

**LAPORAN TUGAS KECIL 1**  
**IF2211 STRATEGI ALGORITMA**



**NAJWA KAHANI FATIMA**

**13523043 - K1**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2025**

## **DAFTAR ISI**

BAB I DESKRIPSI TUGAS.....	2
BAB II ALGORITMA BRUTE FORCE DAN PENERAPANNYA.....	3
2.1 Algoritma Brute Force.....	3
2.2 Penerapan Algoritma Brute Force.....	3
2.3 Pseudo Code.....	4
BAB III SOURCE CODE DAN HASIL TANGKAPAN LAYAR.....	8
3.1 Source Code.....	8
3.2 Tangkapan Layar.....	10
LAMPIRAN.....	14

## **BAB I**

### **DESKRIPSI TUGAS**

IQ Puzzler Pro adalah permainan papan yang diproduksi oleh perusahaan Smart Games. Tujuan dari permainan ini adalah pemain harus dapat mengisi seluruh papan dengan piece (blok puzzle) yang telah tersedia. Komponen penting dari permainan IQ Puzzler Pro terdiri dari:

1. Board (Papan) – Board merupakan komponen utama yang menjadi tujuan permainan dimana pemain harus mampu mengisi seluruh area papan menggunakan blok-blok yang telah disediakan.
2. Blok/Piece – Blok adalah komponen yang digunakan pemain untuk mengisi papan kosong hingga terisi penuh. Setiap blok memiliki bentuk yang unik dan semua blok harus digunakan untuk menyelesaikan puzzle.

Permainan dimulai dengan papan yang kosong. Pemain dapat meletakkan blok puzzle sedemikian sehingga tidak ada blok yang bertumpang tindih (kecuali dalam kasus 3D). Setiap blok puzzle dapat dirotasikan maupun dicerminkan. Puzzle dinyatakan selesai jika dan hanya jika papan terisi penuh dan seluruh blok puzzle berhasil diletakkan.

Persoalan pada tugas kecil ini adalah menemukan cukup satu solusi dari permainan IQ Puzzler Pro dengan menggunakan algoritma Brute Force, atau menampilkan bahwa solusi tidak ditemukan jika tidak ada solusi yang mungkin dari puzzle.

## BAB II

### ALGORITMA BRUTE FORCE DAN PENERAPANNYA

#### 2.1 Algoritma Brute Force

Brute force adalah sebuah pendekatan yang langsung (*straightforward*) untuk memecahkan suatu masalah, biasanya didasarkan pada pernyataan masalah (*problem statement*) dan definisi konsep yang dilibatkan. Algoritma brute force memecahkan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas (Bahan Ajar IF2211 Strategi Algoritma, Rinaldi Munir).

Algoritma brute force memiliki karakteristik, yaitu: 1) Umumnya tidak sangkil karena membutuhkan *cost* komputasi yang besar dan waktu yang lama dalam penyelesaiannya, 2) Algoritma brute force lebih cocok untuk persoalan dengan ukuran masukan kecil, 3) Hampir seluruh persoalan dapat diselesaikan dengan algoritma brute force.

#### 2.2 Penerapan Algoritma Brute Force

Pada tugas kecil ini, digunakan algoritma brute force untuk mencari satu solusi (solusi pertama) dari permainan IQ Puzzler Pro. Input data permainan yang berisi ukuran papan ( $N \times M$ ), banyaknya blok ( $P$ ), model konfigurasi ( $S$ ), dan masing-masing bentuk blok dibaca melalui file .txt. Setiap bagian dari blok akan dibaca lokasinya melalui sistem koordinat kartesian. Dalam penerapannya di sistem matrix, posisi  $x$  akan digunakan sebagai kolom dan posisi  $y$  akan digunakan sebagai baris. Setelah dioperasikan pada algoritma brute force untuk pencarian solusi, hasil pencarian akan ditampilkan dan disimpan pada file .txt (opsional).

Pengecekan pengisian papan dilakukan dengan menggunakan papan matriks bertipe boolean. Jika terisi, maka elemen bernilai *true*. Jika kosong, maka elemen papan bernilai *false*. Blok yang telah digunakan akan disimpan dalam suatu Array beserta keterangan atributnya (orientasi dan bagian blok referensi).

Berikut adalah tahapan algoritma brute force dalam penyelesaian puzzle ini dengan input yang valid menggunakan metode `fitBlocksToBoard()` pada kelas Solver:

1. Program mengecek bagian kotak/elemen papan yang masih belum terisi (nilai false) dengan menggunakan method `curCoord()`.
2. Program melakukan iterasi pengecekan secara berurut untuk setiap:

- a. Blok yang tersisa (belum digunakan)
- b. Bagian/elemen blok sebagai referensi

Setiap bagian dari blok akan dicoba untuk digunakan sebagai referensi/acuan dalam pengisian papan. Dalam kata lain, program akan mentranslasikan koordinat blok sehingga koordinat referensi bernilai (0,0).

- c. Variasi bentuk blok berdasarkan orientasi

Dalam permainan ini, terdapat delapan kemungkinan orientasi atau variasi bentuk untuk masing-masing blok, yaitu 1) Default (sesuai input), 2) Rotasi 90 derajat, 3) Rotasi 180 derajat, 4) Rotasi 270 derajat, 5) Pencerminkan terhadap sumbu-y dari 1), 6) Pencerminkan terhadap sumbu-x dari 1), 7) Pencerminkan terhadap sumbu-y dari 2), dan 8) Pencerminkan terhadap sumbu-x dari 2). Variasi blok didapat menggunakan transformasi koordinat pada method `changeOrientation()` pada kelas `Block`.

3. Program meletakkan blok dengan orientasi dan referensi yang telah diatur pada papan dengan method `placeBlock()`. Jika valid, maka atribut lokasi blok pada papan, bagian referensi/acuan, dan orientasi diperbarui. Lalu, blok akan ditambahkan pada Array blok solusi. Peletakkan blok dikatakan valid jika tidak ada posisi dari bagian/elemen blok yang melewati batas ukuran papan atau kotak pada papan belum terisi.
4. Dilakukan pencocokan blok selanjutnya pada papan dengan sisa blok yang ada.
5. Setiap iterasi, dilakukan pengecekan kondisi papan menggunakan metode `isFull()`. Jika papan sudah terisi penuh (semua elemen bernilai true) namun masih ada blok tersisa atau papan masih belum penuh namun blok sudah habis digunakan, maka program akan mengembalikan nilai false (tidak ada solusi).
6. Jika papan sudah terisi penuh dan tidak ada blok tersisa, maka solusi ditemukan.

## 2.3 Pseudo Code

```

FUNCTION fitBlocksToBoard(remainBlocks, fitBlocks, curBoard)
RETURNS boolean

    IF isFull(curBoard) THEN
        IF remainBlocks IS EMPTY THEN
            solution ← copy of fitBlocks

```

```

        RETURN true
    ELSE
        RETURN false
    END IF
ELSE IF remainBlocks IS EMPTY THEN
    RETURN false
END IF

currCheckCoord ← curCoord(curBoard)

FOR each block IN remainBlocks DO
    newBlock ← copy of block

    FOR ref FROM 0 TO size of newBlock.positions - 1 DO
        FOR or FROM 1 TO 8 DO
            curBlockOrient ←
            Block.changeOrientation(newBlock.positions,
            ref, or)
            counter ← counter + 1

            newBoard ← placeBlock(currCheckCoord.y,
            currCheckCoord.x, curBlockOrient, curBoard)

            IF newBoard IS NOT NULL THEN
                newBlock.setLocation(currCheckCoord)
                newBlock.setRef(ref)
                newBlock.setPosition(curBlockOrient)

                newFitBlocks ← copy of fitBlocks
                newFitBlocks.add(newBlock)

                newRemainBlocks ← copy of remainBlocks
                newRemainBlocks.remove(newBlock)

                IF fitBlocksToBoard(newRemainBlocks,
                newFitBlocks, newBoard) THEN
                    RETURN true
                END IF
            END IF
        END FOR
    END FOR
END FOR

RETURN false
END FUNCTION

```

FUNCTION **isFull(board)** RETURNS boolean

```

    FOR each row IN board DO

```

```

        FOR j FROM 0 TO length of board[0] - 1 DO
            IF row[j] IS false THEN
                RETURN false
            END IF
        END FOR
    END FOR

    RETURN true
END FUNCTION

```

FUNCTION **curCoord(board)** RETURNS Coordinate

```

    FOR i FROM 0 TO length of board - 1 DO
        FOR j FROM 0 TO length of board[0] - 1 DO
            IF board[i][j] IS false THEN
                RETURN Coordinate(j, i)
            END IF
        END FOR
    END FOR

    RETURN null
END FUNCTION

```

FUNCTION **placeBlock(startY, startX, position, board)** RETURNS boolean[][]

```

    newBoard ← copyBoard(board)

    FOR each pos IN position DO
        row ← startY + pos.y
        col ← startX + pos.x

        IF row < 0 OR col < 0 OR row ≥ length of board OR col
        ≥ length of board[0] OR newBoard[row][col] IS true
        THEN
            RETURN null
        END IF

        newBoard[row][col] ← true
    END FOR

    RETURN newBoard
END FUNCTION

```

FUNCTION **changeOrientation(init, ref, ver)** RETURNS ArrayList<Coordinate>

```

    newPos ← copyPosition(init)
    reff ← init[ref] // Koordinat referensi

```

```

// Translasi koordinat berdasarkan elemen referensi
FOR each coord IN newPos DO
    x ← coord.x - reff.x
    y ← coord.y - reff.y

    // Orientasi berdasarkan nilai ver
    SWITCH ver DO
        CASE 1:
            // Tidak ada perubahan
        CASE 2:
            coord.setCoordinate(-x, y)
        CASE 3:
            coord.setCoordinate(x, -y)
        CASE 4:
            coord.setCoordinate(y, -x)
        CASE 5:
            coord.setCoordinate(y, x)
        CASE 6:
            coord.setCoordinate(-y, -x)
        CASE 7:
            coord.setCoordinate(-x, -y)
        CASE 8:
            coord.setCoordinate(-y, x)
    END SWITCH
END FOR

RETURN newPos
END FUNCTION

```



## BAB III

### SOURCE CODE DAN HASIL TANGKAPAN LAYAR

#### 3.1 Source Code

```
1 import java.util.ArrayList;
2
3 public class Solver {
4     public static ArrayList<Block> solution; //storing blocks that fit (sudah disesuaikan dg orientasinya)
5     public static boolean[][] board; //default value is false
6     public static int counter = 0;
7
8     // Check if board is full
9     public static boolean isFull(boolean[][] p){
10         for (boolean[] p1 : p) {
11             for (int j = 0; j < p[0].length; j++){
12                 if (p1[j] == false) {
13                     return false;
14                 }
15             }
16         }
17         return true;
18     }
19
20     // Check the top empty space left -- current coordinate
21     public static Coordinate curCoord(boolean[][] b){
22         for (int i = 0; i < b.length; i++){
23             for (int j = 0; j < b[0].length; j++){
24                 if (b[i][j] == false){
25                     return new Coordinate(j, i);
26                 }
27             }
28         }
29         return null;
30     }
31
32     // Penyelesaian game
33     public static boolean fitBlocksToBoard(ArrayList<Block> remainBlocks, ArrayList<Block> fitBlocks, boolean[][] curBoard){
34
35         if (isFull(curBoard)){
36             if (remainBlocks.isEmpty()){
37                 //Solusi didapatkan
38                 solution = new ArrayList<>(fitBlocks);
39                 return true;
40             } else {
41                 // System.out.println("cek1");
42                 return false;
43             }
44         } else if (remainBlocks.isEmpty()){
45             // System.out.println("cek2");
46             return false;
47         } else {
48
49             Coordinate currCheckCoord = curCoord(curBoard);
50             // System.out.println("x " + currCheckCoord.getX() + " y " + currCheckCoord.getY());
51             for (int i = 0; i < remainBlocks.size(); i++){
52                 Block block = new Block(remainBlocks.get(i));
53
54                 // for each block, try each element as a ref
55                 for (int ref = 0; ref < block.getPosition().size() ; ref++){
56
57                     // for each element as ref, try every orientation
58                     for (int or = 1; or <= 8; or++) {
59                         ArrayList<Coordinate> curBlockOrient = Block.changeOrientation(block.getPosition(), ref, or);
60                         counter++;
61
62                         boolean[][] newBoard = placeBlock(currCheckCoord.getY(), currCheckCoord.getX(), curBlockOrient, curBoard);
63
64                         if (newBoard != null){
```

```

65
66         // System.out.println("fit.");
67         block.setLocation(currCheckCoord);
68         block.setRef(ref);
69         block.setPosition(curBlockOrient);
70
71         ArrayList<Block> newFitBlocks = new ArrayList<>(fitBlocks);
72         newFitBlocks.add(block);
73         ArrayList<Block> newRemainBlocks = new ArrayList<>(remainBlocks);
74         newRemainBlocks.remove(block);
75
76         if (fitBlocksToBoard(newRemainBlocks, newFitBlocks, newBoard)){
77             return true;
78         }
79     }
80 }
81 }
82 }
83 // System.out.println("cek3");
84 return false;
85 }
86 }
87
88 public static boolean[][] placeBlock(int startY, int startX, ArrayList<Coordinate> position, boolean[][] board) {
89     boolean[][] newBoard = copyBoard(board);
90     for (Coordinate pos : position) {
91         int row = startY + pos.getY();
92         int col = startX + pos.getX();
93
94         if (row < 0 || col < 0 || row >= board.length || col >= board[0].length || newBoard[row][col]) {
95             return null; // invalid placement
96         }
97         newBoard[row][col] = true;
98     }
99     return newBoard;
100 }
101
102 public static boolean[][] copyBoard(boolean[][] b){
103     boolean[][] newBoard = new boolean[b.length][b[0].length];
104     for (int i = 0; i < b.length; i++) {
105         for (int j = 0; j < b[0].length; j++){
106             newBoard[i][j] = b[i][j];
107         }
108     }
109     return newBoard;
110 }
111
112 public static void checkBoard(boolean[][] b){
113     for (int i = 0; i < b.length; i++){
114         for (int j = 0; j < b[0].length; j++){
115             if (b[i][j]) {
116                 System.out.print("T");
117             } else {
118                 System.out.print("F");
119             }
120         }
121         System.out.println();
122     }
123 }
124
125 public static void showSolution(Game game, ArrayList<Block> solution){
126     char[][] solBoard = new char[board.length][board[0].length];
127     for (Block block : solution){
128         int locX = block.getLocation().getX();
129         int locY = block.getLocation().getY();
130         for (Coordinate pos : block.getPosition()){
131             solBoard[pos.getY() + locY][pos.getX() + locX] = block.getLetter();
132         }
133     }

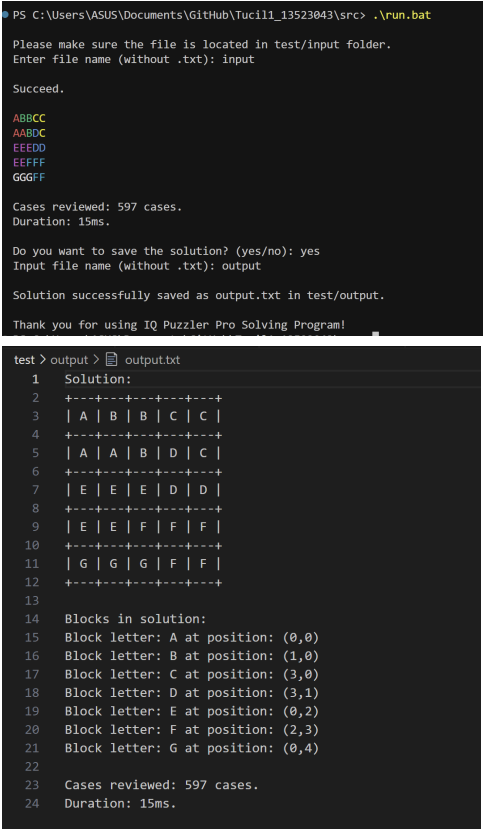
```

```

134     for (int i = 0; i < solBoard.length; i++){
135         for (int j = 0; j < solBoard[0].length; j++){
136             String color = Game.getColorFromLetter(game, solBoard[i][j]);
137             if (solBoard[i][j] >= 'A' && solBoard[i][j] <= 'Z'){
138                 System.out.print(color + solBoard[i][j] + Setup.colors[0]);
139             } else {
140                 System.out.print(" ");
141             }
142         }
143         System.out.println();
144     }
145 }
146 }

```

### 3.2 Tangkapan Layar

Kasus	Input	Output
1 - DEFAULT	5 5 7 DEFAULT A AA B BB C CC D DD EE EE E FF FF F GGG	 <pre> PS C:\Users\ASUS\Documents\GitHub\Tuc111_13523043\src&gt; .\run.bat  Please make sure the file is located in test/input folder. Enter file name (without .txt): input  Succeed.  ABBC AABDC EEED EEFF GGFF  Cases reviewed: 597 cases. Duration: 15ms.  Do you want to save the solution? (yes/no): yes Input file name (without .txt): output  Solution successfully saved as output.txt in test/output.  Thank you for using IQ Puzzler Pro Solving Program!  test &gt; output &gt; output.txt 1  Solution: 2  +---+---+---+---+ 3    A   B   B   C   C   4  +---+---+---+---+ 5    A   A   B   D   C   6  +---+---+---+---+ 7    E   E   E   D   D   8  +---+---+---+---+ 9    E   E   F   F   F   10 +---+---+---+---+ 11   G   G   G   F   F   12 +---+---+---+---+ 13 14 Blocks in solution: 15 Block letter: A at position: (0,0) 16 Block letter: B at position: (1,0) 17 Block letter: C at position: (3,0) 18 Block letter: D at position: (3,1) 19 Block letter: E at position: (0,2) 20 Block letter: F at position: (2,3) 21 Block letter: G at position: (0,4) 22 23 Cases reviewed: 597 cases. 24 Duration: 15ms. </pre>

<p>2 - DEFAULT</p>	<p>8 8 12          DEFAULT          A          A          BBBB          BBB          C          CC          D          EE          EE          FFFF          GGGG          HHHH          H          H          H          H          I I I I I          I I          I I          I I I I I          J J J          J J          J          K K          K          L L          L L</p>	<pre>PS C:\Users\ASUS\Documents\GitHub\Tucill1_13523043\src&gt; .\run.bat  Please make sure the file is located in test/input folder. Enter file name (without .txt): input2  Succeed.  ABBBBCCD AEEBBBCC FGEHMMH FGIIIIH FGJJJJH FGJJJKH LLJJKKH LLIIIIH  Cases reviewed: 435930 cases. Duration: 229ms.  Do you want to save the solution? (yes/no): yes Input file name (without .txt): output2  Solution successfully saved as output2.txt in test/output.  Thank you for using IQ Puzzler Pro Solving Program!</pre> <pre>test &gt; output &gt; output2.txt  1 Solution: 2 +---+---+---+---+---+---+ 3   A   B   B   B   B   C   D   4 +---+---+---+---+---+---+ 5   A   E   E   B   B   C   C   6 +---+---+---+---+---+---+ 7   F   G   E   E   H   H   H   H   8 +---+---+---+---+---+---+ 9   F   G   I   I   I   I   I   H   10 +---+---+---+---+---+---+ 11   F   G   I   J   J   J   I   H   12 +---+---+---+---+---+---+ 13   F   G   I   J   J   K   I   H   14 +---+---+---+---+---+---+ 15   L   L   I   J   K   K   I   H   16 +---+---+---+---+---+---+ 17   L   L   I   I   I   I   I   H   18 +---+---+---+---+---+---+ 19 20 Blocks in solution: 21 Block letter: A at position: (0,0) 22 Block letter: B at position: (1,0) 23 Block letter: C at position: (6,0) 24 Block letter: D at position: (7,0) 25 Block letter: E at position: (1,1) 26 Block letter: F at position: (0,2) 27 Block letter: G at position: (1,2) 28 Block letter: H at position: (4,2) 29 Block letter: I at position: (2,3) 30 Block letter: J at position: (3,4) 31 Block letter: K at position: (5,5) 32 Block letter: L at position: (0,6) 33 34 Cases reviewed: 435930 cases. 35 Duration: 229ms.</pre>
<p>3 - DEFAULT</p>	<p>5 4 8          DEFAULT          XXX          X          OO          A          RRR          R          MMM          M          FF          J          J          H</p>	<pre>PS C:\Users\ASUS\Documents\GitHub\Tucill1_13523043\src&gt; .\run.bat  Please make sure the file is located in test/input folder. Enter file name (without .txt): Input4  Succeed.  XXXX AABO MERR MERR MMJJ  Cases reviewed: 43 cases. Duration: 17ms.  Do you want to save the solution? (yes/no): yes Input file name (without .txt): output4  Solution successfully saved as output4.txt in test/output.  Thank you for using IQ Puzzler Pro Solving Program!</pre> <pre>test &gt; output &gt; output4.txt  1 Solution: 2 +---+---+---+---+ 3   X   X   X   O   4 +---+---+---+---+ 5   A   X   R   O   6 +---+---+---+---+ 7   M   F   R   H   8 +---+---+---+---+ 9   M   F   R   R   10 +---+---+---+---+ 11   M   M   J   J   12 +---+---+---+---+ 13 14 Blocks in solution: 15 Block letter: X at position: (0,0) 16 Block letter: O at position: (3,0) 17 Block letter: A at position: (0,1) 18 Block letter: R at position: (2,1) 19 Block letter: M at position: (0,2) 20 Block letter: F at position: (1,2) 21 Block letter: H at position: (3,2) 22 Block letter: J at position: (2,4) 23 24 Cases reviewed: 43 cases. 25 Duration: 17ms.</pre>

4 - DEFAULT	4 4 7 DEFAULT A AA B BB C CC D DD EE EE E FF FF F GGG	<pre>PS C:\Users\ASUS\Documents\GitHub\Tucill1_13523043\src&gt; .\run.bat</pre> <p>Please make sure the file is located in test/input folder. Enter file name (without .txt): input5</p> <p>Solution not found.</p> <p>Thank you for using IQ Puzzler Pro Solving Program!</p> <p>Kasus elemen blok &gt; kotak pada papan.</p>
5 - DEFAULT	3 3 4 DEFAULT AA B C CC D	<pre>PS C:\Users\ASUS\Documents\GitHub\Tucill1_13523043\src&gt; .\run.bat</pre> <p>Please make sure the file is located in test/input folder. Enter file name (without .txt): input6</p> <p>Solution not found.</p> <p>Thank you for using IQ Puzzler Pro Solving Program!</p> <p>Kasus elemen blok &lt; kotak pada papan.</p>
6 - CUSTOM	5 7 5 CUSTOM ...X... .XXXXX. XXXXXXXX .XXXXX. ...X... A AAA BB BBB CCCC C D EEE E	<pre>PS C:\Users\ASUS\Documents\GitHub\Tucill1_13523043\src&gt; .\run.bat</pre> <p>Please make sure the file is located in test/input folder. Enter file name (without .txt): input3</p> <p>Succeed.</p> <pre> A BBAAA BBBCCCC EEEC E </pre> <p>Cases reviewed: 11677 cases. Duration: 30ms.</p> <p>Do you want to save the solution? (yes/no): yes Input file name (without .txt): output3</p> <p>Solution successfully saved as output3.txt in test/output.</p> <p>Thank you for using IQ Puzzler Pro Solving Program!</p> <pre>test &gt; output &gt; output3.txt</pre> <pre> 1  Solution: 2  +---+---+---+---+---+---+ 3          A       4  +---+---+---+---+---+---+ 5      B   B   A   A   A   6  +---+---+---+---+---+---+ 7    B   B   B   C   C   C   8  +---+---+---+---+---+---+ 9      E   E   E   C   D   10 +---+---+---+---+---+---+ 11         E       12 +---+---+---+---+---+---+ 13 14 Blocks in solution: 15 Block letter: A at position: (3,0) 16 Block letter: B at position: (1,1) 17 Block letter: C at position: (3,2) 18 Block letter: E at position: (1,3) 19 Block letter: D at position: (5,3) 20 21 Cases reviewed: 11677 cases. 22 Duration: 45ms. </pre>

7 - CUSTOM	4 4 4 CUSTOM X.X. XXXX X.XX ..XX LLL L L M EE E NN	<pre> PS C:\Users\ASUS\Documents\GitHub\Tuc111_13523043\src&gt; .\run.bat  Please make sure the file is located in test/input folder. Enter file name (without .txt): input7  Succeed.  L L L L E M EE NN  Cases reviewed: 371 cases. Duration: 19ms.  Do you want to save the solution? (yes/no): yes Input file name (without .txt): output5  Solution successfully saved as output5.txt in test/output.  Thank you for using IQ Puzzler Pro Solving Program! </pre> <pre> test &gt; output &gt; output5.txt 1  Solution: 2  +-----+ 3    L     L     4  +-----+ 5    L   L   L   E   6  +-----+ 7    M     E   E   8  +-----+ 9        N   N   10 +-----+ 11 12 Blocks in solution: 13 Block letter: L at position: (0,0) 14 Block letter: E at position: (3,1) 15 Block letter: M at position: (0,2) 16 Block letter: N at position: (2,3) 17 18 Cases reviewed: 371 cases. 19 Duration: 19ms. </pre>
------------	---	---

## LAMPIRAN

Pranala Repository Github: [https://github.com/najwakanifatima/Tucil1\\_13523043](https://github.com/najwakanifatima/Tucil1_13523043)

No	Poin	Ya	Tidak
1	Program berhasil dikompilasi tanpa kesalahan	✓	
2	Program berhasil dijalankan	✓	
3	Solusi yang diberikan program benar dan mematuhi aturan permainan	✓	
4	Program dapat membaca masukan berkas .txt serta menyimpan solusi dalam berkas .txt	✓	
5	Program memiliki <i>Graphical User Interface</i> (GUI)		✓
6	Program dapat menyimpan solusi dalam bentuk file gambar		✓
7	Program dapat menyelesaikan kasus konfigurasi <i>custom</i>	✓	
8	Program dapat menyelesaikan kasus konfigurasi Piramida (3D)		✓
9	Program dibuat oleh saya sendiri	✓	