Tugas Kecil 1 IF2211 Strategi Algoritma
Semester II tahun 2022/2023

Penyelesaian IQ Puzzler Pro dengan Algoritma Brute Force

Disusun oleh:

Muhammad Iqbal Haidar (13523111)

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2025

DAFTAR ISI

BAB I

DESKRIPSI 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**.

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.

SPESIFIKASI

Wajib

- Buatlah program sederhana dalam bahasa **Java** yang mengimplementasikan ***algoritma Brute Force*** untuk mencari solusi dalam permainan IQ Puzzler Pro.
- Algoritma brute force yang diimplementasikan harus bersifat **"murni"**, tidak boleh memanfaatkan heuristik.
- Papan yang perlu diisi mulanya akan selalu kosong.
- Sebuah blok puzzle bisa saja dirotasi maupun dicerminkan sebelum diletakan pada papan.
- **Input:** program akan memberikan pengguna sebuah prompt untuk memilih file *test case* berekstensi **.txt**, kemudian program membaca file *test case* tersebut yang berisi
    1. **Dimensi Papan** terdiri atas dua buah variabel **N** dan **M** yang membentuk papan berdimensi NxM.
    2. **Banyak blok puzzle** direpresentasikan oleh variabel integer **P**.

3. **Jenis kasus** sebuah variabel string **S** yang digunakan untuk mengidentifikasi kasus konfigurasi, hanya mungkin bernilai salah satu diantara **DEFAULT/CUSTOM/PYRAMID**.
4. **Bentuk blok puzzle** yang dilambangkan oleh konfigurasi *Character* berupa huruf. Akan ada **P** buah blok puzzle berbeda yang dibentuk oleh **P buah huruf berbeda**. *Character* yang digunakan adalah huruf **A-Z dalam kapital**.

File .txt yang akan dibaca memiliki format sebagai berikut

```
N M P
S
puzzle_1_shape
puzzle_2_shape
…
puzzle_P_shape
```

Contoh *Test case*

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

- **Output:**
  1. Tampilkan **konfigurasi blok puzzle** yang berhasil mengisi papan. **Gunakan print berwarna** untuk menunjukkan blok puzzle dengan jelas. Pastikan setiap blok puzzle berbeda memiliki warna berbeda. Beri tahu pengguna apabila puzzle tidak memiliki solusi.

2. **Waktu eksekusi** program dalam *milisecond* (tidak termasuk waktu membaca masukan dan menyimpan solusi, cukup waktu pencarian oleh algoritma).
3. **Banyak kasus** atau jumlah iterasi yang ditinjau oleh algoritma brute force.
4. **Prompt untuk menyimpan solusi** dalam sebuah berkas berekstensi **.txt** (Struktur untuk file output dibebaskan).

Contoh *output* berdasarkan contoh *Input* diatas

```
AGGGD
AABDD
CCBBF
CEEFF
EEEFF

Waktu pencarian: 604 ms

Banyak kasus yang ditinjau: 7387

Apakah anda ingin menyimpan solusi? (ya/tidak)
```

BAB II

TEORI SINGKAT

Brute force merupakan salah satu jenis algoritma yang dapat digunakan untuk mencari solusi dari suatu permasalahan dengan mencoba semua kemungkinan. Algoritma ini memanfaatkan kombinasi serta permutasi untuk membuat seluruh kemungkinan sampel solusi persoalan. Kelebihan dari algoritma ini adalah pendekatannya yang straightforward sehingga mudah bagi orang awam untuk memahaminya.

Pada bidang komputasi, algoritma brute force biasa digunakan sebagai basis pembanding algoritma lain yang dinilai lebih efisien. Algoritma ini juga digunakan untuk memecahkan persoalan yang lebih kompleks seperti travelling salesperson problem dan knapsack problem sebab karakteristik brute force yang dapat memecahkan hampir semua persoalan.

Adapun brute force juga memiliki kelemahan yakni efisiensi yang kurang baik. Hal ini disebabkan brute force pada umumnya memiliki kompleksitas waktu eksponensial sehingga semakin banyak masukkan maka waktu yang diperlukan untuk menjalankan algoritma bertumbuh secara eksponensial.

BAB III

ALGORITMA

1. Ambil sembarang block yang tersedia. Periksa apakah block bisa ditempatkan pada sel papan paling kiri atas (0, 0) [block dikatakan bisa ditempatkan jika seluruh sel block tepat satu berpasangan/bijektif dengan sel papan].
2. Apabila block tidak bisa ditempatkan pada sel papan tersebut, coba untuk pindahkan block sebanyak satu sel papan ke kanan (x+1, y) atau jika posisi sel papan sudah paling kanan maka pindahkan ke baris papan selanjutnya (0, y+1). Apabila sel papan sudah paling kanan bawah, maka coba untuk rotasi block 90 derajat searah jarum jam dan mulai kembali periksa dari sel papan pojok kiri atas. Apabila rotasi block sudah dilakukan sebanyak tiga kali dan masih belum bisa ditempatkan maka cerminkan block terhadap sumbu y, dan kembali periksa dari sel papan kiri atas. Tidak lupa lakukan kemungkinan posisi rotasi sebanyak tiga kali untuk block yang sudah dicerminkan.
3. Apabila block dapat ditempatkan pada sel papan tersebut, maka tempatkan block pada papan sel tersebut dan lanjut kepada block lain dan ulangi langkah nomor 2.
4. Apabila block sudah melewati langkah nomor 2 dan tidak menemukan tempat maka mundur (backtrack) ke block terakhir penempatan dan cari penempatan sel lainnya untuk block terakhir tersebut (langkah nomor 2)
5. Permainan dikatakan tidak memiliki solusi jika block tersedia sudah habis tetapi sel papan belum penuh atau ada block yang tidak dapat terpasang pada papan. Permainan dikatakan memiliki solusi jika semua block terpasang tepat satu sel dengan sel papan.

KODE SUMBER

```
public class papan {
    private int row, col;
    private char[][] buffer;
    private long time = 0;
    private int eval = 0;

    public papan(int row, int col) {
```

```
        this.row = row;
        this.col = col;
        this.buffer = new char[row][col];

        for (char[] rows : buffer) {
            Arrays.fill(rows, '.');
        }
    }
}
```

Gambar 2.1 Kelas papan

```
    public boolean checkPlacement(block b, int tlx, int tly) {
        int[][] buff = b.getBuffer();
        int rows = buff.length;
        int cols = buff[0].length;

        if (tlx + cols > col || tly + rows > row) {
            return false;
        }

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (buff[i][j] == 1 && buffer[tly + i][tlx + j] != '.') {
                    return false;
                }
            }
        }

        return true;
    }
```

Gambar 2.2 Fungsi periksa apakah block bisa ditempatkan pada sel papan (tlx, tly)

```
    public void place(block b, int tlx, int tly) {
        char id = b.getId();
        int[][] buff = b.getBuffer();
        int rows = buff.length;
        int cols = buff[0].length;

        for (int i = 0; i < rows; i++) {
```

```
            for (int j = 0; j < cols; j++) {
                if (buff[i][j] == 1) {
                    buffer[tly + i][tlx + j] = id;
                }
            }
        }
    }
```

Gambar 2.3 Fungsi meletakan block pada sel papan (tlx, tly)

```
    public void remove(block b, int tlx, int tly) {
        int[][] buff = b.getBuffer();
        int rows = buff.length;
        int cols = buff[0].length;

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (buff[i][j] == 1) {
                    buffer[tly + i][tlx + j] = '.';
                }
            }
        }
    }
```

Gambar 2.4 Fungsi menghapus block pada sel papan (tlx, tly)

```
public class block {
    private char id;
    private int[][] buffer;

    public block(char id, int[][] buffer) {
        this.id = id;
        this.buffer = buffer;
    }
}
```

Gambar 2.5 Kelas block

```
    public void rotate() {
        int row = buffer.length;
        int col = buffer[0].length;
        int[][] rotated = new int[col][row];
```

```
        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {
                rotated[j][row - 1 - i] = buffer[i][j];
            }
        }
        buffer = rotated;
    }
```

Gambar 2.6 Fungsi rotasi block sebesar 90 derajat searah jarum jam

```
    public void mirror() {
        int row = buffer.length;
        int col = buffer[0].length;
        int[][] mirrored = new int[row][col];

        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {
                mirrored[i][col - 1 - j] = buffer[i][j];
            }
        }
        buffer = mirrored;
    }
```

Gambar 2.7 Fungsi pencerminan block terhadap sumbu y

```
public class solver {
    public boolean solve(papan p, ArrayList<block> b, int idx) {
        if (idx >= b.size()) {
            return p.isFull();
        }

        for (int m = 0; m < 2; m++) {
            if (m > 0) {
                b.get(idx).mirror();
            }
            for (int r = 0; r < 4; r++) {
                if (r > 0) {
                    b.get(idx).rotate();
                }
                for (int y = 0; y <= (p.getRow() - b.get(idx).getRowLen());
```

```
y++) {
                    for (int x = 0; x <= (p.getCol() -
b.get(idx).getColLen()); x++) {
                        if (p.checkPlacement(b.get(idx), x, y)) {
                            p.incEval();
                            p.place(b.get(idx), x, y);
                            if (solve(p, b, idx + 1)) {
                                return true;
                            }
                            p.remove(b.get(idx), x, y);
                        }
                    }
                }
            }
        }
        return false;
    }
}
```

Gambar 2.8 Kode Solver

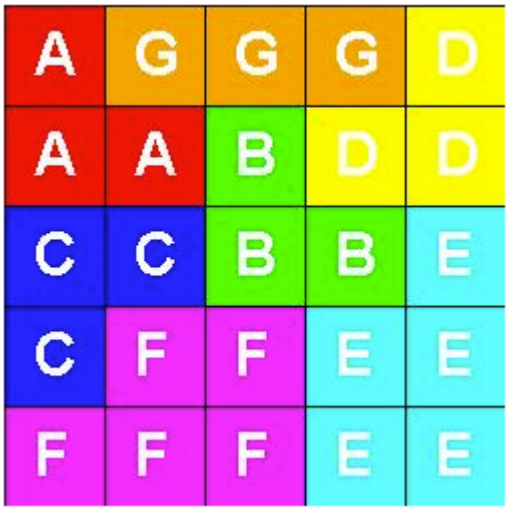
BAB IV

PENGUJIAN

1. Kasus Uji 1

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

```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test1.txt

Solusi ditemukan.

AGGGD
AABDD
CCBBE
CFFEE
FFFEE

Waktu pencarian: 66ms

Banyak kasus yang ditinjau: 27407

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```

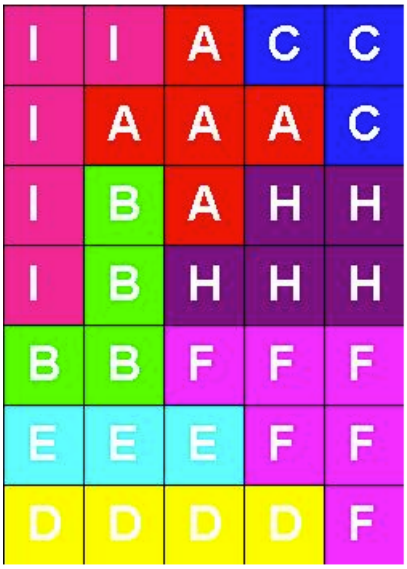


2. Kasus Uji 2

```
7  5  8
DEFAULT
 A
AAA
 A
```

```
B
B
BB
C
CC
DDDD
E
E
E
  F
 FF
FFF
H
HH
HH
II
I
I
I
```

```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test2.txt

Solusi ditemukan.

IIACC
IAAAC
IBAHH
IBHHH
BBFFF
EEEFF
DDDDF

Waktu pencarian: 48947ms

Banyak kasus yang ditinjau: 30651524

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```

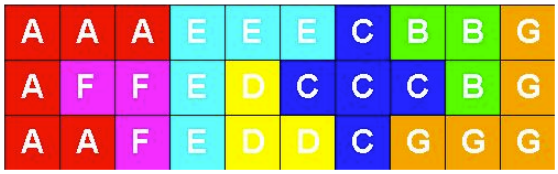


3. Kasus Uji 3

```
3 10 7
DEFAULT
AAA
A
AA
BB
 B
```

```
 C
CCC
 C
D
DD
E
E
EEE
F
FF
G
G
GGG
```

```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test3.txt

Solusi ditemukan.

AAAEEECBBG
AFFEDCCCBG
AAFEDDCGGG

Waktu pencarian: 82ms

Banyak kasus yang ditinjau: 37466

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```

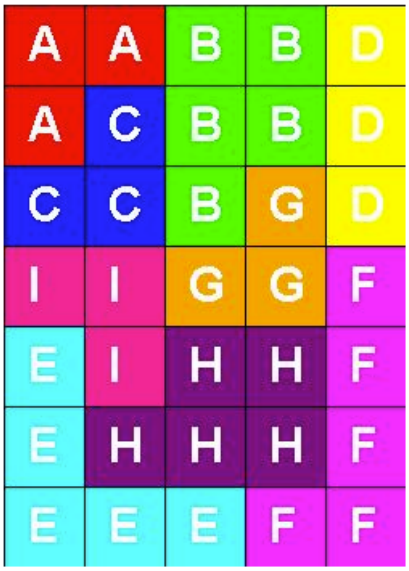


4. Kasus Uji 4

```
7 5 9
DEFAULT
AA
A
BB
BB
B
 C
CC
D
D
D
E
E
EEE
FFFF
```

```
F
G
GG
HH
HH
H
I
II
```

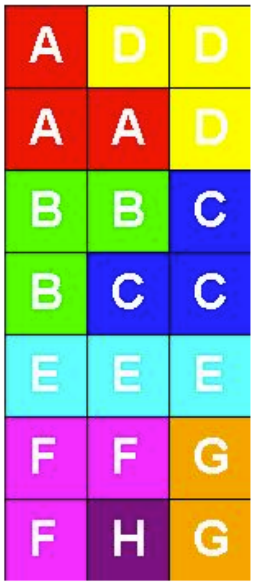
```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test4.txt

Solusi ditemukan.

AABBD
ACBBD
CCBGD
IIGGF
EIHHF
EHHHF
EEEFF

Waktu pencarian: 7ms

Banyak kasus yang ditinjau: 719

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```



5. Kasus Uji 5

```
7 3 8
DEFAULT
A
AA
BB
B
 C
CC
DD
 D
EEE
FF
F
G
G
H
```

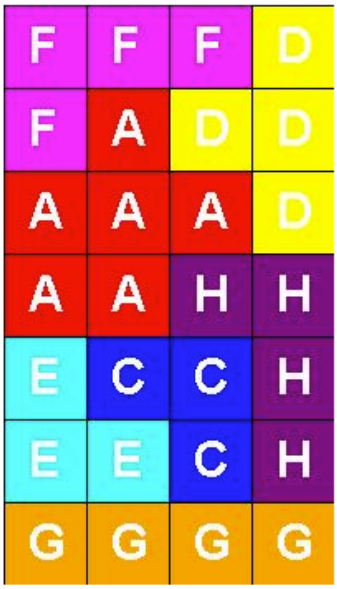
```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test5.txt

Solusi ditemukan.

ADD
AAD
BBC
BCC
EEE
FFG
FHG

Waktu pencarian: 0ms

Banyak kasus yang ditinjau: 8

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```



6. Kasus Uji 6

```
7 4 7
DEFAULT
 A
AAA
AA
CC
 C
D
DD
D
E
EE
F
F
FF
GGGG
HHH
H
```

```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test6.txt

Solusi ditemukan.

FFFD
FADD
AAAD
AAHH
ECCH
EECH
GGGG

Waktu pencarian: 830ms

Banyak kasus yang ditinjau: 538914

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```



7. Kasus Uji 7

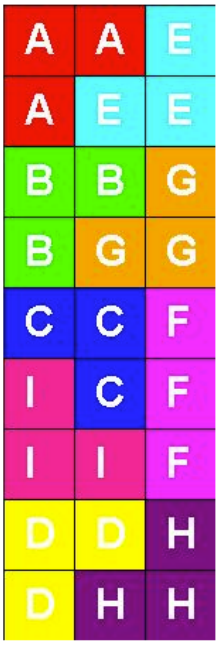
```
2 12 8
DEFAULT
AA
A
BB
B
CC
C
DD
D
EE
E
FF
F
GG
G
HH
H
```

```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test7.txt

Solusi ditemukan.

AAEBBFHCCDDG
AEEBFFHHCDGG

Waktu pencarian: 1163ms

Banyak kasus yang ditinjau: 1068216

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```

8. Kasus Uji 8

```
9 3 9
DEFAULT
AA
A
BB
B
CC
C
DD
D
EE
E
F
F
F
GG
G
HH
H
II
I
```

```
Selamat Datang di IQ Puzzler Pro Solver

Masukkan nama file: test8.txt

Solusi ditemukan.

AAE
AEE
BBG
BGG
CCF
ICF
IIF
DDH
DHH

Waktu pencarian: 20269ms

Banyak kasus yang ditinjau: 14584649

Apakah solusi ingin disimpan? (y/n): y
Solusi berhasil disimpan di ../data/solusi.txt
Apakah gambar ingin disimpan? (y/n): y
Gambar berhasil disimpan di ../data/solusiGambar.jpg
```



BAB V

KESIMPULAN

Berdasarkan hasil pengujian dapat disimpulkan bahwa algoritma brute force cukup efektif untuk mencari solusi permasalahan dengan lingkup yang kecil. Apabila algoritma ini dipaksakan untuk mencari permasalahan yang lebih besar/kompleks maka akan memakan waktu yang lama. Diperlukan algoritma lain yang lebih efisien untuk menghadapi permasalahan seperti ini.

SARAN

Kedepannya boleh ditambahkan fitur lain yang memudahkan pengguna seperti GUI dan algoritma yang digunakan harus di improve agar lebih efisien.

Checklist

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	✔	

Pranala Github

https://github.com/iqbalhaidr/Tucil1_13523111