Tugas Kecil 3 IF2211 Strategi Algoritma

Penyelesaian Persoalan 15-*Puzzle* dengan Algoritma *Branch and Bound*

Dibuat oleh:

Shadiq Harwiz 13520038

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2022

A. Cara Kerja Program *Branch and Bound* dalam Menyelesaikan Persoalan

Langkah-langkah yang dilakukan dalam menyelesaikan persoalan 15-*puzzle* yang telah berhasil memenuhi syarat *reachable goal* dengan memanfaatkan strategi algorima *branch and bound* adalah sebagai berikut.

1. Menyimpan terlebih dahulu susunan awal dalam bentuk matriks 4x4
2. Mencari posisi ubin kosong pada susunan awal dan mengembalikan nilai posisinya (dalam bentuk x,y yang direpresentasikan sebagai array of integer)
3. Memanggil fungsi using_branch_bound
4. Menyimpan susunan akhir dalam bentuk matriks
5. Membuat prioqueue dengan memprioritaskan nilai cost terkecil
6. Menyimpan susunan awal yang telah dalam bentuk matriks dan level node ke-0 $[f(p)]$ ke dalam kelas node
7. Menghitung dan menyimpan ubin tidak kosong $[\hat{g}(p)]$ pada susunan awal ke dalam kelas node
8. Melakukan *enqueue* kelas node ke prioqueue
9. Melakukan iterasi hingga prioquque kosong
10. *Dequque* kelas node pada prioqueue
11. Mengecek apakah nilai cost pada kelas node yang telah di-*dequeue* bernilai 0 atau tidak
12. Jika tidak, akan dilakukan percobaan perpindahan ke atas, ke kanan, ke bawah, dan ke kiri.
13. Pada masing-masing perpindahan akan dilakukan pengecekan apakah setelah berpindah, posisi ubin kosong tidak melewati baris tertinggi dan/atau kolom tertinggi pada matriks 4x4 serta apakah bentuk matriks setelah ubin kosong berpindah tidak seperti matriks pada dua level sebelumnya (dalam konsep *node* pada *tree*).
14. Jika terpenuhi seluruhnya, akan dilakukan penyimpanan ke dalam kelas node dengan level node telah naik menjadi ke-1
15. Melakukan kembali proses ke-7 dan 8
16. Jika iya, akan mengirim output pada layar urutan matriks dari posisi awal ke posisi akhir dan program berakhir
17. Melakukan kembali proses ke-9 hingga 16

B. Source Program

Dalam menyelesaikan persoalan 15-puzzle dengan memanfaatkan strategi algoritma *branch and bound*, penulis membuat program dalam bahasa java.

1. Main.py

```
import java.util.*;
import java.io.*;

public class Main {

    public static void main(String[] args) throws FileNotFoundException {

        int x;
        int y;
        int[] posisi = new int[2];
        int[][] initial = new int [4][4];
        Menu Menu = new Menu();
        Scanner inputFile;

        int menu = Menu.menu();
        if ( menu == 2 ) {
            inputFile = Menu.menu_2();

        } else {
            inputFile = Menu.menu_1();
        }
        initial = Menu.membentuk_matriks_posisi_awal(inputFile);
        posisi = Menu.mencari_posisi_0(initial);
        x = posisi[0];
        y = posisi[1];

        Puzzle puzzle = new Puzzle();
        if (puzzle.cek_dapat_diselesaikan(initial)) {
            System.out.println("Status Tujuan Dapat Dicapai");
            System.out.println("Berikut urutan matriks dari posisi awal ke
posisi akhir");
            puzzle.using_branch_bound(initial, x, y);
        }
        else {
            System.out.println("Status Tidak Tujuan Dapat Dicapai");
        }
        System.out.println("Program telah Berakhir");
    }
}
```

2. Menu.py

```
import java.util.*;
import java.io.*;

public class Menu {

    Scanner input = new Scanner(System.in);
```

```
    int[][] membentuk_matriks_posisi_awal(Scanner inputFile){
        int[][] initial = new int [4][4];
        for ( int i = 0 ; i < 4 ; i++ ){
            for ( int j = 0 ; j < 4 ; j++ ){
                if( inputFile.hasNextInt() ) {
                    initial[i][j]= inputFile.nextInt();
                }
            }
        }
        return initial;
    }

    public int[] mencari_posisi_0(int[][] matriks){
        int[] posisi = new int [2];
        for ( int i = 0 ; i < 4 ; i++ ){
            for ( int j = 0 ; j < 4 ; j++ ){
                if(matriks[i][j] == 0){
                    posisi[0] = i;
                    posisi[1] = j;
                    return(posisi);
                }
            }
        }
        return posisi;
    }

    public boolean cek_file_tersedia(String namaFile){
        File folder1 = new File("../test");
        String contents[] = folder1.list();
        boolean isFound = false;
        int i = 0;
        while (!isFound && i < contents.length) {
            if ( contents[i].equals(namaFile) ){
                isFound = true;
            } else {
                i++;
            }
        }
        return isFound;
    }

    public int menu() {
        input = new Scanner(System.in);
        System.out.println("Ketik 1 jika posisi awal 15-puzzle dibangkitkan
secara acak atau keti
        int inputInt = input.nextInt();
        while ( inputInt != 1 && inputInt != 2 ) {
            System.out.println("Input salah!");
            inputInt = input.nextInt();
        }
        return inputInt;
    }
```

```
    public Scanner menu_1() throws FileNotFoundException{
        Random randNum = new Random();
        File folder1 = new File("../test");
        String contents[] = folder1.list();
        int num = randNum.nextInt(contents.length);
        Scanner inputFile = new Scanner (new File("../test/" + contents[num]));
        return inputFile;
    }

    public Scanner menu_2() throws FileNotFoundException{
        System.out.print("Masukkan nama file yang berada pada folder test (
contoh : test1.txt ) : ");
        input = new Scanner(System.in);
        String inputString = input.nextLine();

        while ( !cek_file_tersedia(inputString) ){
            System.out.print("Nama file tidak ditemukan pada folder test,
masukkan nama file : ");
            inputString = input.nextLine();
        }
        Scanner inputFile = new Scanner (new File("../test/" + inputString));

        return inputFile;
    }
}
```

3. Node.java

```
public class Node {

       // Node Induk
       public Node parent;

       // Informasi perpindahan ( atas, kanan, bawah, atau kiri )
       public String move;

       // Matrix ( puzzle )
       public int[][] matrix;

       // Posisi "0" pada matrix puzzle
       public int x, y;

       // Jumlah ubin tidak kosong yang tidak terdapat pada susunan akhir {g(P)}
       public int ubin_not_empty;

       // Panjang lintasan dari simpul akar ke P {f(P)} dengan memanfaatkan
tingkat level pada tree
       public int level;

       // Constructor dengan parameter
       public Node(int[][] matrix, int x, int y, int newX, int newY, int level,
String move, Node parent) {
              this.parent = parent;

              this.matrix = new int[matrix.length][];
              for (int i = 0; i < matrix.length; i++) {
                     this.matrix[i] = matrix[i].clone();
              }
```

```
            // Swap value
            this.matrix[x][y]       = this.matrix[x][y] +
this.matrix[newX][newY];
            this.matrix[newX][newY] = this.matrix[x][y] -
this.matrix[newX][newY];
            this.matrix[x][y]       = this.matrix[x][y] -
this.matrix[newX][newY];

            this.level = level;
            this.x = newX;
            this.y = newY;

            this.move = move;
      }

      // Fungsi untuk menyimpan ubin_not_empty
      public void inisiasi_ubin_not_empty(int ubin_not_empty){
            this.ubin_not_empty = ubin_not_empty;
      }
}
```

4. Puzzle.java

```
import java.util.*;

public class Puzzle {

      // Dimensi dari matrix puzzle
      public int dimensi = 4;

      // Pergerakan disimpan dalam array yang terdiri dari baris dan kolom
      // Pergerakan ke atas = baris[0] kolom[0]
      // Pergerakan ke kanan = baris[1] kolom[1]
      // Pergerakan ke bawah = baris[2] kolom[2]
      // Pergerakan ke kiri = baris[3] kolom[3]
      int[] baris = { -1, 0, 1, 0 };
      int[] kolom = { 0, 1, 0, -1 };

      // Memberi output isi matrix
      public void print_Matrix(int[][] matrix) {
            System.out.println("====================");
            for (int i = 0; i < matrix.length; i++) {
                  for (int j = 0; j < matrix.length; j++) {
                        if ( j == 0 ) {
                              System.out.print("| ");
                        }
                        if ( matrix[i][j] != 0 ) {
                              if ( matrix[i][j] < 10 ) {
                                    System.out.print(matrix[i][j] + "  | ");
                              } else {
                                    System.out.print(matrix[i][j] + " | ");
                              }
                        } else {
                              System.out.print("   | ");
                        }
                  }
                  System.out.println();
```

```
                if ( i != matrix.length - 1) {
                        System.out.println("---------------------");
                }
            }
            System.out.println("=====================");
        }

        // Cetak jalur dari simpul akar ke simpul tujuan
        public void print_jalur(Node root) {
            if (root == null) {
                return;
            }
            print_jalur(root.parent);
            print_Matrix(root.matrix);
            System.out.println();
        }

        // Menghitung dan mengembalikan jumlah ubin yang tidak kosong yang tidak
terdapat pada susunan akhir -> g(P)
        public int menghitung_ubin_tidak_kosong(int[][] initial, int[][] goal) {
            int count = 0;
            for (int i = 0; i < initial.length; i++) {
                for (int j = 0; j < initial.length; j++) {
                    if (initial[i][j] != 0 && initial[i][j] != goal[i][j])
{
                        count++;
                    }
                }
            }
            return count;
        }

        // Memeriksa apakah setelah berpindah, posisi (x, y) masih koordinat yang
valid
        public boolean cek_perpindahan_valid(int x, int y) {
            return (x >= 0 && x < dimensi && y >= 0 && y < dimensi);
        }

        // Menghitung reachable goal untuk menguji apakah layak atau tidak puzzle
untuk dikerjakan/disusun dengan memanfaatkan KURANG(i) + X
        public boolean cek_dapat_diselesaikan(int[][] matrix) {
            System.out.println("Berikut Puzzle Awal");
            print_Matrix(matrix);
            System.out.println();

            int count = 0;
            Integer[] fungsiKurang = new Integer[matrix.length*matrix.length];
            int k = 0;
            // mencari nilai X ( 0 atau 1 )
            for (int i = 0; i < matrix.length; i++) {
                for (int j = 0; j < matrix.length; j++) {
                    fungsiKurang[k] = matrix[i][j];
                    k++;
                    if (matrix[i][j] == 0) {
                        if ( (i+j) % 2 == 1) {
                            count++;
                        }
                    }
                }
            }
```

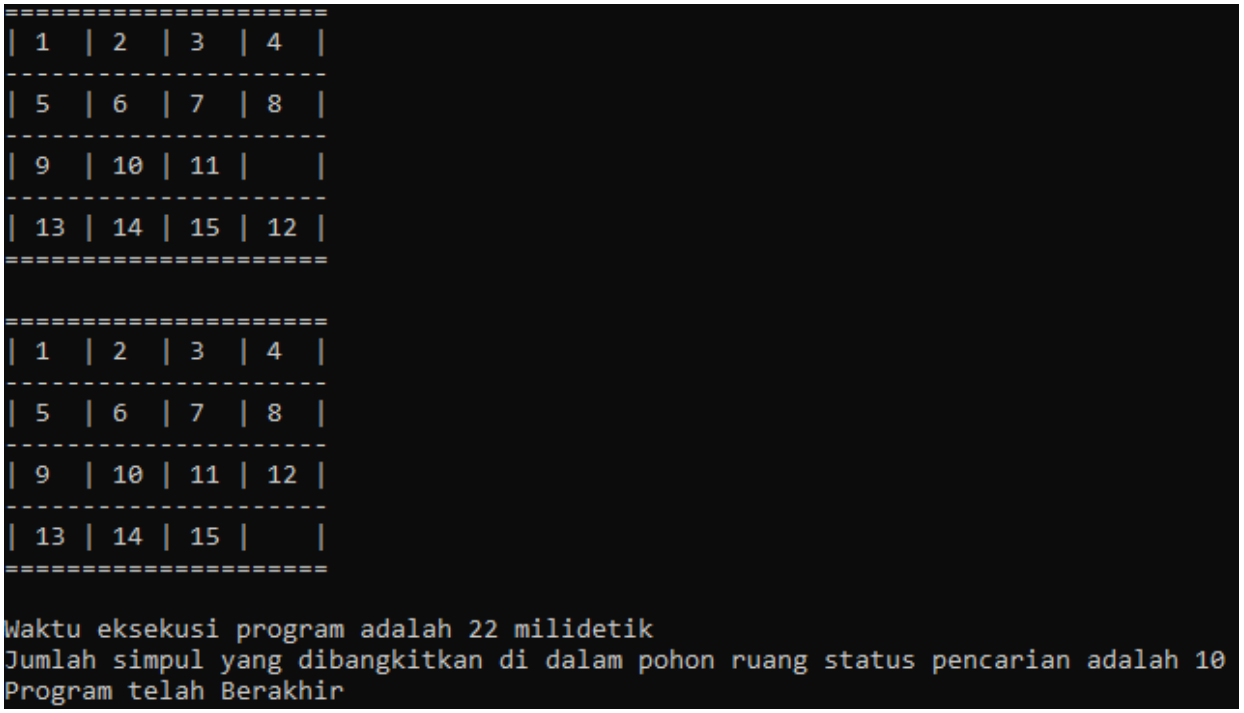
```
            Integer[] nilaiFungsiKurang = new
Integer[matrix.length*matrix.length];
            int count_temp;

            // menghitung Kurang(i)
            for (int i = 0; i < fungsiKurang.length; i++) {
                  count_temp = 0;
                  for (int j = i + 1; j < fungsiKurang.length; j++) {
                        if ( fungsiKurang[i] == 0 ) {
                              count_temp++;
                        } else {
                              if ( fungsiKurang[i] > fungsiKurang[j] &&
fungsiKurang[j] != 0) {
                                    count_temp++;
                              }
                        }
                  }
                  if ( fungsiKurang[i] != 0 ) {
                        nilaiFungsiKurang[fungsiKurang[i] - 1] = count_temp;
                  } else {
                        nilaiFungsiKurang[15] = count_temp;
                  }
                  count = count + count_temp;
            }
            for ( int i = 0; i < nilaiFungsiKurang.length; i++) {
                  System.out.println("Nilai KURANG(" + (i+1) + ") = " +
nilaiFungsiKurang[i] );
            }
            System.out.println("Nilai dari KURANG(i) + X adalah " + count);
            System.out.println();
            return count % 2 == 0;
      }

      // Proses pencarian solusi puzzle matriks dengan menerapkan
algoritma branch and bound
      public void using_branch_bound(int[][] initial, int x, int y) {
            long start = System.currentTimeMillis();
            int[][] goal= { {1, 2, 3, 4},
                              {5, 6, 7, 8},
                              {9, 10, 11, 12},
                              {13, 14, 15, 0} };
            PriorityQueue<Node> pq = new PriorityQueue<Node>(999999, (a, b) ->
(a.ubin_not_empty + a.level) - (b.ubin_not_empty + b.level)); //buat prioqueue
dgn memprioritaskan cost terkecil
            Node root = new Node(initial, x, y, x, y, 0, "Initial", null);
            root.inisiasi_ubin_not_empty(menghitung_ubin_tidak_kosong(initial,
goal));
            pq.add(root); //enquque pq
            int j = 0;
            while (!pq.isEmpty()) {
                  Node min = pq.poll(); //dequque pq
                  j++;
                  if (min.ubin_not_empty == 0) {
                        print_jalur(min);
                        long end = System.currentTimeMillis();
                        System.out.println("Waktu eksekusi program adalah "+
(end - start) + " milidetik");
                        System.out.println("Jumlah simpul yang dibangkitkan di
dalam pohon ruang status pencarian adalah " + (pq.size() + j));
                        return;
```

```
                    for (int i = 0; i < 4; i++) {
                   if (cek_perpindahan_valid(min.x + baris[i], min.y + kolom[i]))
{
                                    // Pergerakan ke atas, kanan, bawah, kiri
                                    String move;
                                    if ( i == 0 && min.move != "Bawah" ){
                                          move = "Atas";
                                          Node child = new Node(min.matrix, min.x,
min.y, min.x + baris[i], min.y + kolom[i], min.level + 1, move, min);
                                          child. inisiasi_ubin_not_empty
(menghitung_ubin_tidak_kosong(child.matrix, goal));
                                          pq.add(child);
                                    } else if ( i == 1 && min.move != "Kiri" ){
                                          move = "Kanan";
                                          Node child = new Node(min.matrix, min.x,
min.y, min.x + baris[i], min.y + kolom[i], min.level + 1, move, min);
                                          child. inisiasi_ubin_not_empty
(menghitung_ubin_tidak_kosong(child.matrix, goal));
                                          pq.add(child);
                                    } else if ( i == 2 && min.move != "Atas" ){
                                          move = "Bawah";
                                          Node child = new Node(min.matrix, min.x,
min.y, min.x + baris[i], min.y + kolom[i], min.level + 1, move, min);
                                          child. inisiasi_ubin_not_empty
(menghitung_ubin_tidak_kosong(child.matrix, goal));
                                          pq.add(child);
                                    } else if ( i == 3 && min.move != "Kanan" ){
                                          move = "Kiri";
                                          Node child = new Node(min.matrix, min.x,
min.y, min.x + baris[i], min.y + kolom[i], min.level + 1, move, min);
                                          child. inisiasi_ubin_not_empty
(menghitung_ubin_tidak_kosong(child.matrix, goal));
                                          pq.add(child);
                                    }
                        }
                     }
                  }
      }
}
```

C. Screenshoot Input dan Output Program Data Uji

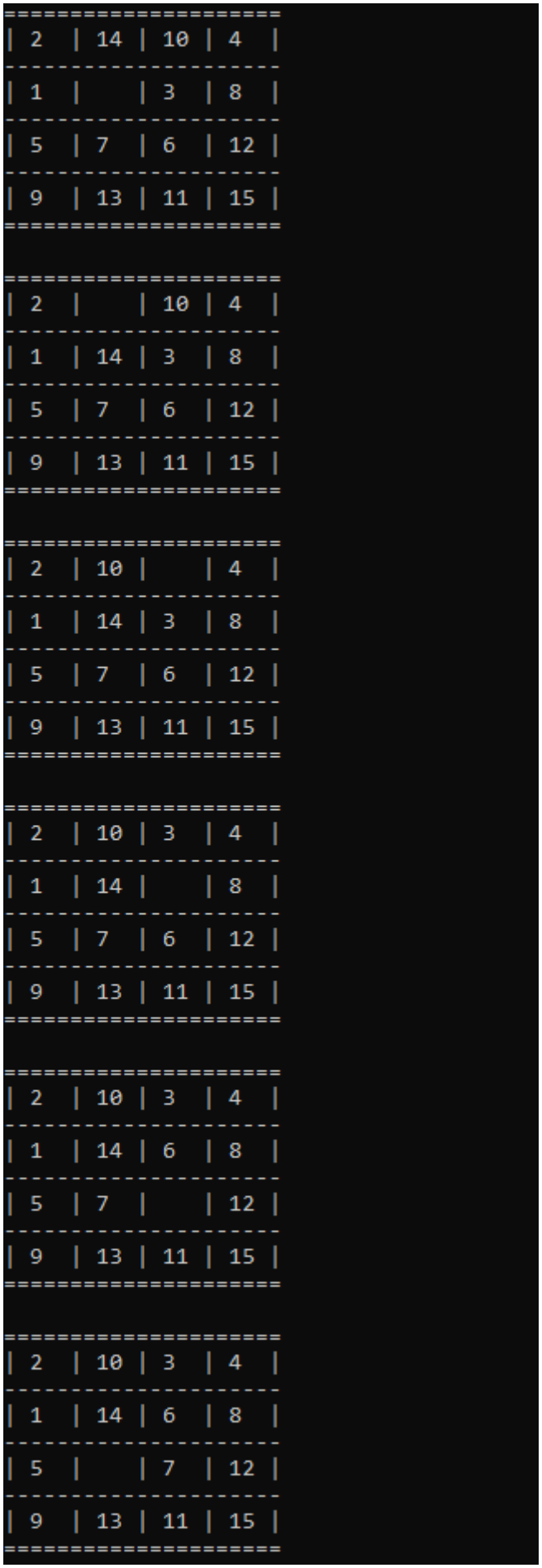
1. Test1.txt

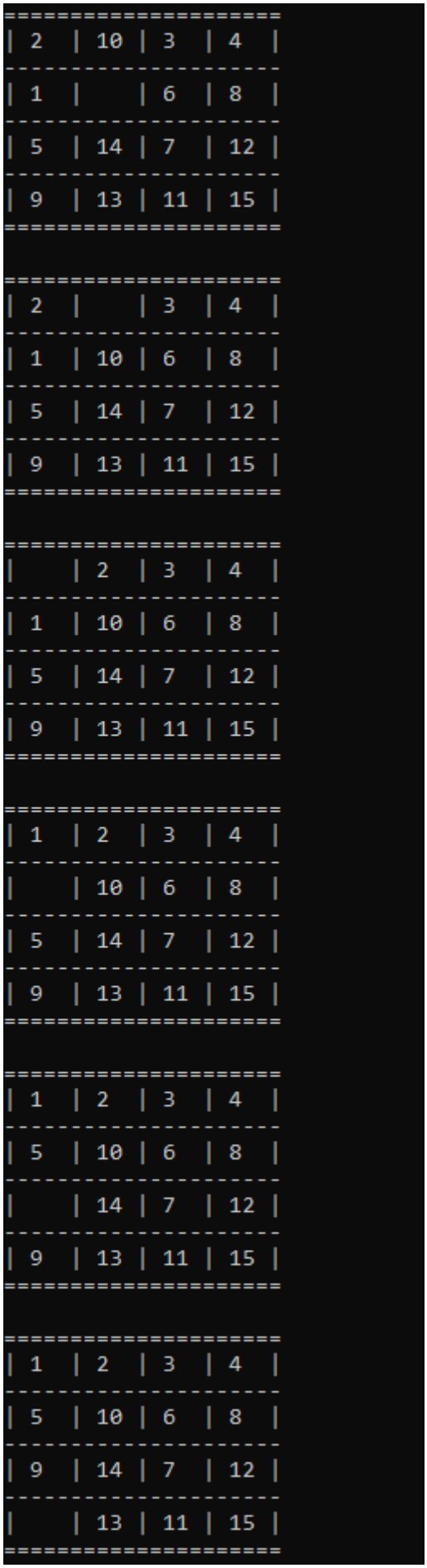
```
E:\Semester 4\IF2211 - Strategi Algoritma\Tucil\Tucil 3\bin>java Main
Ketik 1 jika posisi awal 15-puzzle dibangkitkan secara acak atau ketik 2 menggunakan file teks
2
Masukkan nama file yang berada pada folder test ( contoh : test1.txt ) : test1.txt
Berikut Puzzle Awal
=====================
| 1  | 2  | 3  | 4  |
---------------------
| 5  | 6  |    | 8  |
---------------------
| 9  | 10 | 7  | 11 |
---------------------
| 13 | 14 | 15 | 12 |
=====================

Nilai KURANG(1) = 0
Nilai KURANG(2) = 0
Nilai KURANG(3) = 0
Nilai KURANG(4) = 0
Nilai KURANG(5) = 0
Nilai KURANG(6) = 0
Nilai KURANG(7) = 0
Nilai KURANG(8) = 1
Nilai KURANG(9) = 1
Nilai KURANG(10) = 1
Nilai KURANG(11) = 0
Nilai KURANG(12) = 0
Nilai KURANG(13) = 1
Nilai KURANG(14) = 1
Nilai KURANG(15) = 1
Nilai KURANG(16) = 9
Nilai dari KURANG(i) + X adalah 16

Status Tujuan Dapat Dicapai
Berikut urutan matriks dari posisi awal ke posisi akhir
=====================
| 1  | 2  | 3  | 4  |
---------------------
| 5  | 6  |    | 8  |
---------------------
| 9  | 10 | 7  | 11 |
---------------------
| 13 | 14 | 15 | 12 |
=====================


=====================
| 1  | 2  | 3  | 4  |
---------------------
| 5  | 6  | 7  | 8  |
---------------------
| 9  | 10 |    | 11 |
---------------------
| 13 | 14 | 15 | 12 |
=====================
```

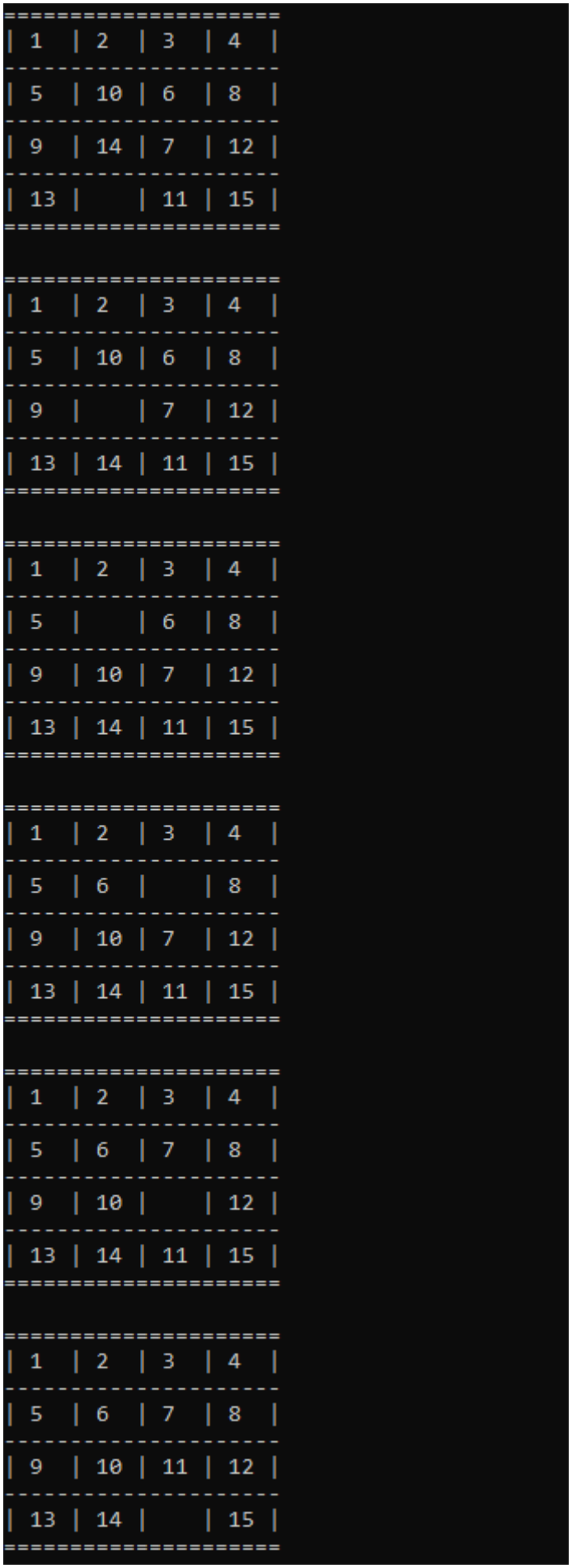

1	2	3	4
5	6	7	8
9	10	11	
13	14	15	12
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	
Waktu eksekusi program adalah 22 milidetik
Jumlah simpul yang dibangkitkan di dalam pohon ruang status pencarian adalah 10
Program telah Berakhir

2. Test3.txt

```
E:\Semester 4\IF2211 - Strategi Algoritma\Tucil\Tucil 3\bin>java Main
Ketik 1 jika posisi awal 15-puzzle dibangkitkan secara acak atau ketik 2 menggunakan file teks
2
Masukkan nama file yang berada pada folder test ( contoh : test1.txt ) : test3.txt
Berikut Puzzle Awal
====================
| 2  | 14 | 10 | 4  |
--------------------
| 1  | 7  | 3  | 8  |
--------------------
|    | 5  | 6  | 12 |
--------------------
| 9  | 13 | 11 | 15 |
====================

Nilai KURANG(1) = 0
Nilai KURANG(2) = 1
Nilai KURANG(3) = 0
Nilai KURANG(4) = 2
Nilai KURANG(5) = 0
Nilai KURANG(6) = 0
Nilai KURANG(7) = 3
Nilai KURANG(8) = 2
Nilai KURANG(9) = 0
Nilai KURANG(10) = 8
Nilai KURANG(11) = 0
Nilai KURANG(12) = 2
Nilai KURANG(13) = 1
Nilai KURANG(14) = 12
Nilai KURANG(15) = 0
Nilai KURANG(16) = 7
Nilai dari KURANG(i) + X adalah 38

Status Tujuan Dapat Dicapai
Berikut urutan matriks dari posisi awal ke posisi akhir
====================
| 2  | 14 | 10 | 4  |
--------------------
| 1  | 7  | 3  | 8  |
--------------------
|    | 5  | 6  | 12 |
--------------------
| 9  | 13 | 11 | 15 |
====================

====================
| 2  | 14 | 10 | 4  |
--------------------
| 1  | 7  | 3  | 8  |
--------------------
| 5  |    | 6  | 12 |
--------------------
| 9  | 13 | 11 | 15 |
====================
```


2	14	10	4
1		3	8
5	7	6	12
9	13	11	15
2		10	4
1	14	3	8
5	7	6	12
9	13	11	15
2	10		4
1	14	3	8
5	7	6	12
9	13	11	15
2	10	3	4
1	14		8
5	7	6	12
9	13	11	15
2	10	3	4
1	14	6	8
5	7		12
9	13	11	15
2	10	3	4
1	14	6	8
5		7	12
9	13	11	15

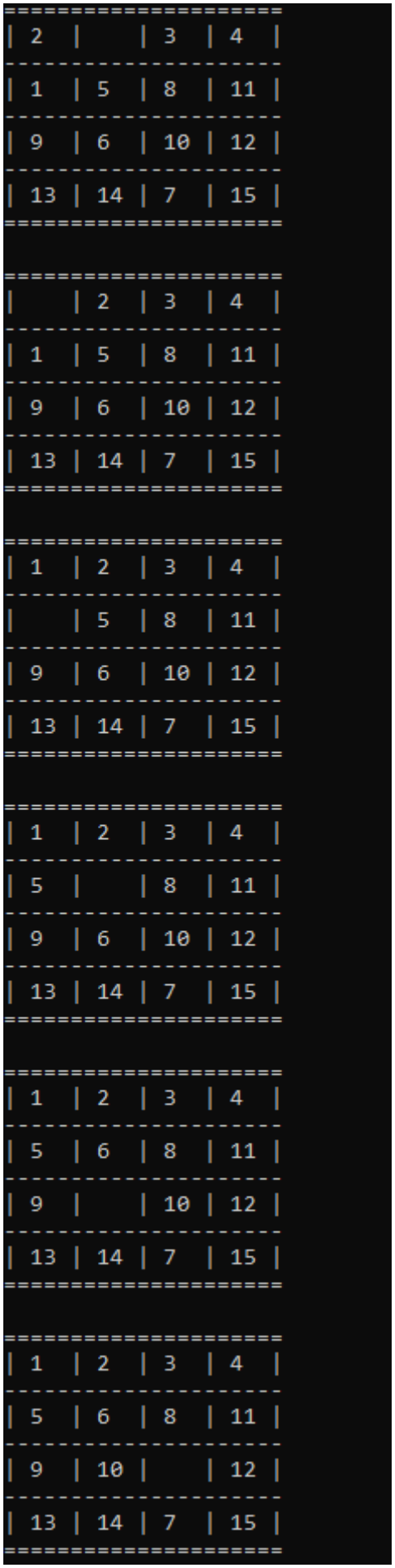

2	10	3	4
1		6	8
5	14	7	12
9	13	11	15
2		3	4
1	10	6	8
5	14	7	12
9	13	11	15
	2	3	4
1	10	6	8
5	14	7	12
9	13	11	15
1	2	3	4
	10	6	8
5	14	7	12
9	13	11	15
1	2	3	4
5	10	6	8
	14	7	12
9	13	11	15
1	2	3	4
5	10	6	8
9	14	7	12
	13	11	15

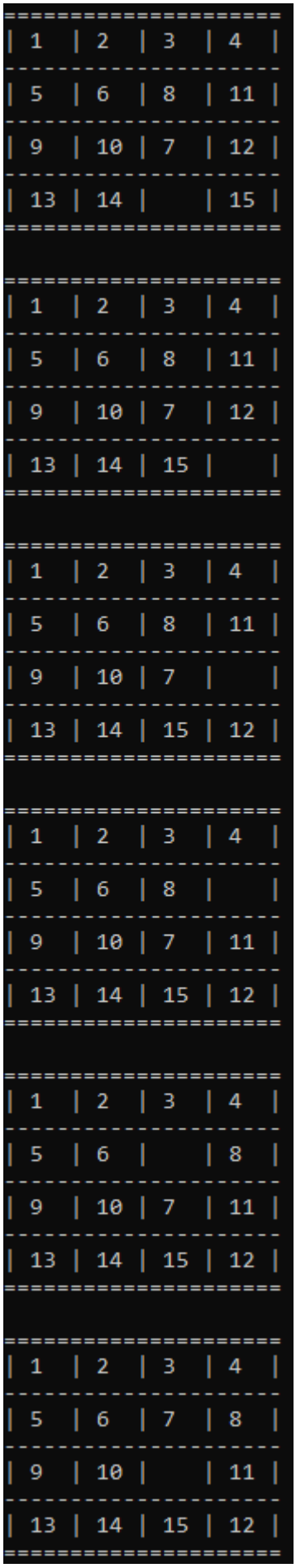

1	2	3	4
5	10	6	8
9	14	7	12
13		11	15
1	2	3	4
5	10	6	8
9		7	12
13	14	11	15
1	2	3	4
5		6	8
9	10	7	12
13	14	11	15
1	2	3	4
5	6		8
9	10	7	12
13	14	11	15
1	2	3	4
5	6	7	8
9	10		12
13	14	11	15
1	2	3	4
5	6	7	8
9	10	11	12
13	14		15

```
=====================
| 1   | 2   | 3   | 4   |
---------------------
| 5   | 6   | 7   | 8   |
---------------------
| 9   | 10  | 11  | 12  |
---------------------
| 13  | 14  | 15  |     |
=====================

Waktu eksekusi program adalah 390 milidetik
Jumlah simpul yang dibangkitkan di dalam pohon ruang status pencarian adalah 5883
Program telah Berakhir
```

3. Test6.txt

```
E:\Semester 4\IF2211 - Strategi Algoritma\Tucil\Tucil 3\bin>java Main
Ketik 1 jika posisi awal 15-puzzle dibangkitkan secara acak atau ketik 2 menggunakan file teks
2
Masukkan nama file yang berada pada folder test ( contoh : test1.txt ) : test6.txt
Berikut Puzzle Awal
====================
| 2  | 3  | 4  |    |
--------------------
| 1  | 5  | 8  | 11 |
--------------------
| 9  | 6  | 10 | 12 |
--------------------
| 13 | 14 | 7  | 15 |
====================

Nilai KURANG(1) = 0
Nilai KURANG(2) = 1
Nilai KURANG(3) = 1
Nilai KURANG(4) = 1
Nilai KURANG(5) = 0
Nilai KURANG(6) = 0
Nilai KURANG(7) = 0
Nilai KURANG(8) = 2
Nilai KURANG(9) = 2
Nilai KURANG(10) = 1
Nilai KURANG(11) = 4
Nilai KURANG(12) = 1
Nilai KURANG(13) = 1
Nilai KURANG(14) = 1
Nilai KURANG(15) = 0
Nilai KURANG(16) = 12
Nilai dari KURANG(i) + X adalah 28

Status Tujuan Dapat Dicapai
Berikut urutan matriks dari posisi awal ke posisi akhir
====================
| 2  | 3  | 4  |    |
--------------------
| 1  | 5  | 8  | 11 |
--------------------
| 9  | 6  | 10 | 12 |
--------------------
| 13 | 14 | 7  | 15 |
====================

====================
| 2  | 3  |    | 4  |
--------------------
| 1  | 5  | 8  | 11 |
--------------------
| 9  | 6  | 10 | 12 |
--------------------
| 13 | 14 | 7  | 15 |
====================
```


2		3	4
1	5	8	11
9	6	10	12
13	14	7	15
	2	3	4
1	5	8	11
9	6	10	12
13	14	7	15
1	2	3	4
	5	8	11
9	6	10	12
13	14	7	15
1	2	3	4
5		8	11
9	6	10	12
13	14	7	15
1	2	3	4
5	6	8	11
9		10	12
13	14	7	15
1	2	3	4
5	6	8	11
9	10		12
13	14	7	15


1	2	3	4
5	6	8	11
9	10	7	12
13	14		15
1	2	3	4
5	6	8	11
9	10	7	12
13	14	15	
1	2	3	4
5	6	8	11
9	10	7	
13	14	15	12
1	2	3	4
5	6	8	
9	10	7	11
13	14	15	12
1	2	3	4
5	6		8
9	10	7	11
13	14	15	12
1	2	3	4
5	6	7	8
9	10		11
13	14	15	12

```
=====================
| 1  | 2  | 3  | 4  |
---------------------
| 5  | 6  | 7  | 8  |
---------------------
| 9  | 10 | 11 |    |
---------------------
| 13 | 14 | 15 | 12 |
=====================

=====================
| 1  | 2  | 3  | 4  |
---------------------
| 5  | 6  | 7  | 8  |
---------------------
| 9  | 10 | 11 | 12 |
---------------------
| 13 | 14 | 15 |    |
=====================

Waktu eksekusi program adalah 301 milidetik
Jumlah simpul yang dibangkitkan di dalam pohon ruang status pencarian adalah 271
Program telah Berakhir
```

4. Test10.txt

```
E:\Semester 4\IF2211 - Strategi Algoritma\Tucil\Tucil 3\bin>java Main
Ketik 1 jika posisi awal 15-puzzle dibangkitkan secara acak atau ketik 2 menggunakan file teks
2
Masukkan nama file yang berada pada folder test ( contoh : test1.txt ) : test10.txt
Berikut Puzzle Awal
=====================
| 2  | 7  | 4  | 14 |
---------------------
| 15 | 5  | 11 |    |
---------------------
| 13 | 3  | 12 | 6  |
---------------------
| 1  | 10 | 9  | 8  |
=====================

Nilai KURANG(1) = 0
Nilai KURANG(2) = 1
Nilai KURANG(3) = 1
Nilai KURANG(4) = 2
Nilai KURANG(5) = 2
Nilai KURANG(6) = 1
Nilai KURANG(7) = 5
Nilai KURANG(8) = 0
Nilai KURANG(9) = 1
Nilai KURANG(10) = 2
Nilai KURANG(11) = 6
Nilai KURANG(12) = 5
Nilai KURANG(13) = 7
Nilai KURANG(14) = 10
Nilai KURANG(15) = 10
Nilai KURANG(16) = 8
Nilai dari KURANG(i) + X adalah 61

Status Tidak Tujuan Dapat Dicapai
Program telah Berakhir
```

5. Test11.txt

```
E:\Semester 4\IF2211 - Strategi Algoritma\Tucil\Tucil 3\bin>java Main
Ketik 1 jika posisi awal 15-puzzle dibangkitkan secara acak atau ketik 2 menggunakan file teks
2
Masukkan nama file yang berada pada folder test ( contoh : test1.txt ) : test11.txt
Berikut Puzzle Awal
====================
| 2  | 12 | 13 | 10 |
--------------------
| 9  | 15 | 6  | 8  |
--------------------
| 7  | 4  | 1  |    |
--------------------
| 11 | 3  | 5  | 14 |
====================

Nilai KURANG(1) = 0
Nilai KURANG(2) = 1
Nilai KURANG(3) = 0
Nilai KURANG(4) = 2
Nilai KURANG(5) = 0
Nilai KURANG(6) = 4
Nilai KURANG(7) = 4
Nilai KURANG(8) = 5
Nilai KURANG(9) = 7
Nilai KURANG(10) = 8
Nilai KURANG(11) = 2
Nilai KURANG(12) = 10
Nilai KURANG(13) = 10
Nilai KURANG(14) = 0
Nilai KURANG(15) = 9
Nilai KURANG(16) = 4
Nilai dari KURANG(i) + X adalah 67

Status Tidak Tujuan Dapat Dicapai
Program telah Berakhir
```

D. Berkas Teks

1. Test1.txt

```
1 2 3 4
5 6 0 8
9 10 7 11
13 14 15 12
```

2. Test2.txt

```
5 1 8 12
0 9 2 3
13 10 4 7
14 11 6 15
```

3. Test6.txt

```
2 14 10 4
1 7 3 8
0 5 6 12
9 13 11 15
```

4. Test4.txt

```
2 3 11 4
1 6 10 8
9 5 12 15
13 14 0 7
```

5. Test5.txt

```
1 6 2 4
5 0 3 8
9 7 15 11
13 14 10 12
```

6. Test6.txt

```
2 3 4 0
1 5 8 11
9 6 10 12
13 14 7 15
```

7. Test7.txt

```
6 0 8 5
4 7 10 3
1 9 2 13
15 12 11 14
```

8. Test8.txt

```
1 3 15 4
10 2 16 11
5 14 8 6
9 7 12 13
```

9. Test9.txt

```
2 3 4 8
1 5 11 0
9 6 10 12
13 14 7 15
```

10. Test10.txt

```
2 7 4 14
15 5 11 0
13 3 12 6
1 10 9 8
```

11. Test11.txt

```
2 12 13 10
9 15 6 8
7 4 1 0
11 3 5 14
```

E. Alamat Kode Program

Program dapat diunduh dari alamat berikut:

https://github.com/shdiqq/Tucil3_13530038.git

Atau dapat melalui

https://drive.google.com/drive/folders/1TbmY0ZmUdhmi4sUBVqf1YV6WLwchm2bX?usp=sharing

F. Tabel Penilaian

Poin	Ya	Tidak
1. Program berhasil dikompilasi	☒	☐
2. Program berhasil running.	☒	☐
3. Program dapat menerima input dan menuliskan output	☒	☐
4. Luaran sudah benar untuk semua data uji	☒	☐
5. Bonus dibuat	☐	☒