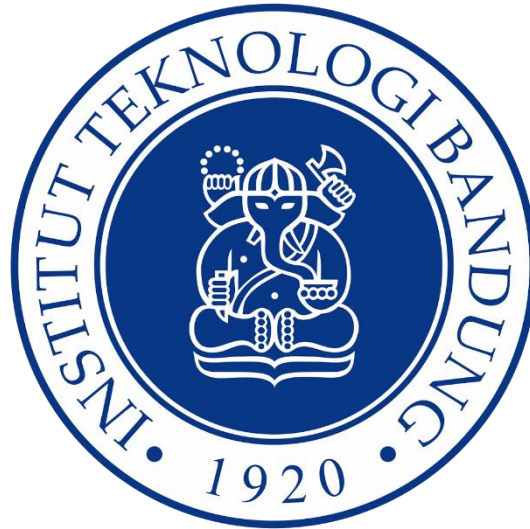


IF2211 – Strategi Algoritma

Tugas Kecil 1

Penyelesaian IQ Puzzler Pro dengan Algoritma Brute Force



Disusun Oleh:

13523001 Wardatul Khoiroh

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2024**

DAFTAR ISI

DAFTAR ISI	2
BAB I.....	3
BAB II	4
BAB III.....	11
LAMPIRAN	18

BAB I

DESKRIPSI MASALAH

1.1 Algoritma Brute Force

Algoritma Brute Force adalah metode eksplorasi yang mencoba semua kemungkinan solusi satu per satu hingga menemukan hasil yang sesuai. Dalam konteks penyelesaian IQ Puzzler Pro Solver, algoritma ini digunakan untuk menemukan solusi dengan mencoba setiap kemungkinan peletakan blok pada papan permainan. Pendekatan yang digunakan dalam implementasi ini adalah rekursif dengan teknik backtracking, yang memungkinkan algoritma untuk kembali ke langkah sebelumnya jika suatu konfigurasi blok tidak menghasilkan solusi yang valid. Teknik ini memastikan bahwa setiap kemungkinan penempatan blok dieksplorasi secara menyeluruh hingga ditemukan solusi yang tepat atau dinyatakan tidak ada solusi yang memungkinkan. Berikut adalah tahapan rinci dari algoritma brute force yang diterapkan dalam penyelesaian IQ Puzzler Pro:

- A. Ambil blok pertama dari daftar blok, lalu lakukan operasi transformasi seperti rotasi dan pencerminan.
- B. Dapatkan semua kemungkinan bentuk dari blok tersebut, kemudian coba setiap bentuk hingga seluruh kemungkinan telah diuji.
- C. Lakukan iterasi pada papan permainan, mulai dari sudut kiri atas hingga sudut kanan bawah, untuk menentukan posisi peletakan blok.
- D. Pada setiap posisi yang diuji, periksa apakah blok dapat ditempatkan dengan syarat:
 - Tidak melebihi batas ukuran papan.
 - Tidak bertumpuk dengan blok lain yang sudah ditempatkan.
- E. Jika blok dapat diletakkan pada posisi tertentu, tempatkan blok tersebut di papan permainan.
- F. Lanjutkan proses dengan mencoba meletakkan blok berikutnya. Jika semua blok sudah ditempatkan tetapi papan belum penuh, hapus blok terakhir yang dipasang dan coba alternatif lain. Hal yang sama berlaku jika suatu blok tidak dapat diletakkan dalam semua kemungkinan bentuknya pada posisi tertentu.
- G. Jika papan sudah penuh, maka solusi telah ditemukan. Namun, jika seluruh kemungkinan penempatan blok telah dicoba tetapi papan masih belum penuh, maka tidak ada solusi yang valid.

BAB II

IMPLEMENTASI ALGORITMA DALAM BAHASA JAVA

2.1 File Blok.java

Pada file ini berisi program untuk melakukan beberapa operasi pada setiap blok/piece. File ini memiliki kelas bernama Blok yang memiliki empat method dan dua atribut.

Tabel 2.1 Tabel Atribut/Method Blok.java

Nama Atribut/Method	Deskripsi
shape	Atribut dengan tipe list of char dua dimensi untuk menyimpan semua kemungkinan bentuk blok setelah dioperasikan.
Blok	Konstruktor yang menerima bentuk asli suatu blok dan memanggil metode perubahanBentuk() sehingga dihasilkan semua kemungkinan orientasi bloknya.
getShape()	Metode untuk mengembalikan daftar shape yang berisi semua variasi bentuk blok.
perubahanBentuk()	Metode untuk menghasilkan semua kemungkinan bentuk blok dengan rotasi sebanyak 90°, 180°, 2° dan pencerminan.
rotasi()	Metode untuk melakukan rotasi blok sebesar 90° searah jarum jam. <pre>public char[][] rotasi(char[][] shape) { int baris = shape.length; int kolom = shape[0].length; char[][] hasil_rotasi = new char[kolom][baris]; for (int i = 0; i < baris; i++) { for (int j = 0; j < kolom; j++) { hasil_rotasi[j][baris - 1 - i] = shape[i][j]; } } return hasil_rotasi; }</pre>
pencerminan()	Metode untuk melakukan refleksi pada blok. <pre>public char[][] pencerminan(char[][] shape) { int baris = shape.length; int kolom = shape[0].length;</pre>

	<pre> char[][] hasil_refleksi = new char[baris][kolom]; for (int i = 0; i < baris; i++) { for (int j = 0; j < kolom; j++) { hasil_refleksi[i][kolom-1-j] = shape[i][j]; } } return hasil_refleksi; } </pre>
--	---

2.2 File Main.java

Pada file ini berisi program utama untuk menjalankan IQ Puzzler Pro Solver. File ini memiliki kelas bernama Main yang memiliki dua method.

Tabel 2.2 Tabel Atribut/Method Main.java

Nama Atribut/Method	Deskripsi
main()	Metode untuk menampilkan welcoming text dan memilih opsi solve puzzle atau exit.
solvePuzzle()	Metode untuk memanggil semua fungsi yang digunakan untuk menyelesaikan IQ Puzzler Pro. Dimulai dengan meminta input nama file, membaca data puzzle dari input file, melakukan inisialisasi papan permainan, membentuk daftar blok, mencatat waktu eksekusi program, mencari solusi menggunakan brute force, dan menyimpan solusi ke dalam file.

2.3 File Papan.java

Pada file ini berisi program untuk melakukan beberapa operasi pada papan/board. File ini memiliki kelas bernama Papan yang memiliki sepuluh method dan enam atribut.

Tabel 2.3 Tabel Atribut/Method Papan.java

Nama Atribut/Method	Deskripsi
N	Atribut untuk menyimpan ukuran baris papan permainan.
M	Atribut untuk menyimpan ukuran kolom papan permainan.
grid	Atribut untuk merepresentasikan keadaan papan permainan, dengan titik sebagai penanda kosong.
COLORS	Array untuk menyimpan kode warna ANSI yang berbeda tiap bloknya.

Papan	<p>Konstruktor untuk inisialisasi Papan berukuran N kali M, dengan alokasi memori untuk grid, serta memanggil createPapan.</p> <pre> public Papan(int N, int M, String S) { this.N = N; this.M = M; this.grid = new char[N][M]; createPapan(); } </pre>
RESET	String ANSI untuk mereset warna teks pada terminal setelah menampilkan warna tertentu.
getN()	Metode untuk mengembalikan jumlah baris papan.
getM()	Metode untuk mengembalikan jumlah kolom papan.
getGrid()	Metode untuk mengembalikan grid papan.
createPapan()	Metode untuk mengisi seluruh papan dengan karakter '.' Sebagai penanda papan masih kosong.
printPapan()	Metode untuk menampilkan papan permainan di terminal menggunakan warna di COLORS.
isFull()	Metode untuk mengecek apakah seluruh papan telah terisi oleh blok.
cekPapan()	<p>Metode untuk memeriksa apakah suatu blok dapat ditempatkan pada posisi (x,y) dimana blok tidak boleh keluar batas papan ataupun tumpang tindih dengan blok lain.</p> <pre> public boolean cekPapan(int x, int y, char[][] shape) { //cek apakah int baris = shape.length; int kolom = shape[0].length; if (x + baris > N y + kolom > M) { return false; } for (int i = 0; i < baris; i++) { for (int j = 0; j < kolom; j++) { if (shape[i][j] != '.' && grid[x + i][y + j] != '.') { return false; } } } return true; } </pre>
taruhBlok()	Metode untuk menempatkan blok dalam papan pada posisi (x,y) dengan karakter yang bukan titik.

	<pre> public void taruhBlok(int x, int y, char[][] shape, char idBlok) { int baris = shape.length; int kolom = shape[0].length; for (int i = 0; i < baris; i++) { for (int j = 0; j < kolom; j++) { if (shape[i][j] != '.') { grid[x + i][y + j] = shape[i][j]; } } } } </pre>
hapusBlok()	<p>Metode untuk menghapus blok dari papan dengan mengembalikan sel yang ditempati menjadi titik.</p> <pre> public void hapusBlok(int x, int y, char[][] shape) { int baris = shape.length; int kolom = shape[0].length; for (int i = 0; i < baris; i++) { for (int j = 0; j < kolom; j++) { if (shape[i][j] != '.') { grid[x + i][y + j] = '.'; } } } } </pre>
gridToString()	<p>Metode untuk mengubah isi grid menjadi string sehingga dapat disimpan dalam file dengan tiap barisnya diakhiri newline.</p>

2.4 File Puzzle.java

Pada file ini berisi program untuk melakukan beberapa operasi pada file input maupun output atau penyimpanan serta algoritma brute force yang digunakan. File ini memiliki kelas bernama Puzzle yang memiliki lima method dan tujuh atribut.

Tabel 2.4 Tabel Atribut/Method Puzzle.java

Nama Atribut/Method	Deskripsi
N	Atribut untuk menyimpan ukuran baris papan permainan.
M	Atribut untuk menyimpan ukuran kolom papan permainan.
P	Atribut untuk menyimpan banyaknya blok yang digunakan.
S	Atribut untuk menyimpan jenis konfigurasi puzzle yang pada kasus ini hanya boleh DEFAULT.
puzzleShape	Atribut list untuk menyimpan representasi bentuk tiap blok dalam array dua dimensi karakter.
count	Atribut untuk menghitung jumlah kasus yang diujikan dalam pencarian solusi.
Puzzle	Konstruktor untuk menginisialisasi objek Puzzle dengan nilai N, M, P, S, dan puzzleShape.

	<pre> public Puzzle (int N, int M, int P, String S, List<char[][]> puzzleShape){ this.N = N; this.M = M; this.P = P; this.S = S; this.puzzleShape = puzzleShape; } </pre>
getKasus()	Metode untuk mengembalikan jumlah kasus yang diujikan dalam pencarian solusi.
inputFile()	<p>Metode untuk membaca input file dengan format baris pertama berisikan nilai N, M, P dan baris kedua berisikan nilai S. Lalu baris ketiga hingga habis merepresentasikan bentuk blok. Mengembalikan objek Puzzle dengan data yang telah diolah.</p> <pre> public static Puzzle inputFile(String filename) throws IOException { BufferedReader reader = new BufferedReader(new FileReader(filename)); // Membaca baris pertama (N M P) String[] barisPertama = reader.readLine().trim().split("\\s+"); if (barisPertama.length < 3) { reader.close(); throw new IOException("Format file salah: Baris pertama harus berisi N M P."); } int N = Integer.parseInt(barisPertama[0]); // Jumlah baris papan int M = Integer.parseInt(barisPertama[1]); // Jumlah kolom papan int P = Integer.parseInt(barisPertama[2]); // Jumlah puzzle if (P < 1 P > 26) { reader.close(); throw new IllegalArgumentException("P harus berada di antara 1 dan 26"); } // Membaca baris kedua (S) String S = reader.readLine().trim(); if (!S.equals("DEFAULT")) { reader.close(); } } </pre>

	<pre> throw new IllegalArgumentException("S hanya berjenis 'DEFAULT'"); } // Membaca P puzzle shapes List<char[][]> blocks = new ArrayList<>(); String line; List<String> currentBlock = new ArrayList<>(); char currentChar = '\0'; while ((line = reader.readLine()) != null) { if (!line.isEmpty()) { if (currentBlock.isEmpty() line.charAt(0) == currentChar) { currentBlock.add(line); currentChar = line.charAt(0); } else { blocks.add(convertToCharArray(current Block, currentChar)); currentBlock.clear(); currentBlock.add(line); currentChar = line.charAt(0); } } } if (!currentBlock.isEmpty()) { blocks.add(convertToCharArray(currentBlock, currentChar)); } reader.close(); return new Puzzle(N,M,P,S,blocks); } </pre>
convertToCharArray()	Metode untuk mengubah daftar string representasi blok menjadi array karakter dua dimensi.
outputFile()	<p>Metode untuk menyimpan solusi dalam file yang ditentukan oleh pengguna.</p> <pre> public static void outputFile (String content, Scanner scanner) throws IOException{ System.out.printf("Masukkan nama file penyimpanan solusi (.txt): "); String fileSimpan = scanner.nextLine(); FileWriter writer = new FileWriter(fileSimpan); writer.write(content); writer.close(); } </pre>

	<pre> System.out.println("Solusi berhasil disimpan dalam file: " + fileSimpan); } </pre>
solve()	<p>Metode untuk menyelesaikan puzzle dengan brute force dengan mencoba tiap blok dalam berbagai kemungkinan bentuk (rotasi dan pencerminan), mencoba setiap posisi di papan, serta menggunakan backtracking saat kombinasi tidak menghasilkan solusi.</p> <pre> public boolean solve(Papan board, List<Blok> pieces, int index) { count++; if (index == pieces.size()) { return board.isFull(); } Blok currentPiece = pieces.get(index); List<char[][]> orientations = currentPiece.getShape(); for (char[][] orientation : orientations) { for (int row = 0; row < this.N; row++) { for (int col = 0; col < this.M; col++) { if (board.cekPapan(row, col, orientation)) { board.taruhBlok(row, col, orientation, (char) ('A' + index)); if (solve(board, pieces, index + 1)) { return true; } board.hapusBlok(row, col, orientation); } } } } return false; } </pre>

BAB III

MASUKAN DAN LUARAN PROGRAM

3.1 Test Case 1

Pada test case kali ini dengan informasi pada baris pertama $N = 13$, $M = 2$, dan $P = 26$ dengan jenis puzzle yaitu DEFAULT. Test case ini bertujuan mengecek warna output dari 26 blok yang berbeda dengan 26 warna yang berbeda.

```
test > ≡ test1.txt
You, 7 hours ago | 1 author (You)
1 13 2 26 You, 7 hours ago
2 DEFAULT
3 A
4 B
5 C
6 D
7 E
8 F
9 G
10 H
11 I
12 J
13 K
14 L
15 M
16 N
17 O
18 P
19 Q
20 R
21 S
22 T
23 U
24 V
25 W
26 X
27 Y
28 Z
```

Gambar 3.1.1 File test1.txt

```
Welcome to IQ Puzzler Pro Solver!
1. Solve Puzzle
2. Exit
Enter your choice: 1
Masukkan nama file input (.txt): test1.txt
Solusi Ditemukan!
AB
CD
EF
GH
IJ
KL
MN
OP
QR
ST
UV
WX
YZ

Waktu pencarian: 33 ms
Banyak kasus yang ditinjau: 27
Apakah anda ingin menyimpan solusi? (ya/tidak)
ya
Masukkan nama file penyimpanan solusi (.txt): output1.txt
Solusi berhasil disimpan dalam file: output1.txt
```

Gambar 3.1.2 Hasil Pengujian test1.txt

3.2 Test Case 2

Pada test case kali ini dengan informasi pada baris pertama $N = 5$, $M = 5$, dan $P = 7$ dengan jenis puzzle yaitu DEFAULT. Test case ini bertujuan mengecek apakah algoritma brute force yang digunakan sudah menunjukkan solusi yang sesuai karena test case ini ada di spesifikasi termasuk jawabannya.

```
test > test2.txt
You, 7 hours ago | 1 aut
1 5 5 7 You,
2 DEFAULT
3 A
4 AA
5 B
6 BB
7 C
8 CC
9 D
10 DD
11 EE
12 EE
13 E
14 FF
15 FF
16 F
17 GGG
18
```

Gambar 3.2.1 File test2.txt

```

Welcome to IQ Puzzler Pro Solver!
1. Solve Puzzle
2. Exit
Enter your choice: 1
Masukkan nama file input (.txt): test2.txt
Solusi Ditemukan!
AGGGC
AABCC
EEBBF
EEDFF
EDDDF

Waktu pencarian: 109 ms
Banyak kasus yang ditinjau: 27024
Apakah anda ingin menyimpan solusi? (ya/tidak)
ya
Masukkan nama file penyimpanan solusi (.txt): output2.txt
Solusi berhasil disimpan dalam file: output2.txt

```

Gambar 3.2.2 Hasil Pengujian test2.txt

3.3 Test Case 3

Pada test case kali ini dengan informasi pada baris pertama $N = 3$, $M = 3$, dan $P = 1$ dengan jenis puzzle yaitu DEFAULT. Test case ini bertujuan mengecek apakah algoritma dapat mengabaikan baris kosong dengan tetap melanjutkan pencarian solusi yang sesuai.

```

test > test3.txt
You, 7 hours ago | 1 a
1 3 3 1 You
2 DEFAULT
3 BBB
4 BBB
5
6 BBB

```

Gambar 3.3.1 File test3.txt

```

Welcome to IQ Puzzler Pro Solver!
1. Solve Puzzle
2. Exit
Enter your choice: 1
Masukkan nama file input (.txt): test3.txt
Solusi Ditemukan!
BBB
BBB
BBB

Waktu pencarian: 0 ms
Banyak kasus yang ditinjau: 2
Apakah anda ingin menyimpan solusi? (ya/tidak)
ya
Masukkan nama file penyimpanan solusi (.txt): output3.txt
Solusi berhasil disimpan dalam file: output3.txt

```

Gambar 3.3.2 Hasil Pengujian test3.txt

3.4 Test Case 4

Pada test case kali ini dengan informasi pada baris pertama $N = 3$, $M = 3$, dan $P = 3$ dengan jenis puzzle yaitu DEFAULT. Test case ini bertujuan mengecek algoritma brute force yang digunakan sudah menunjukkan solusi yang sesuai.

```

test > ≡ test4.txt
You, 7 hours ago | 1 aut
1 3 3 3 You,
2 DEFAULT
3 A
4 AA
5 B
6 BB
7 CCC
8

```

Gambar 2.4.1 File test4.txt

```

Welcome to IQ Puzzler Pro Solver!
1. Solve Puzzle
2. Exit
Enter your choice: 1
Masukkan nama file input (.txt): test4.txt
Solusi Ditemukan!
ABB
AAB
CCC

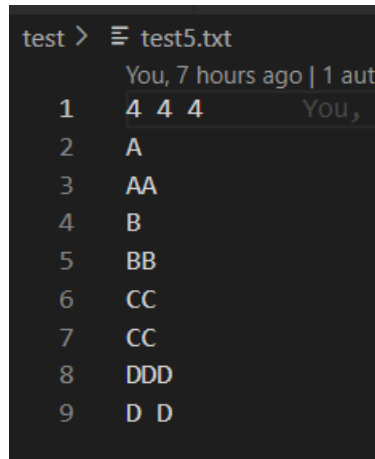
Waktu pencarian: 16 ms
Banyak kasus yang ditinjau: 5
Apakah anda ingin menyimpan solusi? (ya/tidak)
ya
Masukkan nama file penyimpanan solusi (.txt): output4.txt
Solusi berhasil disimpan dalam file: output4.txt

```

Gambar 3.4.2 Hasil Pengujian test4.txt

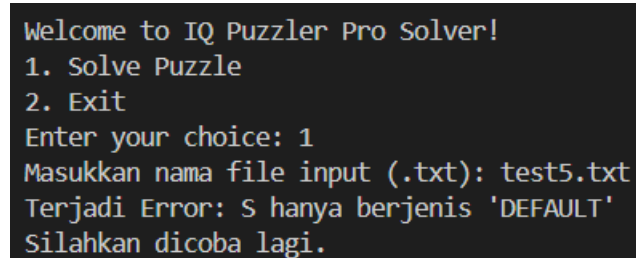
3.5 Test Case 5

Pada test case kali ini dengan informasi pada baris pertama $N = 4$, $M = 4$, dan $P = 4$. Test case ini bertujuan mengecek validasi format file input yaitu harus mengandung jenis puzzlenya yaitu DEFAULT/CUSTOM/PYRAMID.



```
test > test5.txt
You, 7 hours ago | 1 aut
1 4 4 4
2 A
3 AA
4 B
5 BB
6 CC
7 CC
8 DDD
9 D D
```

Gambar 3.5.1 File test5.txt

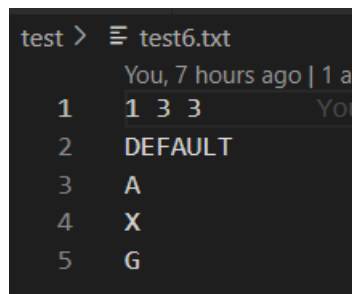


```
Welcome to IQ Puzzler Pro Solver!
1. Solve Puzzle
2. Exit
Enter your choice: 1
Masukkan nama file input (.txt): test5.txt
Terjadi Error: S hanya berjenis 'DEFAULT'
Silahkan dicoba lagi.
```

Gambar 3.5.2 Hasil Pengujian test5.txt

3.6 Test Case 6

Pada test case kali ini dengan informasi pada baris pertama $N = 1$, $M = 3$, dan $P = 3$ dengan jenis puzzle yaitu DEFAULT. Test case ini bertujuan mengecek algoritma brute force yang digunakan sudah menunjukkan solusi yang sesuai dengan bentuk papan tidak persegi dan urutan abjad puzzle tidak urut (A-Z).



```
test > test6.txt
You, 7 hours ago | 1 at
1 1 3 3
2 DEFAULT
3 A
4 X
5 G
```

Gambar 3.6.1 File test6.txt


```

Welcome to IQ Puzzler Pro Solver!
1. Solve Puzzle
2. Exit
Enter your choice: 1
Masukkan nama file input (.txt): test6.txt
Solusi Ditemukan!
AXG

Waktu pencarian: 20 ms
Banyak kasus yang ditinjau: 4
Apakah anda ingin menyimpan solusi? (ya/tidak)
ya
Masukkan nama file penyimpanan solusi (.txt): output6.txt
Solusi berhasil disimpan dalam file: output6.txt

```

Gambar 3.6.2 Hasil Pengujian test6.txt

3.7 Test Case 7

Pada test case kali ini dengan informasi pada baris pertama $N = 5$, $M = 4$, dan $P = 5$ dengan jenis puzzle yaitu DEFAULT. Test case ini bertujuan mengecek algoritma brute force yang digunakan sudah menunjukkan solusi yang sesuai, dimana pada test case kali ini solusi tidak ditemukan.

```

test > ≡ test7.txt
You, 7 hours ago | 1 author (You)
1  5 4 5      You, 7 hou
2  DEFAULT
3  AA
4  A
5  AA
6  BB
7  BBB
8  | C
9  CC
10 C
11 DDD
12 E
13 EE
14 | E

```

Gambar 3.7.1 File test7.txt

```

Welcome to IQ Puzzler Pro Solver!
1. Solve Puzzle
2. Exit
Enter your choice: 1
Masukkan nama file input (.txt): test7.txt
Tidak ada solusi.

```

Gambar 3.7.2 Hasil Pengujian test7.txt

LAMPIRAN

Link Repository Github : https://github.com/wrdtlkhoir/Tucil1_13523001.git

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	✓	