

LAPORAN TUGAS KECIL 1 STRATEGI ALGORITMA IF 2211
SEMESTER II TAHUN 2024/2025

**PENYELESAIAN IQ PUZZLER PRO DENGAN ALGORITMA BRUTE
FORCE**



KENNETH RICARDO CHANDRA
13523022

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2025

BAB I

Deskripsi Masalah



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.

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**.

Laporan ini berisi mengenai penyelesaian permainan IQ Puzzler Pro menggunakan algoritma Brute Force

BAB II

Penjelasan Algoritma Brute Force yang digunakan

Algoritma Brute Force yang digunakan untuk menyelesaikan IQ Puzzler Pro berupa dengan mencoba kombinasi peletakan piece atau blok puzzle ke dalam area yang kosong dengan piece tersebut dapat dirotasi dan dicerminkan. Dengan melakukan hal tersebut, maka pada akhirnya akan ditemukan terdapat minimal 1 solusi pemecahan puzzle, atau tidak ditemukan solusi sama sekali. Selain itu, saat hanya tersisa satu piece yang tersisa dan tidak dapat diletakkan pada area yang tersisa, maka akan dilakukan backtracking dengan mencoba mundur dan mencoba kombinasi yang lain, baik dengan merotasi, mengganti lokasi maupun mencerminkan piece.

BAB III

Source Code Program beserta Penjelasannya.

Program ini dibuat dengan bahasa Java dan dibuat dalam satu package dengan beberapa komponen yaitu Piece.java untuk bagian puzzle, Board.java untuk bagian board atau papan, Solver.java untuk algoritma penyelesaian, Reader.java untuk bagian membaca input, File.java untuk contoh file yang dapat di input dan dibaca oleh Reader, ResultSaver.java untuk menyimpan hasil solving.

Solver.java (dengan Algoritma Brute Force)

```
1  public class Solver {
2      private int iterations = 0;
3
4      public boolean solve(Board board, List<Piece> pieces) {
5          return placeNextPiece(board, pieces, 0);
6      }
7
8      private boolean placeNextPiece(Board board, List<Piece> pieces, int index) {
9          if (index == pieces.size()) {
10             if (board.boardFull()) {
11                 board.printBoard();
12                 return true;
13             }
14             return false;
15         }
16
17         Piece piece = pieces.get(index);
18         for (Piece orientation : piece.getNewPiece()) {
19             if (tryPlacements(board, orientation, index, pieces)) {
20                 return true;
21             }
22         }
23         return false;
24     }
```

```

1  private boolean tryPlacements(Board board, Piece piece, int index, List<Piece> pieces) {
2      for (int i = 0; i < board.getN(); i++) {
3          for (int j = 0; j < board.getM(); j++) {
4              if (board.placePieceAvailable(piece, i, j)) {
5                  iterations++;
6                  board.placePiece(piece, i, j, (char) ('A' + index));
7                  if (placeNextPiece(board, pieces, index + 1)) {
8                      return true;
9                  }
10                 board.removePiece(piece, i, j);
11             }
12         }
13     }
14     return false;
15 }
16
17 public int getIterations() {
18     return iterations;
19 }
20 }

```

Solver ini merupakan solver untuk IQ Puzzler Pro menggunakan algoritma brute force dan menggunakan pendekatan rekursif, dimana sebuah piece akan diletakkan di bagian papan yang masih kosong, dan akan terus dicoba sampai sebuah piece tidak dapat diletakkan di papan karena tidak pas. Kode akan dimulai dari tryPlacements yang akan mengecek apakah dapat memasang piece pada board, dan dapat mencoba juga apabila sudah penuh, maka akan menghapus piece yang telah dipasang untuk diubah cara pemasangannya dengan diputar atau dicerminkan atau diubah tempat penempatannya. Setelah itu, berlanjut ke placeNextPiece yang akan menaruh piece pada board. Jumlah iterasinya dihitung dari iterations dan didapatkan dengan getIterations.

Main.java

```
1 package program;
2
3 import java.io.IOException;
4 import java.util.List;
5 import java.util.Scanner;
6
7 public class Main {
8     public static void main(String[] args) {
9         try (Scanner consoleScanner = new Scanner(System.in)) {
10             System.out.print("Masukkan nama file test case (.txt): ");
11             String fileName = consoleScanner.nextLine();
12             File testCase = Reader.readFile(fileName);
13
14             Board board = new Board(testCase.N, testCase.M);
15             List<Piece> pieces = testCase.puzzlePiece.stream().map(Piece::new).toList();
16
17             Solver solver = new Solver();
18             long startTime = System.nanoTime();
19             boolean result = solver.solve(board, pieces);
20             long endTime = System.nanoTime();
21
22             if (!result) {
23                 System.out.println("Tidak ada solusi yang ditemukan.");
24             }
25
26             System.out.printf("Waktu pencarian: %.3f ms\n", (endTime - startTime) * 1E-6);
27             System.out.println("Jumlah iterasi: " + solver.getIterations() + " kali");
28
29             System.out.print("Apakah hasil ingin disimpan (y/n)? ");
30             if (consoleScanner.nextLine().trim().equalsIgnoreCase("y")) {
31                 System.out.print("Nama file yang disimpan: ");
32                 String saveFileName = consoleScanner.nextLine();
33                 ResultSaver.saveResult(board, startTime, endTime, saveFileName + ".txt", solver.getIterations());
34             }
35         } catch (IOException e) {
36             System.err.println("Gagal membaca file: " + e.getMessage());
37         }
38     }
39 }
```

Main adalah bagian utama dalam program ini yang menjadi tampilan utama program. Sistem akan meminta user untuk memasukkan file test case yang ingin dicoba lalu akan dijalankan solver() dari yang dibuat di solver.java. Selain itu, waktu pengerjaan disimpan dalam nanoTime() dan bila tidak ada solusi, maka akan ditulis tidak ada solusi yang ditemukan dan berapa lama yang diberikan serta berapa jumlah iterasinya. Selain itu, terdapat juga fitur untuk menyimpan hasil dalam bentuk Txt dan nama dari file yang disimpan. Apabila file gagal dibaca, akan ditulis bahwa file gagal dibaca.

Piece.java (bagian piece puzzle)



```
1  package program;
2
3  import java.util.*;
4
5  public class Piece
6  {
7      private char[][] block;
8      private int height;
9      private int width;
10
11     public Piece(String pieceStr) {
12         String[] lines = pieceStr.split("\n");
13         height = lines.length;
14         width = 0;
15         for (String line : lines) {
16             if (line.length() > width) {
17                 width = line.length();
18             }
19         }
20         width -= 1;
21         block = new char[height][width];
22
23         for (int i = 0; i < height; i++) {
24             String line = lines[i];
25             for (int j = 0; j < width; j++) {
26                 try {
27                     char currentchar = line.charAt(j);
28                     if (currentchar >= 65 && currentchar <= 90) {
29                         block[i][j] = currentchar;
30                     }
31                     else block[i][j] = '.';
32                 }
33                 catch (Exception e)
34                 {
35                     block[i][j] = '.';
36                 }
37             }
38         }
39     }
```

```
1  public Piece(char[][] shape)
2  {
3      this.height = shape.length;
4      this.width = shape[0].length;
5      this.block = new char[height][width];
6      for (int i = 0; i < height; i++)
7      {
8          System.arraycopy(shape[i], 0, this.block[i], 0, width);
9      }
10 }
11
12 public int getHeight()
13 {
14     return height;
15 }
16
17 public int getWidth()
18 {
19     return width;
20 }
21
22 public char[][] getBlock()
23 {
24     return block;
25 }
26
27 public Piece getRotatedPiece()
28 {
29     int newHeight = width;
30     int newWidth = height;
31     char[][] newShape = new char[newHeight][newWidth];
32     for (int i = 0; i < newHeight; i++)
33     {
34         for (int j = 0; j < newWidth; j++)
35         {
36             newShape[i][j] = block[height - 1 - j][i];
37         }
38     }
39     return new Piece(newShape);
40 }
```



```
1 public Piece getMirroredPiece()
2 {
3     char[][] newShape = new char[height][width];
4     for (int i = 0; i < height; i++)
5     {
6         for (int j = 0; j < width; j++)
7         {
8             newShape[i][j] = block[i][width - 1 - j];
9         }
10    }
11    return new Piece(newShape);
12 }
13
14 public List<Piece> getNewPiece()
15 {
16     List<Piece> newPiece = new ArrayList<>();
17     Set<String> seen = new HashSet<>();
18     Piece currentPiece = this;
19     for (int i = 0; i < 4; i++)
20     {
21         String rep = currentPiece.toString();
22         if (!seen.contains(rep))
23         {
24             newPiece.add(currentPiece);
25             seen.add(rep);
26         }
27         Piece mirrored = currentPiece.getMirroredPiece();
28         rep = mirrored.toString();
29         if (!seen.contains(rep)) {
30             newPiece.add(mirrored);
31             seen.add(rep);
32         }
33         currentPiece = currentPiece.getRotatedPiece();
34     }
35     return newPiece;
36 }
37
38 public String toString()
39 {
40     StringBuilder sb = new StringBuilder();
41     for (int i = 0; i < height; i++) {
42         sb.append(new String(block[i]));
43         if (i < height - 1) sb.append("\n");
44     }
45     return sb.toString();
46 }
47 }
```

Piece ini dapat dirotasi dan di cerminkan dengan GetNewPiece yang akan mengubah Piece yang didapat menjadi rotasi atau dicerminkan sesuai yang dibutuhkan.

Board.java (papan)



```
1  package program;
2
3  import java.io.BufferedWriter;
4  import java.io.IOException;
5
6  public class Board {
7
8      char[][] board;
9      int N, M;
10
11     private static final String[] Colour = {
12         "\u001B[31m", // Red (A)
13         "\u001B[34m", // Blue (B)
14         "\u001B[32m", // Green (C)
15         "\u001B[33m", // Yellow (D)
16         "\u001B[36m", // Cyan (E)
17         "\u001B[35m", // Magenta (F)
18         "\u001B[37m", // White (G)
19         "\u001B[90m", // Gray (H)
20         "\u001B[91m", // Light Red (I)
21         "\u001B[94m", // Light Blue (J)
22         "\u001B[92m", // Light Green (K)
23         "\u001B[93m", // Light Yellow (L)
24         "\u001B[96m", // Light Cyan (M)
25         "\u001B[95m", // Light Magenta (N)
26         "\u001B[97m", // Light White (O)
27         "\u001B[41m", // Red Background (P)
28         "\u001B[44m", // Blue Background (Q)
29         "\u001B[42m", // Green Background (R)
30         "\u001B[43m", // Yellow Background (S)
31         "\u001B[46m", // Cyan Background (T)
32         "\u001B[45m", // Magenta Background (U)
33         "\u001B[47m", // White Background (V)
34         "\u001B[101m", // Light Red Background (W)
35         "\u001B[104m", // Light Blue Background (X)
36         "\u001B[103m", // Light Green Background (Y)
37         "\u001B[102m" // Light Yellow Background (Z)
38     };
```

```
1 public Board(int N, int M) {
2     this.N = N;
3     this.M = M;
4     board = new char[N][M];
5     for (int i = 0; i < N; i++){
6         for (int j = 0; j < M; j++){
7             board[i][j] = '.';
8         }
9     }
10 }
11
12 public int getN(){
13     return N;
14 }
15
16 public int getM(){
17     return M;
18 }
19
20 public void printBoard() {
21     for (int i = 0; i < N; i++){
22         for (int j = 0; j < M; j++){
23             char tile = board[i][j];
24             if (tile == '.'){
25                 System.out.print("\u001B[0m" + tile + " ");
26             } else {
27                 int colourIndex = (tile - 'A') % Colour.length;
28                 System.out.print(Colour[colourIndex] + tile + " \u001B[0m");
29             }
30         }
31         System.out.println();
32     }
33     System.out.println();
34 }
35
36 public boolean placePieceAvailable(Piece piece, int row, int col){
37     char[][] shape = piece.getBlock();
38     int height = piece.getHeight();
39     int width = piece.getWidth();
40
41     if (row + height > N || col + width > M){
42         return false;
43     }
44     for (int i = 0; i < height; i++){
45         for (int j = 0; j < width; j++){
46             if(shape[i][j] != '.' && board[row + i][col + j] != '.'){
47                 return false;
48             }
49         }
50     }
51     return true;
52 }
```

```
1 public void placePiece(Piece piece, int row, int col, char pieceType){
2     char[][]shape = piece.getBlock();
3     int height = piece.getHeight();
4     int width = piece.getWidth();
5
6     for (int i = 0; i < height; i++){
7         for (int j = 0; j < width; j++){
8             if (shape[i][j] != '.'){
9                 board[row + i][col + j] = pieceType;
10            }
11        }
12    }
13 }
14
15 public void removePiece(Piece piece, int row, int col){
16     char[][]shape = piece.getBlock();
17     int height = piece.getHeight();
18     int width = piece.getWidth();
19
20     for (int i = 0; i < height; i++){
21         for (int j = 0; j < width; j++){
22             if (shape[i][j] != '.'){
23                 board[row + i][col + j] = '.';
24            }
25        }
26    }
27 }
28
29 public boolean boardFull(){
30     for (int i = 0; i < N; i++){
31         for (int j = 0; j < M; j++){
32             if (board[i][j] == '.'){
33                 return false;
34            }
35        }
36    }
37     return true;
38 }
39
40 public void printBoardToFile(BufferedWriter writer) throws IOException{
41     for (int i = 0; i < N; i++){
42         for (int j = 0; j < M; j++){
43             writer.write(board[i][j]);
44         }
45         writer.newLine();
46     }
47     writer.newLine();
48 }
49 }
```

Pada bagian Board ini, terdapat warna yang sudah diassign pada setiap huruf dan terdapat fungsi-fungsi untuk memasang dan melepas piece. Selain itu, terdapat juga printBoardToFile untuk mengeprint Board ke file, selain itu juga ada GetN dan GetM untuk ukuran Board yang ingin dibuat.

Reader.Java

```
1  package program;
2
3  import java.io.*;
4  import java.nio.file.Path;
5  import java.nio.file.Paths;
6  import java.util.*;
7
8  public class Reader {
9
10     public static File readFile(String fileName) throws IOException {
11         Path path = Paths.get("test", fileName);
12         BufferedReader reader = new BufferedReader(new FileReader(path.toFile()));
13
14         String[] firstLine = reader.readLine().split(" ");
15         int N = Integer.parseInt(firstLine[0]);
16         int M = Integer.parseInt(firstLine[1]);
17         int P = Integer.parseInt(firstLine[2]);
18
19         String S = reader.readLine().trim();
20         List<String> puzzlePiece = new ArrayList<>();
21
22         Character last = null;
23         String shape = "";
24         int count = 0;
25         while (count < P) {
26             String line = reader.readLine();
27             if (line == null) {
28                 puzzlePiece.add(shape.substring(0, shape.length() - 1));
29                 break;
30             }
31
32             Character current = null;
33             for (char currentchar : line.toCharArray()) {
34                 if (currentchar >= 65 && currentchar <= 90) {
35                     current = currentchar;
36                     break;
37                 }
38             }
39
40             if (last == null || last.equals(current)) {
41                 shape += line + System.lineSeparator();
42             } else {
43                 puzzlePiece.add(shape.substring(0, shape.length() - 1));
44                 shape = line + System.lineSeparator();
45                 count += 1;
46             }
47
48             last = current;
49         }
50         reader.close();
51         return new File(N, M, P, S, puzzlePiece);
52     }
53 }
```

Merupakan file untuk membaca input yang diberikan oleh user.

File.java

```
1  package program;
2
3  import java.util.*;
4
5  public class File {
6      int N, M, P;
7      String S;
8      List<String> puzzlePiece;
9
10     public File(int N, int M, int P, String S, List<String> puzzlePiece)
11     {
12         this.N = N;
13         this.M = M;
14         this.P = P;
15         this.S = S;
16         this.puzzlePiece = puzzlePiece;
17     }
18 }
```

merupakan file untuk format file yang dimasukkan.

ResultSaver.java

```
1  package program;
2
3  import java.io.BufferedWriter;
4  import java.io.FileWriter;
5  import java.io.IOException;
6  import java.nio.file.Path;
7  import java.nio.file.Paths;
8
9  public class ResultSaver {
10     public static void saveResult(Board board, long startTime, long endTime, String fileName, int iterations) {
11         Path filePath = Paths.get("test", fileName);
12         try (BufferedWriter writer = new BufferedWriter(new FileWriter(filePath.toFile()))) {
13             writer.write("Bentuk Akhir Papan:\n");
14             board.printBoardToFile(writer);
15
16             double elapsedTimeInMillis = (endTime - startTime) * 1E-6;
17             writer.write("\nWaktu Pencarian: " + String.format("%.3f", elapsedTimeInMillis) + " ms\n");
18             writer.write("Jumlah Iterasi: " + iterations + "\n");
19
20             System.out.println("Hasil telah disimpan di " + filePath.toString());
21         } catch (IOException e) {
22             System.out.println("Terjadi kesalahan saat menyimpan hasil: " + e.getMessage());
23         }
24     }
25 }
26
```

Untuk menyimpan file ke txt

BAB IV

Hasil Eksperimen

Test Case 1

```
test > ≡ test1.txt
1 5 5 7
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
```

Solusi (di Terminal)

```
Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)
$ java -cp bin program.Main
Masukkan nama file test case (.txt): test1.txt
A G G G C
A A B C C
E E B B F
E E D F F
E D D F F

Waktu pencarian: 92,514 ms
Jumlah iterasi: 7166 kali
Apakah hasil ingin disimpan (y/n)? y
Nama file yang disimpan: hasil tes1
Hasil telah disimpan di test\hasil tes1.txt
```


Solusi (di Txt)

```
1  Bentuk Akhir Papan:
2  AGGGC
3  AABCC
4  EEBBF
5  EEDFF
6  EDDFF
7
8
9  Waktu Pencarian: 92,514 ms
10 Jumlah Iterasi: 7166
11
```

Test Case 2.

```
test > ≡ test2.txt
1  5 3 6
2  DEFAULT
3  X
4  XX
5  Y
6  YY
7  Z
8  ZZ
9  W
10 WWW
11 V
12 VV
13 U
14 UU
15
```

Solusi (di Terminal)

```
Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)
$ java -cp bin program.Main
Masukkan nama file test case (.txt): test2.txt
Tidak ada solusi yang ditemukan.
Waktu pencarian: 30,741 ms
Jumlah iterasi: 532 kali
Apakah hasil ingin disimpan (y/n)? y
Nama file yang disimpan: hasil tes2
Hasil telah disimpan di test\hasil tes2.txt
```

Solusi (di Txt)

```
1  Bentuk Akhir Papan:
2  ...
3  ...
4  ...
5  ...
6  ...
7
8
9  Waktu Pencarian: 30,741 ms
10 Jumlah Iterasi: 532
11
```

Test Case 3

```
test > ≡ test3.txt
1    6 4 6
2    DEFAULT
3    X
4    XX
5    Y
6    YY
7    Z
8    ZZZ
9    W
10   WW
11   V
12   VVV
13   U
14   UU
15
```

Solusi (di Terminal)

```
Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)
● $ java -cp bin program.Main
Masukkan nama file test case (.txt): test3.txt
Tidak ada solusi yang ditemukan.
Waktu pencarian: 3815,561 ms
Jumlah iterasi: 909816 kali
Apakah hasil ingin disimpan (y/n)? y
Nama file yang disimpan: hasil tes3
Hasil telah disimpan di test\hasil tes3.txt
```

Solusi (di Txt)

```
test > ≡ hasil tes3.txt
1  Bentuk Akhir Papan:
2  ....
3  ....
4  ....
5  ....
6  ....
7  ....
8
9
10 Waktu Pencarian: 3815,561 ms
11 Jumlah Iterasi: 909816
12
```

Test Case 4

```
test > ≡ test4.txt
1  3 3 9
2  DEFAULT
3  A
4  B
5  C
6  D
7  E
8  F
9  G
10 H
11 I
```

Solusi (di Terminal)

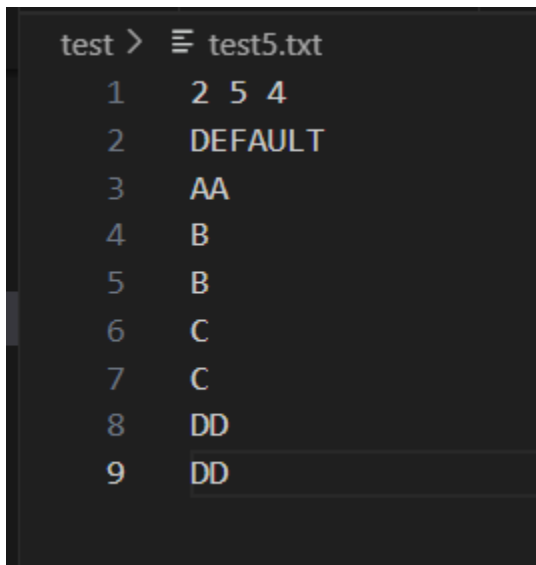
```
Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)
$ java -cp bin program.Main
Masukkan nama file test case (.txt): test4.txt
A B C
D E F
G H I

Waktu pencarian: 7,587 ms
Jumlah iterasi: 9 kali
Apakah hasil ingin disimpan (y/n)? y
Nama file yang disimpan: hasil tes4
Hasil telah disimpan di test\hasil tes4.txt
```

Solusi (di Txt)

```
test > ≡ hasil tes4.txt
1  Bentuk Akhir Papan:
2  ABC
3  DEF
4  GHI
5
6
7  Waktu Pencarian: 7,587 ms
8  Jumlah Iterasi: 9
9
```

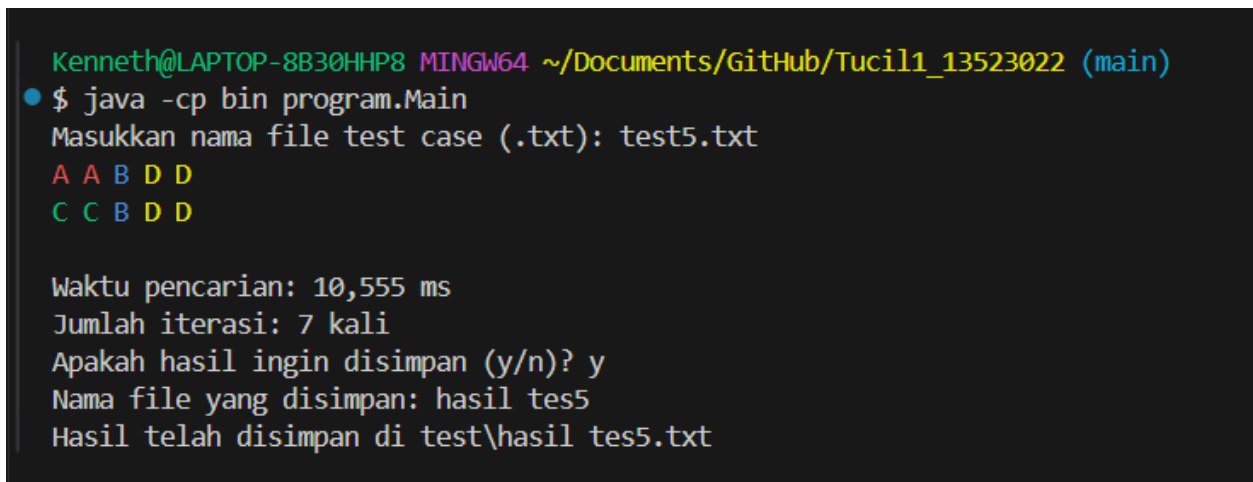
Test Case 5



A screenshot of a text editor window with a dark background. The title bar reads 'test > test5.txt'. The content is a table with two columns: an index from 1 to 9, and a corresponding value. The values are: 2 5 4, DEFAULT, AA, B, B, C, C, DD, and DD.

test >	test5.txt
1	2 5 4
2	DEFAULT
3	AA
4	B
5	B
6	C
7	C
8	DD
9	DD

Solusi (di Terminal)



A screenshot of a terminal window. The prompt is 'Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)'. The user enters '\$ java -cp bin program.Main'. The program prompts for a test case file name, and the user enters 'test5.txt'. The program outputs two lines of results: 'A A B D D' and 'C C B D D'. It then displays 'Waktu pencarian: 10,555 ms' and 'Jumlah iterasi: 7 kali'. It asks 'Apakah hasil ingin disimpan (y/n)?' and the user enters 'y'. The program then says 'Nama file yang disimpan: hasil tes5' and 'Hasil telah disimpan di test\hasil tes5.txt'.

```
Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)
• $ java -cp bin program.Main
Masukkan nama file test case (.txt): test5.txt
A A B D D
C C B D D

Waktu pencarian: 10,555 ms
Jumlah iterasi: 7 kali
Apakah hasil ingin disimpan (y/n)? y
Nama file yang disimpan: hasil tes5
Hasil telah disimpan di test\hasil tes5.txt
```

Solusi (di Txt)

```
test > ≡ hasil tes5.txt
1  Bentuk Akhir Papan:
2  AABDD
3  CCBDD
4
5
6  Waktu Pencarian: 10,555 ms
7  Jumlah Iterasi: 7
8
```

Test Case 6

```
test > ≡ test6.txt
1  5 4 5
2  DEFAULT
3  AAA
4  | A
5  B
6  BB
7  B
8  | C
9  CC
10 | D
11 | D
12 DD
13 | E
14 EEEE
```

Solusi (di Terminal)

```

Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)
● $ java -cp bin program.Main
Masukkan nama file test case (.txt): test6.txt
A A A E
B A E E
B B D E
B C D E
C C D D

Waktu pencarian: 15,990 ms
Jumlah iterasi: 33 kali
Apakah hasil ingin disimpan (y/n)? y
Nama file yang disimpan: hasil tes6
Hasil telah disimpan di test\hasil tes6.txt

```

Solusi (di Txt)

```

test 7 = hasil tes6.txt
1  Bentuk Akhir Papan:
2  AAAE
3  BAEE
4  BBDE
5  BCDE
6  CCDD
7
8
9  Waktu Pencarian: 15,990 ms
10 Jumlah Iterasi: 33
11

```

Test Case 7


```
test > ≡ test7.txt
1      2 2 3
2      DEFAULT
3      A
4      BB
5      C
```

Solusi (di Terminal)

```
Kenneth@LAPTOP-8B30HHP8 MINGW64 ~/Documents/GitHub/Tucil1_13523022 (main)
• $ java -cp bin program.Main
Masukkan nama file test case (.txt): test7.txt
A C
B B

Waktu pencarian: 10,486 ms
Jumlah iterasi: 3 kali
Apakah hasil ingin disimpan (y/n)? y
Nama file yang disimpan: hasil tes7
Hasil telah disimpan di test\hasil tes7.txt
```

Solusi (di Txt)

```
test > ≡ hasil tes7.txt
1      Bentuk Akhir Papan:
2      AC
3      BB
4
5
6      Waktu Pencarian: 10,486 ms
7      Jumlah Iterasi: 3
8
```

LAMPIRAN

•

No	Poin	Ya	Tidak
1	Program berhasil dikompilasi tanpa kesalahan	v	
2	Program berhasil dijalankan	v	
3	Solusi yang diberikan program benar dan mematuhi aturan permainan	v	
4	Program dapat membaca masukan berkas .txt serta menyimpan solusi dalam berkas .txt	v	
5	Program memiliki <i>Graphical User Interface</i> (GUI)		v
6	Program dapat menyimpan solusi dalam bentuk file gambar		v
7	Program dapat menyelesaikan kasus konfigurasi <i>custom</i>		v
8	Program dapat menyelesaikan kasus konfigurasi Piramida (3D)		v
9	Program dibuat oleh saya sendiri	v	