

**Laporan Tugas Kecil 1**  
**IF2211 Strategi Algoritma**  
**Penyelesaian IQ Puzzler Pro dengan Algoritma Brute Force**

**Oleh:**  
**Muhammad Aditya Rahmadeni**  
**13523028**

**Fakultas Sekolah Teknik Elektro dan Informatika**  
**Program Studi Teknik Informatika**  
**Institut Teknologi Bandung**

# **BAB I**

## **Deksripsi Masalah**

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

### Algoritma Brute Force

Program pada awalnya akan menerima masukan berupa file *txt* yang berisi ukuran dari papan serta banyak blok dalam bentuk *integer*, lalu mode papan yang dinyatakan dalam bentuk *string* serta bentuk dari papan jika mode *custom*, lalu terakhir blok sebanyak P yang dinyatakan dalam bentuk karakter yang berbeda untuk setiap blok.

Papan dan blok memiliki *class*-nya masing-masing. Blok akan membuat semua kemungkinan rotasi sebanyak 16 blok pada saat blok diinstansiasi. Papan juga akan diinstansiasi dengan karakter titik "." yang menandakan bahwa sel bisa diisi dengan balok.

Algoritma *brute force* terletak pada *class solver.java*. Program akan melakukan iterasi untuk setiap elemen pada board (N X M) dengan nested loop. Lalu untuk setiap sel, setiap blok sebanyak P akan dicoba dengan mencoba memasukkan setiap kemungkinan rotasi pada blok tersebut

Jika blok dapat diletakkan, maka dilakukan rekursi untuk mencoba blok selanjutnya dengan mencoba setiap kemungkinan rotasi pada blok tersebut. Jika blok tidak dapat diletakkan, akan kembali ke rekursi sebelumnya dan melepas blok pada papan untuk mencoba kemungkinan selanjutnya.

Algoritma ini akan berhenti ketika semua kemungkinan telah dicoba atau pengecekan board dengan memastikan tidak ada karakter "." pada board mengembalikan nilai benar.

Basis rekursi pada algoritma ini adalah ketika index yang diakses sama dengan panjang *array of block* itu sendiri yang menandakan bahwa semua blok telah diletakkan. Lalu dilakukan pengecekan apakah semua sel dalam papan telah terisi. Jika benar, maka papan akan dicetak dan menunjukkan berapa banyak kasus yang ditinjau. Jika tidak, maka kembali ke rekursi sebelumnya.

Pseudocode:

```
Algorithm SolveBruteForce(bl, br, blockIndex, counter)
  if blockIndex == length(bl) then
    if br.checkFull() then
      print new line
      br.printBoard()
      print "Banyak kasus yang ditelusuri: " + counter
      return true
    end if
    return false
  end if

  for i from 0 to length(br.board) - 1 do
    for j from 0 to length(br.board[0]) - 1 do
      if br.board[i][j] == "." then
        for L from 0 to length(bl[blockIndex].calculatedPossBlocks) - 1 do
          counter = counter + 1
          if br.checkValidBlock(i, j,
            bl[blockIndex].calculatedPossBlocks[L]) then
            br.placeBlocks(i, j, bl[blockIndex].calculatedPossBlocks[L])
```

```
        if SolveBruteForce(bl, br, blockIndex + 1, counter) then
            return true
        end if
        br.removeBlocks(i, j, bl[blockIndex].calculatedPossBlocks[L])
    end if
end for
end if
end for
end for

return false
end Algorithm
```

## BAB III

### Source Code

Program ini memiliki 4 objek *class* untuk mempermudah proses *debugging*, yaitu:

- Main
- Solver
- Block
- Board

Setiap *code* ditampilkan dibawah ini

#### 1. Main.java

```
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
import java.util.ArrayList;

public class Main {
    private static int N, M, P;
    private static String mode;
    private static Block[] blocks;
    private static char[] usedBlock;
    private static Board board;

    public static void main(String[] args) {
        readInput();

        Solver solver = new Solver();

        long startTime = System.nanoTime();
        boolean solved = Solver.solveBruteForce(blocks, board, 0, 0);
        long endTime = System.nanoTime();

        if(!solved)
        {
            System.out.println("\nTidak Ada solusi yang ditemukan");
        }

        double duration = (endTime - startTime) / 1_000_000_000.0; // Ubah ke detik
        System.out.println(String.format("\nWaktu Eksekusi = %.5f s\n", duration));

        // Pilihan untuk menyimpan hasil atau tidak
        if(solved)
        {
            System.out.print("Apakah anda ingin menyimpan hasil ? (Y/N) : ");
            Scanner input = new Scanner(System.in);
            String save = input.next();
        }
    }
}
```

```

        if(save.equals("Y") || save.equals("y"))
        {
            System.out.print("\nApakah hasil ingin disimpan dalam bentku gambar ? (Y/N) : ");
            String saveMode = input.next();
            if(saveMode.equals("Y") || saveMode.equals("y"))
            {
                System.out.print("Masukkan nama file : ");
                String fileName = "test/" + input.next();
                board.saveBoardInImage(fileName);
            }
            else
            {
                System.out.print("Masukkan nama file : ");
                String fileName = "test/" + input.next();
                board.saveBoard(fileName);
            }
        }
    }
}

private static void readInput() {
    try {
        // Mengambil input dari cli langsung dari user
        System.out.print("Insert File Name( Make sure its located in test directory) : ");
        Scanner inputNameFile = new Scanner(System.in);
        Scanner fileparser = new Scanner(new File("./test/" + inputNameFile.next()));

        // Mengambil nilai N, M, P dari file dengan split menjadi list
        String[] tokens = fileparser.nextLine().split(" ");
        if(tokens.length != 3)
        {
            System.out.println("Input Tidak Valid");
            System.exit(1);
        }
        N = Integer.parseInt(tokens[0]);
        M = Integer.parseInt(tokens[1]);
        P = Integer.parseInt(tokens[2]);

        // Mengambil "mode" pada line selanjutnya
        mode = fileparser.nextLine().trim();
        String[] mapArray = null;
        if (mode.equals("CUSTOM")) {
            ArrayList<String> mapRow = new ArrayList<>();
            for (int i = 0; i < N; i++) {
                mapRow.add(fileparser.nextLine().trim());
            }
            mapArray = mapRow.toArray(new String[0]);
        }

        board = new Board(N, M, mode, mapArray);
        System.out.println("[+] Inisialisasi Papan berhasil !");
    }
}

```

```

        // Mengambil blok sebanyak P
        blocks = new Block[P];
        usedBlock = new char[P];
        int i = 0;
        String firstLine = fileparser.nextLine();

        while (i < P) {
            ArrayList<String> blockLines = new ArrayList<>();

            blockLines.add(firstLine);
            char charBlok = firstLine.trim().charAt(0);
            // mencegah adanya 2 blok dengan karakter yang sama, jika ada yang sama, program
            langsung keluar
            if (checkUsedBlock(charBlok))
            {
                System.out.println("Blok " + charBlok + " sudah digunakan");
                System.exit(1);
            }
            usedBlock[i] = charBlok;

            while (fileparser.hasNextLine()) {
                String nextLine = fileparser.nextLine();
                if (nextLine.isEmpty() || nextLine.trim().charAt(0) != charBlok) {
                    if (nextLine.trim().charAt(0) != charBlok) {
                        // Jika huruf berubah == blok baru maka line perlu disimpan untuk iterasi
                        berikutnya
                        firstLine = nextLine;
                    }
                    break;
                }
                blockLines.add(nextLine);
            }

            // Memasukkan array of string menjadi blok lalu memasukkan blok kedalam array of block
            String[] blockArray = blockLines.toArray(new String[0]);
            blocks[i] = new Block(blockArray);
            i++;
        }
        System.out.println("[+] Membuat semua kemungkinan blok berhasil !");

        fileparser.close();
    } catch (FileNotFoundException e) {
        System.out.println("Error : " + e.getMessage());
    }
}

private static boolean checkUsedBlock(char c)
{
    for (int i = 0; i < usedBlock.length; i++) {
        if (usedBlock[i] == c)

```

```

        {
            return true;
        }
    }
    return false;
}

```

## 2. Solver.java

```

public class Solver {
    // Bruteforcing menggunakan rekursi untuk setiap blok pada array dan setiap titik pada board
    public static boolean solveBruteForce(Block[] bl, Board br, int blockIndex, int counter){
        // Kalau blok full, maka print board dan return true
        if (blockIndex == bl.length) {
            if (br.checkFull()) {
                System.out.println();
                br.printBoard();
                System.out.println("\nBanyak kasus yang ditelusuri: " + counter);
                return true;
            }
            return false;
        }

        // Iterasi untuk setiap kolom, baris, dan block. lalu untuk memasang setiap kemungkinan
        for (int i = 0; i < br.board.length; i++) {
            for (int j = 0; j < br.board[0].length; j++) {
                // if(br.board[i][j].equals("."))
                // {
                    for (int L = 0; L < bl[blockIndex].calculatedPossBlocks.length; L++) {
                        counter++;
                        if (br.checkValidBlock(i, j, bl[blockIndex].calculatedPossBlocks[L])) {
                            br.placeBlocks(i, j, bl[blockIndex].calculatedPossBlocks[L]);

                            // Rekursi untuk mencoba blok selanjutnya
                            if (solveBruteForce(bl, br, blockIndex + 1, counter)) {
                                return true;
                            }

                            // jika tidak ada blok sesuai, maka blok skarang dihapus dan mencoba
                            kemungkinan selanjutnya
                            br.removeBlocks(i, j, bl[blockIndex].calculatedPossBlocks[L]);
                        }
                    }
                }
            }
        }
        return false;
    }
}

```



### 3. Block.java

```
import java.util.Arrays;

public class Block {
    public String[][] block;
    public String[][][] calculatedPossBlocks;

    public Block(String[] lines) {
        int rows = lines.length;
        int cols = Arrays.stream(lines).mapToInt(String::length).max().orElse(0);

        block = new String[rows][cols];
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (j < lines[i].length())
                {
                    if (lines[i].charAt(j) != ' ')
                    {
                        block[i][j] = String.valueOf(lines[i].charAt(j));
                    }
                    else
                    {
                        block[i][j] = ".";
                    }
                }
                else
                {
                    // Kalau ada Null (jika panjang suatu baris lebih pendek daripada baris terpanjang)
                    // dalam string maka diganti menjadi .
                    block[i][j] = ".";
                }
            }
        }
        generatePossibleBlocks();
    }

    // Buat semua kemungkinan rotasi dalam 2D
    private void generatePossibleBlocks() {
        calculatedPossBlocks = new String[16][3][3]; // satu axis kemungkinan 8 sehingga maka total
        // kemungkinan adalah 16

        String[][] tempBlok;
        if (block.length != block[0].length)
        {
            tempBlok = centerBlock(block);
        }
        else // Jika blok sudah simetris atau nxn maka tidak perlu centering
        {

```

```

        tempBlok = blok;
    }

    for (int i = 0; i < 8; i++) {
        calculatedPossBlocks[i] = tempBlok;
        tempBlok = rotate45(tempBlok);
    }

    for (int i = 0; i < 8; i++) {
        calculatedPossBlocks[i + 8] = flipBlock(calculatedPossBlocks[i]);
    }
}

// Centering blok ke matriks nxn dengan n adalah max(col, row)
private String[][] centerBlock(String[][] blk)
{
    int rows = blk.length;
    int cols = blk[0].length;
    int maxDim = Math.max(rows, cols);
    String[][] centered = new String[maxDim][maxDim];

    for (int i = 0; i < maxDim; i++) {
        for (int j = 0; j < maxDim; j++) {
            centered[i][j] = ".";
        }
    }

    int rowOffset = (maxDim - rows) / 2;
    int colOffset = (maxDim - cols) / 2;

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            centered[i + rowOffset][j + colOffset] = blk[i][j];
        }
    }

    return centered;
}

// Rotasi blok 45 derajat menggunakan teorema pada aljabar linear
private String[][] rotate45(String[][] blk) {
    // blk sudah dipastikan berukuran nxn karena centering atau sudah nxn dari awal
    int n = blk.length;
    String[][] rotated = new String[n][n];

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            rotated[i][j] = ".";
        }
    }
}

```

```

// Perhitungan titik tengah dan besar rotasi dari titik tengah
double centerX = (n - 1) / 2.0;
double centerY = (n - 1) / 2.0;
double angle = Math.PI / 4; // 45 derajat
double cosA = Math.cos(angle);
double sinA = Math.sin(angle);

// Traverse untuk setiap elemen
if (n == 2) {
    // Khusus metriks 2x2
    rotated[0][0] = blk[1][0];
    rotated[0][1] = blk[0][0];
    rotated[1][0] = blk[1][1];
    rotated[1][1] = blk[0][1];
}
else
{
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            // Translasi lalu rotasi blok
            double x = i - centerX;
            double y = j - centerY;
            double newX = cosA * x - sinA * y;
            double newY = sinA * x + cosA * y;
            int resX = (int) Math.round(newX + centerX);
            int resY = (int) Math.round(newY + centerY);

            // Ensure the resX and resY are within bounds
            if (resX >= 0 && resX < n && resY >= 0 && resY < n) {
                rotated[resX][resY] = blk[i][j];
            }
        }
    }

    return rotated;
}

// Mirroring secara harizontal
private String[][] flipBlock(String[][] blk) {
    String[][] flipped = new String[blk.length][blk[0].length];
    for (int i = 0; i < blk.length; i++) {
        for (int j = 0; j < blk[0].length; j++) {
            flipped[i][j] = blk[i][blk[0].length - j - 1];
        }
    }

    return flipped;
}

```

```

// Print block (debugging only)
public void printBlock(String[][] blk) {
    for (String[] row : blk) {
        for (String cell : row) {
            System.out.print(cell + " ");
        }
        System.out.println();
    }
    System.out.println();
}

// Print semua kemungkinan blok (debugging only)
public void printAllBlocks() {
    for (int i = 0; i < calculatedPossBlocks.length; i++) {
        System.out.println("Transformation " + (i + 1) + ":" );
        printBlock(calculatedPossBlocks[i]);
    }
}
}

```

#### 4. Board.java

```

import java.util.HashMap;
import java.util.Map;
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import javax.imageio.ImageIO;

public class Board {
    public String[][] board;
    private static final String RESET = "\033[0m";
    private static final Color[] COLORS = {
        Color.RED, Color.GREEN, Color.YELLOW, Color.BLUE, Color.MAGENTA, Color.CYAN,
        new Color(128, 0, 128), new Color(255, 0, 0), new Color(0, 255, 0), new Color(255, 255, 0),
        new Color(0, 0, 255), new Color(255, 0, 255), new Color(0, 255, 255), new Color(128, 128,
128),
        new Color(255, 165, 0), new Color(255, 140, 0), new Color(255, 69, 0), new Color(255, 255,
224),
        new Color(144, 238, 144), new Color(173, 216, 230), new Color(255, 182, 193),
        new Color(224, 255, 255), new Color(255, 0, 0), new Color(0, 255, 0), new Color(255, 255,
0),
        new Color(0, 0, 255), new Color(255, 0, 255)
    };
    private Map<String, String> colorMap;
}

```

```

public Board(int N, int M, String mode, String[] maplist) {
    board = new String[N][M];
    if (mode.equals("CUSTOM")) {
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < M; j++) {
                board[i][j] = (maplist[i].charAt(j) == 'X') ? "." : "*";
            }
        }
    } else {
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < M; j++) {
                board[i][j] = ".";
            }
        }
    }
    colorMap = new HashMap<>();
}

public void saveBoard(String fileName) {
    try (FileWriter myWriter = new FileWriter(fileName)) {
        for (String[] row : board) {
            for (String cell : row) {
                myWriter.write(cell);
            }
            myWriter.write("\n");
        }
        System.out.println("Board successfully saved to " + new
File(fileName).getAbsolutePath());
    } catch (IOException e) {
        System.err.println("Error saving board: " + e.getMessage());
        e.printStackTrace();
    }
}

public void saveBoardInImage(String fileName) {
    int cellSize = 20;
    int width = board[0].length * cellSize;
    int height = board.length * cellSize;

    BufferedImage image = new BufferedImage(width, height, BufferedImage.TYPE_INT_RGB);
    Graphics g = image.getGraphics();
    g.setFont(new Font("Arial", Font.PLAIN, cellSize));

    for (int i = 0; i < board.length; i++) {
        for (int j = 0; j < board[0].length; j++) {
            String cell = board[i][j];
            int index = cell.charAt(0) - 'A';
            if (cell.equals(".")) {
                g.setColor(Color.WHITE);
            } else if (index >= 0 && index < COLORS.length) {

```

```

        g.setColor(COLORS[index]);
    } else {
        g.setColor(Color.BLACK);
    }
    g.fillRect(j * cellSize, i * cellSize, cellSize, cellSize);
    g.setColor(Color.BLACK);
    g.drawRect(j * cellSize, i * cellSize, cellSize, cellSize);
    g.setColor(Color.BLACK);
    g.drawString(cell, j * cellSize + 5, i * cellSize + cellSize - 5);
}
}
g.dispose();

try {
    File outputFile = new File(fileName);
    outputFile.getParentFile().mkdirs(); // Ensure directories exist
    ImageIO.write(image, "jpg", outputFile);
    System.out.println("Image successfully saved to " + outputFile.getAbsolutePath());
} catch (IOException e) {
    System.err.println("Error saving board image: " + e.getMessage());
    e.printStackTrace();
}
}

public void printBoard() {
    for (int i = 0; i < board.length; i++) {
        for (int j = 0; j < board[0].length; j++) {
            String cell = board[i][j];
            if (cell.equals(".")) {
                System.out.print(cell + " ");
            } else {
                String colorCode;
                if (cell.matches("[A-Z]")) {
                    colorCode = getColorCode(COLORS[cell.charAt(0) - 'A']);
                } else {
                    colorCode = "\033[37m"; // White for any other characters
                }
                System.out.print(colorCode + cell + RESET + " ");
            }
        }
        System.out.println();
    }
}

private String getColorCode(Color color) {
    if (color.equals(Color.RED)) return "\033[31m";
    if (color.equals(Color.GREEN)) return "\033[32m";
    if (color.equals(Color.YELLOW)) return "\033[33m";
    if (color.equals(Color.BLUE)) return "\033[34m";
    if (color.equals(Color.MAGENTA)) return "\033[35m";
    if (color.equals(Color.CYAN)) return "\033[36m";
}

```

```

        // Add more color mappings as needed
        return "\033[37m"; // Default to white
    }

    public boolean checkValidBlock(int x, int y, String[][] bl) {
        // Cek apakah blok valid (dapat diletakkan) pada x dan y dalam board
        for (int i = 0; i < bl.length; i++) {
            for (int j = 0; j < bl[0].length; j++) {
                if (!bl[i][j].equals(".")) {
                    if (x + i >= board.length || y + j >= board[0].length) {
                        return false;
                    }
                    if (!board[x + i][y + j].equals(".")) {
                        return false;
                    }
                }
            }
        }
        return true;
    }

    public boolean checkFull() {
        // Cek kondisi menang atau jika board telah full (tidak ada character .)
        for (int i = 0; i < board.length; i++) {
            for (int j = 0; j < board[0].length; j++) {
                if (board[i][j].equals(".")) {
                    return false;
                }
            }
        }
        return true;
    }

    public void placeBlocks(int i, int j, String[][] bl) {
        for (int x = 0; x < bl.length; x++) {
            for (int y = 0; y < bl[0].length; y++) {
                if (!bl[x][y].equals(".")) {
                    board[i + x][j + y] = bl[x][y];
                }
            }
        }
    }

    public void removeBlocks(int i, int j, String[][] bl) {
        for (int x = 0; x < bl.length; x++) {
            for (int y = 0; y < bl[0].length; y++) {
                if (!bl[x][y].equals(".")) {
                    board[i + x][j + y] = ".";
                }
            }
        }
    }
}

```

}



## BAB IV

### Hasil

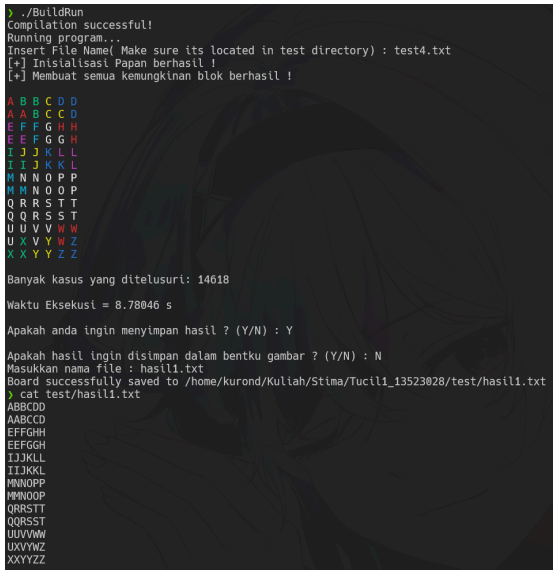
#### 1. Test Case 1 (DEFAULT / Ada solusi )

Input	Output
5 5 7 DEFAULT A AA B BB C CC D DD EE EE E FF FF F GGG	<pre> &gt; ./BuildRun Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : test.txt [+] Inisialisasi Papan berhasil ! [+] Membuat semua kemungkinan blok berhasil !  A B B C C A A B C D G G G D D E E F F F E E F F F  Banyak kasus yang ditelusuri: 725 Waktu Eksekusi = 0.32169 s Apakah anda ingin menyimpan hasil ? (Y/N) : N </pre>

#### 2. Test Case 2 (DEFAULT / Tidak Ada Solusi )


Input	Output
3 3 1 DEFAULT A	<pre> &gt; ./BuildRun Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : test3.txt [+] Inisialisasi Papan berhasil ! [+] Membuat semua kemungkinan blok berhasil !  Tidak Ada solusi yang ditemukan Waktu Eksekusi = 0.00064 s </pre>

3. Test Case 3 (DEFAULT / Simpan di dalam file txt)

Input	Output
<p>13 6 26            DEFAULT            A            AA            B            BB            C            CC            D            DD            E            EE            F            FF            G            GG            H            HH            I            II            J            JJ            K            KK            L            LL            M            MM            N            NN            O            OO            P            PP            Q            QQ            R            RR            S            SS            T            TT            U            UU            V            VV</p>	 <pre> &gt; ./BuildRun Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : test4.txt [*] Inisialisasi Papan berhasil ! [*] Membuat semua kemungkinan blok berhasil !  A B B C D D A A B C C D E F F G H H E E F G H H I J J K L L I J J K K L M N N O P P M M N O O P Q R R S T T Q Q R S S T U U V V W W U X Y Y Z Z X X Y Y Z Z  Banyak kasus yang ditelusuri: 14618 Waktu Eksekusi = 8.78046 s Apakah anda ingin menyimpan hasil ? (Y/N) : Y Apakah hasil ingin disimpan dalam bentuk gambar ? (Y/N) : N Masukkan nama file : hasil1.txt Board successfully saved to /home/kurond/Kuliah/Stima/Tucil1_13523028/test/hasil1.txt &gt; cat test/hasil1.txt ABBCDD AABBCD EEFGHH EEFGHH IJJKLL IJJKLL MNNOPP MNNOPP QRRSTT QRRSTT UUVVWW UUVVWW XXYYZZ XXYYZZ </pre>

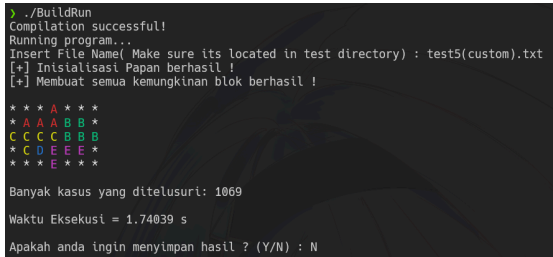
W WW X XX Y YY Z ZZ	
--	--

4. Test Case 4 (DEFAULT / Simpan dalam gambar )

Input	Output
13 6 26 DEFAULT A AA B BB C CC D DD E EE F FF G GG H HH I II J JJ K KK L LL M MM N NN O OO P PP Q	<pre> D:\Study\Tucil1_13523028-Buildrun.bat warning: [options] source value 8 is obsolete and will be removed in a future release warning: [options] target value 8 is obsolete and will be removed in a future release warning: [options] To suppress warnings about obsolete options, use -Xlint:-options. 3 warnings Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : test4.txt [*] Instalikasi Papan berhasil ! [*] Membuat semua kemungkinan blok berhasil !  A B B C D D A A B C C D E F F G H H E E F G G H I J J K L L I I J K K L M N N O P P M M N O O P Q R R S T T Q Q R S S T U U V V W W U X V Y W Z X X Y Y Z Z  Banyak kasus yang ditelusuri: 14618 Waktu Eksekusi = 6.94863 s  Apakah anda ingin menyimpan hasil ? (Y/N) : Y Apakah hasil ingin disimpan dalam bentuk gambar ? (Y/N) : Y Masukkan nama file : hasil2.jpg Image successfully saved to D:\Study\Tucil1_13523028\test\hasil2.jpg </pre> 

QQ R RR S SS T TT U UU V VV W WW X XX Y YY Z ZZ	
---	--


#### 5. Test Case 5 (CUSTOM / Ada solusi)

Input	Output
5 7 5 CUSTOM ...X... .XXXXX. XXXXXXXX .XXXXX. ...X... A AAA BB BBB CCCC C D EEE E	 <pre> ./BuildRun Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : test5(custom).txt [+] Inisialisasi Papan berhasil ! [+] Membuat semua kemungkinan blok berhasil !  ***A*** *AAABBB* CCCCBBB *CDDEE* ***E***  Banyak kasus yang ditelusuri: 1069 Waktu Eksekusi = 1.74039 s Apakah anda ingin menyimpan hasil ? (Y/N) : N </pre>

6. Test Case 6 (CUSTOM / Tidak ada solusi)

Input	Output
5 7 5 CUSTOM ...X... .XXXXX. XXXXXXXX .XXXXX. ...X... A AAA BB BBB CCCC C D EE E	<pre> &gt; ./BuildRun Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : test6(custom).txt [+] Inisialisasi Papan berhasil ! [+] Membuat semua kemungkinan blok berhasil !  Tidak Ada solusi yang ditemukan Waktu Eksekusi = 23.40191 s </pre>

7. Test Case 7 (CUSTOM / Simpan dalam file)

Input	Output
5 7 5 CUSTOM ...X... .XXXXX. XXXXXXXX .XXXXX. ...X... A AAA BB BBB CCCC C D EEE E	<pre> D:\Study\Tucil1_13523028-BuildRun.bat warning: [options] source value 8 is obsolete and will be removed in a future release warning: [options] target value 8 is obsolete and will be removed in a future release warning: [options] To suppress warnings about obsolete options, use -Xlint:-options. 3 warnings Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : test5(custom).txt [+] Inisialisasi Papan berhasil ! [+] Membuat semua kemungkinan blok berhasil !  *** A *** * A A A B B * * C C C B B B * * C D E E E * *** E ***  Banyak kasus yang ditelusuri: 1069 Waktu Eksekusi = 1.39836 s  Apakah anda ingin menyimpan hasil ? (Y/N) : Y  Apakah hasil ingin disimpan dalam bentuk gambar ? (Y/N) : Y Masukkan nama file : hasilcustom.jpg Image successfully saved to D:\Study\Tucil1_13523028\test\hasilcustom.jpg </pre> 

8. Test Case 8 (Input tidak valid)

Input	Output
5 5 DEFAULT A AA B BB C CC D DD EE EE E FF FF F GGG	<pre>&gt; ./BuildRun Compilation successful! Running program... Insert File Name( Make sure its located in test directory) : notvalid.txt Input Tidak Valid</pre>

## LAMPIRAN

Pranala Github : [https://github.com/Kurosue/Tucil1\\_13523028](https://github.com/Kurosue/Tucil1_13523028)

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	✓	