**LAPORAN TUGAS KECIL 1**

**MATA KULIAH IF 2211 STRATEGI ALGORITMA**

**PENYELESAIAN PERMAINAN “IQ PUZZLER PRO”**

**MENGGUNAKAN ALGORITMA**

**BRUTE FORCE**

**A black and white drawing of an elephant

AI-generated content may be incorrect.**

**Dosen Pengampu**

Dr. Nur Ulfa Maulidevi, S.T, M.Sc.

**Disusun Oleh:**

|  |  |
| --- | --- |
| Adhimas Aryo Bimo | 13523052 |

**SEKOLAH TEKNIK INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**FEBRUARI 2025**

**DAFTAR ISI**

Isi

**No table of contents entries found.**

# BAB I

# LATAR BELAKANG

## IQ PUZZLER PRO



### Gambar 1.1

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.

A colorful balls on a black tray

AI-generated content may be incorrect.

### Gambar 1.2 Mainan Terselesaikan

## BRUTE FORCE

A screen shot of a computer

AI-generated content may be incorrect.

### Gambar 1.3 Contoh Algoritma Brute Force

Dalem dunia komputer sains, brute force merupakan suatu metode pemecahan persoalan yang cukup umum. Brute force memiliki ciri-ciri yakni mengecek semua kemungkinan kejadian tak peduli apakah kejadian tersebut memenuhi persoalan atau tidak. Algoritma brute force akan mencari setiap opsi hingga jawaban dari suatu masalah dapat ditemukan.

Pendekatan bruteforce cukup umum ditemukan, bahkan tidak disadari kita lakukan dalam menyelesaikan berbagai persoalan pemrograman. Namun, karena pencarian solusi yang terlalu mendalam, algoritma ini cenderung memiliki kompleksitas yang cukup tinggi dan tidak efisien untuk menyelesaikan permasalahan yang besar.

# BAB II

# DESKRIPSI & IMPLEMENTASI ALGORITMA

## Alur Berpikir

Dalam menyelesaikan permasalahan yang tidak memiliki langkah pasti, algoritma Brute-Force menjadi salah satu alternatif yang dapat digunakan dalam menyelesaikan persoalan tersebut. Seperti yang dipaparkan pada penjelasan mengenai peraturan permainan IQ PUZZLER, kita dapat fokus pada beberapa aspek yang krusial untuk menjadi landasan berpikir dalam merencanakan program ini.

Pertama, blok. Bentuk tiap blok pada permainan ini acak, Penempatan blok tidak bergantung pada posisi mana pun dan akan valid jika tiap satuan blok memenuhi tempat pada papan. Dalam hal ini, perlu untuk mengecek semua kemungkinan posisi yang valid bagi blok untuk ditempatkan pada papan. Semua posisi berpotensi untuk menjadi tempat yang pas bagi blok

Kedua, transformasi blok. Blok tidak hanya ditempatkan, melainkan dapat diubah orientasi dan juga dapat dicerminkan. Dalam hal ini, dalam semua kondisi bagi yang valid ataupun tidak, blok memiliki alternatif cara penempatan. Oleh karena itu, diperlukan untuk mengecek transformasi orientasi tiap blok pada tiap posisi untuk mencari solusi pada permainan ini.

Ketiga, metode penempatan blok. Dalam hal ini ada beberapa alternatif bagaimana cara menyusun blok yang baik. Misal, dengan meletakkan posisi yang rumit pada pojok papan, atau mulai dengan bentuk blok yang paling besar. Namun, karena algoritma yang digunakan pada persoalan ini berfokus pada Brute-Force, metode penempatan yang paling mendekati algoritma itu adalah dengan menggunakan Backtracking. Jika semua posisi yang telah dicoba oleh blok ternyata tidak memenuhi aturan penempatan blok. blok sebelumnya akan diubah cara penempatannya dengan mencari tempat penempatan lain yang valid. Hal ini dapat digunakan dengan menggunakan backtracing pada posisi terakhir blok ditempatkan secara valid.

Ibaratkan ketika manusia mencoba memainkan puzzle, kita akan mencoba memasangkan blok yang pas pada papan dan sesuai dengan aturan permainan. Jika ada blok yang tidak dapat dipasang, maka blok lain akan dilepas dan diubah entah posisi maupun orientasinya agar blok yang tidak dipasang itu dapat dipasang pada papan.

## Kode Program

A screenshot of a computer program

AI-generated content may be incorrect.

### Gambar 2.1 Constructor, Getter, dan Metode Pemanggilan BruteForce

A screen shot of a computer program

AI-generated content may be incorrect.

### Gambar 2.2 Rekursif dan Backtracking untuk Mencari Semua Solusi

Pada kedua gambar diatas, dituliskan program dari hasil pemikiran pada bagian Alur Berpikir. Berikut merupakan rincian penjelasan program.

1. Inisiasi

Untuk melalukan pemnggilan kelas Brute Force, perlu diinisasi dengan koleksi dari semua blok (listBlocks) dan papan (Board). Hal ini diperlukan sebagai wahana untuk melakukan percobaan untuk mencari solsui pada permainan

1. Cek Tile Block dan Area Board

Sebelum melakukan operasi, kita dapat mengecek terlebih dahulu apakah solusi yang ada pada input valid. Setidaknya, dengan mengecek total kepingan blok dengan area papan, kita dapat mengetahui bahwa jika keduanya cocok, kemungkinan akan ada solusi yang bisa dicari. Jika terdapat perbedaan, maka tidak mungkin akan ada solusi bagi permainan.

1. Rekursif

Program akan mencari solusi secara rekursi melalui metode tryAllPossibility. Basis pada metode ini ada 2 kemungkinan, yakni. Jika semua kotak pada papan sudah terisi, dan jika semua blok sudah dicoba. Jika semua kotak pada papan sudah terisi, maka akan menghasilkan solusi yang valid, dan kebalikannya jika semua blok sudah dicoba dan papan belum terisi, maka tidak mungkin ada solusi.

Lalu, program dilanjurkan dengan mengecek posisi yang valid pada tiap kolom dan alternatif orientasi blok. Untuk mengecek apakah posisi pada blok valid, dapat dilakukan dengan mengecek index yang valid untuk blok. Dalam hal ini, mengurangi baris dan kolom pada papan dengan baris dan kolom pada blok akan menghasilkan posisi yang valid bagi blok. Jika terdapat blok dengan ukuran 2 x 2 pada papan berukuran 4 x4, maka kita hanyak perlu untuk memosisikan indeks blok tersebut hingga baris dan kolom 2 karena pada posisi itulah blok akan valid ditempatkan.

Selanjutnya, blok akan dicek apakah pada posisi tersebut blok tidak bertabrakan dengan blok lain melalui metode isValidPlace. Jika valid blok akan diletakkan dan dilanjutkan dengan rekursif untuk mencari posisi lain yang valid.

1. Transformasi blok

Jika pada penempatan blok gagal, kita dapat mencari alternatif penempatan blok dengan mengubah orientasi blok. Orientasi dilakukan sebesar 90o pada sekali iterasi (berarti akan ada 4 kali rotasi, 0o , 90o , 180o , 270o ). Selain itu, dilakukan juga pencerminan pada blok. Kedua metode tersebut jika ditotal akan memberikan 8 alternatif kemungkinan posisi valid blok pada satu tempat.

Untuk megeksekusi kode diatas, terdapat kode utilitas untuk membantu menjalankan kode. Berikut merupakan kode utilitas yang digunakan.

A screen shot of a computer code

AI-generated content may be incorrect.

### Gambar 2.3 Utilitas pada Kelas Brute Force

# BAB III

# SEKILAS KODE PROGRAM

## Kelas Blok

**A screen shot of a computer

AI-generated content may be incorrect.**

### Gambar 3.1 Constructor dan Getter pada Blok

A screen shot of a computer

AI-generated content may be incorrect.

### Gambar 3.2 Setter pada Blok

A screen shot of a computer

AI-generated content may be incorrect.

### Gambar 3.3 Fungsi Rotasi pada Blok

A screen shot of a computer

AI-generated content may be incorrect.

### Gambar 3.4 Fungsi Mirror dan Utilitas pada Blok

## Kelas Papan

A screen shot of a computer program

AI-generated content may be incorrect.

### Gambar 3.5 Constructor dan Getter pada kelas Papan

**A screen shot of a computer program

AI-generated content may be incorrect.**

### Gambar 3.6 Setter dan Utilitas pada kelas Papan

# BAB IV

# PERCOBAAN PROGRAM

## Eksperimen test1.txt

A screenshot of a computer

AI-generated content may be incorrect.

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

## Eksperimen test2.txt

**A screenshot of a computer game

AI-generated content may be incorrect.**

|  |
| --- |
| 8 5 10 DEFAULT AA BBBBB CC DDDDD D EEEE  FF  F  F  F FF GGGG H JJJ  K KKKKK |

## Eksperimen test3.txt

A screenshot of a computer game

AI-generated content may be incorrect.

|  |
| --- |
| 6 5 7 DEFAULT AAA A A A A AAA BBBB B CCC DD DD EE E FF F GG |

## Eksperimen test4.txt

**A screenshot of a puzzle game

AI-generated content may be incorrect.**

|  |
| --- |
| 3 5 5 DEFAULT AAAAA BB CC  C DD D E  E |

## Eksperimen test5.txt

**A screenshot of a puzzler

AI-generated content may be incorrect.**

|  |
| --- |
| 12 1 3 DEFAULT A AA B BBBB C CC C |

## Eksperimen test6.txt

**A screenshot of a computer game

AI-generated content may be incorrect.**

|  |
| --- |
| 10 12 8 DEFAULT AAAAAAAAA AAAAAAAAA  AAAA  AAAAAA  AAAAAA AAAAAAAAAAA AAAAAAAAAAA AAAAAAAAA AAA AAA AAA AAA HHH H  BBB  BBBBB BBBBBB RR KKKKK VVV VVV III III NNN NNN |

## Eksperimen test7.txt

A screenshot of a computer

AI-generated content may be incorrect.

|  |
| --- |
| 13 2 26 DEFAULT 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 |

# BAB V

# TAUTAN PROGRAM

Program dapat diakses pada .......

# BAB VI

# 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 | ✓ |  |