

IF2211 STRATEGI ALGORITMA
TUGAS KECIL 1
Penyelesaian IQ Puzzler Pro dengan
Algoritma Brute Force



Diusun Oleh:
13523029 - Bryan Ho

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
JL. GANESA 10, BANDUNG 40132
2025

DAFTAR ISI

DAFTAR ISI	2
BAB I	3
PENDAHULUAN	3
I. Algoritma Brute Force	3
II. IQ Puzzler Pro	3
BAB II	
IMPLEMENTASI	4
I. Algoritma Penyelesaian IQ Puzzler Pro dengan Algoritma Brute Force	4
II. Class yang Digunakan	5
1. Class Piece	5
2. Class Board	8
3. Class Solver	12
4. Class IO	16
5. Class GUI	18
6. Class Main	19
II. Tampilan User (User Interface)	20
BAB III	
EKSPERIMEN	21
LAMPIRAN	29
I. Referensi	29
II. Link Repository	29
III. Tabel Evaluasi Program	29

BAB I

PENDAHULUAN

I. Algoritma Brute Force

Algoritma brute force adalah algoritma dengan pendekatan yang lurus atau *straightforward* untuk memecahkan suatu persoalan. Tujuan dari algoritma ini adalah mencoba semua kemungkinan secara berurutan hingga menemukan solusi yang optimal. Terdapat beberapa kelebihan dan kekurangan dari algoritma brute force. Kelebihan dari algoritma ini adalah kepastian solusi dan kesederhanaan dalam mengimplementasi (tidak perlu pengetahuan khusus). Sedangkan kekurangannya adalah kecepatan yang rendah dan tidak efisien untuk masalah yang kompleks.

II. IQ Puzzler Pro



Permainan IQ Puzzler Pro

Sumber: https://m.media-amazon.com/images/I/71+9bXpvamL._AC_SL1280_.jpg

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.

BAB II

IMPLEMENTASI

I. Algoritma Penyelesaian IQ Puzzler Pro dengan Algoritma Brute Force

Permainan IQ Puzzler Pro dapat diselesaikan dengan menggunakan algoritma brute force. Berikut merupakan algoritma yang digunakan.

```
procedure Solve (input index: integer)
{Menyusun puzzle secara rekursif menggunakan brute-force}

Deklarasi:
  piece, variant : Piece
  variants : list of Piece
  i, j : integer

Algoritma:
if isBoardComplete() and index = size of pieces then
  isSolved ← true
  call printBoard()
  return
endif

if index = size of pieces then
  return
endif

piece ← pieces[index]
variants ← getAllVariants(piece)

for each variant in variants do
  for i ← 0 to boardRows - 1 do
    for j ← 0 to boardCols - 1 do
      if canPlacePiece(variant, i, j) then
        caseChecked ← caseChecked + 1
        placePiece(variant, i, j)
        Solve(index + 1)
        if isSolved then
          return
        endif
        removePiece(variant, i, j)
      endif
    endfor
  endfor
endfor
end procedure
```

Langkah-langkah pendekatan yang dilakukan adalah sebagai berikut.

1. Mengambil potongan puzzle dan menyimpan semua variasi bentuknya (rotasi dan *flip/mirror*). Terdapat 8 variasi bentuk potongan puzzle apabila tidak ada bentuk potongan yang sama.
2. Menempatkan potongan pertama puzzle pada seluruh posisi di papan dan mencoba semua variasi bentuk untuk setiap posisi di papan.

3. Jika potongan puzzle bisa dipasang, lanjut ke potongan puzzle berikutnya. Namun jika potongan puzzle tidak bisa dipasang, hapus potongan terakhir yang sudah ditempatkan dan mencoba posisi atau variasi bentuk yang lain.
4. Jika semua potongan puzzle berhasil dipasang di papan tanpa ada celah yang kosong, maka solusi ditemukan. Namun jika semua variasi bentuk potongan puzzle telah dicoba dan papan masih terdapat celah, maka tidak ada solusi.

Program juga akan menghitung jumlah kasus yang telah ditinjau. Program akan memeriksa terlebih dahulu apakah potongan puzzle bisa ditempatkan di suatu posisi di papan. Jika potongan puzzle bisa ditempatkan, maka jumlah kasus akan bertambah.

II. Class yang Digunakan

Detail masing-masing kelas dapat dilihat sebagai berikut.

1. Class Piece

- **Attribute**

Nama	Tipe	Deskripsi
piece	private char[][]	Array Char 2D yang menyimpan bentuk puzzle piece
width	private int	Nilai yang menyimpan lebar maksimum puzzle piece
height	private int	Nilai yang menyimpan tinggi maksimum puzzle piece

```

public class Piece {
    private char[][] piece;
    private int width;
    private int height;

```

- **Method**

Nama	Tipe	Parameter	Deskripsi
Piece	public Constructor	char[][] piece	Membuat puzzle piece sesuai dengan ukuran piece

```

public Piece(char[][] piece) {
    this.width = piece[0].length;
    this.height = piece.length;
    this.piece = new char[height][width];

```

<pre> for (int i = 0; i < height; i++) { for (int j = 0; j < width; j++) { this.piece[i][j] = piece[i][j]; } } } </pre>			
getWidth	public int		Mengambil jumlah kolom (lebar) dari puzzle piece
<pre> public int getWidth() { return width; } </pre>			
getHeight	public int		Mengambil jumlah baris (tinggi) dari puzzle piece
<pre> public int getHeight() { return height; } </pre>			
getPiece	public char[][]		Mengembalikan sebuah <i>copy</i> dari puzzle piece
<pre> public char[][] getPiece() { char [][] copy = new char[height][width]; for (int i = 0; i < height; i++) { for (int j = 0; j < width; j++) { copy[i][j] = piece[i][j]; } } return copy; } </pre>			
rotate	public Piece		Membuat dan mengembalikan puzzle piece yang telah dirotasi

```

public Piece rotate() {
    char[][] rotated = new char[width][height];

    for (int i = 0; i < width; i++) {
        for (int j = 0; j < height; j++) {
            rotated[i][j] = piece[height - j - 1][i];
        }
    }

    return new Piece(rotated);
}

```

flipHorizontal

public Piece

Membuat dan mengembalikan puzzle piece yang telah dicerminkan secara horizontal

```

public Piece flipHorizontal() {
    char[][] flipped = new char[height][width];

    for (int i = 0; i < height; i++) {
        for (int j = 0; j < width; j++) {
            flipped[i][j] = piece[i][width - j - 1];
        }
    }

    return new Piece(flipped);
}

```

getAllVariants

public
List<Piece>

Mengembalikan semua transformasi dari puzzle piece berdasarkan metode rotate dan flip

```

public List<Piece> getAllVariants() {
    List<Piece> variants = new ArrayList<>();
    Piece current = this;

    for (int i = 0; i < 4; i++) {
        variants.add(current);
        variants.add(current.flipHorizontal());
        current = current.rotate();
    }
}

```

<pre> } return variants; } </pre>			
printPiece	public void		Mencetak puzzle piece
<pre> void printPiece() { for (int i = 0; i < height; i++) { for (int j = 0; j < width; j++) { System.out.print(piece[i][j]); } System.out.println(); } } </pre>			

2. Class Board

- **Attribute**

Nama	Tipe	Deskripsi
board	private char[][]	Array 2D yang menyimpan bentuk papan puzzle
rows	private int	Nilai yang menyimpan jumlah baris pada papan puzzle
cols	private int	Nilai yang menyimpan jumlah kolom pada papan puzzle
<pre> public class Board { private char[][] board; private int rows; private int cols; } </pre>		

- **Method**

Nama	Tipe	Parameter	Deskripsi
Board	public Constructor	int rows, int cols	Membentuk papan puzzle sesuai ukuran

			baris dan kolom, dan menginisialisasi isi papan dengan '#'
<pre> public Board(int rows, int cols) { this.rows = rows; this.cols = cols; board = new char[rows][cols]; for (int i = 0; i < rows; i++) { for (int j = 0; j < cols; j++) { board[i][j] = '#'; } } } </pre>			
getRows	public int		Mengambil jumlah baris pada papan puzzle
<pre> public int getRows() { return rows; } </pre>			
getCols	public int		Mengambil jumlah kolom pada papan puzzle
<pre> public int getCols() { return cols; } </pre>			
getBoard	public char[][]		Mengembalikan papan puzzle
<pre> public char[][] getBoard() { return board; } </pre>			
canPlacePiece	public boolean	Piece piece, int x, int y	Mengecek apakah puzzle piece bisa diletakkan pada

			papan puzzle
<pre> public boolean canPlacePiece(Piece piece, int x, int y){ char[][] shape = piece.getPiece(); int pieceWidth = piece.getWidth(); int pieceHeight = piece.getHeight(); for (int i = 0; i < pieceHeight; i++) { for (int j = 0; j < pieceWidth; j++) { if (shape[i][j] != '#' && (x + i >= rows y + j >= cols board[x + i][y + j] != '#')) { return false; } } } return true; } </pre>			
placePiece	public void	Piece piece, int x, int y	Meletakkan puzzle piece pada papan puzzle
<pre> public void placePiece(Piece piece, int x, int y) { if (canPlacePiece(piece, x, y)) { char[][] shape = piece.getPiece(); int pieceWidth = piece.getWidth(); int pieceHeight = piece.getHeight(); for (int i = 0; i < pieceHeight; i++) { for (int j = 0; j < pieceWidth; j++) { if (shape[i][j] != '#') { board[x + i][y + j] = shape[i][j]; } } } } } </pre>			
removePiece	public void	Piece piece, int x, int y	Menghapus puzzle piece pada papan puzzle

```

public void removePiece(Piece piece, int x, int y) {
    char[][] shape = piece.getPiece();
    int pieceHeight = piece.getHeight();
    int pieceWidth = piece.getWidth();

    for (int i = 0; i < pieceHeight; i++) {
        for (int j = 0; j < pieceWidth; j++) {
            if (shape[i][j] != '#') {
                board[x + i][y + j] = '#';
            }
        }
    }
}

```

isComplete

public boolean

Mengecek
apakah papan
puzzle sudah
terisi penuh

```

public boolean isComplete() {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            if (board[i][j] == '#') {
                return false;
            }
        }
    }

    return true;
}

```

printBoard

public void

Mencetak papan
puzzle beserta
isinya

```

public void printBoard() {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            System.out.print(getColor(board[i][j]) +
board[i][j] + "\u001B[0m");
        }
        System.out.println();
    }
}

```

getColor	public String	char c	Mengembalikan warna yang unik untuk huruf 'A-Z'
----------	---------------	--------	---

```

public String getColor(char c) {
    if (c == '#') return "\u001B[37m";
    if (c < 'A' || c > 'Z') return "\u001B[0m";

    switch(c) {
        case 'A': return "\u001B[31m";
        case 'B': return "\u001B[32m";
        case 'C': return "\u001B[33m";
        case 'D': return "\u001B[34m";
        case 'E': return "\u001B[35m";
        case 'F': return "\u001B[36m";
        case 'G': return "\u001B[91m";
        case 'H': return "\u001B[92m";
        case 'I': return "\u001B[93m";
        case 'J': return "\u001B[94m";
        case 'K': return "\u001B[95m";
        case 'L': return "\u001B[96m";
        case 'M': return "\u001B[35m";
        case 'N': return "\u001B[90m";
        case 'O': return "\u001B[32m";
        case 'P': return "\u001B[41m\u001B[30m";
        case 'Q': return "\u001B[42m\u001B[30m";
        case 'R': return "\u001B[43m\u001B[30m";
        case 'S': return "\u001B[44m\u001B[30m";
        case 'T': return "\u001B[45m\u001B[30m";
        case 'U': return "\u001B[46m\u001B[30m";
        case 'V': return "\u001B[101m\u001B[30m";
        case 'W': return "\u001B[102m\u001B[30m";
        case 'X': return "\u001B[103m\u001B[30m";
        case 'Y': return "\u001B[104m\u001B[30m";
        case 'Z': return "\u001B[105m\u001B[30m";
        default: return "\u001B[0m";
    }
}

```

3. Class Solver

- Attribute

Nama	Tipe	Deskripsi
board	private Board	Objek Board yang menyimpan papan puzzle
pieces	private List<Piece>	List yang menyimpan piece puzzle
isSolved	private boolean	Nilai yang menyimpan apakah papan telah berhasil diselesaikan
caseChecked	private long	Nilai yang menyimpan jumlah kasus yang telah ditinjau
duration	private long	Nilai yang menyimpan lamanya waktu pencarian solusi

```

public class Solver {
    private Board board;
    private List<Piece> pieces;
    private boolean isSolved;
    private long caseChecked;
    private long duration;

```

- **Method**

Nama	Tipe	Parameter	Deskripsi
Solver	public Constructor	Board board, List<Piece> pieces	Menginisialisasi objek Solver dengan papan puzzle dan potongan puzzle piece. isSolved bernilai false, caseChecked dan duration bernilai 0

```

    public Solver(Board board, List<Piece> pieces) {
        this.board = board;
        this.pieces = pieces;
        this.isSolved = false;
        this.caseChecked = 0;

```

<pre>this.duration = 0; }</pre>			
isSolved	public boolean		Mengecek apakah papan telah berhasil diselesaikan
<pre>public boolean isSolved() { return isSolved; }</pre>			
getCaseChecked	public long		Mengembalikan jumlah kasus yang telah ditinjau
<pre>public long getCaseChecked() { return caseChecked; }</pre>			
getDuration	public long		Mengembalikan lamanya waktu pencarian kasus
<pre>public long getDuration() { return duration; }</pre>			
solve	public void	int idx	Menyelesaikan permainan IQ Puzzler Pro dengan algoritma brute force
<pre>public void solve(int idx) { if (board.isComplete() && idx == pieces.size()) { isSolved = true; board.printBoard(); return; } if (idx == pieces.size()) { return; } }</pre>			

```

        Piece piece = pieces.get(idx);
        List<Piece> variants = piece.getAllVariants();

        for (Piece variant : variants) {
            for (int i = 0; i < board.getRows(); i++) {
                for (int j = 0; j < board.getCols(); j++){
                    if (board.canPlacePiece(variant, i, j)){
                        caseChecked++;
                        board.placePiece(variant, i, j);

                        solve(idx + 1);
                        if (isSolved) {
                            return;
                        }

                        board.removePiece(variant, i, j);
                    }
                }
            }
        }
    }
}

```

solvePuzzle

public void

Menggunakan metode solve dan mencetak beberapa detail seperti apakah permainan dapat diselesaikan, waktu pencarian, dan banyak kasus yang ditinjau

```

public void solvePuzzle() {
    long startTime = System.currentTimeMillis();
    solve(0);
    long endTime = System.currentTimeMillis();
    duration = endTime - startTime;

    if (isSolved) {
        System.out.println("Solution found");
    } else {
        System.out.println("No solution found.");
    }
}

```

```

    }

    System.out.println("Waktu pencarian: " + duration
+ " ms");
    System.out.println("Banyak kasus yang ditinjau: "
+ caseChecked);
}

```

4. Class IO

- **Attribute**

Nama	Tipe	Deskripsi
N	private int	Nilai yang menyimpan jumlah baris pada papan puzzle
M	private int	Nilai yang menyimpan jumlah kolom pada papan puzzle
P	private int	Nilai yang menyimpan jumlah puzzle piece
board	private Board	Objek Board yang menyimpan papan puzzle
pieces	private List<Piece>	List yang menyimpan puzzle piece
solver	private Solver	Objek Solver yang menyimpan rincian solusi dari puzzle
shape	private char[][]	Array Char 2D yang menyimpan bentuk puzzle piece
caseType	private String	String yang menyimpan jenis kasus yang ditinjau

- **Method**

Nama	Tipe	Parameter	Deskripsi
IO	public Constructor		Menginisialisasi objek IO dengan nilai default. N, M, dan P bernilai 0, sementara board, pieces, solver, shape, dan caseType bernilai null.
getBoard	public Board		Mengembalikan papan puzzle
getPieces	public List<Piece>		Mengembalikan list yang berisi piece puzzle
setSolver	public void	Solver solver	Menyimpan referensi solver
readInputFile	public void	String filename	Membaca input berupa file .txt
saveBoardToFile	public void	String filename	Menyimpan solusi dari papan puzzle ke dalam bentuk file .txt
saveBoardAsImage	public void	String filename	Menyimpan solusi dari papan puzzle ke dalam bentuk gambar
readCustomMatrix	public char[][]	List<String> customLine	Membaca dan mengembalikan papan puzzle bertipe "custom"
firstCharPiece	private char	String line	Mengembalikan char piece puzzle pertama pada file .txt
stringListToCharArray	private char[][]	List<String> shapeLine	Mengubah baris puzzle piece pada file .txt

			menjadi array char 2D
getColorFromChar	public Color	char piece	Mengembalikan warna yang unik untuk setiap puzzle piece

5. Class GUI

- **Attribute**

Nama	Tipe	Deskripsi
openButton	private JButton	Tombol untuk membuka file input puzzle
solveButton	private JButton	Tombol untuk memulai proses penyelesaian puzzle
saveTxtButton	private JButton	Tombol untuk menyimpan hasil solusi dalam file .txt
saveImgButton	private JButton	Tombol untuk menyimpan solusi dalam bentuk gambar
boardPanel	private JPanel	Tempat menampilkan representasi visual dari papan puzzle
statusLabel	private JLabel	Tempat menampilkan status atau pesan kepada user
IO	private IO	Objek kelas IO yang menangani pembacaan dan penyimpanan file puzzle
solver	private Solver	Objek kelas Solver yang digunakan untuk menyelesaikan puzzle
currentFilePath	private String	Mengambil nama path file input puzzle yang

		sedang digunakan
--	--	------------------

- **Method**

Nama	Tipe	Parameter	Deskripsi
GUI	public		Membentuk tampilan GUI
openFile	private void		Membuka dan membaca file .txt
solvePuzzle	private void		Menyelesaikan puzzle dan menampilkan rincian puzzle
saveTxtFile	private void		Menyimpan solusi puzzle dalam bentuk file .txt
saveImgFile	private void		Menyimpan solusi puzzle dalam bentuk gambar
displayBoard	private void	Board board	Menampilkan papan puzzle
main	public static void	String[] args	Menjalankan program GUI

6. Class Main

- **Attribute**

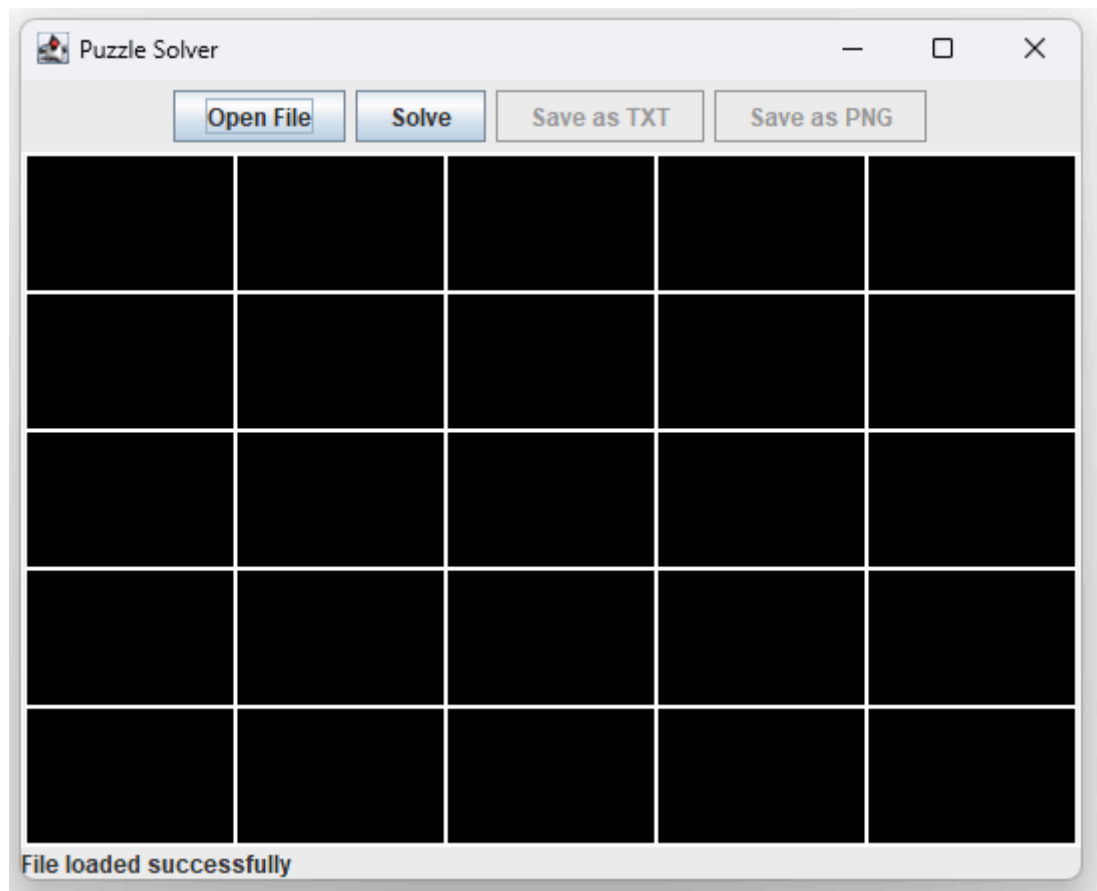
Kelas ini tidak memiliki atribut.

- **Method**

Nama	Tipe	Parameter	Deskripsi
main	public static void	String[] args	Menjalankan program CLI

II. Tampilan User (User Interface)

User Interface dari program ini menggunakan CLI (Command Line Interface), dengan tampilan sebagai berikut.



Tampilan GUI

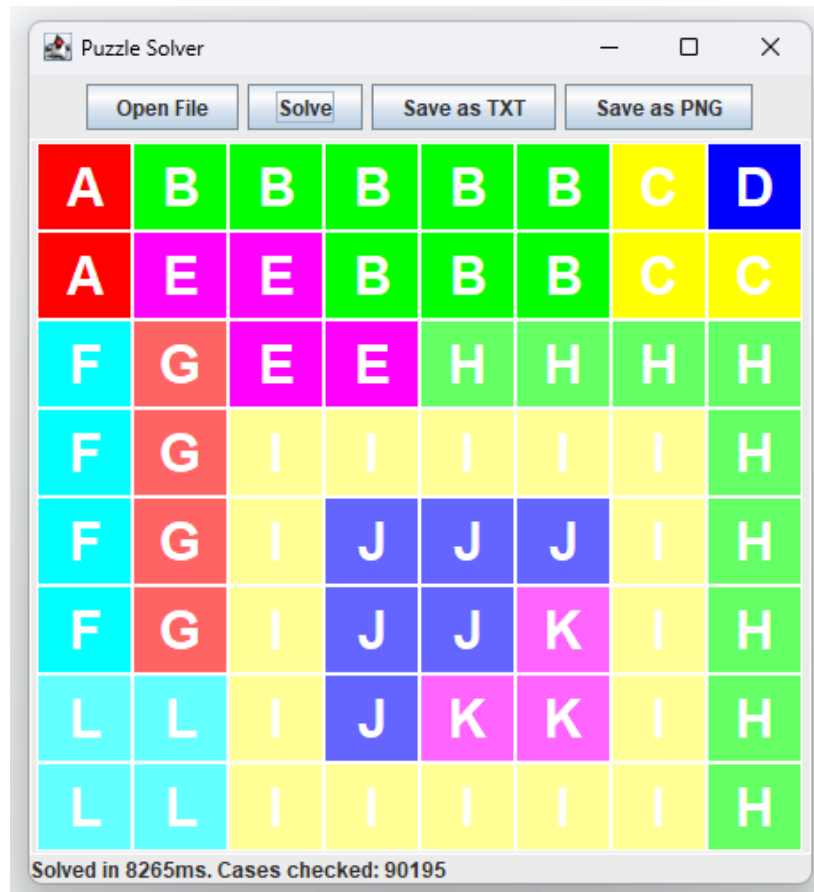
BAB III

EKSPERIMEN

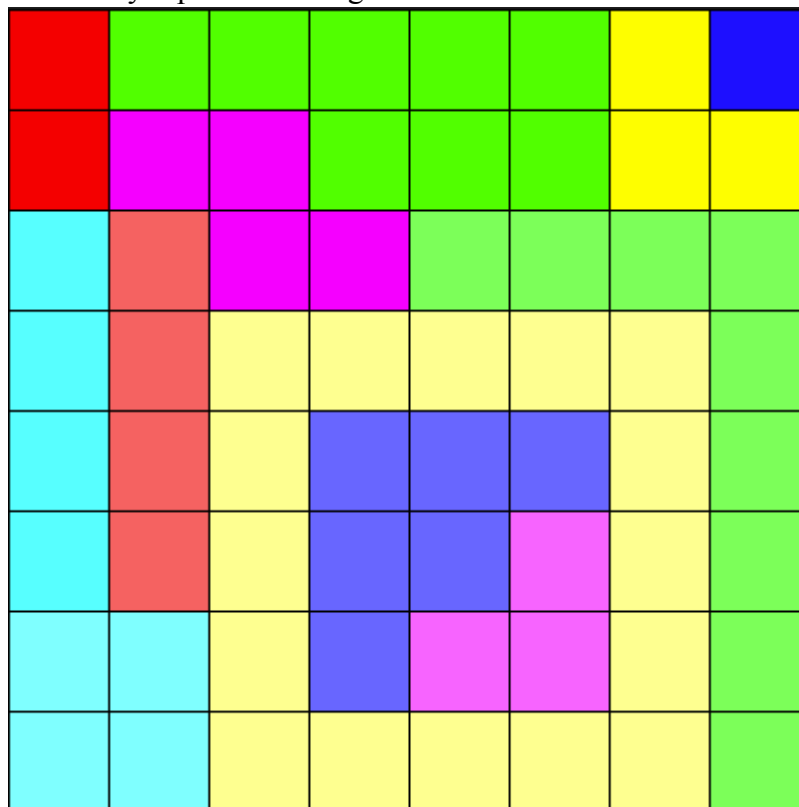
Bab ini berisi hasil eksekusi program terhadap contoh-contoh kasus yang diberikan. Tabel berikut berisikan analisis dan hasil eksekusi dari kasus-kasus tersebut.

1	<p>Input file 1.txt</p> <pre>5 5 7 DEFAULT A AA B BB C CC D DD EE EE E FF FF F GGG</pre>
	<p>Output file 1.txt</p> 

	<p>Hasil Penyimpanan file .txt</p> <pre> AGGGC AABCC EEBBF EEDFF EDDFF Waktu pencarian: 773 ms Banyak kasus yang ditinjau: 27023 </pre>
2	<p>Input file 2.txt</p> <pre> 8 8 12 DEFAULT A A BBBBB BBB C CC D EE EE FFFF GGGG HHHH H H H H H IIIII I I I I I I IIIII JJJ JJ J KK K LL LL </pre>
	<p>Output file 2.txt</p>



Hasil Penyimpanan dalam gambar



3

Input file 3.txt

```

7 7 9
DEFAULT
AAA
AAAAA
BBBBB
CCC
C
C
CCC
EEE
FFF
F
F
F
F
GGGG
GGGG
HHH
I
II
DD
DD

```

Output file 3.txt



4

Input file 4.txt

```
13 2 26
DEFAULT
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
```

Output file 4.txt



5

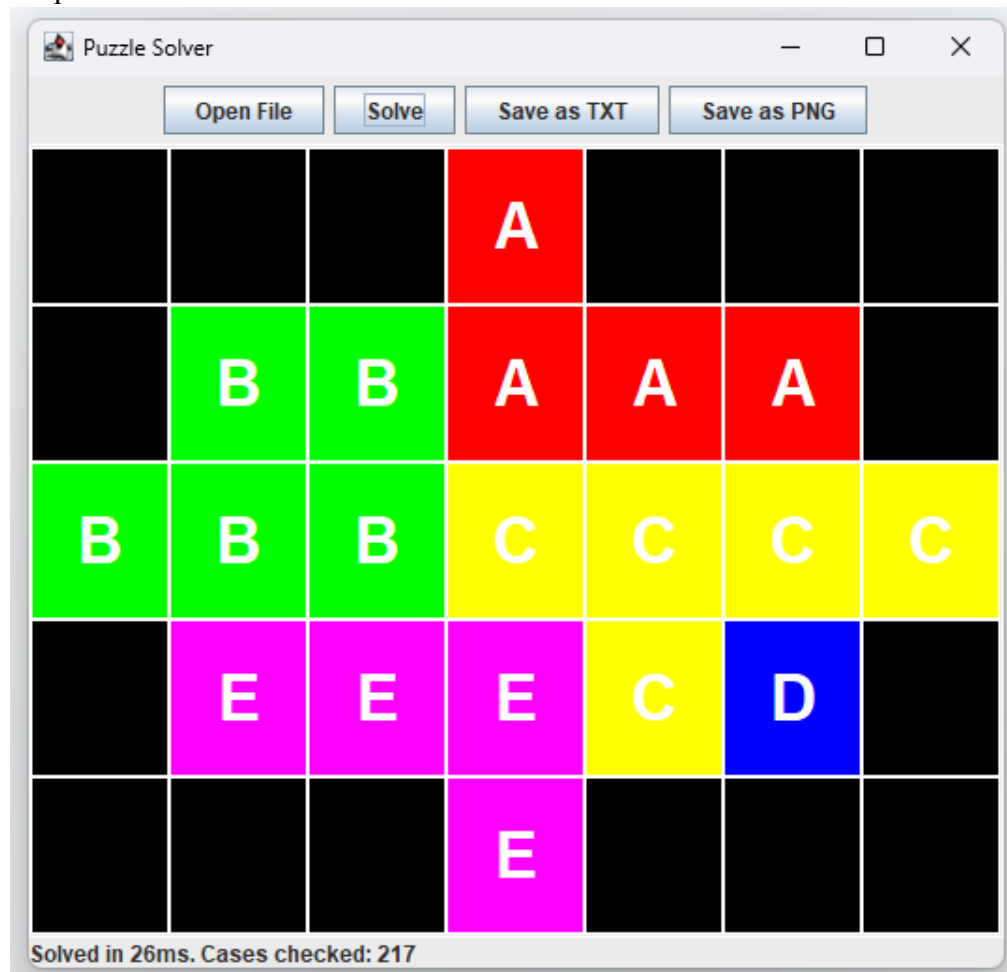
Input file 5.txt

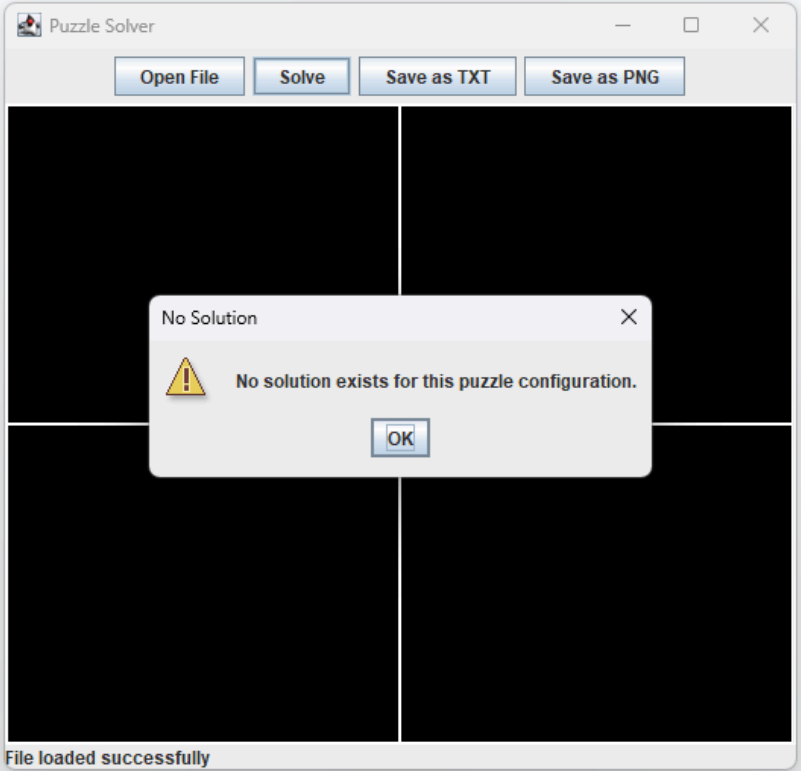
```

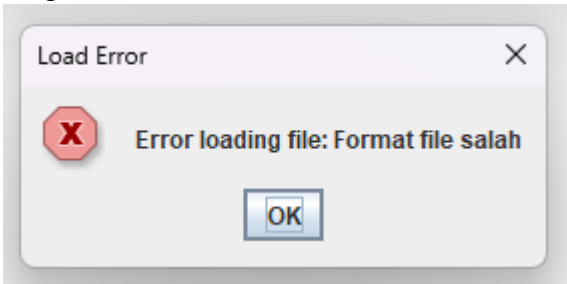
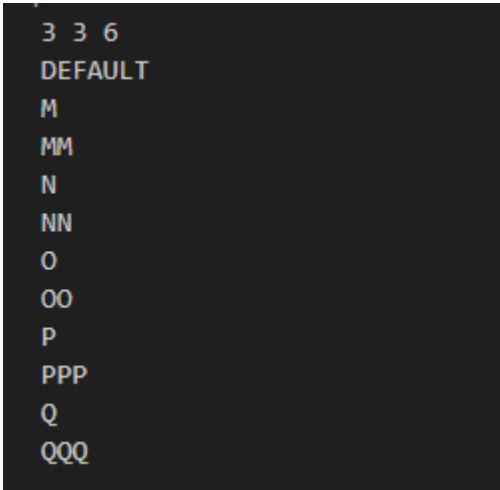
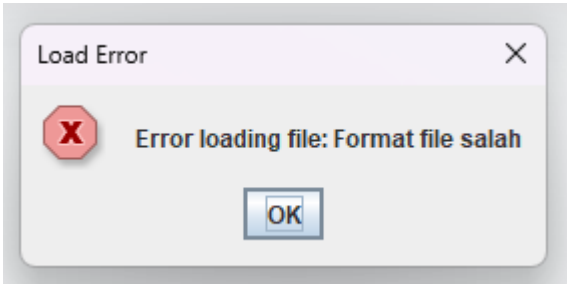
5 7 5
CUSTOM
...X...
.XXXXX.
XXXXXXX
.XXXXX.
...X...
A
AAA
BB
BBB
CCCC
| C
D
EEE
E

```

Output file 5.txt



6	<p>Input file 6.txt</p> <pre> 2 2 3 DEFAULT AA BB CC </pre>
	<p>Output file 6.txt</p> 
7	<p>Input file 7.txt</p> <pre> 5 7 5 CUSTOM XXX.XXX X.....X X.....X XXX.XXX A AAA BB BBB CCCC C D EEE E </pre>

	<p>Output file 7.txt</p> 
8	<p>Input file 8.txt</p> 
	<p>Output file 8.txt</p> 

LAMPIRAN

I. Referensi

- [https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/02-Algoritma-Brute-Force-\(2025\)-Bag1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/02-Algoritma-Brute-Force-(2025)-Bag1.pdf) (Slide Kuliah Pak Rinaldi Munir)
- <https://www.cloudeka.id/id/berita/web-sec/cara-kerja-algoritma-brute-force/>

II. Link Repository

https://github.com/bry-ho/Tucil1_13523029

III. Tabel Evaluasi Program

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	✓	