LAPORAN TUGAS KECIL 1
STRATEGI ALGORITMA

Disusun Oleh:

13523034 Rafizan Muhammad Syawalazmi

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2025

A. ALGORITMA BRUTE FORCE

0. Pilih piece pertama dan tempatkan pada koordinat (0, 0) atau di paling atas kiri board
1. Cek apakah piece tersebut valid untuk ditempatkan pada koordinatnya. Hal-hal yang dapat membuat pengecekan invalid adalah antara terdapat piece lain yang menempatkan tempat tersebut atau tempat tersebut di luar area permainan.
2. Jika bisa, tempatkan piece tersebut, pilih piece selanjutnya dalam list, dan tempatkan pada koordinat (0, 0) atau di paling atas kiri board.
3. Jika tidak bisa ditempatkan, lanjutkan ke piece type selanjutnya. Piece type adalah semua 8 orientasi yang bisa dilakukan untuk setiap piece dengan cara melakukan rotasi dan mencerminkan piecenya.
4. Jika semua orientasi di tempat yang sama sudah dilakukan, geser penempatan piece satu unit ke kanan.
5. Jika Piece sudah tidak bisa digeser ke kanan (sudah melewati board) geser piece satu unit ke bawah dan pada posisi paling kiri board.
6. Jika piece sudah tidak bisa digeser ke bawah, 'backtrack' ke piece sebelumnya dan lakukan langkah ketiga.
7. Ulangi langkah 1-6 hingga semua piece sudah ditempatkan secara valid atau sudah dicek semua kemungkinan penempatan piece.
8. Cek apakah kondisi board sudah terisi penuh (ada solusi) atau tidak (tidak ada solusi).

B. SOURCE CODE

Main.java

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

public class Main {
   public static void main(String[] args) throws IOException {
      int N;
      int M;
      int P;
      String S;
      Piece[][] pieces;
      int[][] positions;
      int[] types;
      int cases;

      //#region Input
      Scanner scanner = new Scanner(System.in);
      System.out.print("Masukkan nama file input: ");

      BufferedReader reader = new BufferedReader(new
FileReader(scanner.nextLine()));

      String[] inputNums = reader.readLine().trim().split("\\s+");

      N = Integer.parseInt(inputNums[0]);
```

```
M = Integer.parseInt(inputNums[1]);
P = Integer.parseInt(inputNums[2]);

S = reader.readLine();

Board board = new Board(N, M, S);

if (S.equals("CUSTOM")) {
   for (int i = 0; i < N; i++) {
      char[] mapRow = reader.readLine().toCharArray();
      for (int j = 0; j < M; j++) {
         if (mapRow[j] == 'X') {
            board.map[i][j] = ' ';
         } else {
            board.map[i][j] = '.';
         }
      }
   }
}

pieces = new Piece[P][8];

String line = reader.readLine();
char[] row = line.toCharArray();
int symbolIdx = 0;
while (row[symbolIdx] == ' ') {
   symbolIdx++;
}
char currentSymbol = row[symbolIdx];
int currentPiece = 0;
pieces[currentPiece][0] = new Piece(0, 0);
pieces[currentPiece][0].addShapeRow(row);

while ((line = reader.readLine()) != null) {
   row = line.toCharArray();
   if (CustomUtils.arrayContains(row, currentSymbol)) {
      pieces[currentPiece][0].addShapeRow(row);
   } else {
      symbolIdx = 0;
      while (row[symbolIdx] == ' ') {
         symbolIdx++;
      }
      currentSymbol = row[symbolIdx];
      currentPiece++;
      pieces[currentPiece][0] = new Piece(0, 0);
      pieces[currentPiece][0].addShapeRow(row);
   }
}

reader.close();
```

```
        //#endregion

        //#region Test Input
        // System.out.println(N);
        // System.out.println(M);
        // System.out.println(P);
        // System.out.println(S);

        // for (int k = 0; k < 8; k++) {
        //     for (i = 0; i < pieces[k][0].N; i++) {
        //         for (int j = 0; j < pieces[k][0].M; j++) {
        //             System.out.print(pieces[k][0].shape[i][j]);
        //         }
        //         System.out.println();
        //     }
        // }
        //#endregion

        //#region Pre-Process
        for (int i = 0; i < P; i++) {
            for (int j = 1; j < 8; j++) {
                if (j % 4 == 0) {
                    pieces[i][j] = pieces[i][j - 1].mirror();
                } else {
                    pieces[i][j] = pieces[i][j - 1].rotate();
                }
            }
        }

        positions = new int[P][2];
        types = new int[P];
        for (int i = 0; i < P; i++) {
            positions[i][0] = 0;
            positions[i][1] = 0;
            types[i] = 0;
        }

        cases = 0;
        //#endregion

        long start = System.currentTimeMillis();
        //#region Solve
        int piecesPlaced = 0;
        while (piecesPlaced < P) {
            cases++;
            // check if the current piece will fit at the current position
            if (board.isPieceFit(pieces[piecesPlaced][types[piecesPlaced]],
    positions[piecesPlaced][0], positions[piecesPlaced][1])) {
                // place the current piece
```

```
                board.placePiece(pieces[piecesPlaced][types[piecesPlaced]],
positions[piecesPlaced][0], positions[piecesPlaced][1]);
                piecesPlaced++;
            // else rotate, mirror, or slide the current piece
            } else {
                // next iteration
                types[piecesPlaced] = (types[piecesPlaced] + 1) % 8;
                if (types[piecesPlaced] == 0) {
                    positions[piecesPlaced][0] = (positions[piecesPlaced][0] + 1) %
board.M;
                    if (positions[piecesPlaced][0] == 0) {
                        positions[piecesPlaced][1] = (positions[piecesPlaced][1] +
1);
                    }
                }

                while (positions[piecesPlaced][1] == board.N) {
                    // go back to the previous piece
                    positions[piecesPlaced][1] = 0;
                    piecesPlaced--;

                    // check if it went past the first piece
                    if (piecesPlaced == -1) {
                        break;
                    }

                    board.takePiece(pieces[piecesPlaced][types[piecesPlaced]],
positions[piecesPlaced][0], positions[piecesPlaced][1]);

                    // next iteration
                    types[piecesPlaced] = (types[piecesPlaced] + 1) % 8;
                    if (types[piecesPlaced] == 0) {
                        positions[piecesPlaced][0] = (positions[piecesPlaced][0] + 1)
% board.M;
                        if (positions[piecesPlaced][0] == 0) {
                            positions[piecesPlaced][1] = (positions[piecesPlaced][1] +
1);
                        }
                    }
                }

                if (piecesPlaced == -1) {
                    break;
                }
            }
        }
        //#endregion
        long end = System.currentTimeMillis();

        if (board.isBoardFilled()) {
```

```
            System.out.println();
            board.printBoard();
            System.out.println("\nWaktu pencarian: " + (end - start) + " ms");
            System.out.println("Banyak kasus yang ditinjau: " + cases + "");

            System.out.print("Apakah anda ingin menyimpan solusi dalam .txt
        file? (ya/tidak) ");
            if (scanner.nextLine().equals("ya")) {
               BufferedWriter writer = new BufferedWriter(new
        FileWriter("output.txt"));
               board.writeBoard(writer);
               writer.close();
            }

            System.out.print("Apakah anda ingin menyimpan solusi dalam .png
        file? (ya/tidak) ");
            if (scanner.nextLine().equals("ya")) {
               board.drawBoard("output.png");
            }
         } else {
            System.out.println("\nTidak ada solusi yang ditemukan :(");
            System.out.println("Waktu pencarian: " + (end - start) + " ms");
            System.out.println("Banyak kasus yang ditinjau: " + cases);
         }
      }
   }
```

Board.java

```
import java.awt.Color;
import java.awt.Graphics2D;
import java.awt.image.BufferedImage;
import java.io.BufferedWriter;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;

public class Board {
   int N;
   int M;
   String S;
   char[][] map;

   public static final String[] ANSI_COLORS = {"\u001b[38;2;0;0;255m",
"\u001b[38;2;0;255;0m", "\u001b[38;2;255;128;0m",
"\u001b[38;2;166;66;200m", "\u001b[38;2;101;55;0m",
```

```
                                    "\u001b[38;2;128;128;128m",
"\u001b[38;2;255;0;0m", "\u001b[38;2;128;0;0m",
"\u001b[38;2;143;0;255m", "\u001b[38;2;205;127;50m",
                                    "\u001b[38;2;255;255;204m",
"\u001b[38;2;254;220;86m", "\u001b[38;2;0;0;128m",
"\u001b[38;2;248;131;121m", "\u001b[38;2;224;176;255m",
                                    "\u001b[38;2;255;204;153m",
"\u001b[38;2;255;215;0m", "\u001b[38;2;0;255;255m",
"\u001b[38;2;0;127;255m", "\u001b[38;2;0;128;128m",
                                    "\u001b[38;2;112;130;56m",
"\u001b[38;2;245;245;220m","\u001b[38;2;54;69;79m",
"\u001b[38;2;0;0;255m", "\u001b[38;2;36;122;253m",
                                    "\u001b[38;2;255;209;220m"};

   public static final Color[] COLORS = {new Color(0, 0, 255), new
Color(0, 255, 0), new Color(255, 128, 0), new Color(166,66,200), new
Color(101,55,0),
                                    new Color(128, 128, 128), new Color(255, 0, 0),
new Color(128,0,0), new Color(143, 0, 255), new Color(205, 127, 50),
                                    new Color(255, 255, 204), new Color(254, 220,
86), new Color(0, 0, 128), new Color(248, 131, 121), new Color(224, 176,
255),
                                    new Color(255, 204, 153), new Color(255, 215,
0), new Color(0, 255, 255), new Color(0, 127, 255), new Color(0, 128,
128),
                                    new Color(112, 130, 56), new Color(245, 245,
220), new Color(54, 69, 79), new Color(0, 0, 255), new Color(36, 112,
253),
                                    new Color(255, 209, 220)};

   public static final String ANSI_RESET = "\u001b[0m";

   Board(int N, int M, String S) {
      this.N = N;
      this.M = M;
      if (S.equals("DEFAULT") || S.equals("CUSTOM")) {
         this.map = new char[N][M];
         for (int i = 0; i < N; i++) {
            for (int j = 0; j < M; j++) {
               this.map[i][j] = ' ';
            }
         }
      }
   }

   public boolean isPieceFit(Piece piece, int x, int y) {
      if (x + piece.M > this.M || y + piece.N > this.N) {
         return false;
      }
```

```
            for (int i = 0; i < piece.N; i++) {
                for (int j = 0; j < piece.M; j++) {
                    if (piece.shape[i][j] != ' ' && this.map[y + i][x + j] != ' ') {
                        return false;
                    }
                }
            }
            return true;
        }

        public void placePiece(Piece piece, int x, int y) {
            for (int i = 0; i < piece.N; i++) {
                for (int j = 0; j < piece.M; j++) {
                    if (piece.shape[i][j] != ' ') {
                        this.map[y + i][x + j] = piece.shape[i][j];
                    }
                }
            }
        }

        public void takePiece(Piece piece, int x, int y) {
            for (int i = 0; i < piece.N; i++) {
                for (int j = 0; j < piece.M; j++) {
                    if (piece.shape[i][j] != ' ') {
                        this.map[y + i][x + j] = ' ';
                    }
                }
            }
        }

        public boolean isBoardFilled() {
            for (int i = 0; i < this.N; i++) {
                for (int j = 0; j < this.M; j++) {
                    if (this.map[i][j] == ' ') {
                        return false;
                    }
                }
            }
            return true;
        }

        public void printBoard() {
            for (int i = 0; i < this.N; i++) {
                for (int j = 0; j < this.M; j++) {
                    if (this.map[i][j] == '.') {
                        System.out.print(' ');
                    } else {
                        System.out.print(ANSI_COLORS[(int) this.map[i][j] - (int)
    'A'] + this.map[i][j] + ANSI_RESET);
                    }
```

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

    public void writeBoard(BufferedWriter writer) throws IOException {
        for (int i = 0; i < this.N; i++) {
            for (int j = 0; j < this.M; j++) {
                if (this.map[i][j] == '.') {
                    writer.write(' ');
                } else {
                    writer.write(this.map[i][j]);
                }
            }
            writer.write('\n');
        }
    }

    public void drawBoard(String output) throws IOException {
        int offset = 100;
        int radius = 50;
        int distance = 20;
        int connector = 10;

        BufferedImage image = new BufferedImage(2*offset +
2*radius*this.M + distance*(this.M-1), 2*offset + 2*radius*this.N +
distance*(this.N-1), BufferedImage.TYPE_INT_ARGB);
        Graphics2D graphics = image.createGraphics();
        for (int i = 0; i < this.N; i++) {
            for (int j = 0; j < this.M; j++) {
                if (this.map[i][j] != '.') {
                    graphics.setColor(COLORS[(int) this.map[i][j] - (int) 'A']);
                    graphics.fillOval(offset + (2 * radius + distance) * j, offset + (2
* radius + distance) * i, 2*radius, 2*radius);
                    if (i < this.N - 1 && this.map[i][j] == this.map[i+1][j]) {
                        graphics.fillRect(offset + (2 * radius + distance) * j + radius
- connector, offset + (2 * radius + distance) * i + radius, 2*connector,
2*radius + distance);
                    }
                    if (j < this.M - 1 && this.map[i][j] == this.map[i][j+1]) {
                        graphics.fillRect(offset + (2 * radius + distance) * j + radius,
offset + (2 * radius + distance) * i + radius - connector, 2*radius + distance,
2*connector);
                    }
                }
            }
        }
        graphics.dispose();
        ImageIO.write(image, "png", new File(output));
    }
```

```
    }
```

Piece.java

```
public class Piece {
    int N;
    int M;
    char[][] shape;
    int x;
    int y;

    Piece(int N, int M) {
        this.N = N;
        this.M = M;
        this.shape = new char[N][M];
    }

    public void addShapeRow(char[] row) {
        int newN = this.N + 1;
        int newM;

        if (this.M < row.length) {
            newM = row.length;
        } else {
            newM = this.M;
        }

        char[][] newShape = new char[newN][newM];
        for (int i = 0; i < newN; i++) {
            for (int j = 0; j < newM; j++) {
                newShape[i][j] = ' ';
            }
        }

        for (int i = 0; i < this.N; i++) {
            for (int j = 0; j < this.M; j++) {
                newShape[i][j] = this.shape[i][j];
            }
        }

        for (int i = 0; i < row.length; i++) {
            newShape[newN-1][i] = row[i];
        }

        this.N = newN;
        this.M = newM;
        this.shape = newShape;
    }

    public Piece mirror() {
        Piece mirrored = new Piece(this.N, this.M);
```

```
            for (int i = 0; i < this.N; i++) {
                for (int j = 0; j < this.M; j++) {
                    mirrored.shape[i][j] = this.shape[i][this.M-j-1];
                }
            }

            return mirrored;
        }

        public Piece rotate() {
            Piece rotated = new Piece(this.M, this.N);

            Piece mirrored = this.mirror();
            for (int i = 0; i < rotated.N; i++) {
                for (int j = 0; j < rotated.M; j++) {
                    rotated.shape[i][j] = mirrored.shape[j][i];
                }
            }

            return rotated;
        }

        public Piece transform3D() {
            return this;
        }

        public Piece rotate3D() {
            return this;
        }
    }
```

CustomUtils.java

```
    public class CustomUtils {
        public static boolean arrayContains(char[] array, char x) {
            for (char c: array) {
                if (c == x) {
                    return true;
                }
            }
            return false;
        }
    }
```

C. TESTING

1. Test 1

Input:

Tucil1_13523034 > bin > input1.txt

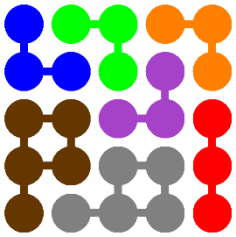
```
5 5 7
DEFAULT
A
AA
B
BB
C
CC
D
DD
EE
EE
E
FF
FF
F
GGG
```

Output:

```
PS C:\Users\rafis\Documents\Kuliah\Semester 4\STIMA\Tucil1_13523034\bin> java Main
Masukkan nama file input: input.txt

ABBCC
AABDC
EEDDG
EEFFG
EFFFG

Waktu pencarian: 10 ms
Banyak kasus yang ditinjau: 487427
Apakah anda ingin menyimpan solusi dalam .txt file? (ya/tidak) ya
Apakah anda ingin menyimpan solusi dalam .png file? (ya/tidak) ya
```

Tucil1_13523034 > test > output1.txt

```
ABBCC
AABDC
EEDDG
EEFFG
EFFFG

```



2. Test 2

Input:

Tucil1_13523034 > bin > input.txt

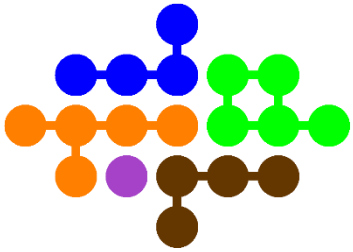
```
5 7 5
CUSTOM
...X...
.XXXXX.
XXXXXXX
.XXXXX.
...X...
A
AAA
BB
BBB
CCCC
C
D
EEE
E
```

Output:

```
PS C:\Users\rafis\Documents\Kuliah\Semester 4\STIMA\Tucil1_13523034\bin> java Main
Masukkan nama file input: input.txt

   A
 AAABB
CCCCBBB
 CDEEE
   E

Waktu pencarian: 3 ms
Banyak kasus yang ditinjau: 35596
Apakah anda ingin menyimpan solusi dalam .txt file? (ya/tidak) ya
Apakah anda ingin menyimpan solusi dalam .png file? (ya/tidak) ya
```

Tucil1_13523034 > bin > output.txt

```
   A
 AAABB
CCCCBBB
 CDEEE
   E

```



3. **Test 3**

Input:

Tucil1_13523034 > bin > input3.txt

```
6 5 26
CUSTOM
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
X....
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:

```
PS C:\Users\rafis\Documents\Kuliah\Semester 4\STIMA\Tucil1_13523034\bin> java Main
Masukkan nama file input: input3.txt

ABCDE
FGHIJ
KLMNO
PQRST
UVWXY
Z

Waktu pencarian: 1 ms
Banyak kasus yang ditinjau: 2626
Apakah anda ingin menyimpan solusi dalam .txt file? (ya/tidak) ya
Apakah anda ingin menyimpan solusi dalam .png file? (ya/tidak) ya
```

Tucil1_13523034 > bin > output.txt

```
1  ABCDE
2  FGHIJ
3  KLMNO
4  PQRST
5  UVWXY
6  Z
```

4. **Test 4**

Input:

Tucil1_13523034 > bin > input4.txt

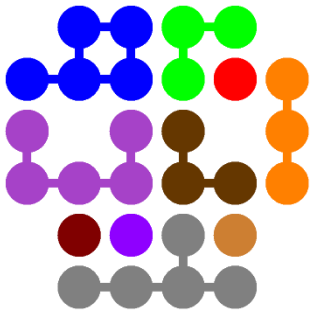
```
1  6 6 10
2  CUSTOM
3  .XXXX.
4  XXXXXX
5  X.XX.X
6  XXXXXX
7  .XXXX.
8  .XXXX.
9  AAA
10 AA
11 BB
12 B
13 CCC
14 D D
15 DDD
16 EE
17 E
18 F
19 F
20 FF
21 F
22 G
23 H
24 I
25 J
```

Output:

```
PS C:\Users\rafis\Documents\Kuliah\Semester 4\STIMA\Tucil1_13523034\bin> java Main
Masukkan nama file input: input4.txt

 AABB
AAABGC
D DE C
DDDEEC
 HIFJ
 FFFF

Waktu pencarian: 0 ms
Banyak kasus yang ditinjau: 1255
Apakah anda ingin menyimpan solusi dalam .txt file? (ya/tidak) ya
Apakah anda ingin menyimpan solusi dalam .png file? (ya/tidak) ya
```

Tucil1_13523034 > bin > output4.txt

```
 AABB
AAABGC
D DE C
DDDEEC
 HIFJ
 FFFF

```



5. **Test 5**

Input:

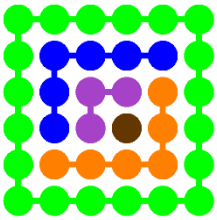
```
6 6 5
DEFAULT
BBBBBB
B    B
B    B
B    B
B    B
BBBBBB
AAAA
A
A
CCCC
C
C
DD
D
E
```

Output:

```
PS C:\Users\rafis\Documents\Kuliah\Semester 4\STIMA\Tucil1_13523034\bin> java Main
Masukkan nama file input: input5.txt

BBBBBB
BAAAAB
BADDCB
BADECB
BCCCCB
BBBBBB

Waktu pencarian: 0 ms
Banyak kasus yang ditinjau: 447
Apakah anda ingin menyimpan solusi dalam .txt file? (ya/tidak) ya
Apakah anda ingin menyimpan solusi dalam .png file? (ya/tidak) ya
```

Tucil1_13523034 > bin > output5.txt

```
1  BBBBBB
2  BAAAAB
3  BADDCB
4  BADECB
5  BCCCCB
6  BBBBBB
7
```



6. **Test 6**

   Input:

Tucil1_13523034 > bin > input6.txt

```
1   4 4 5
2   DEFAULT
3   AAA
4    A
5   BB
6   B
7   CCCC
8   C
9   DDD
10  EE
```

   Output:

```
PS C:\Users\rafis\Documents\Kuliah\Semester 4\STIMA\Tucil1_13523034\bin> java Main
Masukkan nama file input: input6.txt

Tidak ada solusi yang ditemukan :(
Waktu pencarian: 15 ms
Banyak kasus yang ditinjau: 657536
```

7. **Test 7**

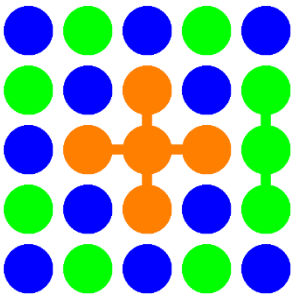
Input:

```
Tucil1_13523034 > bin > input7.txt
1   5 5 3
2   DEFAULT
3   A A A
4    A A
5   A
6    A A
7   A A A
8    B B
9   B   B
10      B
11  B   B
12   B B
13   C
14  CCC
15   C
```

Output:

```
PS C:\Users\rafis\Documents\Kuliah\Semester 4\STIMA\Tucil1_13523034\bin> java Main
Masukkan nama file input: input7.txt

ABABA
BACAB
ACCCB
BACAB
ABABA

Waktu pencarian: 0 ms
Banyak kasus yang ditinjau: 51
Apakah anda ingin menyimpan solusi dalam .txt file? (ya/tidak)
```

```
Tucil1_13523034 > bin > output7.txt
1   ABABA
2   BACAB
3   ACCCB
4   BACAB
5   ABABA
6
```



D. REPOSITORY

https://github.com/Rafizan46/Tucil1_13523034

E. LAMPIRAN

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	✓	