LAPORAN TUGAS KECIL 1

IF2211 Strategi Algoritma

Penyelesaian IQ Puzzler Pro dengan Algoritma Brute Force

Disusun oleh:

Muh. Rusmin Nurwadin

NIM : 13523068

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2025

A. Pendahuluan

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. Setiap blok puzzle dapat dirotasikan maupun dicerminkan. Puzzle dinyatakan selesai jika dan hanya jika papan terisi penuh dan seluruh blok puzzle berhasil diletakkan.

B. Algoritma Brute Force

Algoritma Brute Force adalah suatu penyelesaian maslaah dengan mencoba semua kemungkinan solusi yang ada secara sistematis, sampai menemukan suatu Solusi yang benar. Algoritma ini menjamin terdapat solusi pada permasalahan, karena meninjau semua kasus yang ada. Berdasarkan karakteristiknya yang meninjau semua kasus, tentunya hal tersebut berpengaruh pada komplesitas algoritma ini, walaupun memberi solusi pasti, tidak jarang membutuhkan waktu yang sangat lama, sehingga diperlukan cara lain yang lebih efisien lagi, dalam hal kompleksitas waktu.

Secara umum algoritma bruteforce dalam menyelesaikan persoalan IQ Puzzle adalah sebagai berikut:

1. Ambil piece/block pertama yang akan ditempatkan. Tempatkan pada posisi paling kiri atas papan yang kosong.
2. Jika penempatan valid, lanjut pada piece berikutnya. Tempatkan pada posisi paling kiri atas yang tersedia.
3. Jika penempatan tidak valid (keluar papan atau bertumpuk), maka coba posisi berikutnya pada papan. Jika semua posisi sudah dicoba, maka:
    - Coba rotasi (90 derajat).
    - Jika sudah 4 rotasi, coba flip/mirror.
    - Setelah flip, coba 4 rotasi lagi.
    - Ulangi proses dari awal untu setiap bentuk yang mungkin.
4. Jika semua posisi dan transformasi untuk suatu piece sudah dicoba dan tidak ada yang valid, maka:
    - Mundur ke piece berikutnya
    - Hapus piece sebelumnya dari board
    - Coba transformasi atau posisi berikutnya untuk piece sebelumnya
    - Jika piece sebelumnya belum bisa juga, maka mundur pada piece sebelumnya lagi
5. Ulangi langkah 1-4 sampai ditemukan suatu solusi atau tidak menemukan solusi sama sekali.

Berdasarkan algoritma yang digunakan, maka kompleksitas maksimum adalah $O(8^N N!)$. Hal ini dikarenakan untuk tiap piece memiliki 8 kemungkinan maksimum dari rotasi dan pencerminan. Sedangkan N! adalah banyaknya kemungkinan pemasangan N piece berbeda pada papan.

C. Souce Code

```java
import java.util.*;

public class MainCLI {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.println("==================================================");
        System.out.println("                SELAMAT DATANG :))");
        System.out.println("==================================================\n");

        try {
            String file;
            do {
                System.out.print("Masukkan nama file input (.txt): ");
                file = scan.nextLine().trim();

                if (!Utils.cekFile("test/" + file)) {
                    System.out.println("File tidak ditemukan atau bukan file .txt");
                }
            } while (!Utils.cekFile("test/" + file));

            System.out.println("Memulai pencarian solusi...\n");

            Object[] dataPuzzle = Init.initPuzzle("test/" + file);
            if (dataPuzzle == null) return;

            char[][] board = (char[][]) dataPuzzle[0];
            char[][][] pieces = (char[][][]) dataPuzzle[1];

            Solver solver = new Solver();
            char[][] solution = solver.solve(board, pieces);

            if (solution.length > 0) {
                System.out.println("Solusi ditemukan pada kombinasi ke-" + (solver.getSolveAt()+1) + ":\n");
                IOHandler.showSolution(solution);

                System.out.print("\nApakah anda ingin menyimpan solusi? (Y/N): ");
                String input = scan.nextLine().trim().toUpperCase();
                if (input.equals("Y")) {
                    IOHandler.saveSolution(solution, "test/output_" + file);
                }
            } else {
                System.out.println("Total waktu : " + (solver.getTime()) + "ms\n");
                System.out.println("Jumlah pengujian : " + (solver.getTotalCheck()) + "\n");
                System.out.println("Tidak ditemukan solusi untuk puzzle ini.\n");
            }

            System.out.println("==================================================\n");

        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        } finally {
            scan.close();
        }
    }
}
```

Gambar 1. MainCLI,java

```java
import java.io.*;
import java.util.*;

public class IOHandler {
    private static final Map<Character, String> colors = new HashMap<>();
    private static final String RESET = "\u001B[0m";

    static {
        String[] colorCode = {
            "\u001B[38;5;196m", "\u001B[38;5;46m", "\u001B[38;5;226m",
            "\u001B[38;5;21m", "\u001B[38;5;201m", "\u001B[38;5;51m",
            "\u001B[38;5;208m", "\u001B[38;5;165m", "\u001B[38;5;118m",
            "\u001B[38;5;199m", "\u001B[38;5;111m", "\u001B[38;5;202m",
            "\u001B[38;5;92m", "\u001B[38;5;76m", "\u001B[38;5;209m",
            "\u001B[38;5;135m", "\u001B[38;5;198m", "\u001B[38;5;160m",
            "\u001B[38;5;33m", "\u001B[38;5;154m", "\u001B[38;5;91m",
            "\u001B[38;5;124m", "\u001B[38;5;23m", "\u001B[38;5;166m",
            "\u001B[38;5;57m", "\u001B[38;5;106m"
        };

        for (int i = 0; i < 26; i++) {
            colors.put((char)('A' + i), colorCode[i]);
        }
    }

    public static String[] readFile(String file) {
        ArrayList<String> lines = new ArrayList<>();
        try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
            String line;
            while ((line = reader.readLine()) != null) {
                lines.add(line);
            }
        } catch (IOException e) {
            System.out.println("Error membaca file: " + e.getMessage());
            return null;
        }

        if (lines.isEmpty()) {
            System.out.println("File kosong");
            return null;
        }

        return lines.toArray(new String[0]);
    }

    public static void showSolution(char[][] board) {
        for (int i = 0; i < board.length; i++) {
            for (int j = 0; j < board[i].length; j++) {
                if (board[i][j] != '.') {
                    String color = colors.getOrDefault(board[i][j], RESET);
                    System.out.print(color + board[i][j] + " " + RESET);
                } else {
                    System.out.print(board[i][j] + " ");
                }
            }
            System.out.println();
        }
    }

    public static void saveSolution(char[][] board, String file) {
        try (PrintWriter writer = new PrintWriter(new FileWriter(file))) {
            for (int i = 0; i < board.length; i++) {
                for (int j = 0; j < board[i].length; j++) {
                    writer.print(board[i][j] + " ");
                }
                writer.println();
            }
            System.out.println("Solusi berhasil disimpan di " + file);
        } catch (IOException e) {
            System.out.println("Error menyimpan file: " + e.getMessage());
        }
    }
}
```

Gambar 2. IOHandler.java

```java
import java.util.*;

public class Init {
    public static Object[] initPuzzle(String file) {
        String[] input = IOHandler.readFile(file);

        if (input == null || input.length < 2) {
            System.out.println("File input tidak valid. Minimal harus memiliki 2 baris");
            return null;
        }

        String[] size = input[0].trim().split("\\s+");
        if (size.length != 3) {
            System.out.println("Format dimensi tidak valid. Harus berisi 3 angka");
            return null;
        }

        try {
            int rows = Integer.parseInt(size[0]);
            int cols = Integer.parseInt(size[1]);
            int totalPiece = Integer.parseInt(size[2]);

            if (rows <= 0 || cols <= 0 || totalPiece <= 0) {
                System.out.println("Dimensi dan jumlah piece harus positif");
                return null;
            }

            char[][] board = new char[rows][cols];
            for (int i = 0; i < rows; i++) {
                Arrays.fill(board[i], '.');
            }

            if (!input[1].trim().equals("DEFAULT")) {
                System.out.println("Mode tidak valid. Saat ini hanya mendukung mode DEFAULT");
                return null;
            }

            char[][][] pieces = new char[totalPiece][][];
            ArrayList<String> tempLine = new ArrayList<>();
            int idxPiece = 0;
            char pieceChar = ' ';

            for (int i = 2; i < input.length; i++) {
                String line = input[i];

                if (line.isEmpty()) continue;

                char first = ' ';
                for (int j = 0; j < line.length(); j++) {
                    char c = line.charAt(j);
                    if (!Character.isWhitespace(c)) {
                        first = c;
                        break;
                    }
                }

                if (first != pieceChar && first != ' ') {
                    if (!tempLine.isEmpty()) {
                        pieces[idxPiece] = buatPiece(tempLine);
                        idxPiece++;
                        tempLine.clear();
                    }
                    pieceChar = first;
                }

                tempLine.add(line);
            }

            if (!tempLine.isEmpty()) {
                pieces[idxPiece] = buatPiece(tempLine);
                idxPiece++;
            }

            if (idxPiece != totalPiece) {
                System.out.println("Jumlah piece tidak sesuai dengan input");
                return null;
            }

            return new Object[]{board, pieces};

        } catch (NumberFormatException e) {
            System.out.println("Dimensi harus berupa angka");
            return null;
        }
    }
```

Gambar 3. Init.java (1)

```java
    private static char[][] buatPiece(ArrayList<String> lines) {
        if (lines.isEmpty()) {
            System.out.println("Piece kosong");
            return null;
        }

        int maxLen = 0;
        for (int i = 0; i < lines.size(); i++) {
            maxLen = Math.max(maxLen, lines.get(i).length());
        }

        char[][] piece = new char[lines.size()][maxLen];

        for (int i = 0; i < piece.length; i++) {
            Arrays.fill(piece[i], '.');
        }

        for (int i = 0; i < lines.size(); i++) {
            String line = lines.get(i);
            for (int j = 0; j < line.length(); j++) {
                char c = line.charAt(j);
                piece[i][j] = Character.isWhitespace(c) ? '.' : c;
            }
        }

        // System.out.println("Piece dibaca sebagai:\n");
        // for (int i = 0; i < piece.length; i++) {
        //     for (int j = 0; j < piece[i].length; j++) {
        //         System.out.print(piece[i][j]);
        //     }
        //     System.out.println();
        // }

        return piece;
    }
}
```

Gambar 4. Init.java (2)

```java
import java.util.*;

public class Solver {
    private long count = 0;
    private long solveAt;
    private long startTime;

    public char[][] solve(char[][] board, char[][][] pieces) {
        count = 0;
        solveAt = 0;

        if (bruteSolve(board, pieces, 0)) {
            return board;
        }
        return new char[0][0];
    }

    private boolean bruteSolve(char[][] board, char[][][] pieces, int index) {
        if (index == 0) startTime = System.nanoTime();

        if (index == pieces.length) {
            if (checkBoard(board)) {
                solveAt = count;
                long elapsed = System.nanoTime() - startTime;
                System.out.println("Solusi ditemukan setelah " + (elapsed / 1_000_000.0) + " ms\n");
                return true;
            }
            return false;
        }

        List<char[][]> transforms = getTransforms(pieces[index]);
        List<int[]> positions = getPositions(board);
        char pieceId = getId(pieces[index]);

        for (int i = 0; i < transforms.size(); i++) {
            char[][] trans = transforms.get(i);
            for (int j = 0; j < positions.size(); j++) {
                int[] pos = positions.get(j);
                if (tryPiece(board, pieces, trans, pos[0], pos[1], pieceId, index)) {
                    return true;
                }
            }
        }

        count++;
        // System.out.println("Mencoba kombinasi ke-" + count + "\n");
        // printBoard(board);
        return false;
    }
```

Gambar 5. Solver.java (1)

```java
private boolean tryPiece(char[][] board, char[][][] pieces, char[][] piece, int row, int col, char id, int idx) {
    if (canPlace(board, piece, row, col)) {
        putPiece(board, piece, row, col, id);
        if (bruteSolve(board, pieces, idx + 1)) {
            return true;
        }
        removePiece(board, piece, row, col);
    }
    return false;
}

private boolean canPlace(char[][] board, char[][] piece, int row, int col) {
    if (row + piece.length > board.length || col + piece[0].length > board[0].length) {
        return false;
    }

    for (int i = 0; i < piece.length; i++) {
        for (int j = 0; j < piece[0].length; j++) {
            if (piece[i][j] != '.' && board[row + i][col + j] != '.') {
                return false;
            }
        }
    }
    return true;
}

private void putPiece(char[][] board, char[][] piece, int row, int col, char id) {
    for (int i = 0; i < piece.length; i++) {
        for (int j = 0; j < piece[0].length; j++) {
            if (piece[i][j] != '.') {
                board[row + i][col + j] = id;
            }
        }
    }
}

private void removePiece(char[][] board, char[][] piece, int row, int col) {
    for (int i = 0; i < piece.length; i++) {
        for (int j = 0; j < piece[0].length; j++) {
            if (piece[i][j] != '.') {
                board[row + i][col + j] = '.';
            }
        }
    }
}
```

Gambar 6. Solver.java (2)

```java
    private boolean checkBoard(char[][] board) {
        for (int i = 0; i < board.length; i++) {
            for (int j = 0; j < board[i].length; j++) {
                if (board[i][j] == '.') return false;
            }
        }
        return true;
    }

    private char getId(char[][] piece) {
        for (int i = 0; i < piece.length; i++) {
            for (int j = 0; j < piece[i].length; j++) {
                if (piece[i][j] != '.') return piece[i][j];
            }
        }
        return '.';
    }

    private char[][] rotate(char[][] matrix) {
        int rows = matrix.length;
        int cols = matrix[0].length;
        char[][] rotated = new char[cols][rows];

        for (int i = 0; i < cols; i++) {
            Arrays.fill(rotated[i], '.');
        }

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                rotated[j][rows - 1 - i] = matrix[i][j];
            }
        }
        return rotated;
    }

    private char[][] flip(char[][] matrix) {
        int rows = matrix.length;
        int cols = matrix[0].length;
        char[][] flipped = new char[rows][cols];

        for (int i = 0; i < rows; i++) {
            Arrays.fill(flipped[i], '.');
            for (int j = 0; j < cols; j++) {
                flipped[i][cols - 1 - j] = matrix[i][j];
            }
        }
        return flipped;
    }

    private List<char[][]> getTransforms(char[][] piece) {
        List<char[][]> transforms = new ArrayList<>();
        char[][] current = piece;

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

        current = flip(piece);
        for (int i = 0; i < 4; i++) {
            transforms.add(current);
            current = rotate(current);
        }

        return transforms;
    }

    private List<int[]> getPositions(char[][] board) {
        List<int[]> positions = new ArrayList<>();
        for (int i = 0; i < board.length; i++) {
            for (int j = 0; j < board[0].length; j++) {
                positions.add(new int[]{i, j});
            }
        }
        return positions;
    }

    public long getSolveAt() {
        return solveAt;
    }

   public long getTotalCheck() {
        return count;
    }

    public double getTime() {
        long elapsed = System.nanoTime() - startTime;
        return elapsed / 1_000_000.0;
    }
    // private void printBoard(char[][] board) {
    //     for (int i = 0; i < board.length; i++) {
    //         for (int j = 0; j < board[i].length; j++) {
    //             System.out.print(board[i][j] + " ");
    //         }
    //         System.out.println();
    //     }
    //     System.out.println();
    // }
}
```

Gambar 7. Solver.java (3)

```java
import java.io.*;

public class Utils {
    public static boolean cekFile(String path) {
        File file = new File(path);
        return file.exists() && file.isFile() && path.toLowerCase().endsWith(".txt");
    }
}
```

Gambar 8. Utils.java

D. Hasil Test

1. Test 1 : Kasus normal, penyimpanan berhasil

   Input :

test > test1.txt

```
5 5 7
DEFAULT
A
AA
B
BB
C
CC
D
DD
EE
EE
E
FF
FF
F
GGG
```

   Output:

```
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> java -cp bin MainCLI
               SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test1.txt
Memulai pencarian solusi...

Solusi ditemukan setelah 129.6585 ms

Solusi ditemukan pada kombinasi ke-27401:

A G G G D
A A B D D
C C B B E
C F F E E
F F F E E

Apakah anda ingin menyimpan solusi? (Y/N): Y
Solusi berhasil disimpan di test/output_test1.txt
================================================
```

Output file:

test > output_test1.txt

```
1  A G G G D
2  A A B D D
3  C C B B E
4  C F F E E
5  F F F E E
6
```

2. Test 2 : Kasus normal.
   Input :

test > test2.txt

```
1  2 2 4
2  DEFAULT
3  X
4  M
5  O
6  P
```

Output :

```
                SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test2.txt
Memulai pencarian solusi...

Solusi ditemukan setelah 0.0372 ms

Solusi ditemukan pada kombinasi ke-1:

X M
O P

Apakah anda ingin menyimpan solusi? (Y/N): n
================================================
```

3. Test 3 : Kasus normal.
   Input :

test > test5.txt

```
4 6 8
DEFAULT
AAA
BB
B
B
CC
C
CC
D
DD
D
E
EE
 E
F
G
HH
```

Output :

```
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> javac -d bin src/*.java
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> java -cp bin MainCLI
               SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test5.txt
Memulai pencarian solusi...

Solusi ditemukan setelah 1.2742 ms

Solusi ditemukan pada kombinasi ke-28:

A A A B B H
C C F B D H
C E E B D D
C C E E D G

Apakah anda ingin menyimpan solusi? (Y/N): n
================================================
```

4. Test 4 : Kasus normal.
   Input :

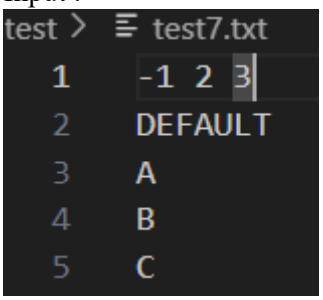
test > test6.txt

```
4 5 7
DEFAULT
AA
B
BBB
CC
 C
 C
DD
D
EE
F
FFF
G
```

   Output :

```
                SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test6.txt
Memulai pencarian solusi...

Solusi ditemukan setelah 7.5971 ms

Solusi ditemukan pada kombinasi ke-728:

A A B E E
D D B B B
D F F F C
G F C C C

Apakah anda ingin menyimpan solusi? (Y/N): y
Solusi berhasil disimpan di test/output_test6.txt
================================================
```

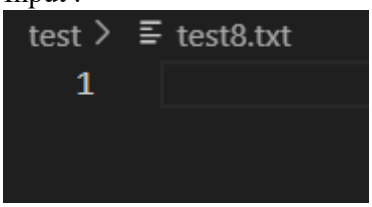
5. Test 5 : Input negatif (kurang dari sama dengan nol).
   Input :



   Output :

```
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> java -cp bin MainCLI
                SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test7.txt
Memulai pencarian solusi...

Dimensi dan jumlah piece harus positif
```

6. Test 6 : File kosong.
   Input :



   Output :

```
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> java -cp bin MainCLI
               SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test8.txt
Memulai pencarian solusi...

File kosong
File input tidak valid. Minimal harus memiliki 2 baris
```

7. Test 7 : Tidak ada file.
   Input : File tidak ada

   Output :

```
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> java -cp bin MainCLI
               SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test99.txt
File tidak ditemukan atau bukan file .txt
Masukkan nama file input (.txt): test100.txt
File tidak ditemukan atau bukan file .txt
Masukkan nama file input (.txt):
```

   *Mengulang hingga file valid

8. Test 8 : Mode selain DEFAULT.
   Input :

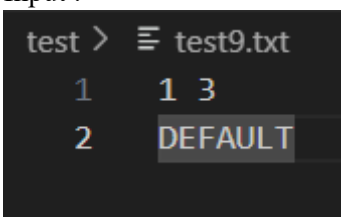
test > test8.txt

```
1   5 7 5
2   CUSTOM
3   ...X...
4   .XXXXX.
5   XXXXXXX
6   .XXXXX.
7   ...X...
8   A
9   AAA
10  BB
11  BBB
12  CCCC
13   C
14  D
15  EEE
16  E
```

Output :

```
                SELAMAT DATANG :))
==============================================

Masukkan nama file input (.txt): test8.txt
Memulai pencarian solusi...

Mode tidak valid. Saat ini hanya mendukung mode DEFAULT
```

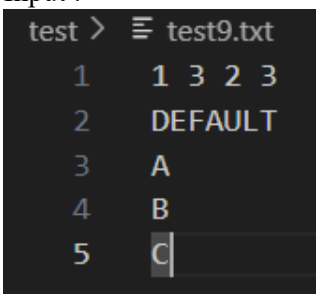
9. Test 9 : Dimensi input kurang.
   Input :



   Output :

```
                SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test9.txt
Memulai pencarian solusi...

Format dimensi tidak valid. Harus berisi 3 angka
```

10. Test 10 : Dimensi input lebihan.
    Input :



    Output :

```
                SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test9.txt
Memulai pencarian solusi...

Format dimensi tidak valid. Harus berisi 3 angka
```

11. Test 11 : Piece kurang (hanya ada 2 piece padahal harusnya 3).
    Input :

test > test10.txt

```
1 3 2
DEFAULT
A
B
```

    Output :

```
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> java -cp bin MainCLI
               SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test10.txt
Memulai pencarian solusi...

Tidak ditemukan solusi untuk puzzle ini.

================================================
```

12. Test 12 : Jumlah piece tidak sama dengan input.
    Input :

test > test11.txt

```
2 2 2
DEFAULT
A
```

    Output :

```
                SELAMAT DATANG :))
================================================

Masukkan nama file input (.txt): test11.txt
Memulai pencarian solusi...

Jumlah piece tidak sesuai dengan input
```

13. Test 13 : Kasus tidak ada solusi
    Input :

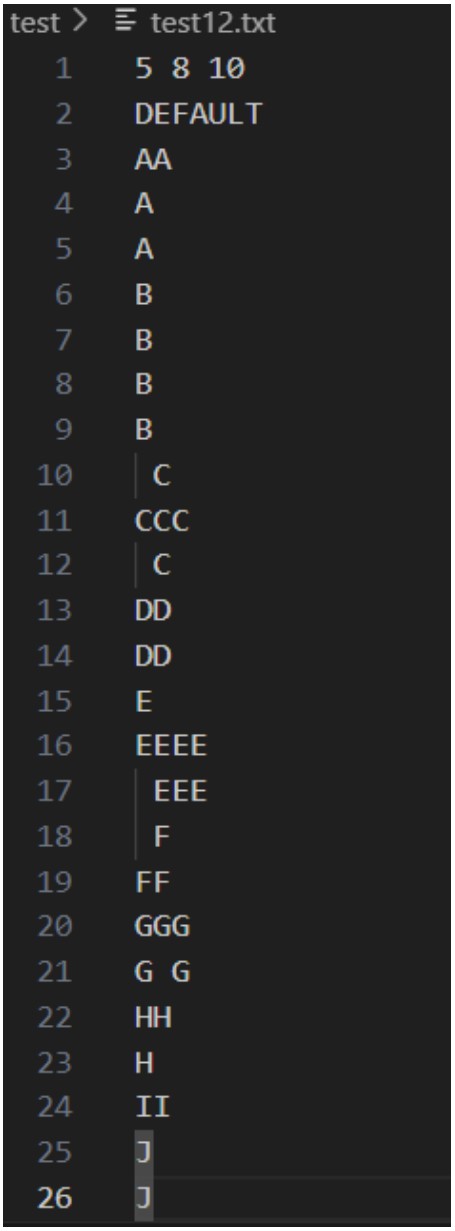
test > test16.txt

```
5 6 7
DEFAULT
AAA
AA
B
BB
BB
CC
CCC
DDD
D
F
F
F
G
GGG
H
HH
```

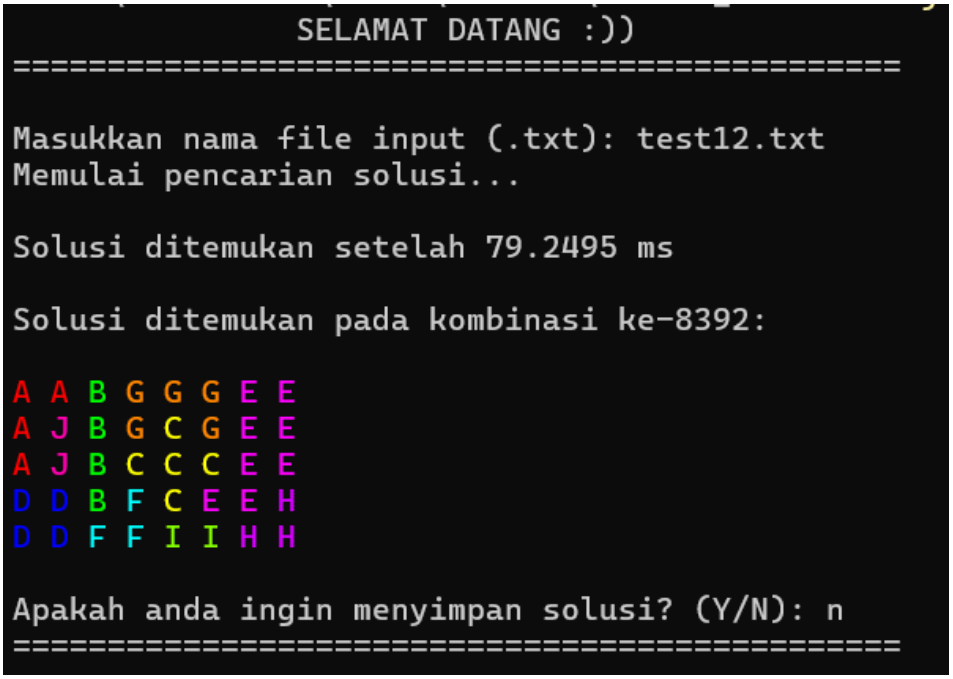
Output :

```
=====================================================
                SELAMAT DATANG :))
=====================================================

Masukkan nama file input (.txt): test16.txt
Memulai pencarian solusi...

Total waktu : 163536.0908ms

Jumlah pengujian : 50734757

Tidak ditemukan solusi untuk puzzle ini.

=====================================================
```

14. Test 14 : Kasus normal 10 block
    Input :


test > test12.txt
5 8 10
DEFAULT
AA
A
A
B
B
B
B
C
CCC
C
DD
DD
E
EEEE
EEE
F
FF
GGG
G G
HH
H
II
J
J

Output :



15. Test 15 : Kasus normal
    Input :

test14.txt

```
5 8 10
DEFAULT
AA
A
A
B
BB
B
BB
C
CCC
 C
DD
DD
EEE
EE
 F
FFF
GGG
G G
HHH
II
J
J
```

Output :

```
PS D:\SEMESTER 4\STIMA\TUCIL 1\Tucil1_13523068> java -cp bin MainCLI
==================================================
                SELAMAT DATANG :))
==================================================

Masukkan nama file input (.txt): test14.txt
Memulai pencarian solusi...

Solusi ditemukan setelah 35.6136 ms

Solusi ditemukan pada kombinasi ke-3300:

A A B G G C I I
A J B B G C C C
A J B G G F C E
D D B B F F E E
D D H H H F E E

Apakah anda ingin menyimpan solusi? (Y/N): n
==================================================
```

E. Kesimpulan

Berdasarkan uraian diatas, dapat dilihat jikalau permainan IQ Puzzle Block dapat diselesaikan dengan algoritma brute force. Jika ”beruntung”, maka solusi dapat ditemukan lebih cepat. Kelemahan algoritma ini justru ketika puzzle tidak memiliki solusi, dimana tinjauan kemungkinan yang begitu luas memberikan kompleksitas waktu yang lama.

F. Lampiran

Link Github : https://github.com/Rusmn/Tucil1_13523068

Tabel :

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 *Graphical User Interface* (GUI)		✓
6	Program dapat menyimpan solusi dalam bentuk file gambar		✓
7	Program dapat menyelesaikan kasus konfigurasi *custom*		✓
8	Program dapat menyelesaikan kasus konfigurasi Piramida (3D)		✓
9	Program dibuat oleh saya sendiri	✓	