IF2211 – Strategi Algoritma

Tugas Kecil

IQ Puzzler Pro Solver

Disusun Oleh:

SALMAN HANIF

13523056

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

Bandung 2025

BAB 1: Deskripsi Tugas

Gambar 1 Permainan IQ Puzzler Pro

(Sumber: https://www.smartgamesusa.com)

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.

Tugas anda 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 2: Algoritma Brute Force

2.1. Langkah-Langkah Ide Algoritma

1) Inisialisasi board dan list of pieces
2) Buat list dari semua kemungkinan rotasi dan mirror dari setiap piece.
3) Cek prasyarat apakah board dan piece memungkinkan untuk solusi (tidak ada piece yang Panjang melebihi board, jumlah dot piece dan slot board sama banyak)
4) Algoritma untuk menempatkan pieces:
   - Coba tempatkan piece pertama di semua koordinat yang memungkinkan.

   Untuk setiap piece berikutnya:
   - Coba semua posisi kosong di board, mulai dari (0,0).
   - Jika tidak bisa, coba semua rotasi dan mirror.
   - Jika masih gagal, geser ke kanan. Jika mencapai tepi board, geser ke baris bawah.
5) Jika tidak dapat menempatkan piece ke-n, maka:
   - Coba alternatif rotasi/mirror lain piece terakhir atau geser piece terakhir ke posisi lain
   - Jika semua kemungkinan gagal, kembali ke piece sebelumnya dan coba alternatif baru
6) Jika semua pieces berhasil ditempatkan, cek apakah board terisi penuh
7) Jika tidak ada solusi ditemukan setelah semua kemungkinan, maka return no solution.

2.2. Pseudocode Algoritma Utama

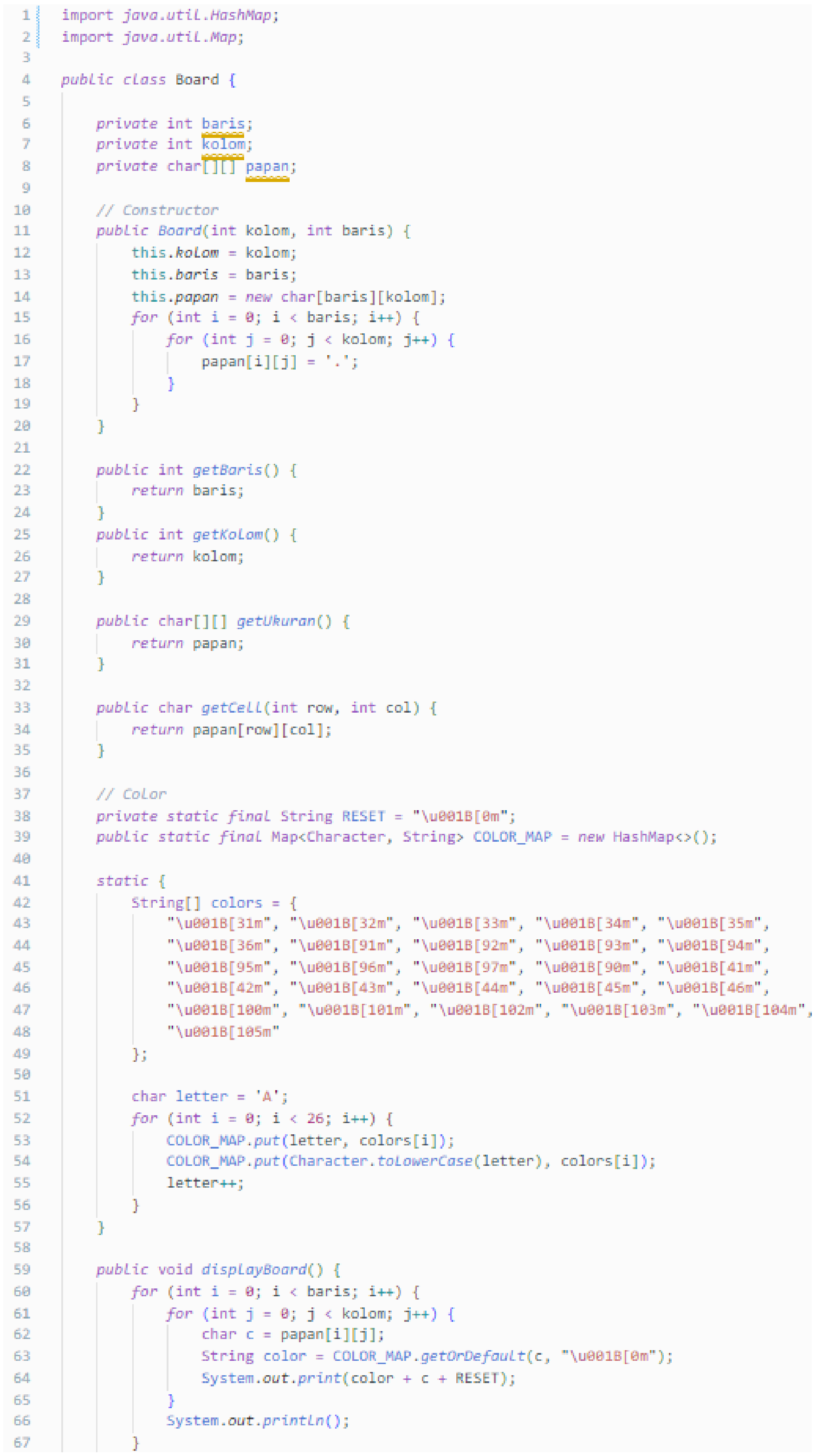
```
Algorithm IQPuzzlerProSolver
   Input: pieces (list of pieces), board (game board)
   Output: Board with placed pieces or indication of no solution

   DECLARE Stack posisiPiece  // Stack to store last placed piece positions
   DECLARE Array indices OF INTEGER WITH SIZE pieces.size()  // Array to track indices
of pieces
   DECLARE Integer pieceIndex = 0
   DECLARE Integer x = 0, y = 0, rotationIndex = 0
   DECLARE Boolean placed
   DECLARE List pieceCombination
   DECLARE Piece piece
   DECLARE Array lastPosition OF INTEGER WITH SIZE 4
   DECLARE Piece lastPiece
```

```
  WHILE pieceIndex >= 0 DO
    IF pieceIndex >= pieces.size() THEN
      IF board.isEndGame() THEN
        BREAK  // Solution found, exit loop
      END IF
      pieceIndex = pieceIndex - 1  // Backtrack
      CONTINUE  // Retry previous step
    END IF

    placed = FALSE
    pieceCombination = allPieceCombinations[pieceIndex]  // Get all rotations of the current
piece

    WHILE rotationIndex < pieceCombination.size() DO
      piece = pieceCombination[rotationIndex]

      FOR y = 0 TO board.getBaris() - 1 DO
        FOR x = 0 TO board.getKolom() - 1 DO
          IF board.canPlaced(piece, x, y) THEN
            board.placePiece(piece, x, y)
            posisiPiece.push([pieceIndex, rotationIndex, x, y])

            // Move to the next piece
            pieceIndex = pieceIndex + 1
            rotationIndex = 0
            placed = TRUE
            BREAK
          END IF
        END FOR
        IF placed THEN BREAK
      END FOR
      IF placed THEN BREAK

      rotationIndex = rotationIndex + 1  // Try next rotation
    END WHILE

    IF placed == FALSE THEN
      IF posisiPiece IS NOT EMPTY THEN
        lastPosition = posisiPiece.pop()
        pieceIndex = lastPosition[0]
        rotationIndex = lastPosition[1] + 1
        x = lastPosition[2]
```

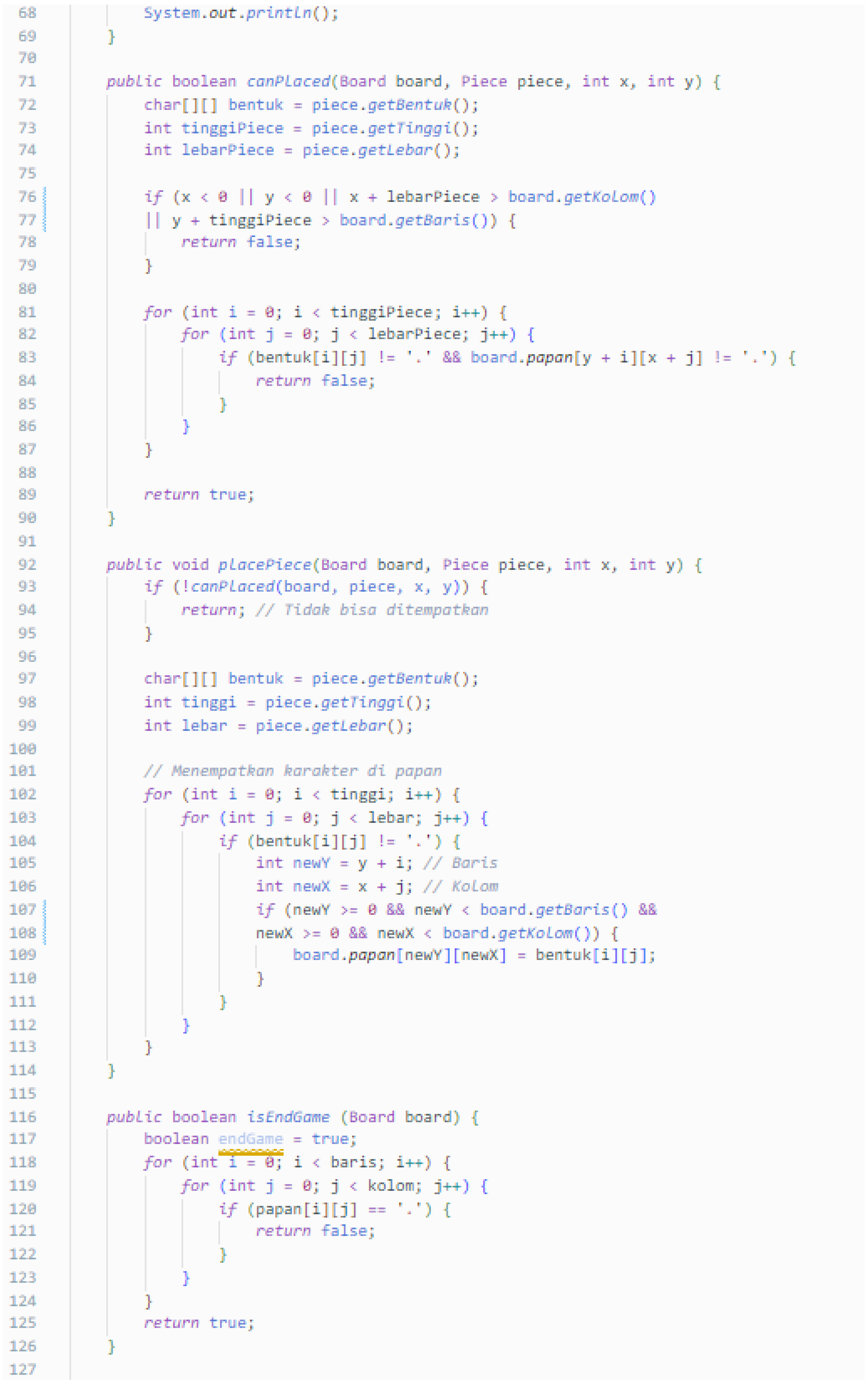
```
            lastPiece = allPieceCombinations[pieceIndex][lastPosition[1]]
            board.removePiece(lastPiece, x, y)  // Undo last placement
         ELSE
            BREAK  // Exit if no solution
         END IF
      END IF
   END WHILE
END Algorithm
```

BAB 3: Source Code

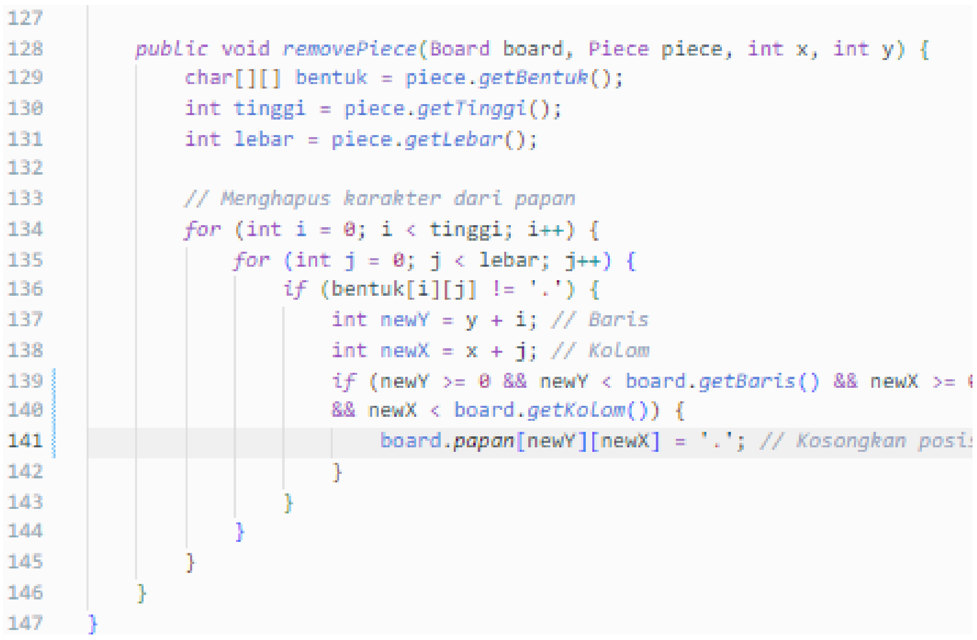
3.1 Board.java



```java
import java.util.HashMap;
import java.util.Map;

public class Board {

    private int baris;
    private int kolom;
    private char[][] papan;

    // Constructor
    public Board(int kolom, int baris) {
        this.kolom = kolom;
        this.baris = baris;
        this.papan = new char[baris][kolom];
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                papan[i][j] = '.';
            }
        }
    }

    public int getBaris() {
        return baris;
    }
    public int getKolom() {
        return kolom;
    }

    public char[][] getUkuran() {
        return papan;
    }

    public char getCell(int row, int col) {
        return papan[row][col];
    }

    // Color
    private static final String RESET = "\u001B[0m";
    public static final Map<Character, String> COLOR_MAP = new HashMap<>();

    static {
        String[] colors = {
            "\u001B[31m", "\u001B[32m", "\u001B[33m", "\u001B[34m", "\u001B[35m",
            "\u001B[36m", "\u001B[91m", "\u001B[92m", "\u001B[93m", "\u001B[94m",
            "\u001B[95m", "\u001B[96m", "\u001B[97m", "\u001B[90m", "\u001B[41m",
            "\u001B[42m", "\u001B[43m", "\u001B[44m", "\u001B[45m", "\u001B[46m",
            "\u001B[100m", "\u001B[101m", "\u001B[102m", "\u001B[103m", "\u001B[104m",
            "\u001B[105m"
        };

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

    public void displayBoard() {
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                char c = papan[i][j];
                String color = COLOR_MAP.getOrDefault(c, "\u001B[0m");
                System.out.print(color + c + RESET);
            }
            System.out.println();
        }
```

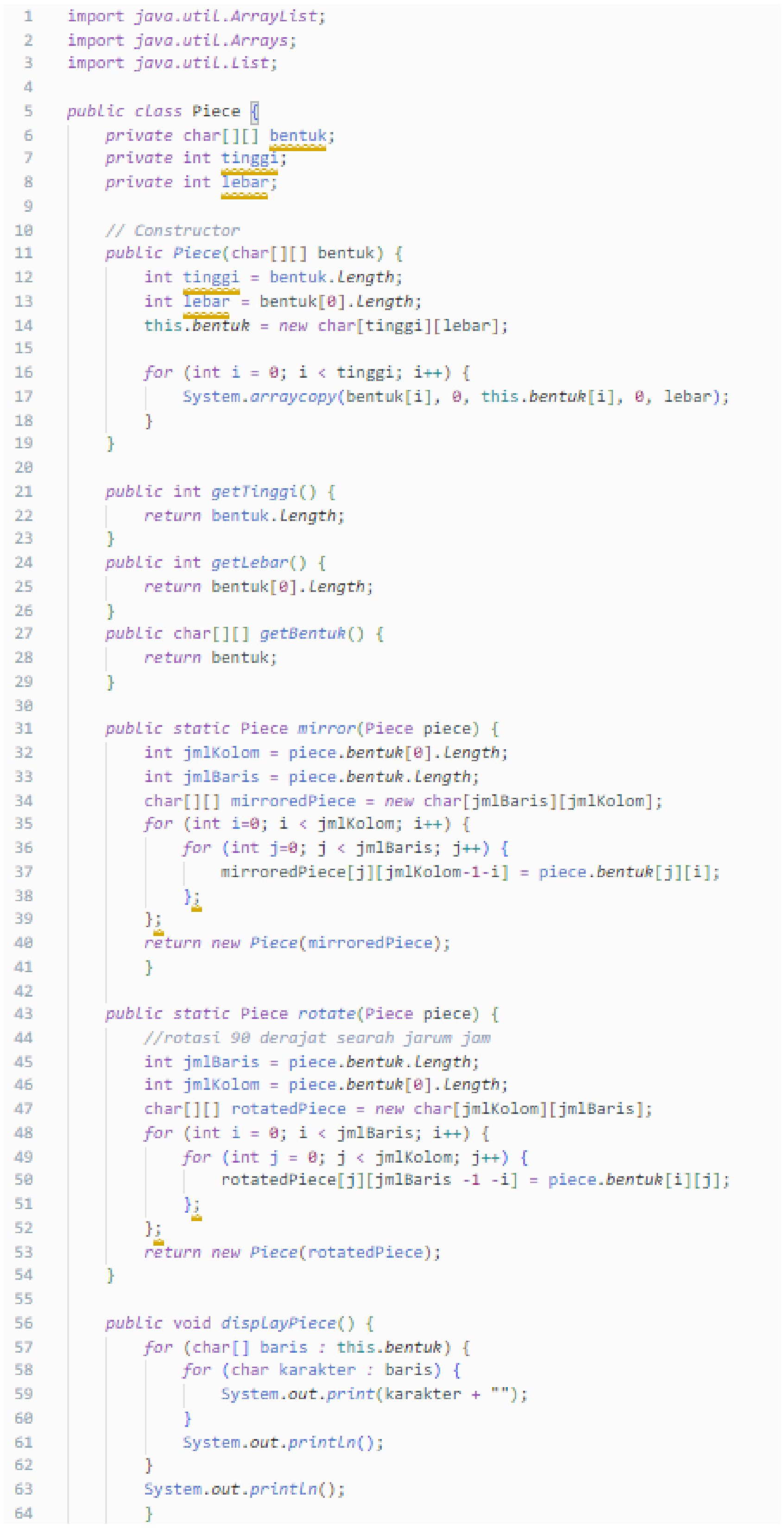


```java
        System.out.println();
    }

    public boolean canPlaced(Board board, Piece piece, int x, int y) {
        char[][] bentuk = piece.getBentuk();
        int tinggiPiece = piece.getTinggi();
        int lebarPiece = piece.getLebar();

        if (x < 0 || y < 0 || x + lebarPiece > board.getKolom()
        || y + tinggiPiece > board.getBaris()) {
            return false;
        }

        for (int i = 0; i < tinggiPiece; i++) {
            for (int j = 0; j < lebarPiece; j++) {
                if (bentuk[i][j] != '.' && board.papan[y + i][x + j] != '.') {
                    return false;
                }
            }
        }

        return true;
    }

    public void placePiece(Board board, Piece piece, int x, int y) {
        if (!canPlaced(board, piece, x, y)) {
            return; // Tidak bisa ditempatkan
        }

        char[][] bentuk = piece.getBentuk();
        int tinggi = piece.getTinggi();
        int lebar = piece.getLebar();

        // Menempatkan karakter di papan
        for (int i = 0; i < tinggi; i++) {
            for (int j = 0; j < lebar; j++) {
                if (bentuk[i][j] != '.') {
                    int newY = y + i; // Baris
                    int newX = x + j; // Kolom
                    if (newY >= 0 && newY < board.getBaris() &&
                    newX >= 0 && newX < board.getKolom()) {
                        board.papan[newY][newX] = bentuk[i][j];
                    }
                }
            }
        }
    }

    public boolean isEndGame (Board board) {
        boolean endGame = true;
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                if (papan[i][j] == '.') {
                    return false;
                }
            }
        }
        return true;
    }

```

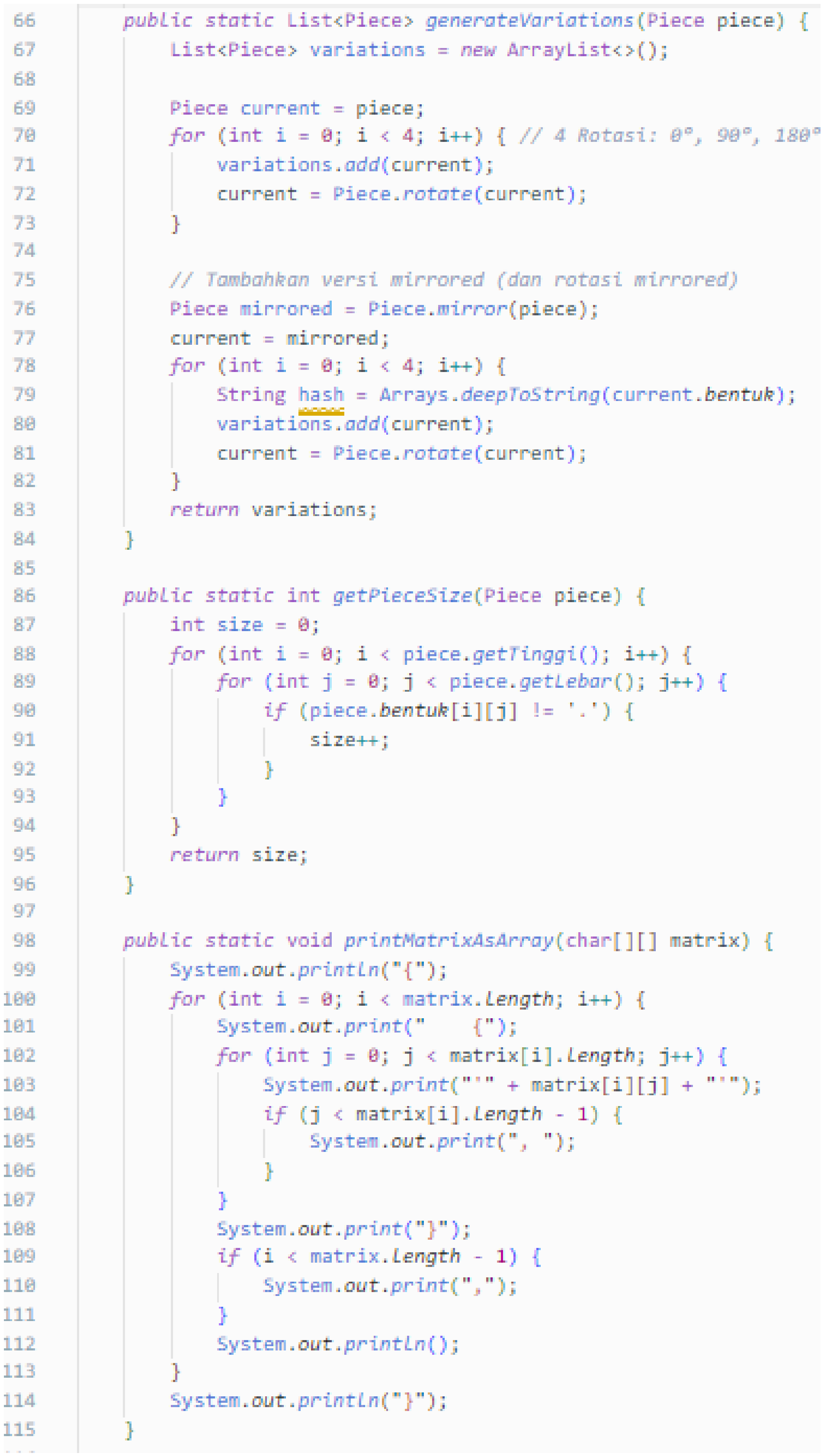


```java

    public void removePiece(Board board, Piece piece, int x, int y) {
        char[][] bentuk = piece.getBentuk();
        int tinggi = piece.getTinggi();
        int lebar = piece.getLebar();

        // Menghapus karakter dari papan
        for (int i = 0; i < tinggi; i++) {
            for (int j = 0; j < lebar; j++) {
                if (bentuk[i][j] != '.') {
                    int newY = y + i; // Baris
                    int newX = x + j; // Kolom
                    if (newY >= 0 && newY < board.getBaris() && newX >= 
                    && newX < board.getKolom()) {
                        board.papan[newY][newX] = '.'; // Kosongkan posi
                    }
                }
            }
        }
    }
}
```

3.2 Piece.java

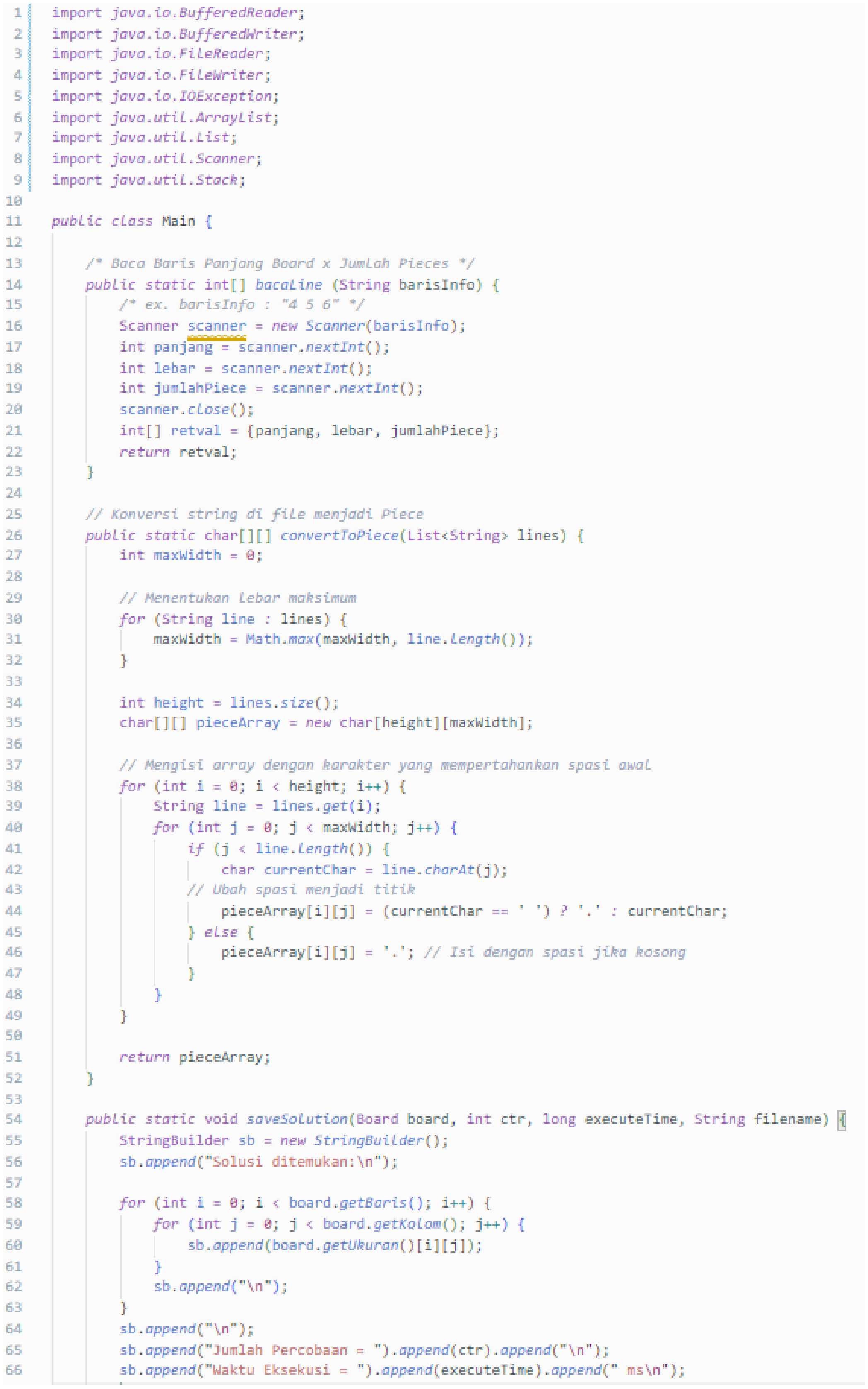


```java
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class Piece {
    private char[][] bentuk;
    private int tinggi;
    private int lebar;

    // Constructor
    public Piece(char[][] bentuk) {
        int tinggi = bentuk.length;
        int lebar = bentuk[0].length;
        this.bentuk = new char[tinggi][lebar];

        for (int i = 0; i < tinggi; i++) {
            System.arraycopy(bentuk[i], 0, this.bentuk[i], 0, lebar);
        }
    }

    public int getTinggi() {
        return bentuk.length;
    }
    public int getLebar() {
        return bentuk[0].length;
    }
    public char[][] getBentuk() {
        return bentuk;
    }

    public static Piece mirror(Piece piece) {
        int jmlKolom = piece.bentuk[0].length;
        int jmlBaris = piece.bentuk.length;
        char[][] mirroredPiece = new char[jmlBaris][jmlKolom];
        for (int i=0; i < jmlKolom; i++) {
            for (int j=0; j < jmlBaris; j++) {
                mirroredPiece[j][jmlKolom-1-i] = piece.bentuk[j][i];
            };
        };
        return new Piece(mirroredPiece);
    }

    public static Piece rotate(Piece piece) {
        //rotasi 90 derajat searah jarum jam
        int jmlBaris = piece.bentuk.length;
        int jmlKolom = piece.bentuk[0].length;
        char[][] rotatedPiece = new char[jmlKolom][jmlBaris];
        for (int i = 0; i < jmlBaris; i++) {
            for (int j = 0; j < jmlKolom; j++) {
                rotatedPiece[j][jmlBaris -1 -i] = piece.bentuk[i][j];
            };
        };
        return new Piece(rotatedPiece);
    }

    public void displayPiece() {
        for (char[] baris : this.bentuk) {
            for (char karakter : baris) {
                System.out.print(karakter + "");
            }
            System.out.println();
        }
        System.out.println();
    }
```

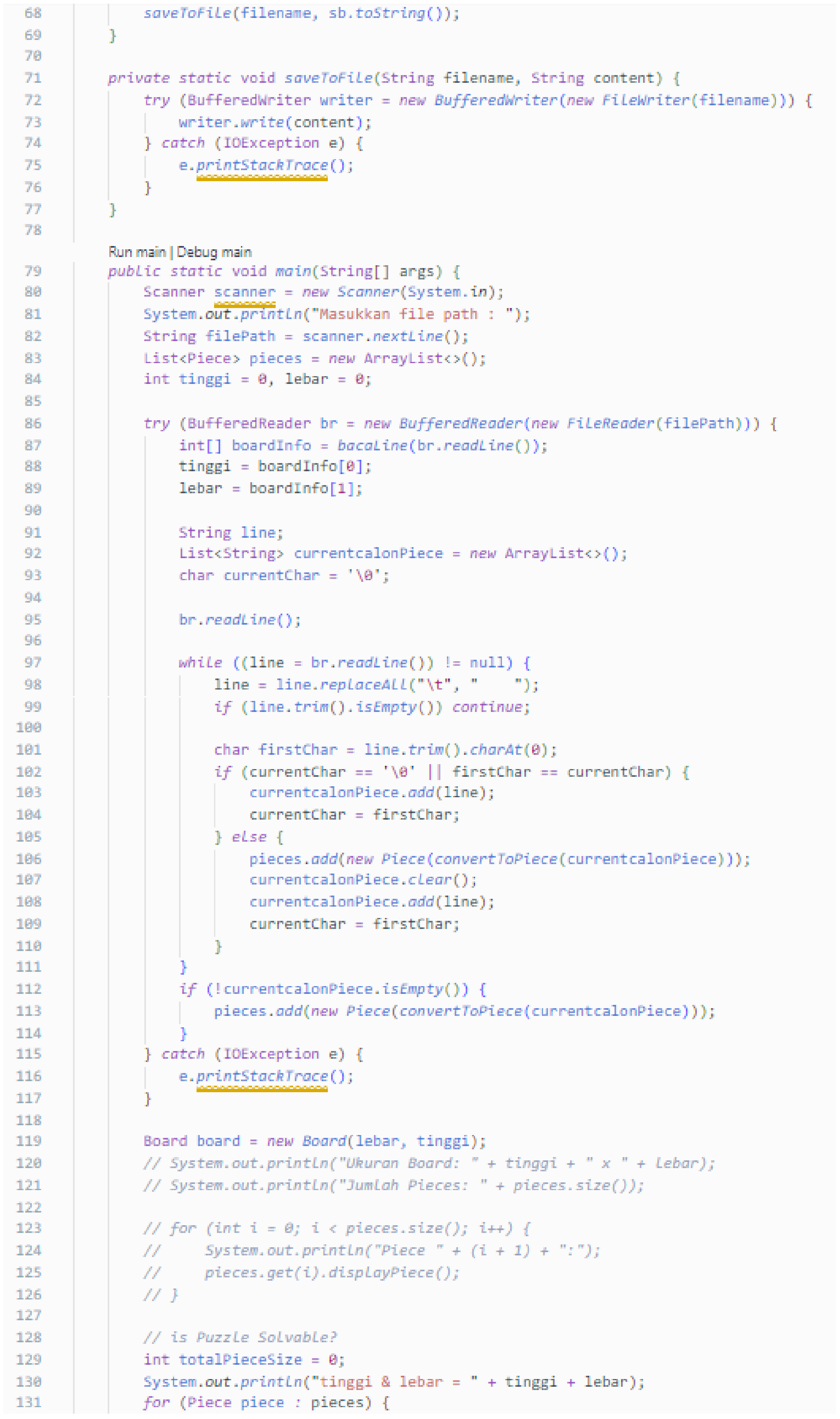


```java
    public static List<Piece> generateVariations(Piece piece) {
        List<Piece> variations = new ArrayList<>();

        Piece current = piece;
        for (int i = 0; i < 4; i++) { // 4 Rotasi: 0°, 90°, 180°
            variations.add(current);
            current = Piece.rotate(current);
        }

        // Tambahkan versi mirrored (dan rotasi mirrored)
        Piece mirrored = Piece.mirror(piece);
        current = mirrored;
        for (int i = 0; i < 4; i++) {
            String hash = Arrays.deepToString(current.bentuk);
            variations.add(current);
            current = Piece.rotate(current);
        }
        return variations;
    }

    public static int getPieceSize(Piece piece) {
        int size = 0;
        for (int i = 0; i < piece.getTinggi(); i++) {
            for (int j = 0; j < piece.getLebar(); j++) {
                if (piece.bentuk[i][j] != '.') {
                    size++;
                }
            }
        }
        return size;
    }

    public static void printMatrixAsArray(char[][] matrix) {
        System.out.println("{");
        for (int i = 0; i < matrix.length; i++) {
            System.out.print("    {");
            for (int j = 0; j < matrix[i].length; j++) {
                System.out.print("'" + matrix[i][j] + "'");
                if (j < matrix[i].length - 1) {
                    System.out.print(", ");
                }
            }
            System.out.print("}");
            if (i < matrix.length - 1) {
                System.out.print(",");
            }
            System.out.println();
        }
        System.out.println("}");
    }
```

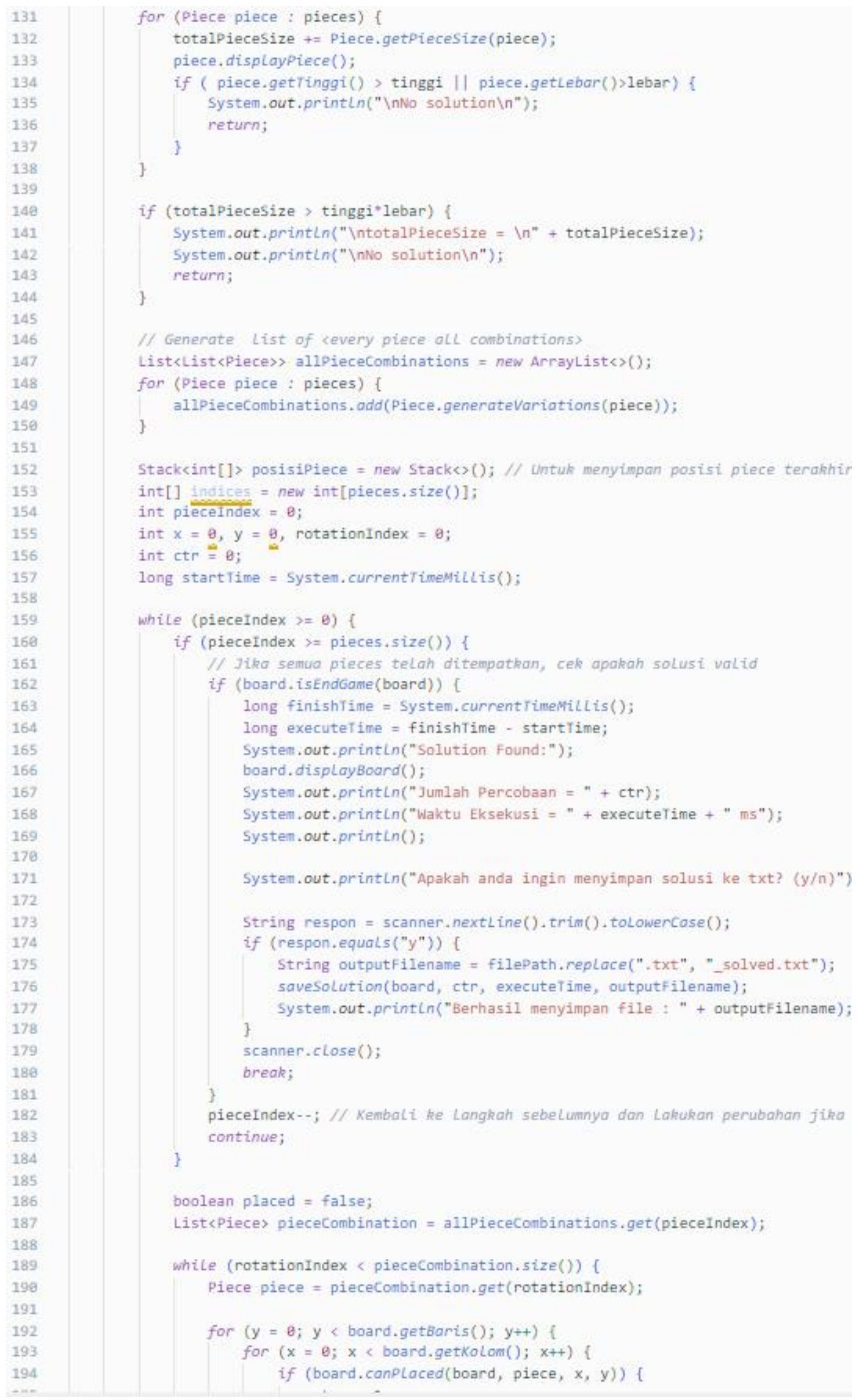
3.3 Main.java



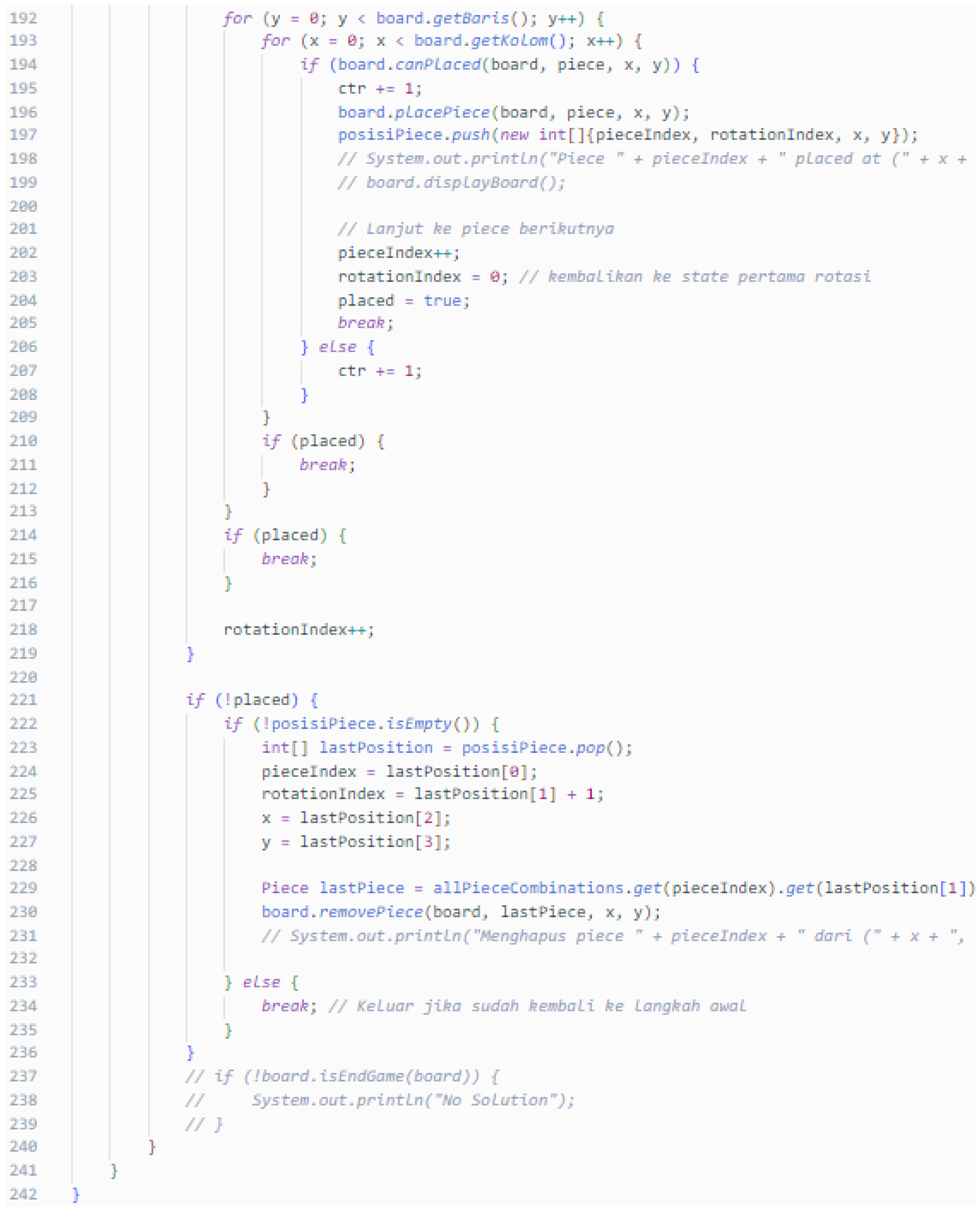
```java
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
import java.util.Stack;

public class Main {

    /* Baca Baris Panjang Board x Jumlah Pieces */
    public static int[] bacaLine (String barisInfo) {
        /* ex. barisInfo : "4 5 6" */
        Scanner scanner = new Scanner(barisInfo);
        int panjang = scanner.nextInt();
        int lebar = scanner.nextInt();
        int jumlahPiece = scanner.nextInt();
        scanner.close();
        int[] retval = {panjang, lebar, jumlahPiece};
        return retval;
    }

    // Konversi string di file menjadi Piece
    public static char[][] convertToPiece(List<String> lines) {
        int maxWidth = 0;

        // Menentukan lebar maksimum
        for (String line : lines) {
            maxWidth = Math.max(maxWidth, line.length());
        }

        int height = lines.size();
        char[][] pieceArray = new char[height][maxWidth];

        // Mengisi array dengan karakter yang mempertahankan spasi awal
        for (int i = 0; i < height; i++) {
            String line = lines.get(i);
            for (int j = 0; j < maxWidth; j++) {
                if (j < line.length()) {
                    char currentChar = line.charAt(j);
                // Ubah spasi menjadi titik
                    pieceArray[i][j] = (currentChar == ' ') ? '.' : currentChar;
                } else {
                    pieceArray[i][j] = '.'; // Isi dengan spasi jika kosong
                }
            }
        }

        return pieceArray;
    }

    public static void saveSolution(Board board, int ctr, long executeTime, String filename) {
        StringBuilder sb = new StringBuilder();
        sb.append("Solusi ditemukan:\n");

        for (int i = 0; i < board.getBaris(); i++) {
            for (int j = 0; j < board.getKolom(); j++) {
                sb.append(board.getUkuran()[i][j]);
            }
            sb.append("\n");
        }
        sb.append("\n");
        sb.append("Jumlah Percobaan = ").append(ctr).append("\n");
        sb.append("Waktu Eksekusi = ").append(executeTime).append(" ms\n");
```



```java
        saveToFile(filename, sb.toString());
    }

    private static void saveToFile(String filename, String content) {
        try (BufferedWriter writer = new BufferedWriter(new FileWriter(filename))) {
            writer.write(content);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Masukkan file path : ");
        String filePath = scanner.nextLine();
        List<Piece> pieces = new ArrayList<>();
        int tinggi = 0, lebar = 0;

        try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
            int[] boardInfo = bacaLine(br.readLine());
            tinggi = boardInfo[0];
            lebar = boardInfo[1];

            String line;
            List<String> currentcalonPiece = new ArrayList<>();
            char currentChar = '\0';

            br.readLine();

            while ((line = br.readLine()) != null) {
                line = line.replaceAll("\t", "    ");
                if (line.trim().isEmpty()) continue;

                char firstChar = line.trim().charAt(0);
                if (currentChar == '\0' || firstChar == currentChar) {
                    currentcalonPiece.add(line);
                    currentChar = firstChar;
                } else {
                    pieces.add(new Piece(convertToPiece(currentcalonPiece)));
                    currentcalonPiece.clear();
                    currentcalonPiece.add(line);
                    currentChar = firstChar;
                }
            }
            if (!currentcalonPiece.isEmpty()) {
                pieces.add(new Piece(convertToPiece(currentcalonPiece)));
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

        Board board = new Board(lebar, tinggi);
        // System.out.println("Ukuran Board: " + tinggi + " x " + lebar);
        // System.out.println("Jumlah Pieces: " + pieces.size());

        // for (int i = 0; i < pieces.size(); i++) {
        //     System.out.println("Piece " + (i + 1) + ":");
        //     pieces.get(i).displayPiece();
        // }

        // is Puzzle Solvable?
        int totalPieceSize = 0;
        System.out.println("tinggi & lebar = " + tinggi + lebar);
        for (Piece piece : pieces) {
```

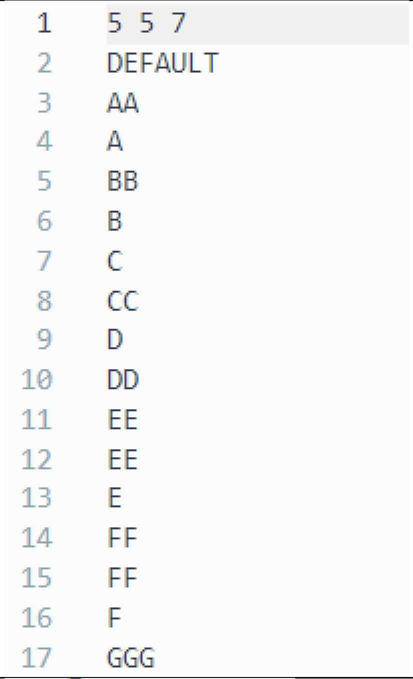
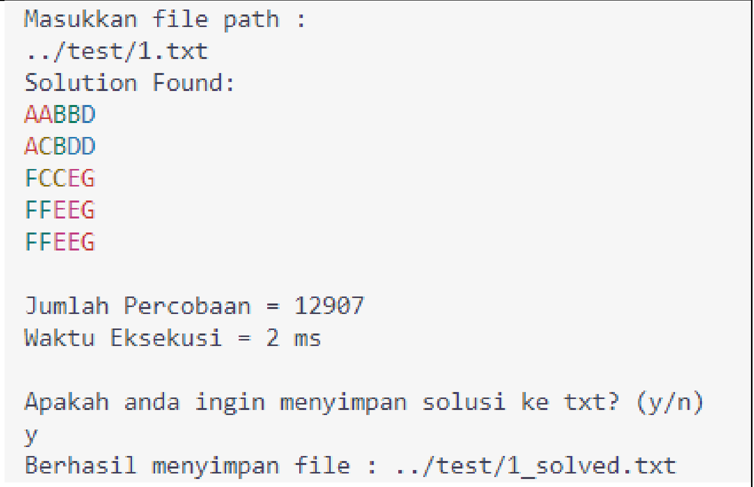
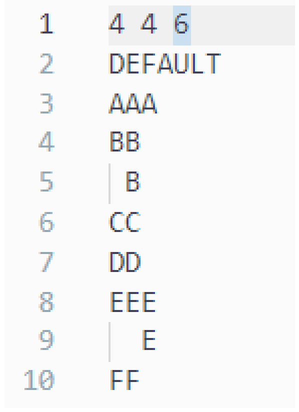
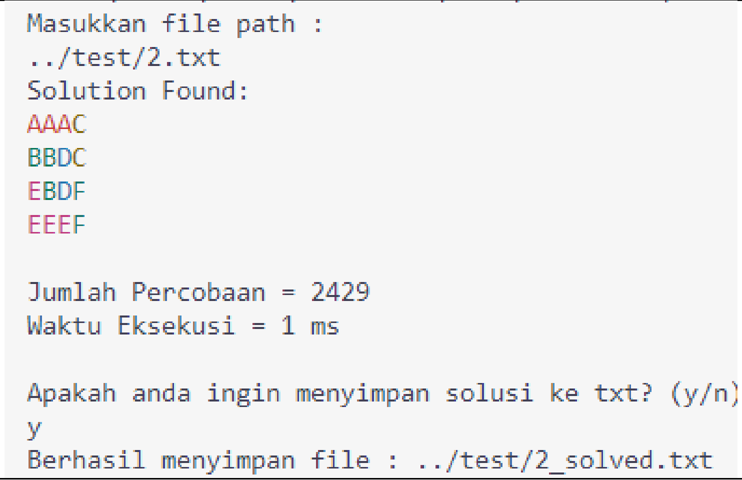


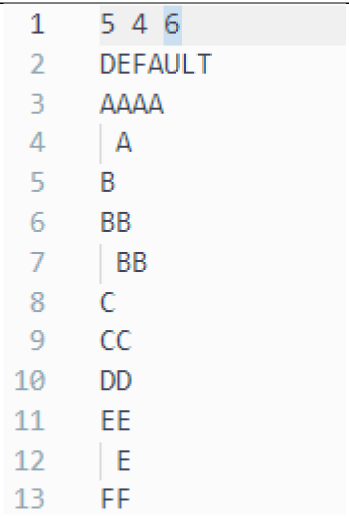
```java
        for (Piece piece : pieces) {
            totalPieceSize += Piece.getPieceSize(piece);
            piece.displayPiece();
            if ( piece.getTinggi() > tinggi || piece.getLebar()>lebar) {
                System.out.println("\nNo solution\n");
                return;
            }
        }

        if (totalPieceSize > tinggi*lebar) {
            System.out.println("\ntotalPieceSize = \n" + totalPieceSize);
            System.out.println("\nNo solution\n");
            return;
        }

        // Generate  List of <every piece all combinations>
        List<List<Piece>> allPieceCombinations = new ArrayList<>();
        for (Piece piece : pieces) {
            allPieceCombinations.add(Piece.generateVariations(piece));
        }

        Stack<int[]> posisiPiece = new Stack<>(); // Untuk menyimpan posisi piece terakhir
        int[] indices = new int[pieces.size()];
        int pieceIndex = 0;
        int x = 0, y = 0, rotationIndex = 0;
        int ctr = 0;
        long startTime = System.currentTimeMillis();

        while (pieceIndex >= 0) {
            if (pieceIndex >= pieces.size()) {
                // Jika semua pieces telah ditempatkan, cek apakah solusi valid
                if (board.isEndGame(board)) {
                    long finishTime = System.currentTimeMillis();
                    long executeTime = finishTime - startTime;
                    System.out.println("Solution Found:");
                    board.displayBoard();
                    System.out.println("Jumlah Percobaan = " + ctr);
                    System.out.println("Waktu Eksekusi = " + executeTime + " ms");
                    System.out.println();

                    System.out.println("Apakah anda ingin menyimpan solusi ke txt? (y/n)")

                    String respon = scanner.nextLine().trim().toLowerCase();
                    if (respon.equals("y")) {
                        String outputFilename = filePath.replace(".txt", "_solved.txt");
                        saveSolution(board, ctr, executeTime, outputFilename);
                        System.out.println("Berhasil menyimpan file : " + outputFilename);
                    }
                    scanner.close();
                    break;
                }
                pieceIndex--; // Kembali ke Langkah sebelumnya dan Lakukan perubahan jika
                continue;
            }

            boolean placed = false;
            List<Piece> pieceCombination = allPieceCombinations.get(pieceIndex);

            while (rotationIndex < pieceCombination.size()) {
                Piece piece = pieceCombination.get(rotationIndex);

                for (y = 0; y < board.getBaris(); y++) {
                    for (x = 0; x < board.getKolom(); x++) {
                        if (board.canPlaced(board, piece, x, y)) {
```



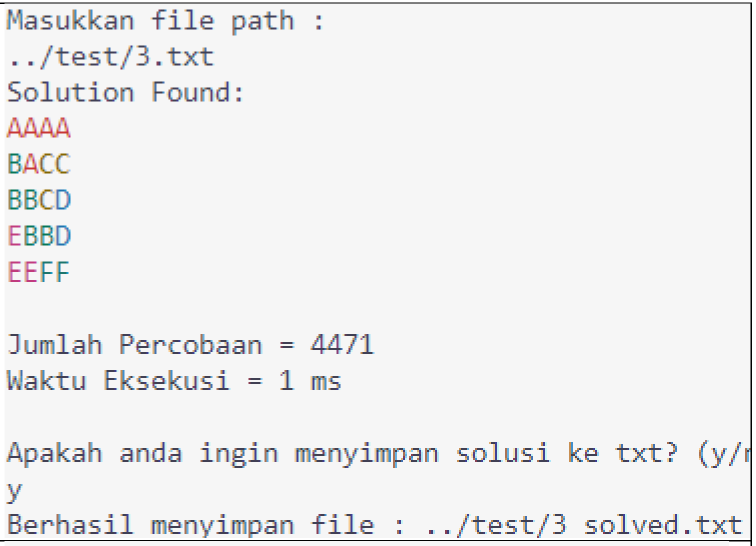
```java
                for (y = 0; y < board.getBaris(); y++) {
                    for (x = 0; x < board.getKolom(); x++) {
                        if (board.canPlaced(board, piece, x, y)) {
                            ctr += 1;
                            board.placePiece(board, piece, x, y);
                            posisiPiece.push(new int[]{pieceIndex, rotationIndex, x, y});
                            // System.out.println("Piece " + pieceIndex + " placed at (" + x +
                            // board.displayBoard();

                            // Lanjut ke piece berikutnya
                            pieceIndex++;
                            rotationIndex = 0; // kembalikan ke state pertama rotasi
                            placed = true;
                            break;
                        } else {
                            ctr += 1;
                        }
                    }
                    if (placed) {
                        break;
                    }
                }
                if (placed) {
                    break;
                }

                rotationIndex++;
            }

            if (!placed) {
                if (!posisiPiece.isEmpty()) {
                    int[] lastPosition = posisiPiece.pop();
                    pieceIndex = lastPosition[0];
                    rotationIndex = lastPosition[1] + 1;
                    x = lastPosition[2];
                    y = lastPosition[3];

                    Piece lastPiece = allPieceCombinations.get(pieceIndex).get(lastPosition[1])
                    board.removePiece(board, lastPiece, x, y);
                    // System.out.println("Menghapus piece " + pieceIndex + " dari (" + x + ",

                } else {
                    break; // Keluar jika sudah kembali ke langkah awal
                }
            }
            // if (!board.isEndGame(board)) {
            //     System.out.println("No Solution");
            // }
        }
    }
}
```

BAB 4: Test Case

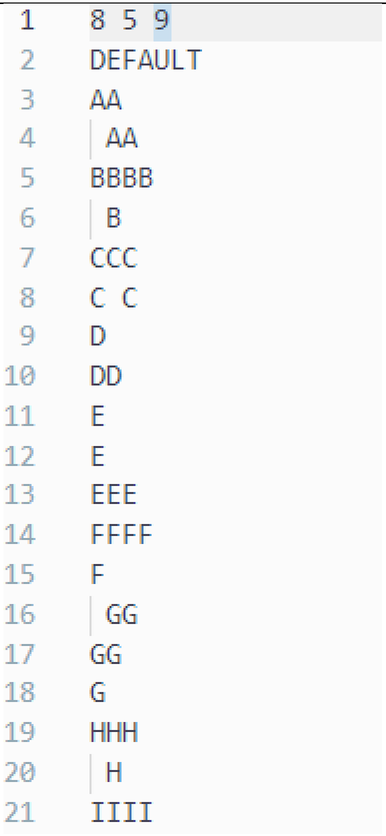
Test Case	Solusi
 5 5 7 DEFAULT AA A BB B C CC D DD EE EE E FF FF F GGG	 Masukkan file path : ../test/1.txt Solution Found: AABBD ACBDD FCCEG FFEEG FFEEG Jumlah Percobaan = 12907 Waktu Eksekusi = 2 ms Apakah anda ingin menyimpan solusi ke txt? (y/n) y Berhasil menyimpan file : ../test/1_solved.txt
 4 4 6 DEFAULT AAA BB B CC DD EEE E FF	 Masukkan file path : ../test/2.txt Solution Found: AAAC BBDC EBDF EEEF Jumlah Percobaan = 2429 Waktu Eksekusi = 1 ms Apakah anda ingin menyimpan solusi ke txt? (y/n) y Berhasil menyimpan file : ../test/2_solved.txt



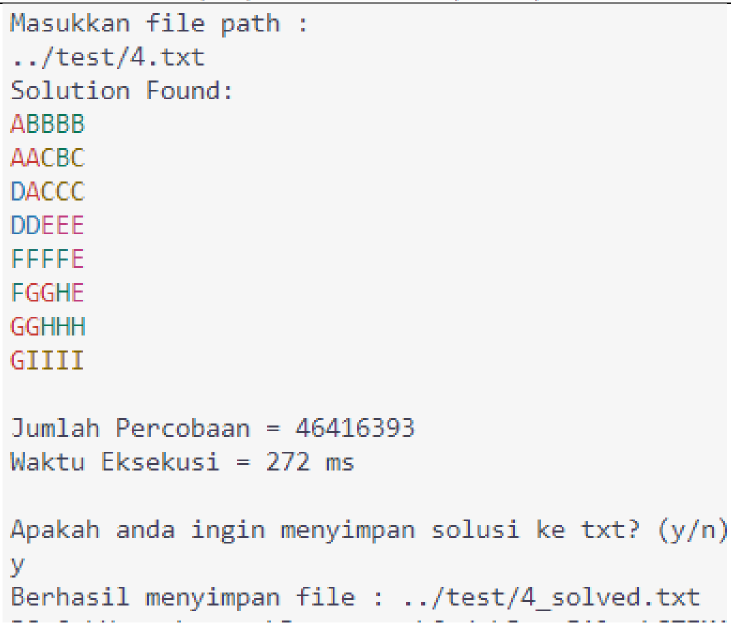
```
5 4 6
DEFAULT
AAAA
  A
B
BB
  BB
C
CC
DD
EE
  E
FF
```



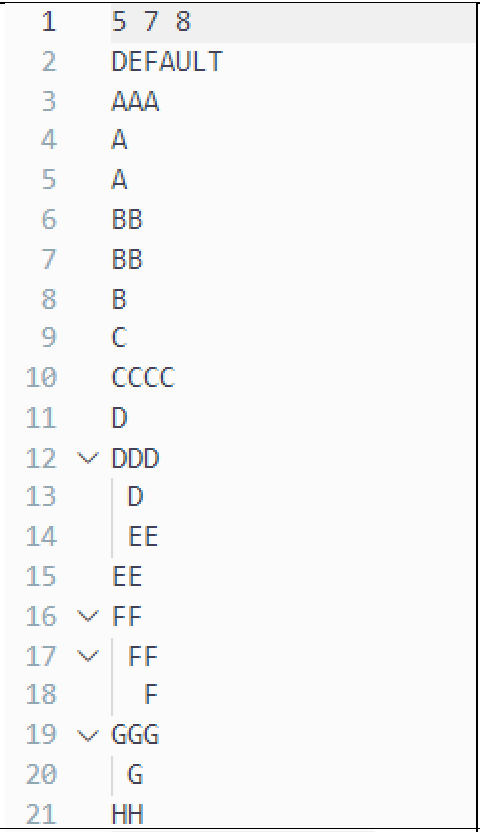
```
Masukkan file path :
../test/3.txt
Solution Found:
AAAA
BACC
BBCD
EBBD
EEFF

Jumlah Percobaan = 4471
Waktu Eksekusi = 1 ms

Apakah anda ingin menyimpan solusi ke txt? (y/r
y
Berhasil menyimpan file : ../test/3 solved.txt
```



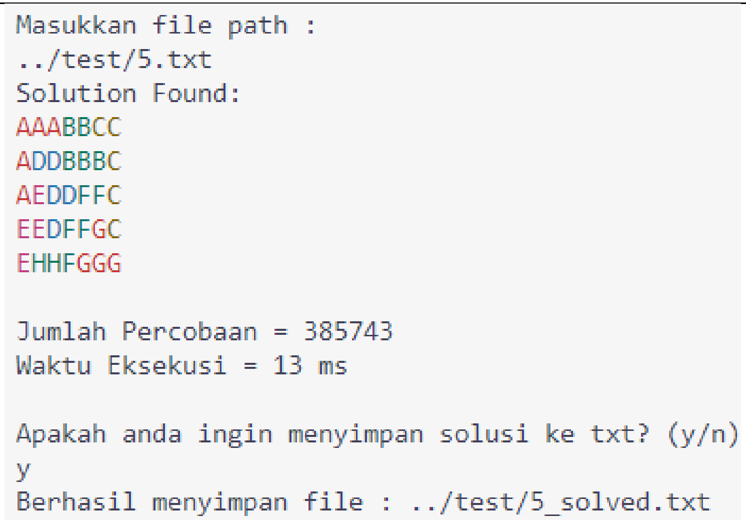
```
8 5 9
DEFAULT
AA
  AA
BBBB
  B
CCC
C C
D
DD
E
E
EEE
FFFF
F
  GG
GG
G
HHH
  H
IIII
```



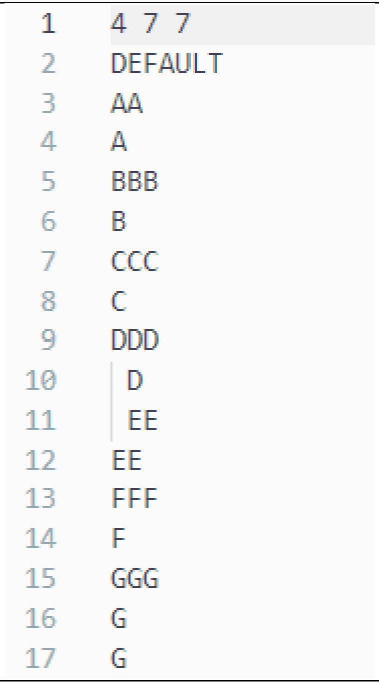
```
Masukkan file path :
../test/4.txt
Solution Found:
ABBBB
AACBC
DACCC
DDEEE
FFFFE
FGGHE
GGHHH
GIIII

Jumlah Percobaan = 46416393
Waktu Eksekusi = 272 ms

Apakah anda ingin menyimpan solusi ke txt? (y/n)
y
Berhasil menyimpan file : ../test/4_solved.txt
```



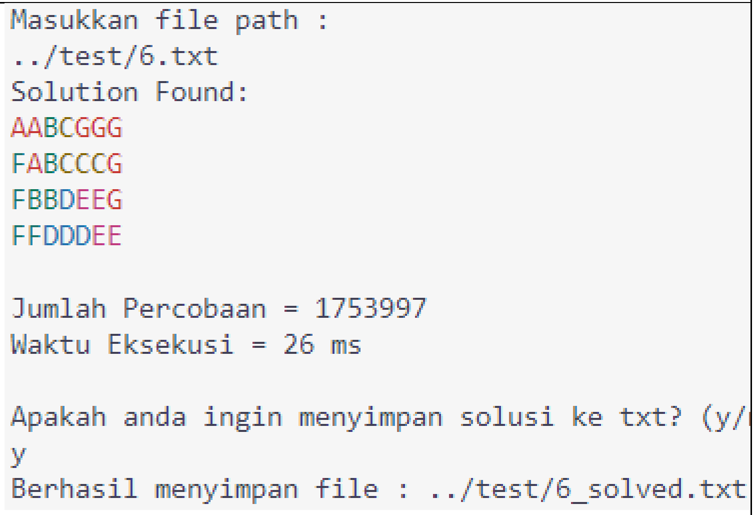
```
5 7 8
DEFAULT
AAA
A
A
BB
BB
B
C
CCCC
D
DDD
 D
 EE
EE
FF
 FF
  F
GGG
 G
HH
```



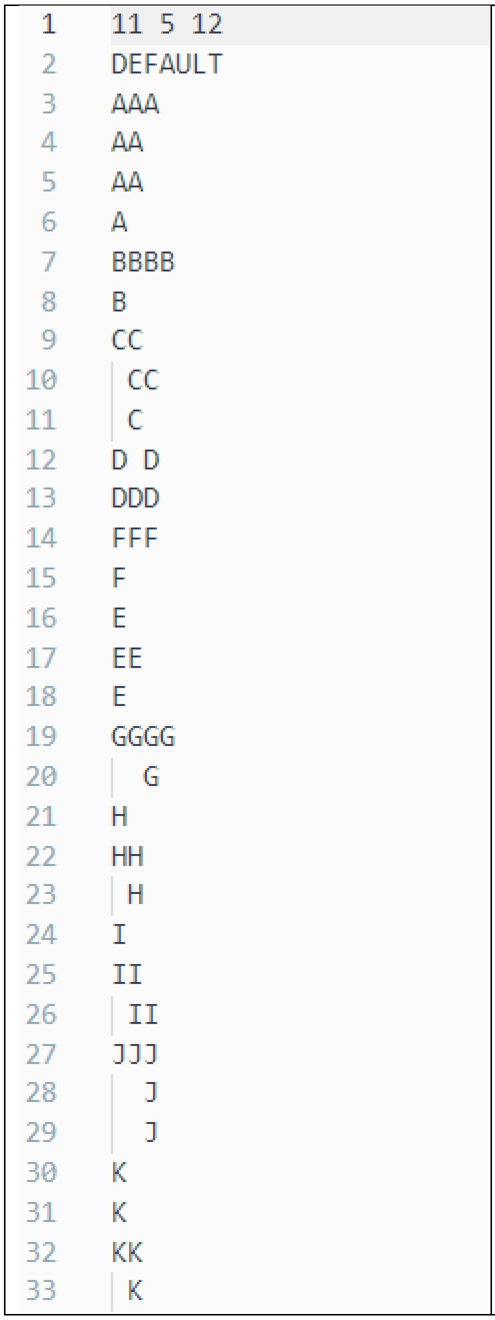
```
Masukkan file path :
../test/5.txt
Solution Found:
AAABBCC
ADDBBBC
AEDDFFC
EEDFFGC
EHHFGGG

Jumlah Percobaan = 385743
Waktu Eksekusi = 13 ms

Apakah anda ingin menyimpan solusi ke txt? (y/n)
y
Berhasil menyimpan file : ../test/5_solved.txt
```



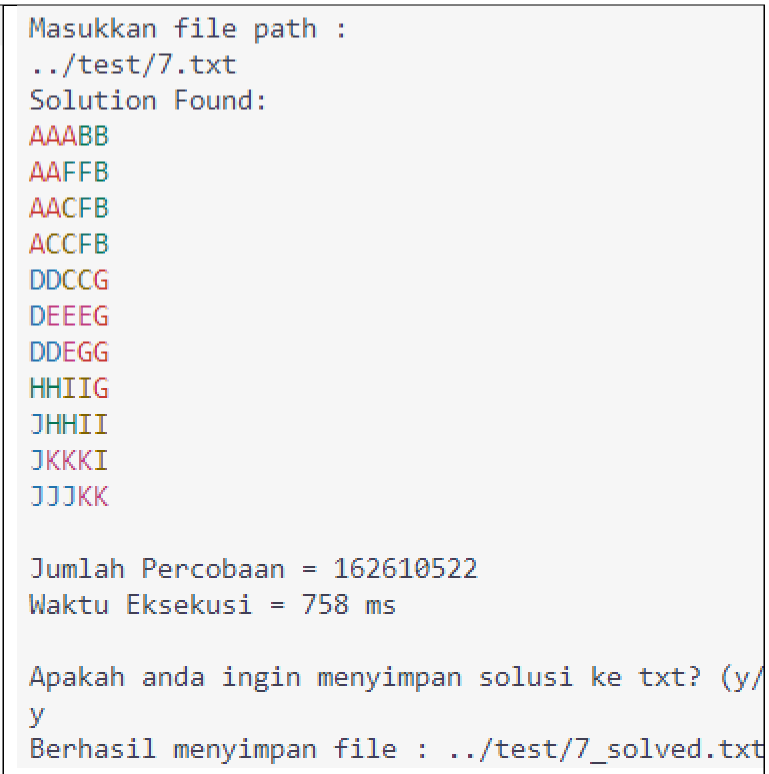
```
4 7 7
DEFAULT
AA
A
BBB
B
CCC
C
DDD
 D
 EE
EE
FFF
F
GGG
G
G
```



```
Masukkan file path :
../test/6.txt
Solution Found:
AABCGGG
FABCCCG
FBBDEEG
FFDDDEE

Jumlah Percobaan = 1753997
Waktu Eksekusi = 26 ms

Apakah anda ingin menyimpan solusi ke txt? (y/
y
Berhasil menyimpan file : ../test/6_solved.txt
```



```
11 5 12
DEFAULT
AAA
AA
AA
A
BBBB
B
CC
 CC
 C
D D
DDD
FFF
F
E
EE
E
GGGG
  G
H
HH
 H
I
II
 II
JJJ
  J
  J
K
K
KK
 K
```



```
Masukkan file path :
../test/7.txt
Solution Found:
AAABB
AAFFB
AACFB
ACCFB
DDCCG
DEEEG
DDEGG
HHIIG
JHHII
JKKKI
JJJKK

Jumlah Percobaan = 162610522
Waktu Eksekusi = 758 ms

Apakah anda ingin menyimpan solusi ke txt? (y/
y
Berhasil menyimpan file : ../test/7_solved.txt
```

Lampiran

Repository Github :

https://github.com/salmaanhaniif/Tucil1_13523056

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	√	