col_num));
99                 }
100                 else
101                 {
102                     found = false;
103                     break;
104                 }
105             }
106             if(found)
107             {
108                 output.emplace_back(output_temp);
109             }
110         }
111     }
112 }
```

```

109         if(found)
110         {
111             output.emplace_back(std::move(output_temp));
112             continue;
113         }
114
115         output_temp.clear();
116
117         found = true;
118         // Check to the left
119         for(int64_t index = 0; index < word.size(); ++index)
120         {
121             if(col_num - index < 0)
122             {
123                 found = false;
124                 break;
125             }
126             compare_count++;
127             if(word[index] == word_matrix[row_num][col_num-index])
128             {
129                 output_temp.emplace_back(std::make_tuple(word[index], row_num, col_num-index));
130             }
131             else
132             {
133                 found = false;
134                 break;
135             }
136         }
137         if(found)
138         {
139             output.emplace_back(std::move(output_temp));
140             continue;
141         }
142
143         output_temp.clear();
144
145         found = true;
146         // Check to the bottom
147         for(int64_t index = 0; index < word.size(); ++index)
148         {
149             if(row_num + index >= word_matrix.size())
150             {
151                 found = false;
152                 break;
153             }
154             compare_count++;
155             if(word[index] == word_matrix[row_num+index][col_num])
156             {
157                 output_temp.emplace_back(std::make_tuple(word[index], row_num + index, col_num));
158             }
159             else
160             {
161                 found = false;
162                 break;
163             }
164         }
165         if(found)
166         {
167             output.emplace_back(std::move(output_temp));
168             continue;
169         }
170
171         output_temp.clear();
172
173         found = true;
174         // Check to the right
175         for(int64_t index = 0; index < word.size(); ++index)
176         {
177             if(col_num + index >= word_matrix[0].size())
178             {
179                 found = false;
180                 break;
181             }
182             compare_count++;
183             if(word[index] == word_matrix[row_num][col_num+index])
184             {
185                 output_temp.emplace_back(std::make_tuple(word[index], row_num, col_num+index));
186             }
187             else
188             {
189                 found = false;
190                 break;
191             }
192         }
193         if(found)
194         {
195             output.emplace_back(std::move(output_temp));
196             continue;
197         }
198
199         output_temp.clear();
200
201         found = true;
202         // Check to top-left
203         for(int64_t index = 0; index < word.size(); ++index)
204         {
205             if(col_num - index < 0 || row_num - index < 0)
206             {
207                 found = false;
208                 break;
209             }
210             compare_count++;
211             if(word[index] == word_matrix[row_num-index][col_num-index])
212             {
213                 output_temp.emplace_back(std::make_tuple(word[index], row_num-index, col_num-index));
214             }
215             else
216             {

```

```

217         found = false;
218         break;
219     }
220 }
221 if(found)
222 {
223     output.emplace_back(std::move(output_temp));
224     continue;
225 }
226
227 output_temp.clear();
228
229 found = true;
230 // Check to top-right
231 for(int64_t index = 0; index < word.size(); ++index)
232 {
233     if(col_num + index >= word_matrix[0].size() || row_num - index < 0)
234     {
235         found = false;
236         break;
237     }
238     compare_count++;
239     if(word[index] == word_matrix[row_num-index][col_num+index])
240     {
241         output_temp.emplace_back(std::make_tuple(word[index], row_num-index, col_num+index));
242     }
243     else
244     {
245         found = false;
246         break;
247     }
248 }
249 if(found)
250 {
251     output.emplace_back(std::move(output_temp));
252     continue;
253 }
254
255 output_temp.clear();
256
257 found = true;
258 // Check to bottom-left
259 for(int64_t index = 0; index < word.size(); ++index)
260 {
261     if(col_num - index < 0 || row_num + index >= word_pos.size())
262     {
263         found = false;
264         break;
265     }
266     compare_count++;
267     if(word[index] == word_matrix[row_num+index][col_num-index])
268     {
269         output_temp.emplace_back(std::make_tuple(word[index], row_num+index, col_num-index));
270     }
271     else
272     {
273         found = false;
274         break;
275     }
276 }
277 if(found)
278 {
279     output.emplace_back(std::move(output_temp));
280     continue;
281 }
282
283 output_temp.clear();
284
285 found = true;
286 // Check to bottom-right
287 for(int64_t index = 0; index < word.size(); ++index)
288 {
289     if(col_num + index >= word_matrix[0].size() || row_num + index >= word_matrix.size())
290     {
291         found = false;
292         break;
293     }
294     compare_count++;
295     if(word[index] == word_matrix[row_num+index][col_num+index])
296     {
297         output_temp.emplace_back(std::make_tuple(word[index], row_num+index, col_num+index));
298     }
299     else
300     {
301         found = false;
302         break;
303     }
304 }
305 if(found)
306 {
307     output.emplace_back(std::move(output_temp));
308     continue;
309 }
310 output_temp.clear();
311 }
312 }
313
314 // Return hasil
315 for(auto& result : output)
316 {
317     for(int i = 0; i < word_matrix.size(); ++i)
318     {
319         for(int j = 0; j < word_matrix[i].size(); ++j)
320         {
321             auto iter = std::find_if(result.begin(), result.end(), [i, j](std::tuple<char, int64_t, int64_t> el)
322             {
323                 return std::get<0>(el) == i && std::get<2>(el) == j;
324             });

```

```

325         if(iter != result.end())
326         {
327             std::cout << std::get<0>(*iter);
328         }
329         else
330         {
331             std::cout << '-';
332         }
333         std::cout << ' ';
334     }
335     std::cout << '\n';
336 }
337 std::cout << "-----\n\n";
338 }
339 auto time_end = std::chrono::steady_clock::now();
340 std::cout << "Waktu yang diperlukan untuk mengeksekusi program adalah " << std::chrono::duration_cast<std::chrono::duration<double>>(time_end - time_start).count() << " detik\n";
341 std::cout << "Banyak operasi perbandingan kata yang dilakukan program ini adalah " << compare_count << " perbandingan \n\n";
342 }
343

```


BAB IV

INPUT/OUTPUT PROGRAM

Test Case 1 (Small)

Input



tc1.txt - Notepad

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

```
AUTO
EXTERN
SIZEOF
UNION
CONTINUE
GOTO
STATIC
VOID
DEFAULT
REGISTER
STRUCT
VOLATILE
```

Output

```
Masukkan nama file
Contoh: ../test/tc1.txt

>>> ../test/tc1.txt
```

OTUA

A 10x10 grid with letters placed at the following intersections (row, column):

- (3, 4): E
- (4, 5): X
- (5, 6): T
- (6, 7): E
- (7, 8): R
- (8, 9): N

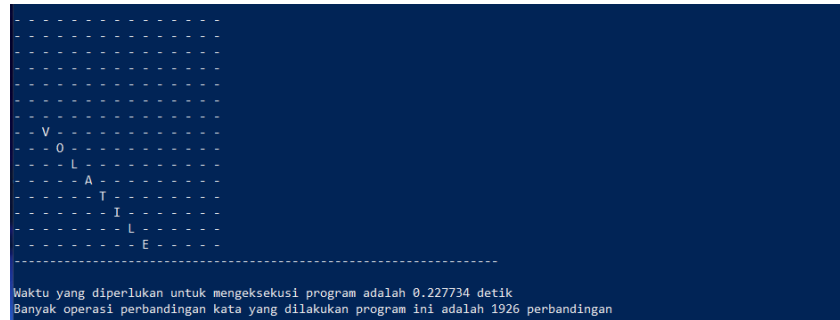


FOEZIS

A 10x10 grid of dots. In the bottom right corner, a small cluster of dots forms the word 'UNION'. The 'U' is at row 8, column 8; 'N' is at row 8, column 9; 'I' is at row 7, column 9; 'O' is at row 6, column 9; 'N' is at row 6, column 10; and 'U' is at row 5, column 10.

CONSTITUTION

T
C
U
R
T
S



Waktu yang diperlukan untuk mengeksekusi program adalah 0.227734 detik
Banyak operasi perbandingan kata yang dilakukan program ini adalah 1926 perbandingan

Test Case 2 (Small)

Input



ASM
EXPLICIT
PUBLIC
VIRTUAL
CATCH
MUTABLE
THROW
CLASS
NAMESPACE
TYPEID

Output

```
Masukkan nama file
Contoh: ../test/tc1.txt
>>> ../test/tc2.txt
```

M
S
A

EXPLICIT

P
U
B
L
I
C

C
A
T
C
H

E
 L
 B
 A
 T
 U
 M

Waktu yang diperlukan untuk mengeksekusi program adalah 0.198526 detik
Banyak operasi perbandingan kata yang dilakukan program ini adalah 1647 perbandingan

Test Case 3 (Small)

Input



tc3.txt - Notepad

File Edit Format View Help

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

ADONIS
ALPINE
ANGULAR
BACKBONE
CHART
CYTOSCAPE
EMBER
NEST
NEXT
NODE
NUXT
ROLLUP
THREE
VUE
WIKI

Output

```
Masukkan nama file
Contoh: ../test/tc1.txt

>>> ../test/tc3.txt
```

A
D
O
N
I
S

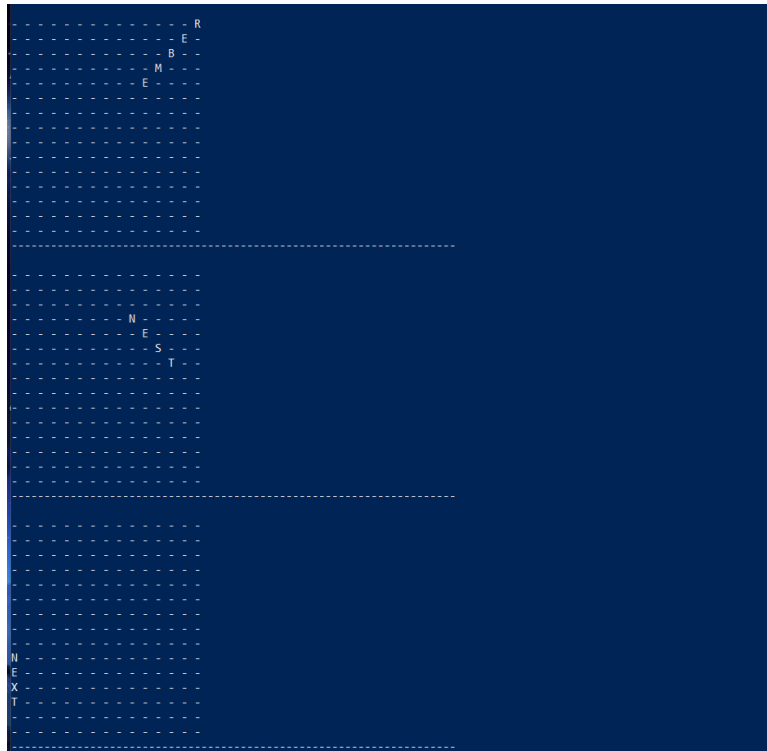
A	-	-	-	-	-	-	-	-
L	-	-	-	-	-	-	-	-
P	-	-	-	-	-	-	-	-
I	-	-	-	-	-	-	-	-
N	-	-	-	-	-	-	-	-
E	-	-	-	-	-	-	-	-

- - - R - - -
 - - - A - - -
 - - - L - - -
 - - - U - - -
 - - - G - - -
 - - - N - - -
 - - - A - - -

E	-	-	-	-	-	-
N	-	-	-	-	-	-
O	-	-	-	-	-	-
B	-	-	-	-	-	-
K	-	-	-	-	-	-
C	-	-	-	-	-	-
A	-	-	-	-	-	-
B	-	-	-	-	-	-

T	-	-	-	-	-	-	-	-	-
R	-	-	-	-	-	-	-	-	-
A	-	-	-	-	-	-	-	-	-
H	-	-	-	-	-	-	-	-	-
C	-	-	-	-	-	-	-	-	-

CYTOSCAPE



Output

```
Masukkan nama file  
Contoh: ../test/tc1.txt
```

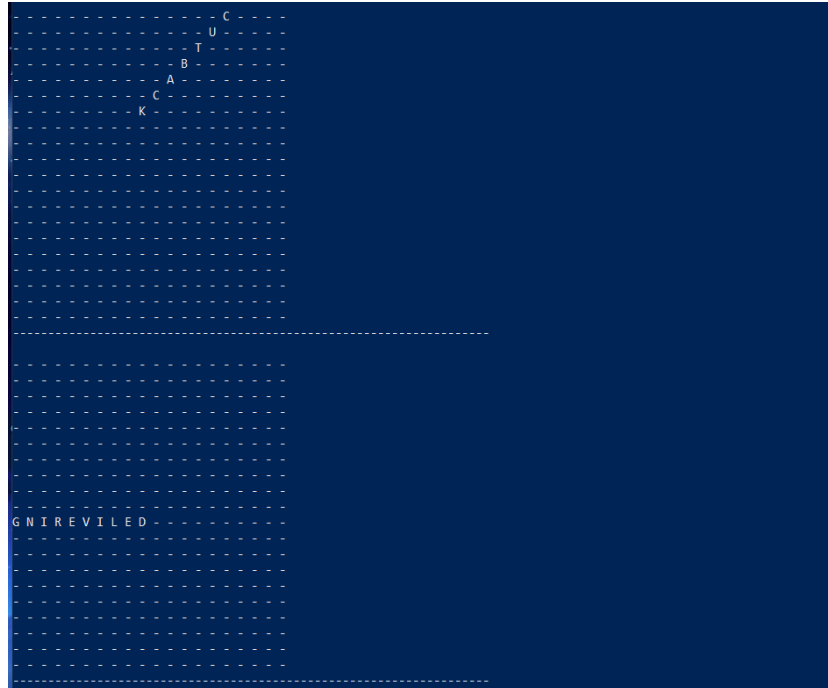
```
>>> ../test/tc4.txt
```

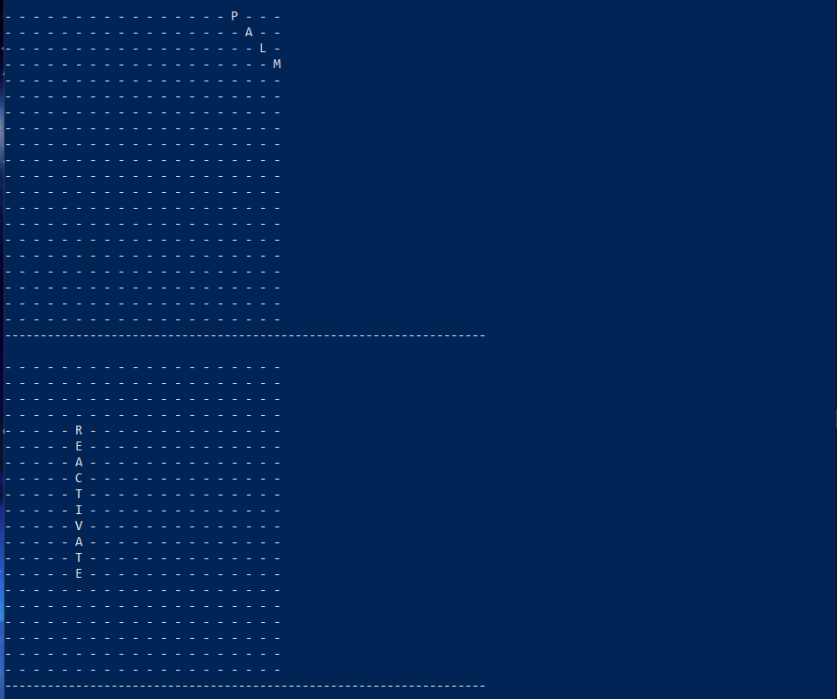
```
-- -- -- E --  
-- -- -- L --  
-- -- B --  
-- A --  
- R -  
E -  
W -  
S -  
N -  
A -  
  
-----  
  
-- -- -- A V E R T -- --  
  
-----  

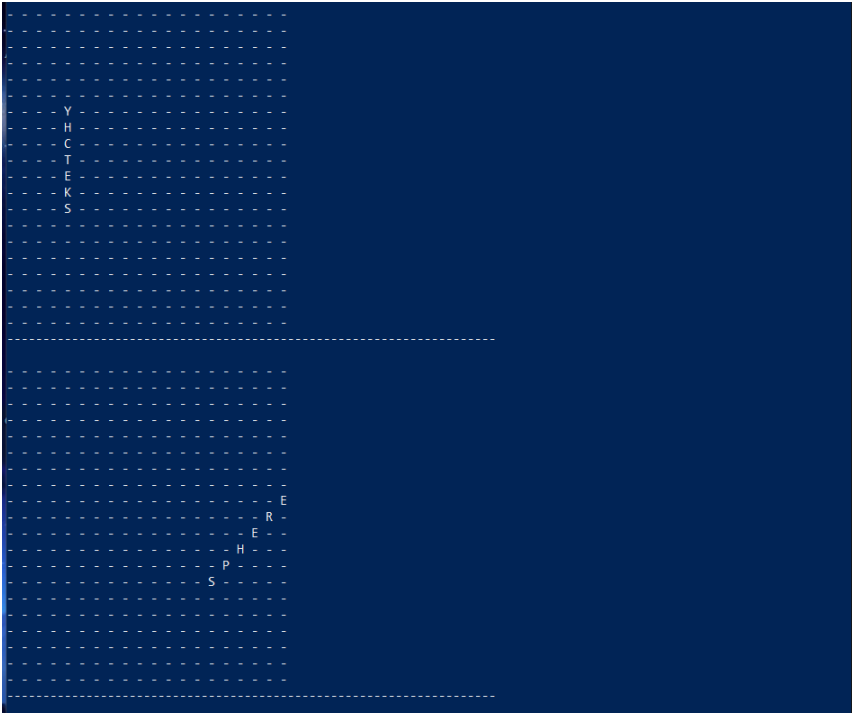
```

B
 E
 D
 A
 Z
 Z
 L
 E

C
 R
 E
 D
 U
 L
 T
 T
 Y







Test Case 5 (Medium)

Input

tc5.txt - Notepad

File Edit Format View Help

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

ANGOLA
ARGENTINA
BAHAMAS
BANGLADESH
BRAZIL
CHINA
CUBA
DENMARK
EGYPT
FRANCE
GUINEA
INDONESIA
ISRAEL
JAPAN
LEBANON
LUXEMBOURG
NORWAY
POLAND

Output

```
Masukkan nama file
Contoh: ../test/tc1.txt
```

```
>>> ../test/tc5.txt
```

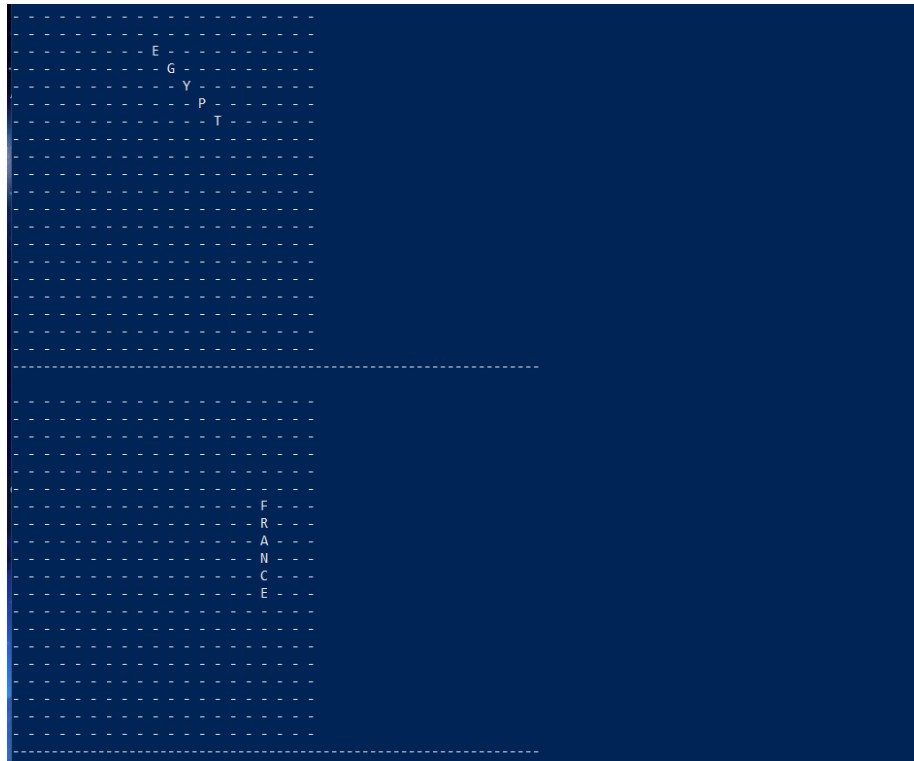
A
N
G
O
L
A

A
R
G
E
N
T
I
N
A

A B
H A
M A
S

H S E D A L G N A B







Output

```
Masukkan nama file
Contoh: ../test/tc1.txt

>>> ../test/tc6.txt

-----
- D -
- N -
- O -
- M -
- L -
- A -
-----

-----
- S -
- U -
- G -
- A -
- R -
- A -
- P -
- S -
- A -
-----

-----
- B -
- A -
- G -
- E -
- L -
-----

-----
- B E E R -
-----
```





[illegible]

Test Case 7 (Large)

Input



tc7.txt - Notepad

File Edit Format View Help

```
ECXLESISJPLNGJKNIMUQXCQJQABOPDIEW
ULWRGIUYOOEBGZUXCDIPWUYNTJAUENCGA
SHBLXCOIOMTTPZZEHGIDEDROCCATSOORF
BHPAUHSYWRSTCYGUARIOI IWKGDA PNTUIF
YSTMRODETAGRUPXEISWZCJIWEIEOBTISOL
VEHANORUNJEEYCZDNHHHFFYJBLCMRIQBBE
ARPEYCVTUVLGDPLETIONXBITIDGUILLED
WFDRLPLFANEGATTIVEDEAJDAFICNRHYSZVD
ALCVIREAFCCCTIJJOYFOZNCFOWTA AHZYWAN
TROTAGITSEVNIRROSSJBSANAHRMFNKFIMK
AIHQRCAWUAPAVMEQCWWJSSVGWDSSTSGPOAR
PDEXUACDJC MVXIUTQOYUERNNEEIPSUONC
FQVFYASZNEADXA OYSLPRWCIHHLNLFHPKL
BEEQRLLOYRKJASWFLIETIYATUQAGAWCJEP
AIZSJJEQOMMRHREW CADGLILAIDSETKMPHR
MULIRWTNQAHFAZPMVTLA OBRVNJHTQHECK
RTTFSNWNNCJSUMDYLUENMJGLPGYEGQLIS
LFWKAEJNUEVASYZSFTNDHRI DSYURXRLEM
ZIXRFMHOUOXXSQVTUHSIHDMCRECREXRWH
FKAWVURTVICEILHXZOLPPBCTAMCUTOQTD
EUSEEMEDNORTLGOYFRLCAHIGUVOVWLBZY
GTTKHL LK YFAUPEAA MCGQRIDCHTJR XGHX
SBEIQMXKKIKSOWPDYTTQQDLRKLDNUMDLPDI
QECSEFAWUGHGETQJCD OXZADLI INRSS LIS
BAREBCODJTSZOLT LKQZAMRGLIFATTGUNG
SRLRPBPDUYSNGDFOWSUNLZVNMNZGNTE SP
LTSMA MHHLRIMIQKAKZVHUPITXUGKRPBCJ
FKSJNZRPZJB IHBUMTMQVULCKAPIENAPRP
TCXEOCPKFR IWTALTRWSOXAXFDEXUCRMIQ
JMGZCSULOIGXBWPPEY GADTLBBEOALHRBD
GNITOUQGNIMRAWSPXOB PZTXOCRRCPDTIB
HQQUNFASTENEDPYLIMAFKEDOZCOHLXJNT
ZHUAENOI TCNUPMOC P AGGKRJTMAQHVKRGF
```

ACCORDED
ACUTELY
ADVANTAGE
AFFILIATED
BARE
BUSTS
CHAINED
CHIEFS
COMPUNCTION
COUNTERFEIT
CREEP
CREWMEN
DEPLETION
DIAGRAM
DRAFTING
ECONOMIC
EXPURGATED
FAVORABLE
FRESH
GUARANTOR
GUILLE
IDIOCY
INSCRIBING
INSPECTION
INVESTIGATOR
MAGISTERIAL
MIGRATING
MUCUS
NEGATIVE
OBJECTING
PLATTER
POISONED
QUOTING
REPUGNANT
SALESMAN
SCABBED
SEEMED
SPARKLING
SPLATTER
SQUASH
SWARMING
SYNTHESIZE
TEXAS
THOUGHTFULLY
UNFASTENED
VIOLATED
WAFFLED
WAVERED

Output

```
Masukkan nama file
Contoh: ../test/tc1.txt
>>> ../test/tc7.txt
```

```
-----
- D E D R O C C A -
-----
```

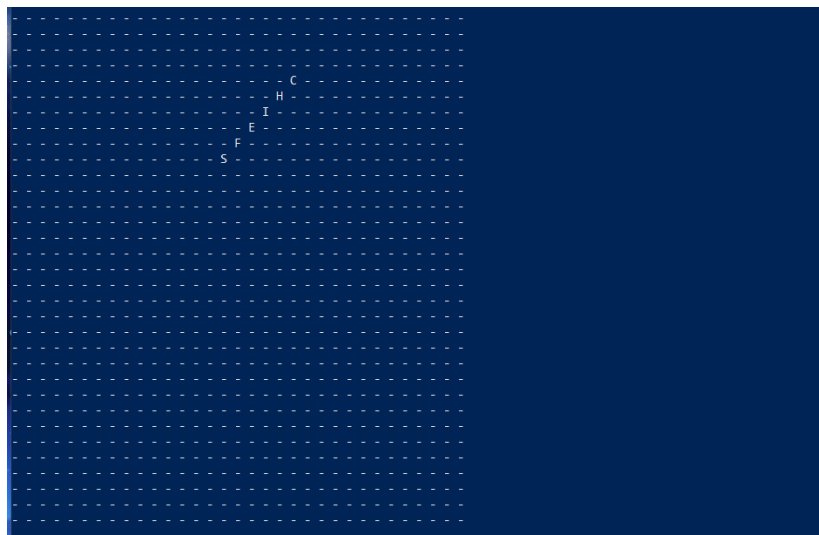
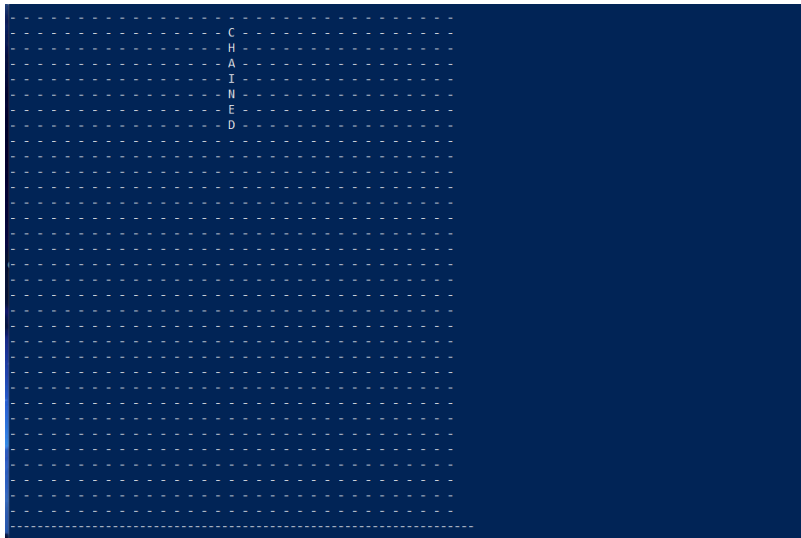
```
-----
- Y -
- L -
- E -
- T -
- U -
- C -
- A -
-----
```

```
-----
- E -
- G -
- A -
- T -
- N -
- A -
- V -
- D -
- A -
-----
```

- D -
- E -
- T -
- A -
- I -
- L -
- I -
- F -
- F -
- A -

- B -
- A -
- R -
- E -

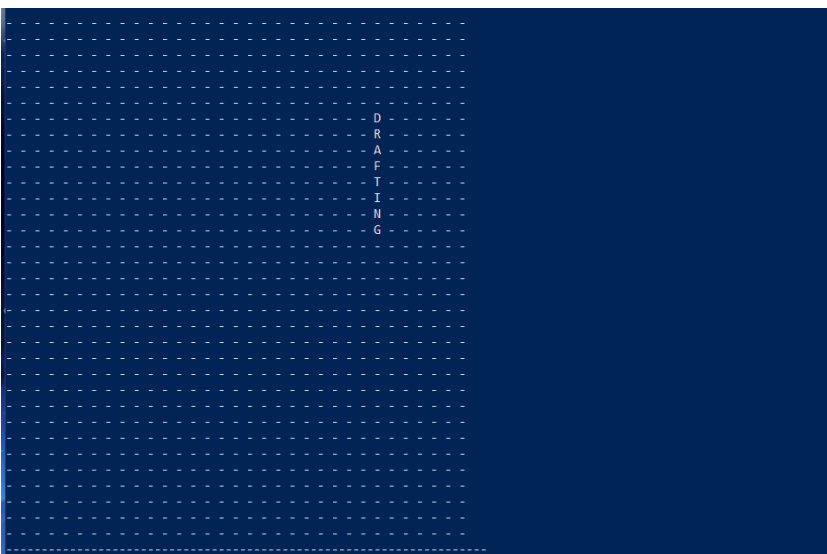
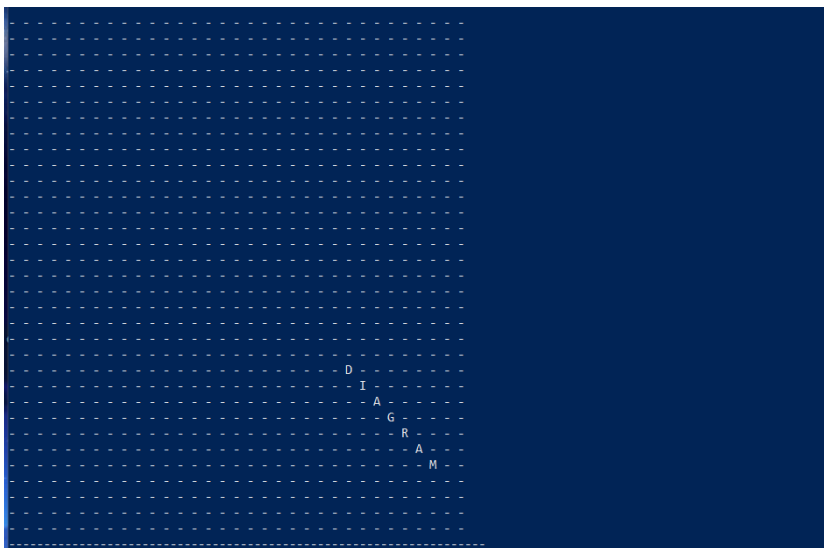
- B -
- U -
- S -
- T -
- S -



T
I
E
F
R
E
T
N
U
O
C

P
E
E
R
C

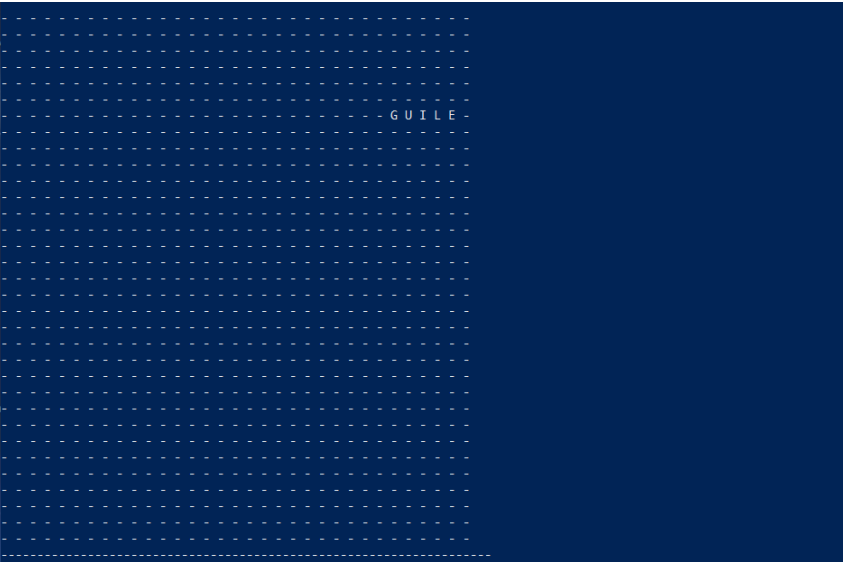
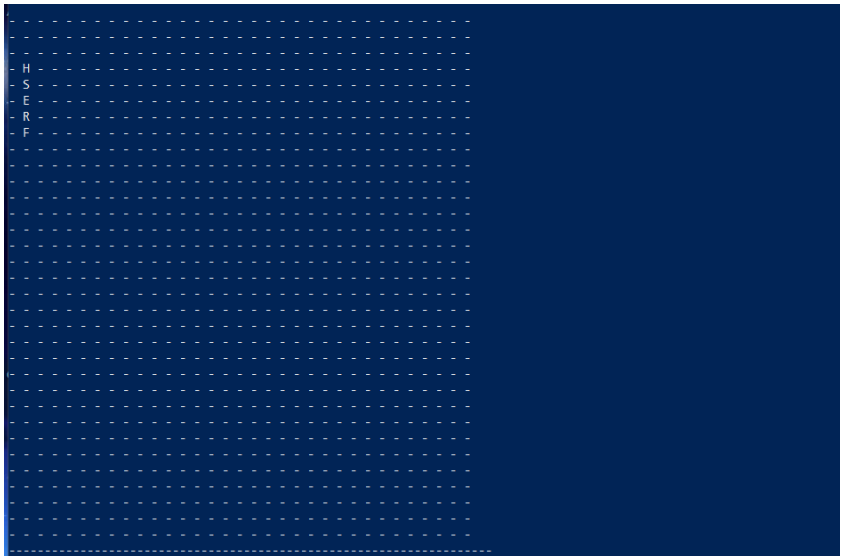
N
E
M
W
E
R
C



E
C
O
N
O
M
I
C

DETAGRUPXE

E
L
B
A
R
O
V
A
F



I
D
I
O
C
Y

I
N
S
C
R
I
B
I
N
G

I
N
S
P
E
C
T
I
O
N

ROTAGITSEVNI

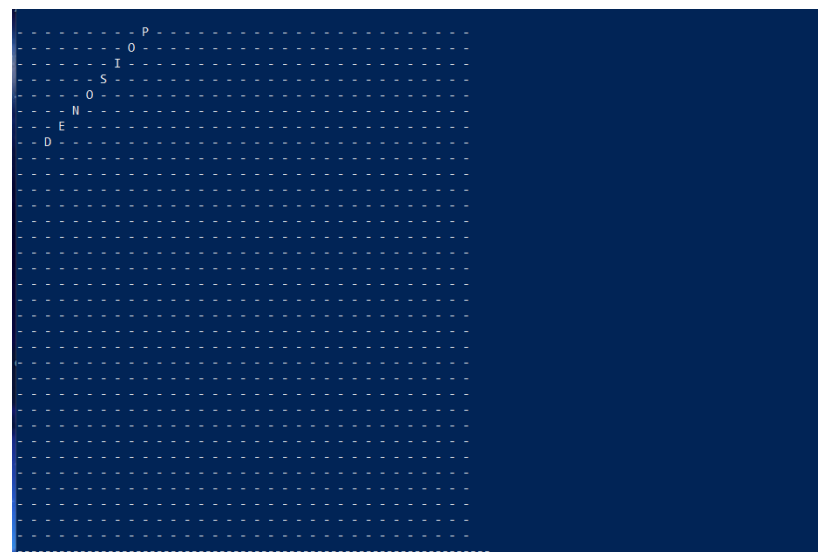
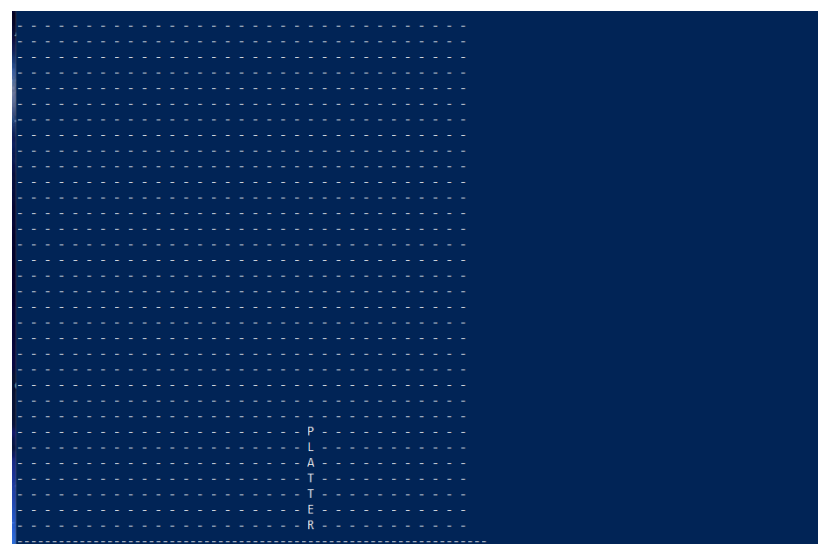
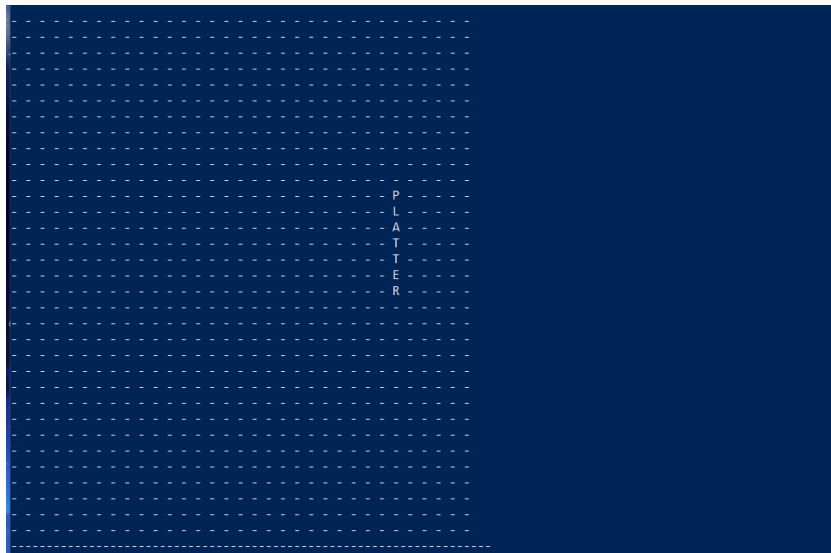
L
A
I
R
E
T
S
I
G
A
M

G
N
I
T
A
R
G
I
M

S
U
C
U
M

NEGATIVE

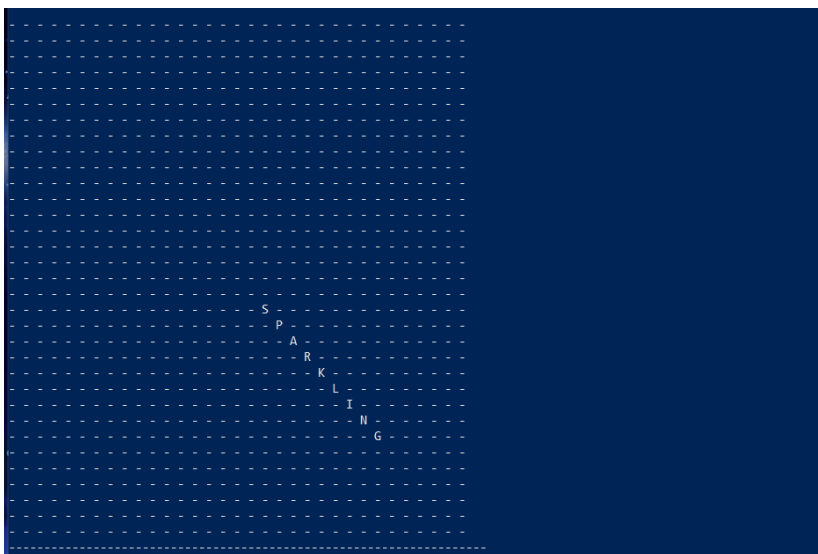
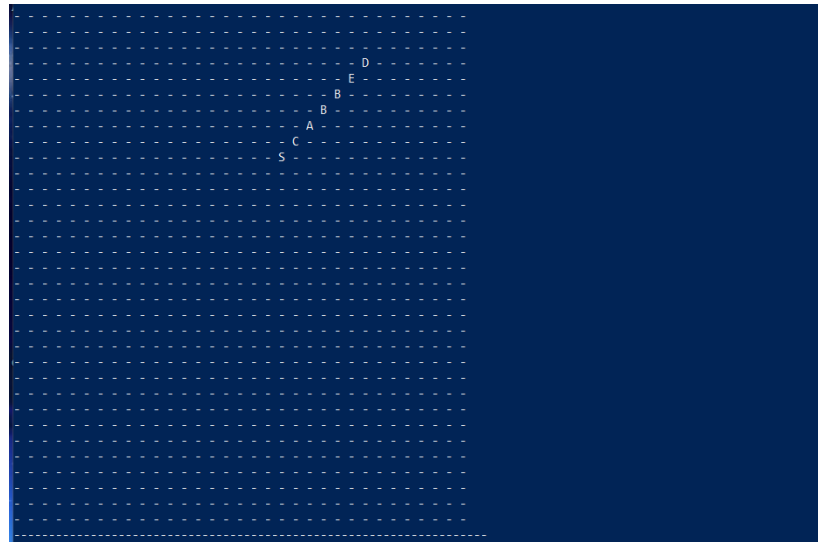
O
B
J
E
C
T
I
V
E



G N I T O U Q

T
N
A
N
G
U
P
E
R

N
A
M
S
E
L
A
S



S
P
L
A
T
T
E
R

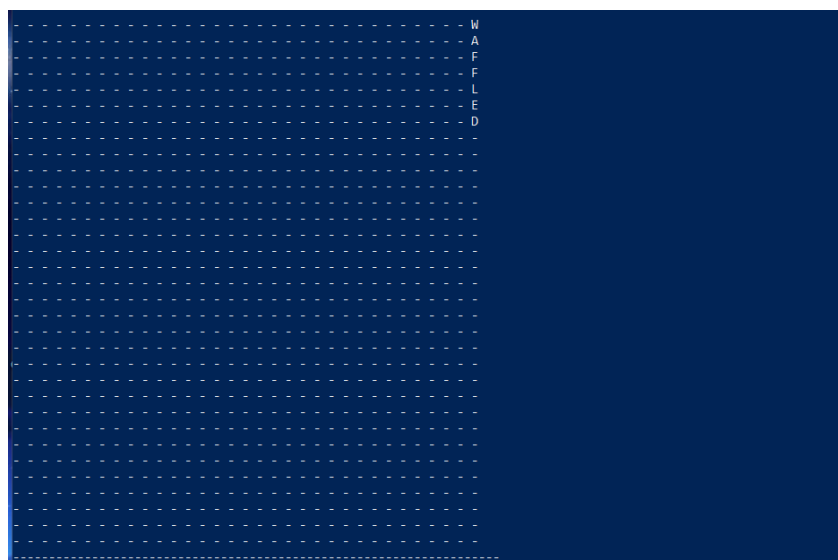
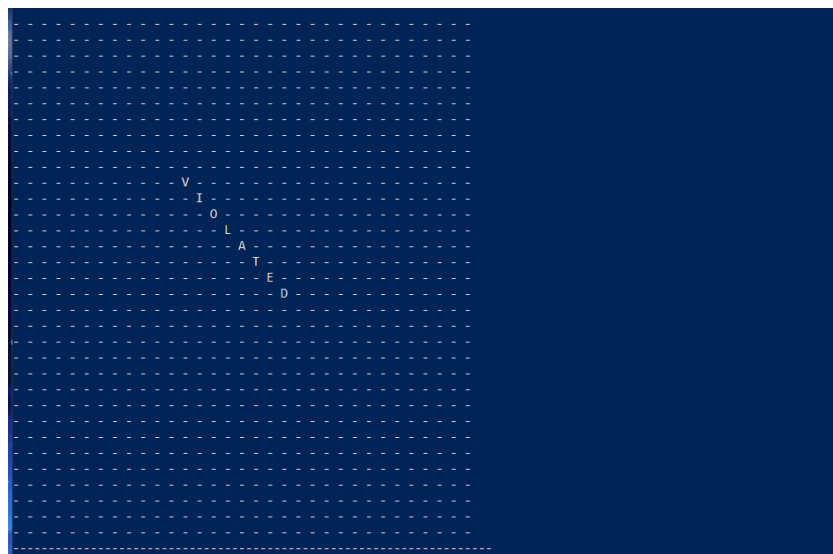
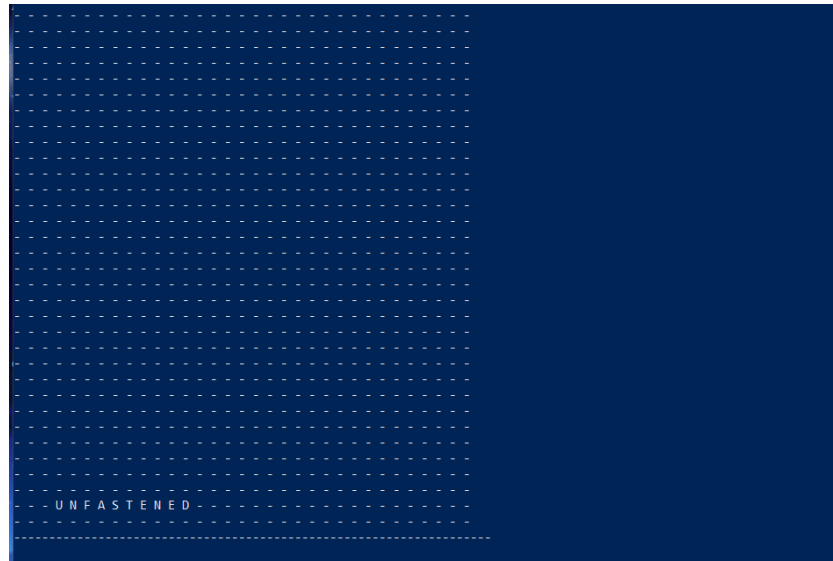
S
Q
U
A
S
H

G N I M R A W S

E
Z
I
S
E
H
T
N
Y
S

S
A
X
E
T


Y
L
U
F
T
H
G
U
O
H
T



[illegible]

Test Case 8 (Large)

Input

 tc8.txt - Notepad

File Edit Format View Help

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

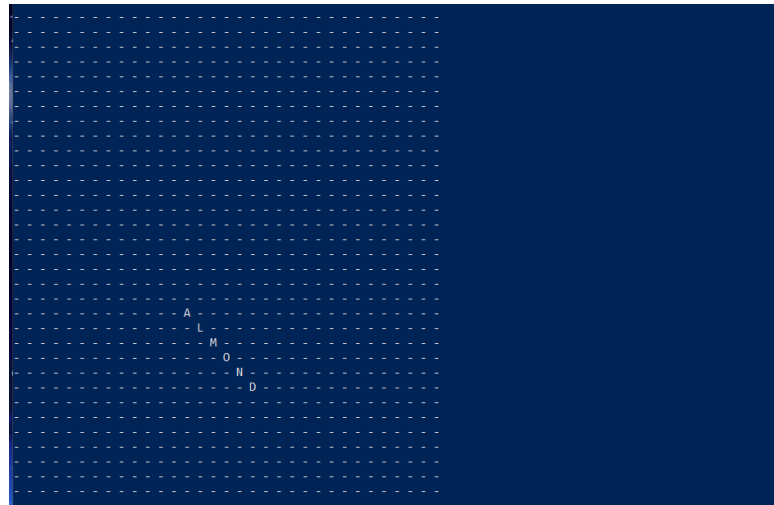
AGAVE
ALMOND
BASMATI
BOKCHOY
BREAD
CACAO
CAPSICUM
CASHEW
CHIVES
CUCUMBER
DRAGONFRUIT
GINGER
GRAPE
LEMONGRASS
OATMEAL
OYSTER
PASTA
POTATO
QUINOLA
RADISH
WATERMELON

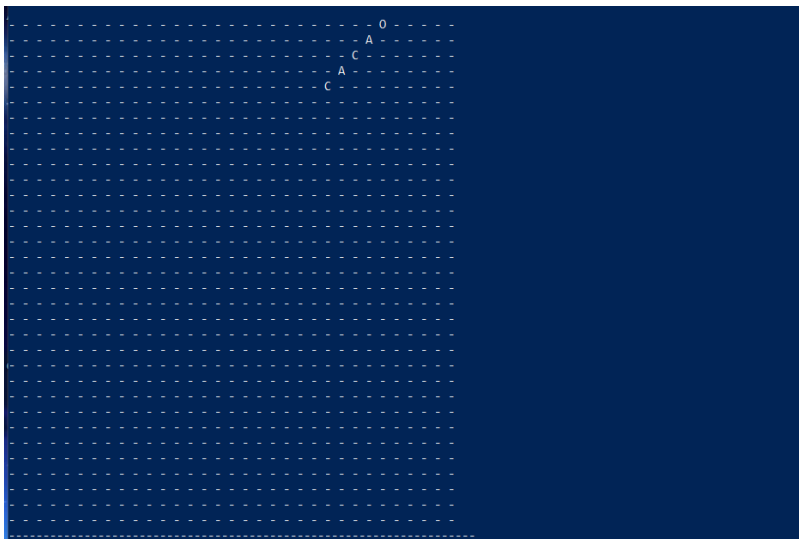
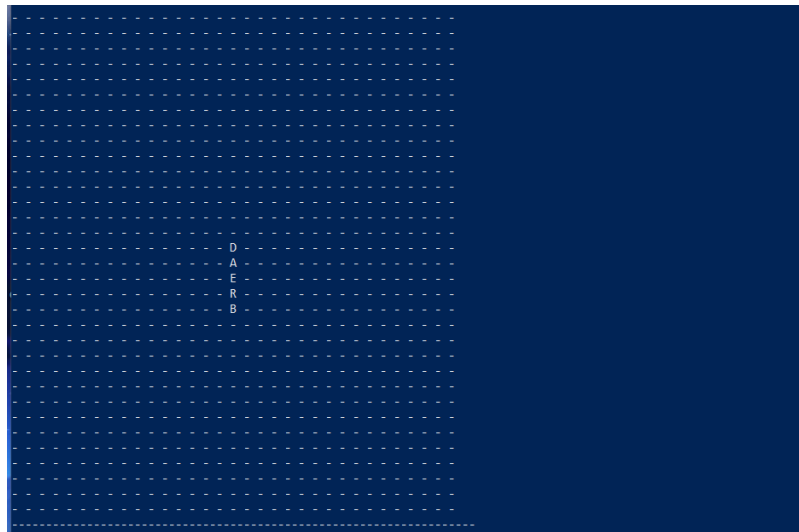
Output

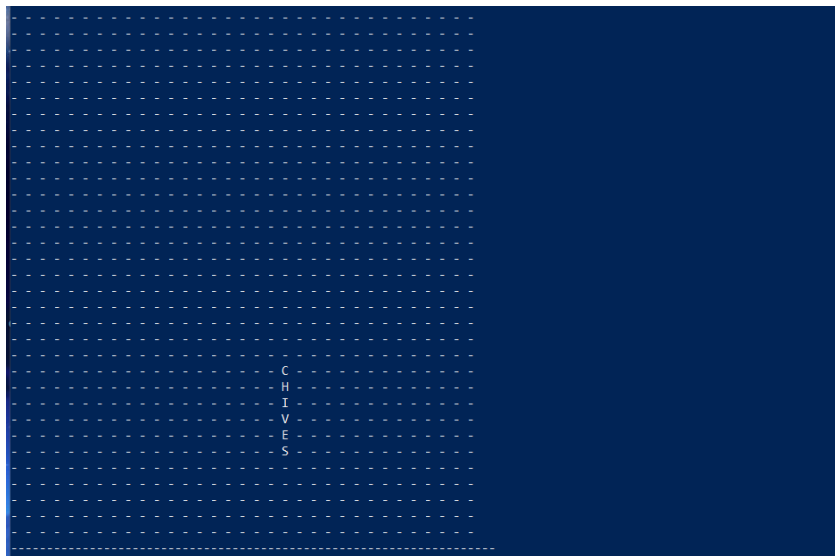
```
Masukkan nama file
Contoh: ../test/tc1.txt
```

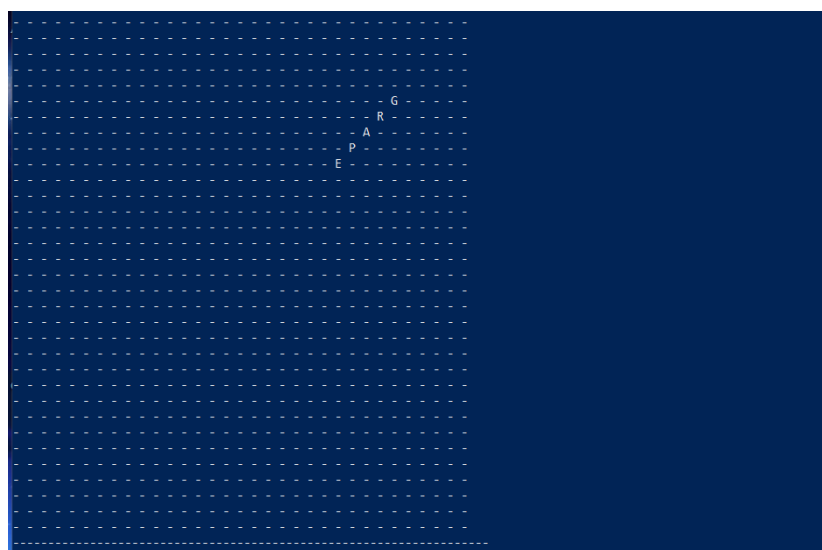
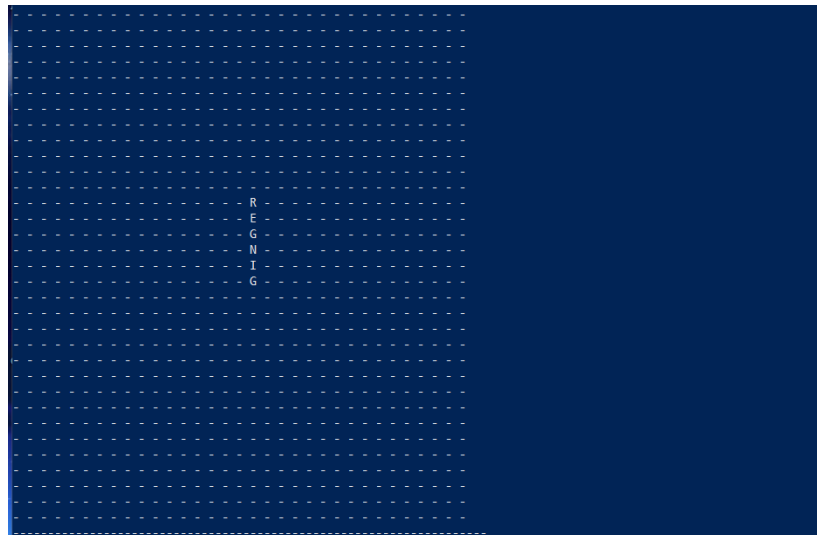
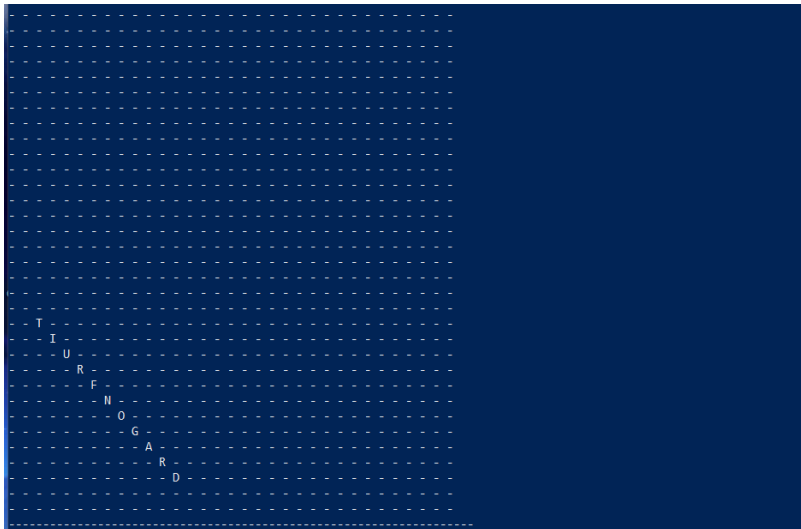
```
>>> ../test/tc8.txt
```

AGAVE









SSARGNOMEL

O
A
T
M
E
A
L

R
E
T
S
Y
O

P
A
S
T
A

O
T
A
T
O
P

A
L
O
N
I
U
Q

[illegible]

Test Case 9 (Large)

Input



tc9.txt - Notepad

File Edit Format View Help

```
CHLQCTVDSEMCNWUUXDQDLFIYPQZTIVKNR
TCCESUKTDWREANAQNOELOATSSENEROSUW
ZPLEFZSAFAKXMOIAJDDKINTLZJJJWHNK
R3WONSMMSMAKEOVERUWERFOEXIKESFEWH
COQATDSSIUHNLOIHKDFEGTYKMBLALLFOA
LZDUNTNDNKNJQZTRGZMRINDKWOALMWKRX
IKCAQEEEDTUMBBLEND EADFZITIWNOBOLKF
CAHYSSLDEJNEESIJTKUVANLRZNTERQGAX
WIMSHGZJRJXZDCNZCAWLCDBOFONMHESBL
EHMAKUKEFKUIMVOLUMEIMVEVDRROSIVPULZ
SNVHXTETAZQLGGLSMLTXULDCEISCDCTGEO
VELNTATSCCKBOHSKTIILRTIQDHTVADYIRVA
ICVIRIXJEOTLDIEQVNBTFVSNMTRL SRGHT
HDMOSQRNMVYOPDQDZUGZTUINEOEETGVRG
RNOGETIOHPCPPTDEVXUTCANPYBDMNYQKM
KETVNVMBGUMA ZBB C PRGNKFJTPDZEDPRIM
IVXMNIXEYLJOQCZNEEDITORIALLYNIAPEO
SYSP TZKINKAIAHYAUPWQSFEGMERYQECNO
IBAFOCIAZTQUWEETABUGLWGNIVLOVEDIR
YIIMMOVNETTBIEKSZUWOSQGL OYPQJOETA
ZEKAHMC AEPHFQMPIEASUKWUYBVIPZR DSE
PKGLVMDHFVSRQGGIDFSHJGGMKWNAEPDJIT
CQUSOXBPPIKJAWBIKBGPNBFUEJTOBFLF
FFLZPNLASWOTLJINDSVNNIMZOXDTINBIW
HNOITAMALCCAAFGREDNIKHTRAPPEDOFHG
XNKMGHDVMBXBBWB I KCLPDJCREFILEICNPV
BLGNIETSKASWTOLENNCAWNKSSSGRGBCKVE
ZIGOLXRSHLCUZDOYWHUWTULQRSUTPSXEW
GCGLWOSZPEJ CXMP PJNEX TAXEMUXUPEKSU
JUIGNLQWKRLNXDVL DNAUXLQBLBNSACARF
NRPSTWWETNETVIKICKCXVGYXOLGTOCGIE
DEWQFELAYRAKZOCOHOUN DINGVUDNLL OFY
FXOWUXMDTTZL OERKHQHFBKHEYQFWUNMTZ
```

ACCLAMATION
ALGORITHMIC
APOLOGIZE
AUDIT
BLEND
CLOTTED
COMMON
CRASSNESS
DEVOLVING
DEWY
DISTANCE
DRILLED
EDITORIALY
ENLISTMENT
FACELIFT
FRACAS
FRINGED
GRIT
HANDMADE
HOUNDING
INTERFACE
KINDER|
LANK
LAUNCHING
LOWEST
MAKEOVER
MEMOIR
MIDDLEMEN
MOTORED
PAIN
PHENOMENAL
PHILISTINE
PINCUSHION
REFILE
RENOVATION
SCARED
SHAVE
SORENESS
SPEAKING
TEAROOM
TEMERITY
TRAPPED
UNBECOMING
UNWORKABLE
WILTED

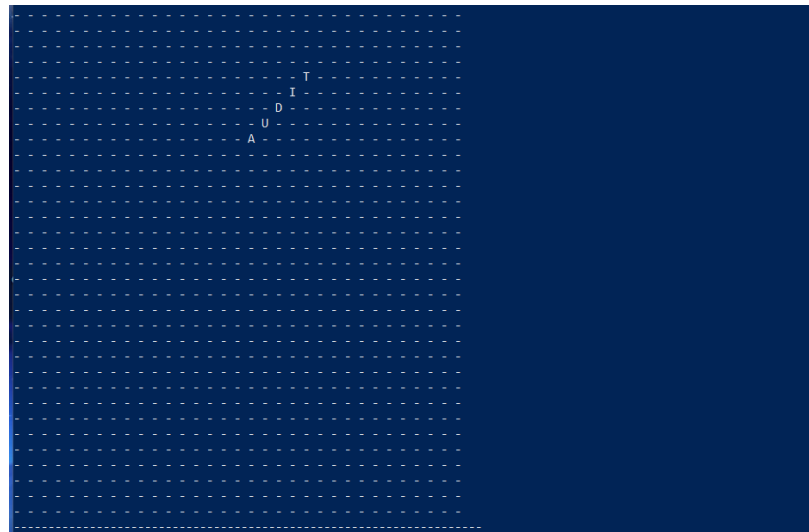
Output

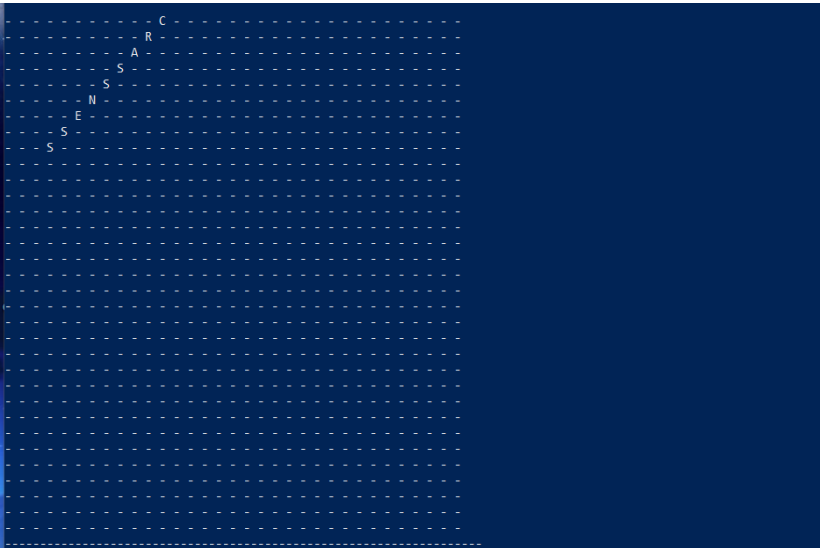
```
Masukkan nama file
Contoh: ../test/tc1.txt
>>> ../test/tc9.txt
```

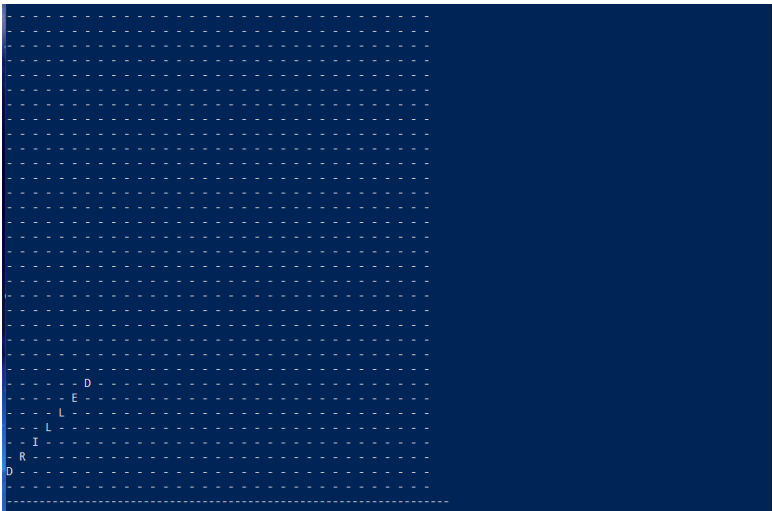
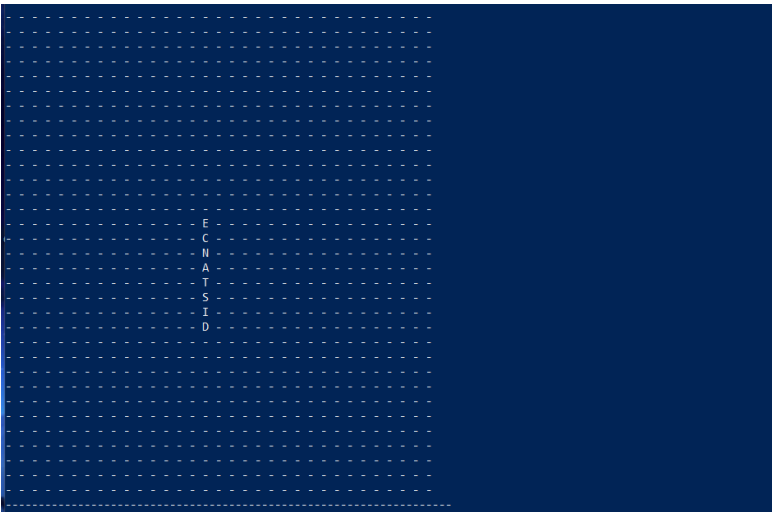
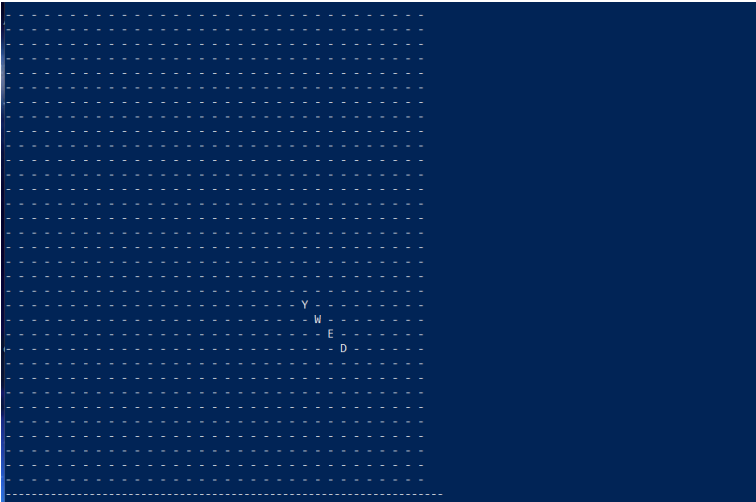
```
- NOITAMALCCA -
```

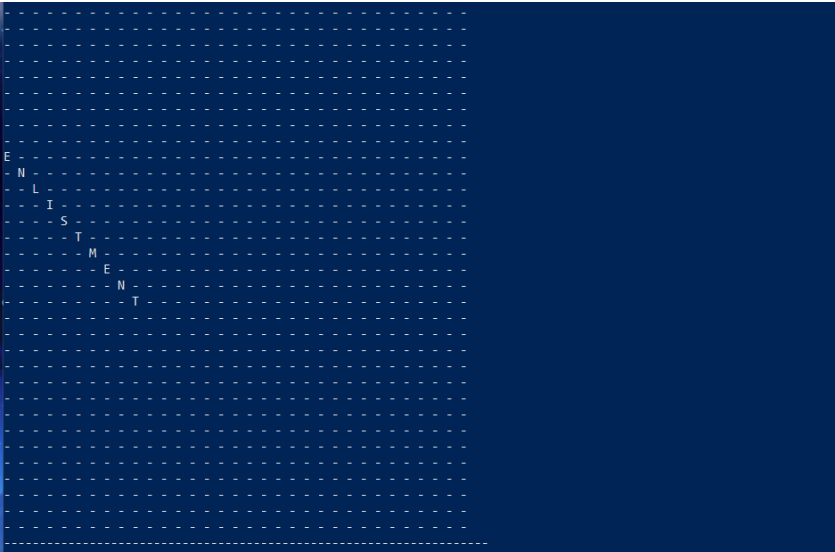
```
C
I
- M -
- H -
T
I
R
O
G
L
A
```

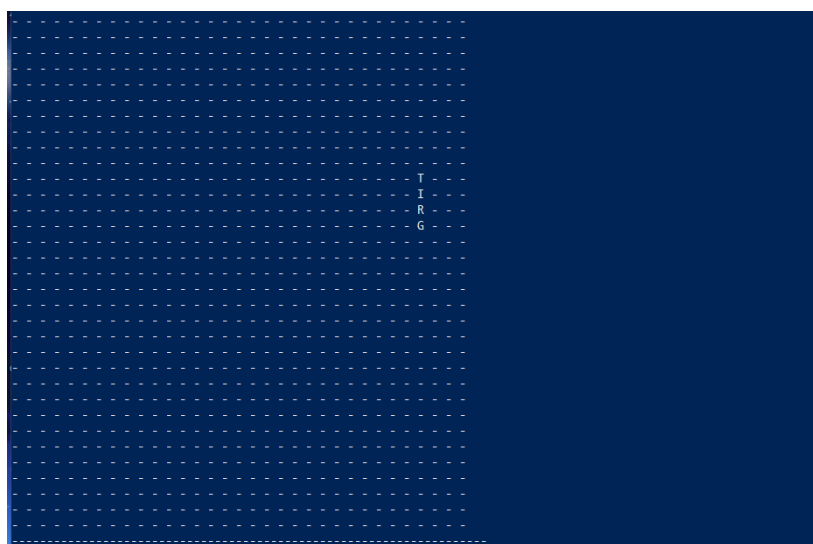
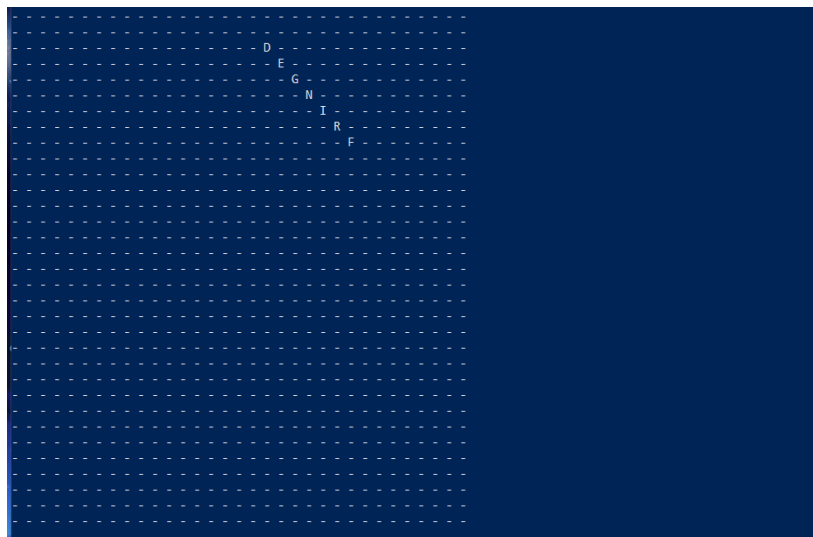
```
E
Z
I
G
O
L
O
P
A
```





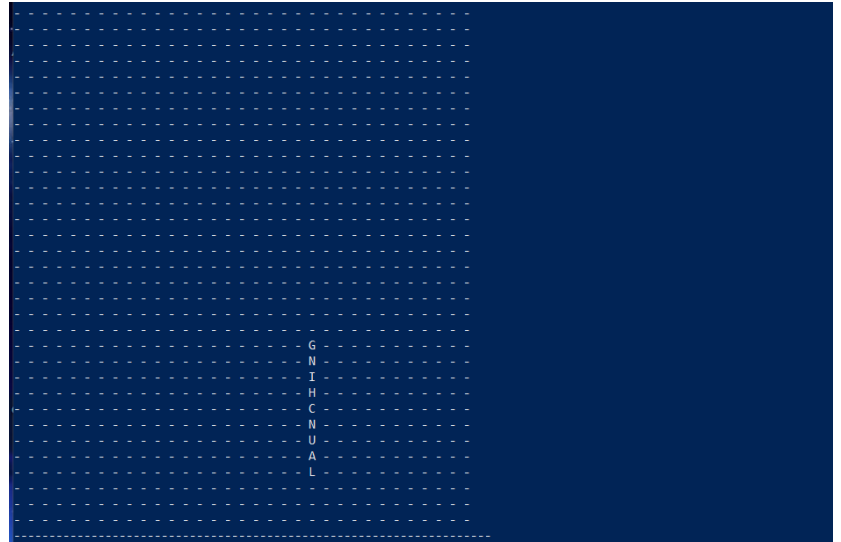
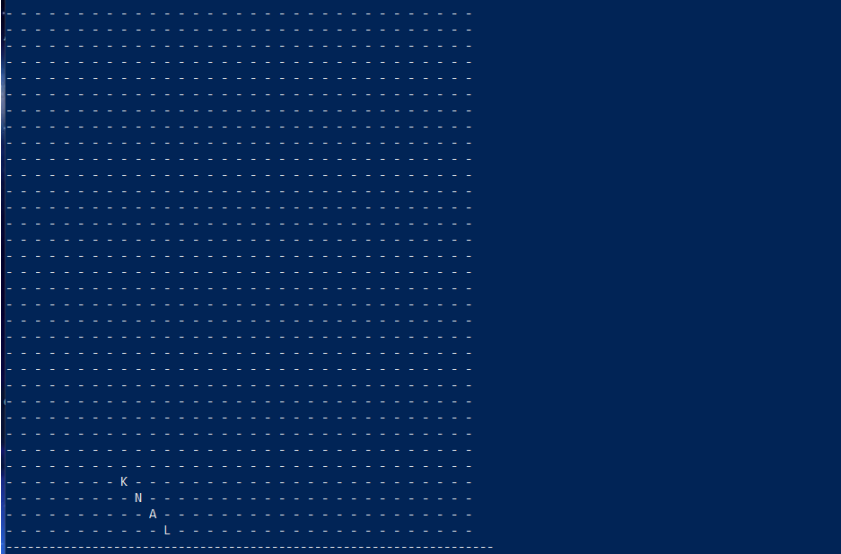


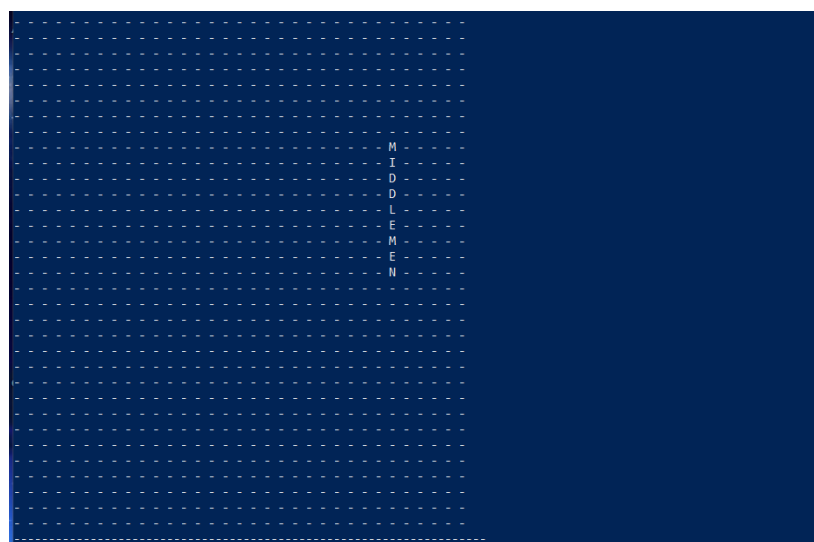
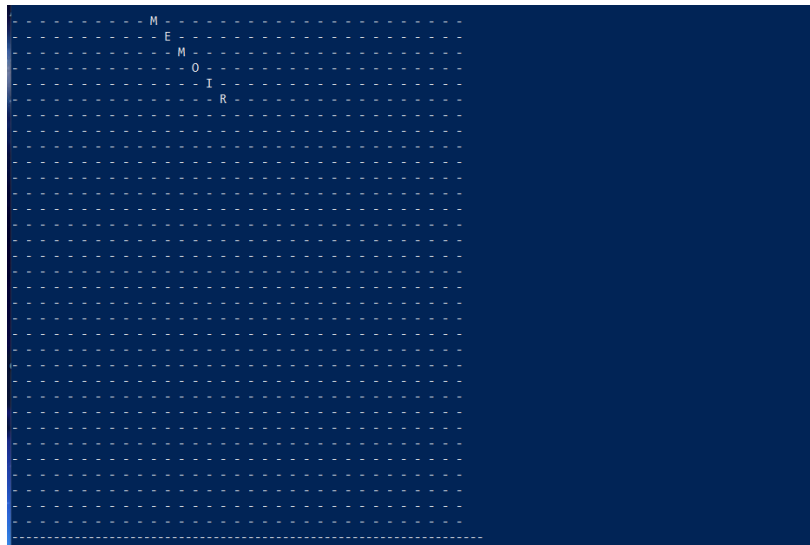


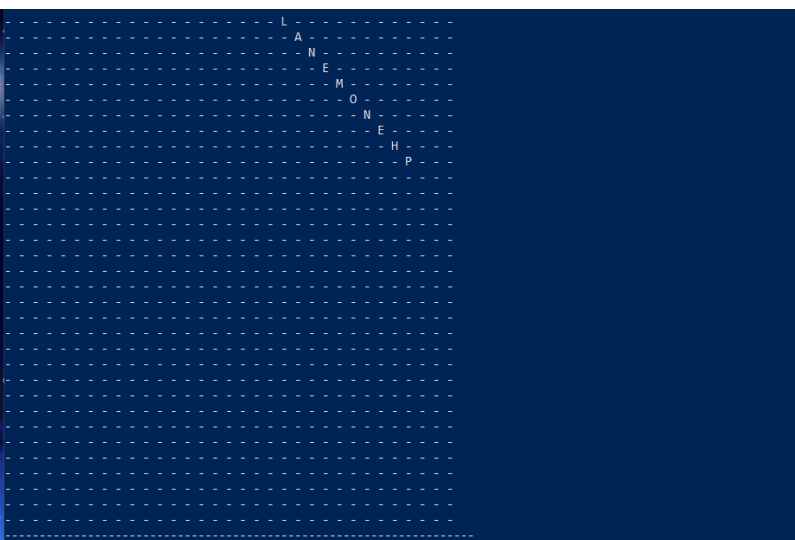
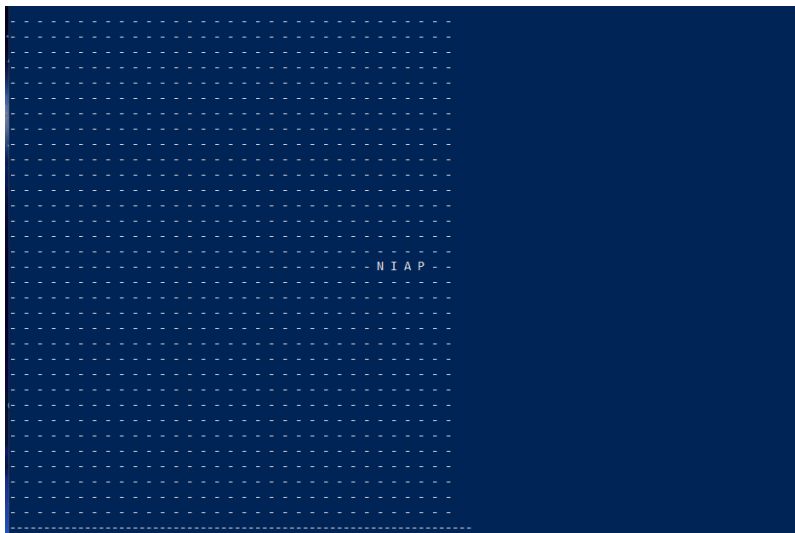
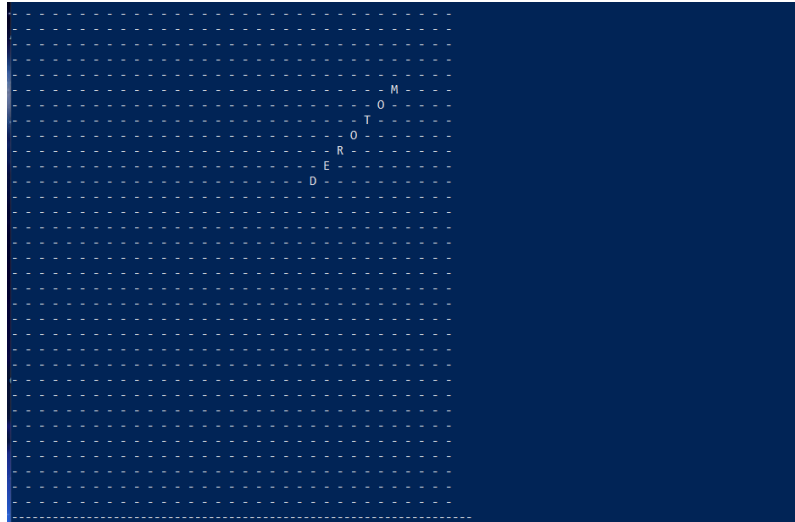


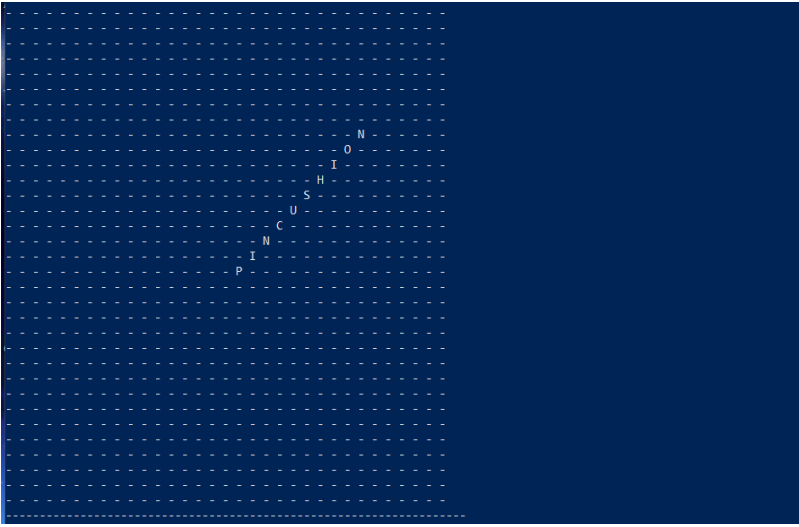


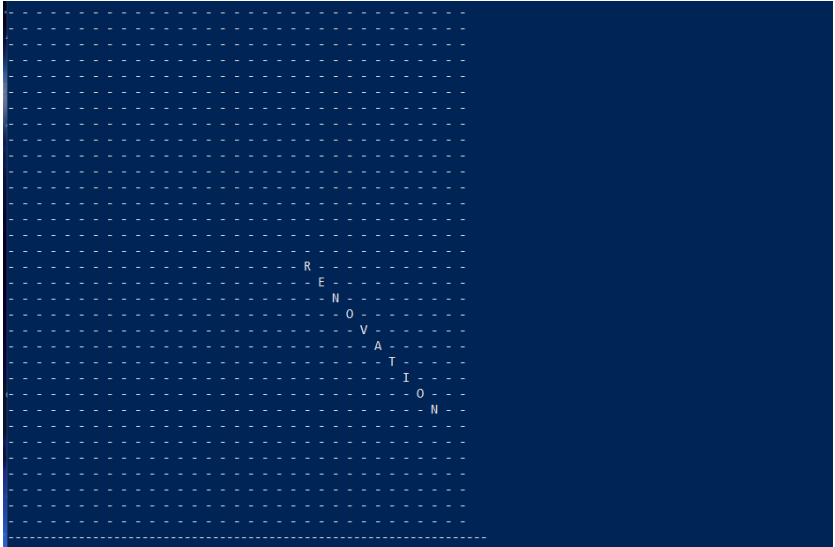


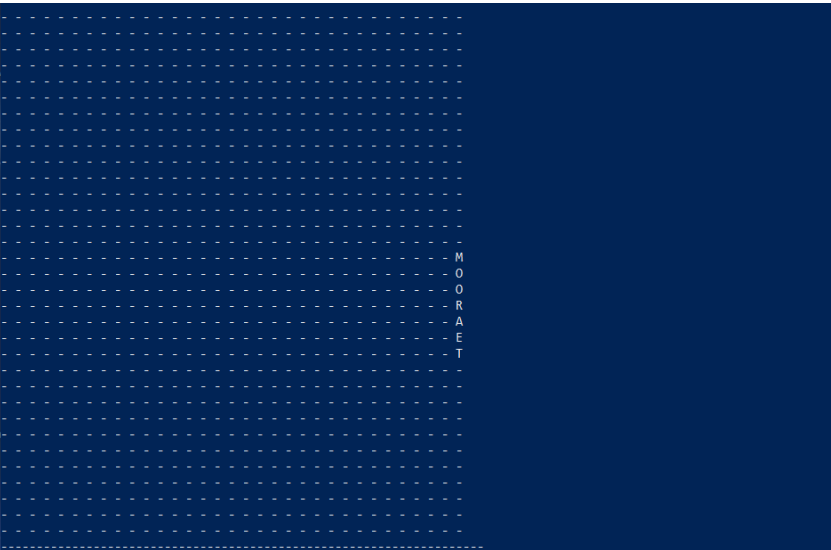


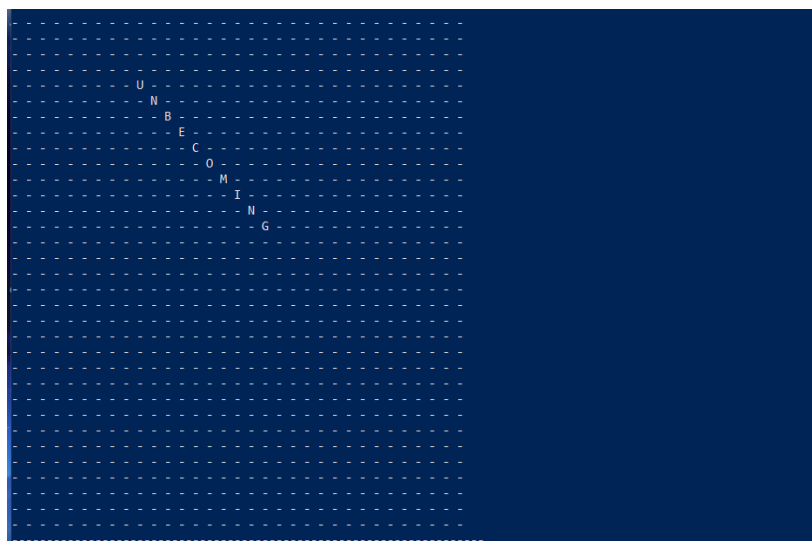
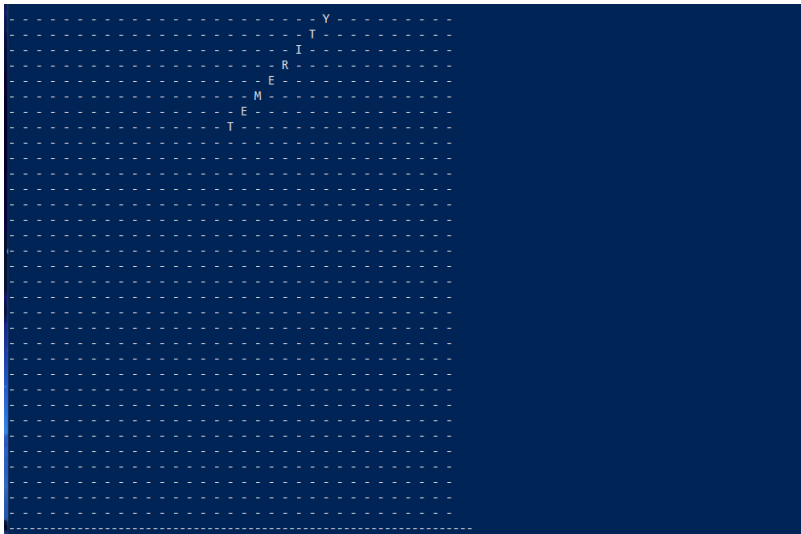












```
U
N
W
O
R
K
A
B
L
E
```

```
W
I
L
T
E
D
```

Waktu yang diperlukan untuk mengeksekusi program adalah 2.89278 detik
Banyak operasi perbandingan kata yang dilakukan program ini adalah 31924 perbandingan

BAB V

TABEL PENILAIAN

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