Laporan Tugas Kecil 3

IF2211 Strategi Algoritma

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

Rizky Ramadhana P. K.

13520151

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

A. Pendahuluan

Permasalahan 15-Puzzle adalah sebuah permainan di petak berukuran 4x4 dimana 15 kotak diantaranya berisi ubin dengan angka 1-15 yang bisa digeser-geser. Sedangkan 1 kotak lainnya tidak terisi oleh ubin apapun. Untuk setiap giliran permainan, pemain bisa memilih ingin menggeser ubin yang mana ke arah mana. Perlu diingat bahwa ubin yang bisa digeser hanyalah ubin yang bersebelahan dengan kotak yang tidak terisi oleh ubin apapun.

Permasalahan ini bisa diselesaikan dengan algoritma *branch and bound* yang serupa dengan algoritma *breadth first search* (bfs) hanya saja untuk mengelola antrian simpul yang akan dikunjungi digunakan struktur data *priority queue* yang menggunakan *cost* dari suatu simpul sebagai prioritasnya. Semakin kecil cost dari suatu simpul, maka akan lebih diprioritaskan pada antrian tersebut.

B. Cara Kerja Program

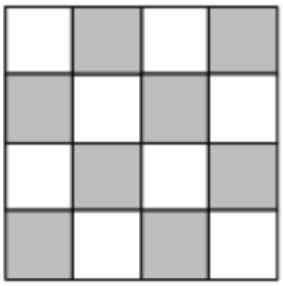
Program mula-mula akan meminta informasi dari pengguna apakah ingin menggunakan puzzle yang dibangkitkan secara random atau menggunakan puzzle buatan sendiri yang terdapat di sebuah file teks. Kedua metode tersebut akan memberikan program sebuah array berukuran 4x4 dengan elemen berupa angka 1-16, di mana 16 mewakili ubin kosong. Puzzle yang dibangkitkan secara random didapat dari konfigurasi puzzle tujuan yang digerakkan secara random sebanyak n kali. Besaran n bervariasi dari 15, 30, atau 60 sesuai level yang dipilih oleh pengguna.

Setelah didapat array 4x4 yang menjadi puzzle untuk diselesaikan, program akan menghitung nilai dari fungsi KURANG(i) untuk tiap ubin yang ada. Fungsi tersebut menyatakan banyaknya ubin j sedemikian hingga j<i tetapi POSISI(j) > posisi(i). Program kemudian akan mencetaknya ke layar dan menentukan apakah puzzle dapat diselesaikan atau tidak menurut besaran di bawah ini

$$\sum_{i=1}^{16} KURANG(i) \; + \; X$$

dengan X bernilai 1 bila posisi ubin kosong berada di kotak yang diarsir. Bila besaran tersebut bernilai genap, maka puzzle bisa diselesaikan dan program akan mencari

solusinya. Namun bila besaran tersebut bernilai ganjil, maka puzzle tidak bisa diselesaikan dan program akan memberi pesan peringatan dan berhenti.



Bila puzzle bisa diselesaikan, maka algoritma *branch and bound* akan dimulai. Dalam tugas ini, akan di buat kelas Node yang berperan sebagai simpul dalam pohon pencarian. Kelas Node akan memiliki atribut nama, nama orang tua, matriks isi, posisi ubin kosong, kedalaman, dan cost. Untuk menjalankan algoritma *branch and bound* juga akan dibuat kelas bernama BranchNBound yang memiliki atribut simpul yang telah dikunjungi, simpul awal, daftar simpul yang telah dibangkitkan, dan *priority queue* untuk menyimpan antrian simpul. Mula-mula akan dicek apakah kondisi awal merupakan goal atau bukan. Bila iya maka program akan berhenti, tetapi bila tidak maka program akan meneruskan pencarian. Di setiap langkah akan di *dequeue* sebuah simpul. Untuk setiap simpul yang dikunjungi, akan dibuat empat anak yang masing-masing berkaitan dengan alternatif gerakan ubin kosong. Perlu diingat simpul yang dibangkitkan bisa saja kurang dari empat pada kasus ubin kosong berada di ujung atau pojok. Untuk setiap simpul yang baru saja dibangkitkan, akan dicek apakah simpul tersebut merupakan simpul tujuan atau bukan. Bila iya, maka pencarian dihentikan. Bila tidak, maka simpul tersebut akan di *enqueue* ke dalam *priority queue.*

Bila solusi telah ditemukan, maka akan dicari langkah-langkah yang bersesuaian dengan solusi tersebut. Dari simpul tujuan, akan ditelusuri orang tua simpul tersebut, yang di simpan dalam salah satu atribut kelas Node, sampai ditemukan akar. Kemudian langkah tersebut ditampilkan secara terbalik dari akar sampai simpul tujuan. Langkah-langkah penyelesaian akan ditampilkan dalam dua versi yaitu versi animasi (simulasi animasi dapat dilihat pada video di lampiran) dan versi statis.

C. Source Code Program

main.py

```
from Node import *
from BranchNBound import *
from utils import *

source = input("Ketik 1 untuk menggunakan puzzle random, ketik 2 untuk
menggunakan puzzle dari file : ")
if source=='1':
    level = input("Ketik level puzzle yang diinginkan (1, 2, atau 3) : ")
    puzzle = randomPuzzle(level)
else:
    filename = input("Masukkan nama file (harus berada dalam folder test) : ")
    puzzle = readFile("test/"+filename)

bnb = BranchNBound(puzzle)

if bnb.check()==1:
    bnb.searchSolution()
```

BranchNBound.py

```
from Node import *
from queue import PriorityQueue
from utils import *
from time import sleep,time

class BranchNBound :
    def __init__(self,puzzle):
        self.visited={}

        found = False
        i=0
        while(not found and i<4):
            j=0
            while(not found and j<4):
                if puzzle[i][j]==16:
                    found=True
                else:
                    j+=1
            if not found :
                i+=1

        root = Node("NO",puzzle,(i,j),0)
        self.start = root
        self.nodeList={root.name : root}
        self.prioQ = PriorityQueue()
        self.prioQ.put((root.cost,root))

    def check(self):
        kurangs = [0 for i in range(16)]
        flat =
```

```
self.start.matrix[0]+self.start.matrix[1]+self.start.matrix[2]+self.start.matr
ix[3]

        for i in range(16):
            kurang=0
            for j in range(i+1,16):
                if flat[j]<flat[i]:
                    kurang+=1
            kurangs[self.start.matrix[i//4][i%4]-1]=kurang

        print("PUZZLE AWAL")
        self.start.print()
        sum=0
        for i,kurang in enumerate(kurangs):
            print(f"Nilai KURANG({i+1})={kurang}")
            sum=sum+kurang
        if (self.start.nullPos[0]+self.start.nullPos[1])%2==1:
            sum+=1
        print(f"Jumlah dari KURANG(i) ditambah X : {sum}")
        if sum%2==0:
            print("Puzzle dapat diselesaikan !")
            return 1
        else :
            print("Puzzle tidak dapat diselesaikan !")
            return 0

    def searchSolution(self):
        print("Mencari solusi...")
        start = time()
        if(self.start.isGoal()):
            ans=self.start
        else:
            found = False
            while(self.prioQ and not found):
                now = self.prioQ.get()[1]
                try:
                    newNode = now.UP()
                    self.nodeList[newNode.name]=newNode
                    newString = toString(newNode.matrix)
                    if not self.visited.get(newString,False) :
                        if newNode.isGoal():
                            found=True
                            ans=newNode
                        else:
                            self.prioQ.put((newNode.cost,newNode))
                except :
                    pass


                try:
                    newNode = now.DOWN()
                    self.nodeList[newNode.name]=newNode
                    newString = toString(newNode.matrix)
                    if not self.visited.get(newString,False) :
                        if newNode.isGoal():
                            found=True
```

```
                            ans=newNode
                        else:
                            self.prioQ.put((newNode.cost,newNode))
                except :
                    pass


                try:
                    newNode = now.RIGHT()
                    self.nodeList[newNode.name]=newNode
                    newString = toString(newNode.matrix)
                    if not self.visited.get(newString,False) :
                        if newNode.isGoal():
                            found=True
                            ans=newNode
                        else:
                            self.prioQ.put((newNode.cost,newNode))
                except :
                    pass

                try:
                    newNode = now.LEFT()
                    self.nodeList[newNode.name]=newNode
                    newString = toString(newNode.matrix)
                    if not self.visited.get(newString,False) :
                        if newNode.isGoal():
                            found=True
                            ans=newNode
                        else:
                            self.prioQ.put((newNode.cost,newNode))
                except :
                    pass

                self.visited[toString(now.matrix)] = True

        end = time()
        current = ans.name
        steps=[]
        while current!="NO":
            steps.insert(0,self.nodeList[current])
            current=self.nodeList[current].prev
        input("Tekan ENTER untuk memulai animasi")
        n = len(steps)
        for i in range(n-1):
            steps[i].print()
            print
("\033[A\033[A\033[A\033[A\033[A\033[A\033[A\033[A\033[A\033[A")
            sleep(1)
        steps[n-1].print()
        sleep(1)
        choice = input("Ingin mencetak semua langkah (y/n) ? ")
        if choice=='y':
            for i in range(n):
                steps[i].print()

        print("Jumlah node yang dibangkitkan : ",Node.numOfNodes)
```

```
        print("Waktu yang dibutuhkan : ", end-start, " detik")
```

Node.py

```
class Node:
    numOfNodes = 0
    def __init__(self,prev,matrix,nullPos,level):
        Node.numOfNodes+=1
        self.name="n"+str(Node.numOfNodes)
        self.prev=prev
        self.matrix=matrix.copy()
        self.nullPos=nullPos
        self.level=level
        self.cost=self.countCost(matrix)


    def UP(self):
        if self.nullPos[0]==0:
            raise Exception("Cannot move it up !")
        else:
            temp = [[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]]
            for k in range(4):
                for l in range(4):
                    temp[k][l]=self.matrix[k][l]
            i=self.nullPos[0]
            j=self.nullPos[1]
            temp[i][j] = temp[i-1][j]
            temp[i-1][j] = 16
            cost = self.countCost(temp)
            return Node(self.name,temp,(i-1,j),self.level+1)

    def DOWN(self):
        if self.nullPos[0]==3:
            raise Exception("Cannot move it down !")
        else:
            temp = [[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]]
            for k in range(4):
                for l in range(4):
                    temp[k][l]=self.matrix[k][l]
            i=self.nullPos[0]
            j=self.nullPos[1]
            temp[i][j] = temp[i+1][j]
            temp[i+1][j] = 16
            cost = self.countCost(temp)
            return Node(self.name,temp,(i+1,j),self.level+1)
    def LEFT(self):
        if self.nullPos[1]==0:
            raise Exception("Cannot move it left !")
        else:
            temp = [[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]]
            for k in range(4):
                for l in range(4):
                    temp[k][l]=self.matrix[k][l]
```

```
            i=self.nullPos[0]
            j=self.nullPos[1]
            temp[i][j] = temp[i][j-1]
            temp[i][j-1] = 16
            cost = self.countCost(temp)
            return Node(self.name,temp,(i,j-1),self.level+1)

    def RIGHT(self):
        if self.nullPos[1]==3:
            raise Exception("Cannot move it down !")
        else:
            temp = [[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]]
            for k in range(4):
                for l in range(4):
                    temp[k][l]=self.matrix[k][l]
            i=self.nullPos[0]
            j=self.nullPos[1]
            temp[i][j] = temp[i][j+1]
            temp[i][j+1] = 16
            cost = self.countCost(temp)
            return Node(self.name,temp,(i,j+1),self.level+1)

    def __lt__(self,other):
        return self.cost < other.cost

    #UBAH DI SINI
    def countCost(self,puzzle):
        cost=0
        for i in range(4):
            for j in range(4):
                if puzzle[i][j]!=4*i+j+1 and puzzle[i][j]!=16:
                    cost+=1

        return cost+self.level

    def isGoal(self):
        goal = [[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]]
        return self.matrix == goal

    def print(self):
        print("====================")
        for i in range(3):
            print("* ",end="")
            for j in range(3):
                if self.matrix[i][j]!=16 :
                    print(str(self.matrix[i][j]).ljust(2),end="")
                else:
                    print("  ",end="")
                print(" | ",end="")
            if self.matrix[i][3]!=16 :
                print(str(self.matrix[i][3]).ljust(2),end="")
            else:
                 print("  ",end="")
            print(" *")
            print("*-------------------*")
        print("* ",end="")
```

```
        for j in range(3):
            if self.matrix[3][j]!=16 :
                print(str(self.matrix[3][j]).ljust(2),end="")
            else:
                print("  ",end="")
            print(" | ",end="")

        if self.matrix[3][3]!=16 :
           print(str(self.matrix[3][3]).ljust(2),end="")
        else:
            print("  ",end="")
        print(" *")
        print("====================")
```

utils.py

```
from random import randint
from Node import *

def toString(puzzle):
    return ','.join(str(item) for innerlist in puzzle for item in innerlist)

def readFile(filename):
    with open(filename) as f:
        lines = f.readlines()
    lines = [line.strip().split() for line in lines]
    for i in range(4):
        for j in range(4):
            lines[i][j] = int(lines[i][j])
    return lines

def randomPuzzle(n):
    if(n==1):
        steps=15
    elif(n==2):
        steps=30
    else:
        steps=60

    root = Node("NO",[[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]],(3,3),0)

    i=0
    while i<steps :
        y = randint(1,4)
        if y==1 :
            try:
                root=root.UP()
                i+=1
            except:
                pass
        elif y==2 :
            try:
                root=root.DOWN()
```

```
                i+=1
            except:
                pass
        elif y==3 :
            try:
                root=root.LEFT()
                i+=1
            except:
                pass
        else :
            try:
                root=root.RIGHT()
                i+=1
            except:
                pass
    Node.numOfNodes = 0
    return root.matrix
```

D. Contoh Instansiasi Permasalahan

Catatan : pada tugas ini diasumsikan ubin kosong diwakili dengan angka 16

1.txt

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

2.txt

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

3.txt

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

unsolvable1.txt

```
2 1 3 7
```

```
5 16 8 4
9 6 12 14
13 10 11 15
```

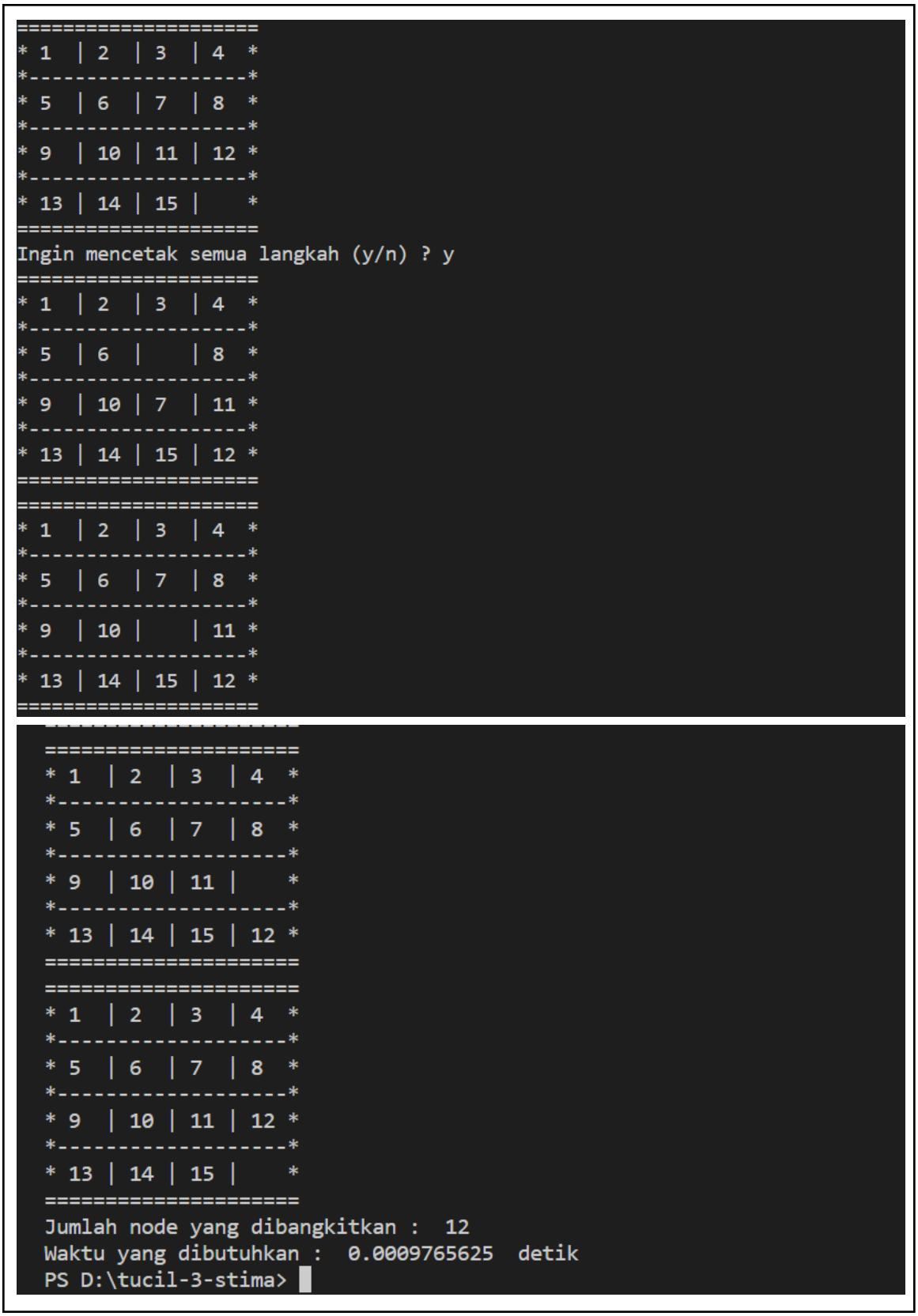
unsolvable2.txt

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

E. Hasil Eksekusi Program

1.txt

```
Jumlah node yang dibangkitkan : 52849
Waktu yang dibutuhkan& C:/Users/user/AppData/Local/Programs/Python/Python310/python.exe d:/tucil-3-stima/src/ma
in.py\tucil-3-stima>
Ketik 1 untuk menggunakan puzzle random, ketik 2 untuk menggunakan puzzle dari file : 2
Masukkan nama file (harus berada dalam folder test) : 1.txt
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
Jumlah dari KURANG(i) ditambah X : 16
Puzzle dapat diselesaikan !
Mencari solusi...
Tekan ENTER untuk memulai animasi
====================
```

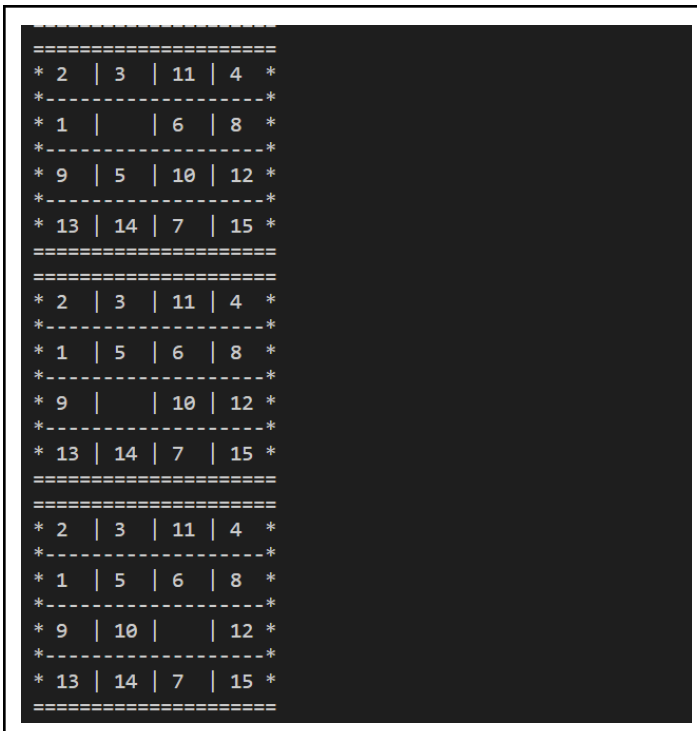

====================
* 1 | 2 | 3 | 4 *
----
* 5 | 6 | 7 | 8 *
----
* 9 | 10 | 11 | 12 *
----
* 13 | 14 | 15 | *
====================
Ingin mencetak semua langkah (y/n) ? y
====================
* 1 | 2 | 3 | 4 *
----
* 5 | 6 | | 8 *
----
* 9 | 10 | 7 | 11 *
----
* 13 | 14 | 15 | 12 *
====================
====================
* 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 | *
====================
Jumlah node yang dibangkitkan : 12
Waktu yang dibutuhkan : 0.0009765625 detik
PS D:\tucil-3-stima>

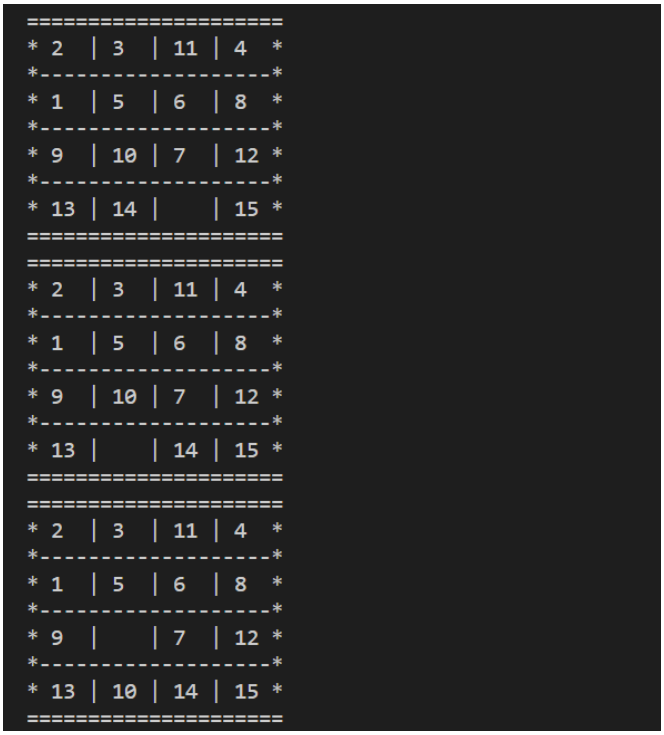
2.txt

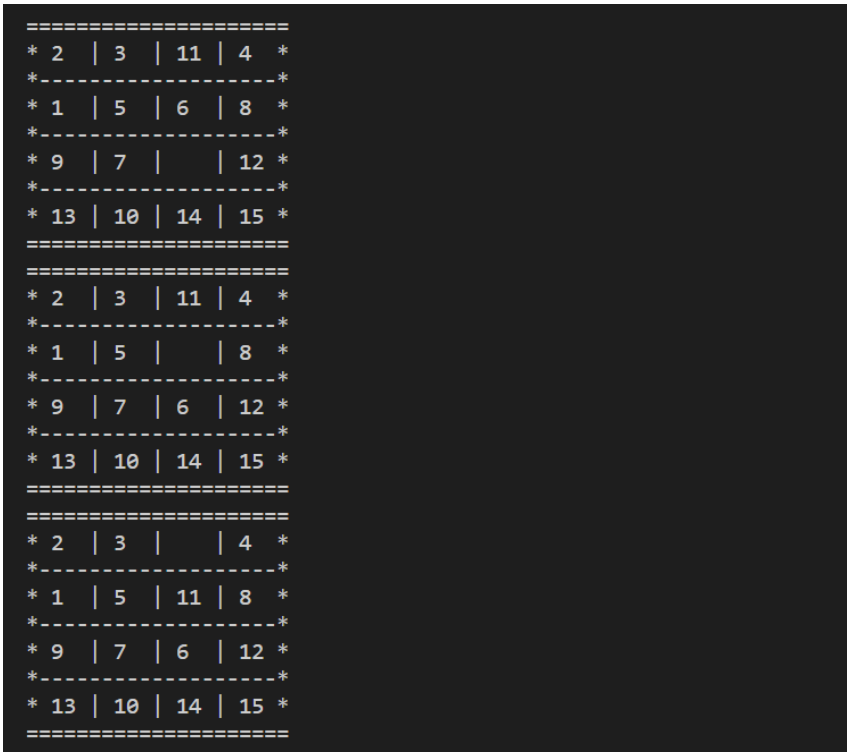
```
in.py
Ketik 1 untuk menggunakan puzzle random, ketik 2 untuk menggunakan puzzle dari file : 2
Masukkan nama file (harus berada dalam folder test) : 2.txt
PUZZLE AWAL
====================
* 2  | 3  | 11 | 4  *
*-------------------*
* 1  | 6  | 10 | 8  *
*-------------------*
* 9  | 5  | 12 | 15 *
*-------------------*
* 13 | 14 |    | 7  *
====================
Nilai KURANG(1)=0
Nilai KURANG(2)=1
Nilai KURANG(3)=1
Nilai KURANG(4)=1
Nilai KURANG(5)=0
Nilai KURANG(6)=1
Nilai KURANG(7)=0
Nilai KURANG(8)=2
Nilai KURANG(9)=2
Nilai KURANG(10)=4
Nilai KURANG(11)=8
Nilai KURANG(12)=1
Nilai KURANG(13)=1
Nilai KURANG(14)=1
Nilai KURANG(15)=3
Nilai KURANG(16)=1
Jumlah dari KURANG(i) ditambah X : 28
Puzzle dapat diselesaikan !
Mencari solusi...
Tekan ENTER untuk memulai animasi
```

```
Tekan ENTER untuk memulai animasi
 ====================
 * 1  | 2  | 3  | 4  *
 *-------------------*
 * 5  | 6  | 7  | 8  *
 *-------------------*
 * 9  | 10 | 11 | 12 *
 *-------------------*
 * 13 | 14 | 15 |    *
 ====================
 Ingin mencetak semua langkah (y/n) ? y
 ====================
 * 2  | 3  | 11 | 4  *
 *-------------------*
 * 1  | 6  | 10 | 8  *
 *-------------------*
 * 9  | 5  | 12 | 15 *
 *-------------------*
 * 13 | 14 |    | 7  *
 ====================
 ====================
 * 2  | 3  | 11 | 4  *
 *-------------------*
 * 1  | 6  | 10 | 8  *
 *-------------------*
 * 9  | 5  | 12 | 15 *
 *-------------------*
 * 13 | 14 | 7  |    *
 ====================
```

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

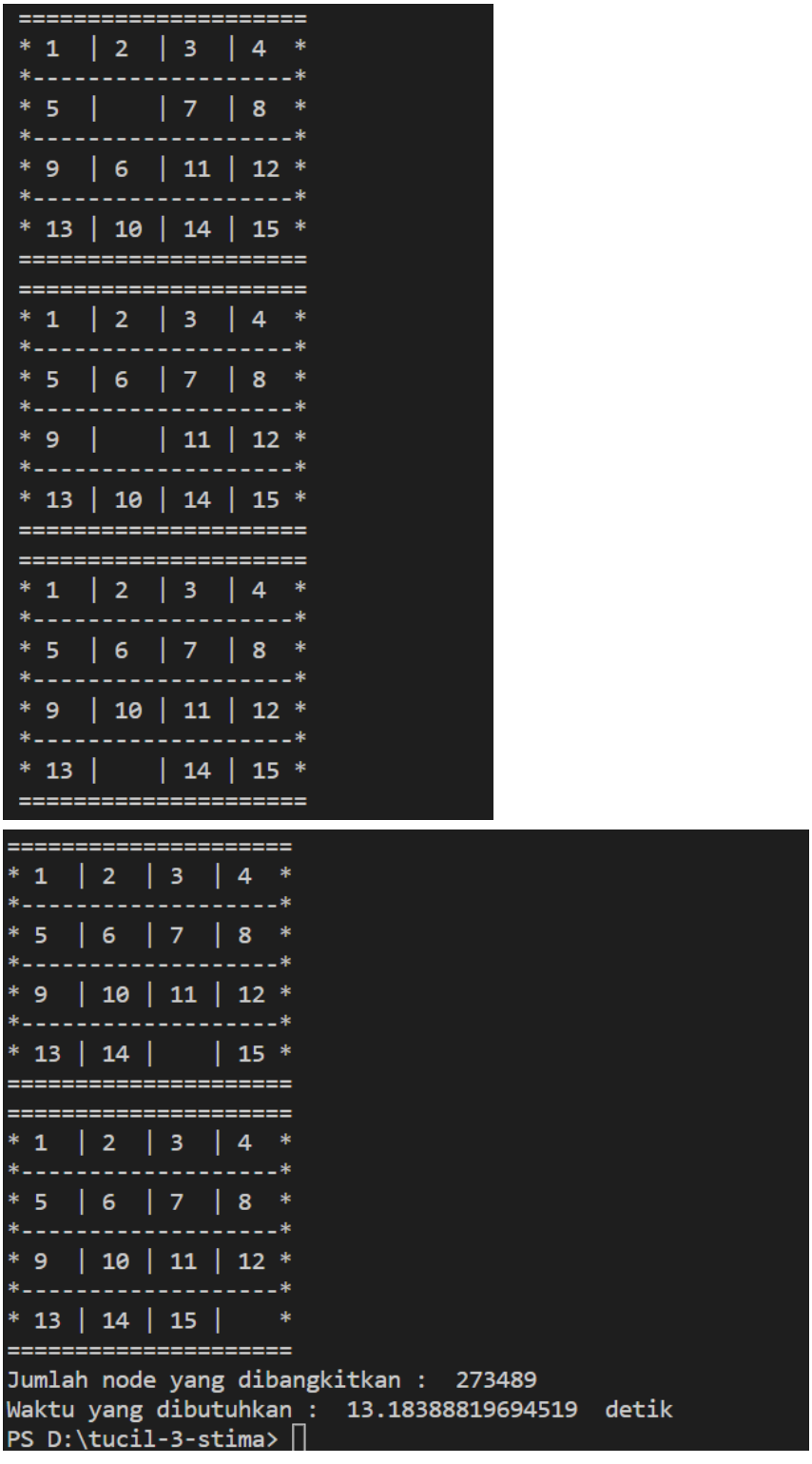

* 2 | 3 | 11 | 4 *
* 1 | | 6 | 8 *
* 9 | 5 | 10 | 12 *
* 13 | 14 | 7 | 15 *
* 2 | 3 | 11 | 4 *
* 1 | 5 | 6 | 8 *
* 9 | | 10 | 12 *
* 13 | 14 | 7 | 15 *
* 2 | 3 | 11 | 4 *
* 1 | 5 | 6 | 8 *
* 9 | 10 | | 12 *
* 13 | 14 | 7 | 15 *


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


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

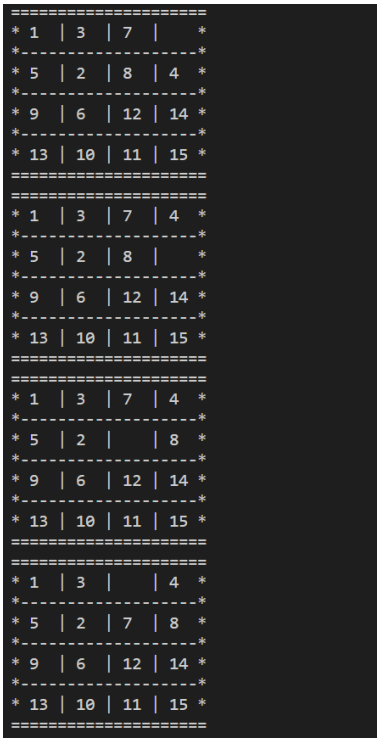
```
====================
* 2  |    | 3  | 4  *
*-------------------*
* 1  | 5  | 11 | 8  *
*-------------------*
* 9  | 7  | 6  | 12 *
*-------------------*
* 13 | 10 | 14 | 15 *
====================
====================
*    | 2  | 3  | 4  *
*-------------------*
* 1  | 5  | 11 | 8  *
*-------------------*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

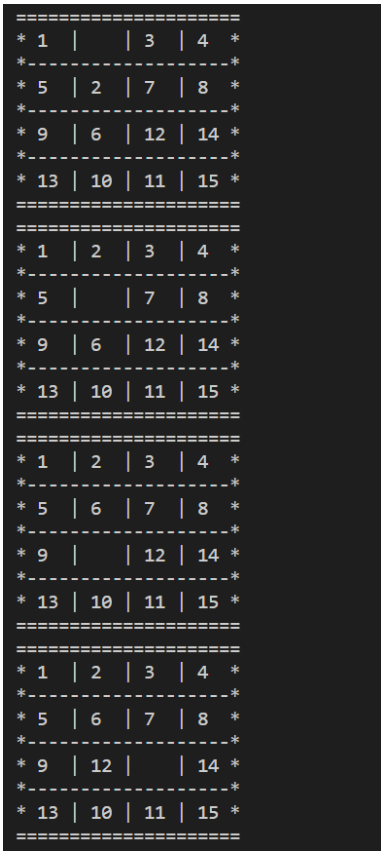
```
====================
* 1  | 2  | 3  | 4  *
*-------------------*
* 5  | 7  | 11 | 8  *
*-------------------*
* 9  |    | 6  | 12 *
*-------------------*
* 13 | 10 | 14 | 15 *
====================
====================
* 1  | 2  | 3  | 4  *
*-------------------*
* 5  | 7  | 11 | 8  *
*-------------------*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

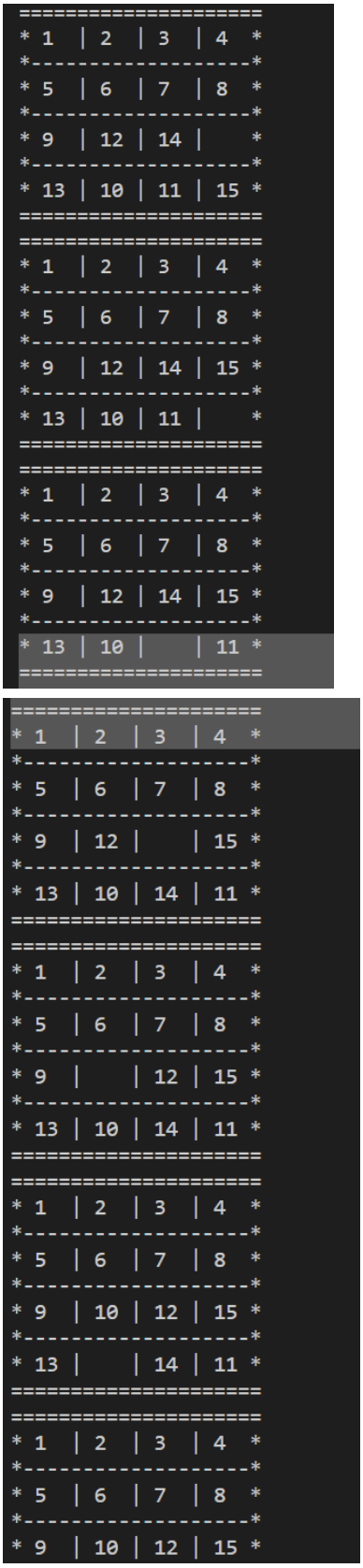

=====================
* 1 | 2 | 3 | 4 *
----
* 5 | | 7 | 8 *
----
* 9 | 6 | 11 | 12 *
----
* 13 | 10 | 14 | 15 *
=====================
=====================
* 1 | 2 | 3 | 4 *
----
* 5 | 6 | 7 | 8 *
----
* 9 | | 11 | 12 *
----
* 13 | 10 | 14 | 15 *
=====================
=====================
* 1 | 2 | 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 *
=====================
=====================
* 1 | 2 | 3 | 4 *
----
* 5 | 6 | 7 | 8 *
----
* 9 | 10 | 11 | 12 *
----
* 13 | 14 | 15 | *
=====================
Jumlah node yang dibangkitkan : 273489
Waktu yang dibutuhkan : 13.18388819694519 detik
PS D:\tucil-3-stima>

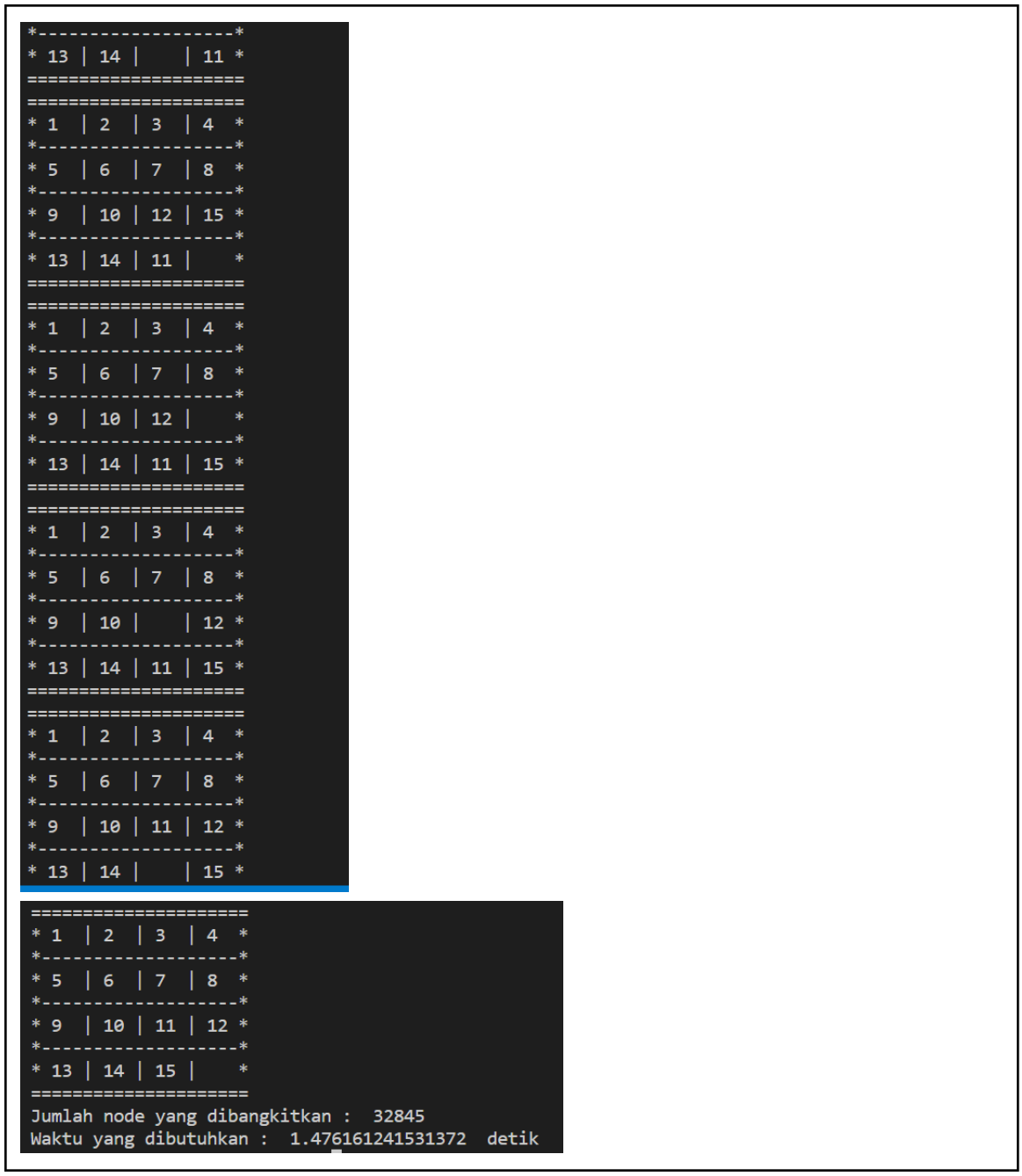
3.txt

```
PS D:\tucil-3-stima> & C:/Users/user/AppData/Local/Programs/Python/Python310/python.exe d:/tucil-3-stima/src/ma
in.py
Ketik 1 untuk menggunakan puzzle random, ketik 2 untuk menggunakan puzzle dari file : 2
Masukkan nama file (harus berada dalam folder test) : 3.txt
PUZZLE AWAL
====================
*    | 1  | 3  | 7  *
*------------------*
* 5  | 2  | 8  | 4  *
*------------------*
* 9  | 6  | 12 | 14 *
*------------------*
* 13 | 10 | 11 | 15 *
====================
Nilai KURANG(1)=0
Nilai KURANG(2)=0
Nilai KURANG(3)=1
Nilai KURANG(4)=0
Nilai KURANG(5)=2
Nilai KURANG(6)=0
Nilai KURANG(7)=4
Nilai KURANG(8)=2
Nilai KURANG(9)=1
Nilai KURANG(10)=0
Nilai KURANG(11)=0
Nilai KURANG(12)=2
Nilai KURANG(13)=2
Nilai KURANG(14)=3
Nilai KURANG(15)=0
Nilai KURANG(16)=15
Jumlah dari KURANG(i) ditambah X : 32
Puzzle dapat diselesaikan !
Mencari solusi...
Tekan ENTER untuk memulai animasi
====================
* 1  | 2  | 3  | 4  *
*------------------*
* 5  | 6  | 7  | 8  *
*------------------*
*------------------*
* 9  | 10 | 11 | 12 *
*------------------*
* 13 | 14 | 15 |    *
====================
Ingin mencetak semua langkah (y/n) ? y
====================
*    | 1  | 3  | 7  *
*------------------*
* 5  | 2  | 8  | 4  *
*------------------*
* 9  | 6  | 12 | 14 *
*------------------*
* 13 | 10 | 11 | 15 *
====================
====================
* 1  |    | 3  | 7  *
*------------------*
* 5  | 2  | 8  | 4  *
*------------------*
* 9  | 6  | 12 | 14 *
*------------------*
* 13 | 10 | 11 | 15 *
====================
====================
* 1  | 3  |    | 7  *
*------------------*
* 5  | 2  | 8  | 4  *
*------------------*
* 9  | 6  | 12 | 14 *
*------------------*
* 13 | 10 | 11 | 15 *
====================
```

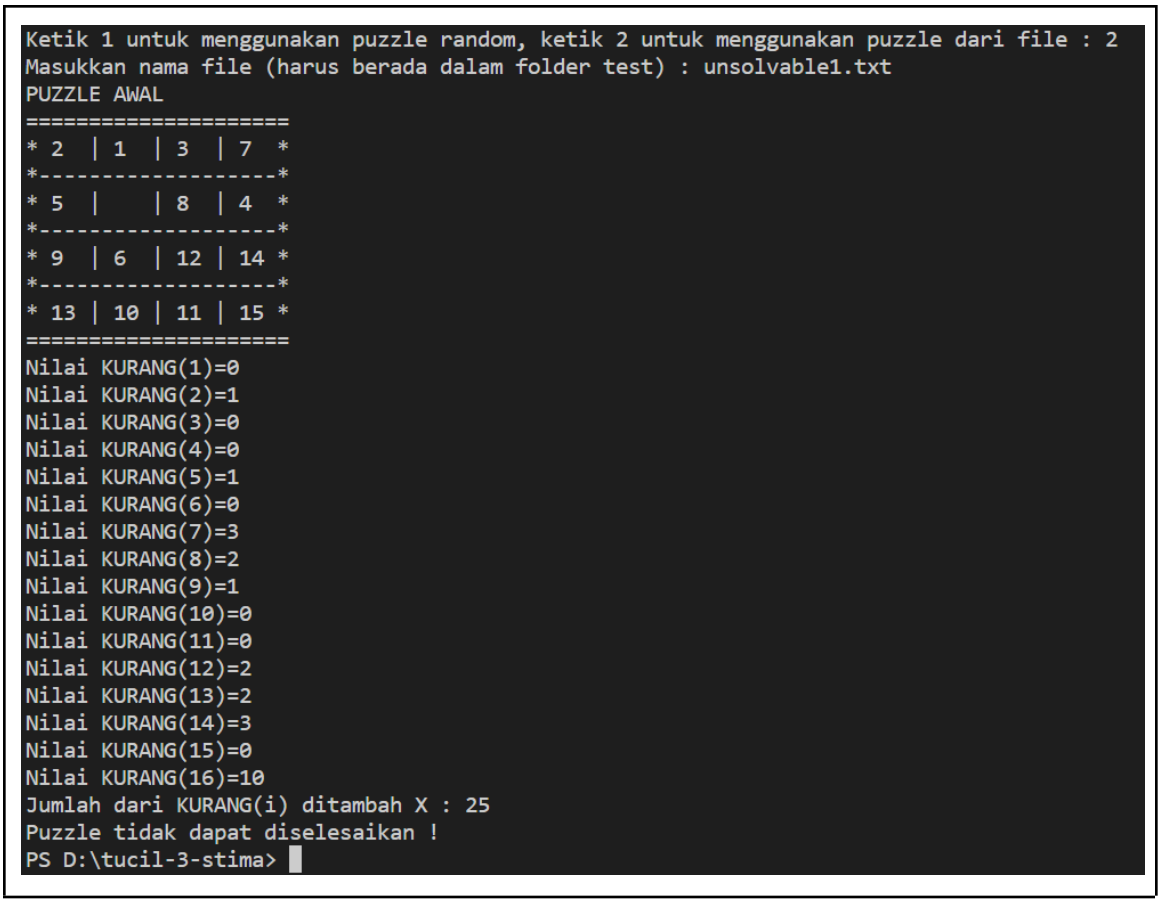

==========
* 1 | 3 | 7 | *
----
* 5 | 2 | 8 | 4 *
----
* 9 | 6 | 12 | 14 *
----
* 13 | 10 | 11 | 15 *
==========
==========
* 1 | 3 | 7 | 4 *
----
* 5 | 2 | 8 | *
----
* 9 | 6 | 12 | 14 *
----
* 13 | 10 | 11 | 15 *
==========
==========
* 1 | 3 | 7 | 4 *
----
* 5 | 2 | | 8 *
----
* 9 | 6 | 12 | 14 *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


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

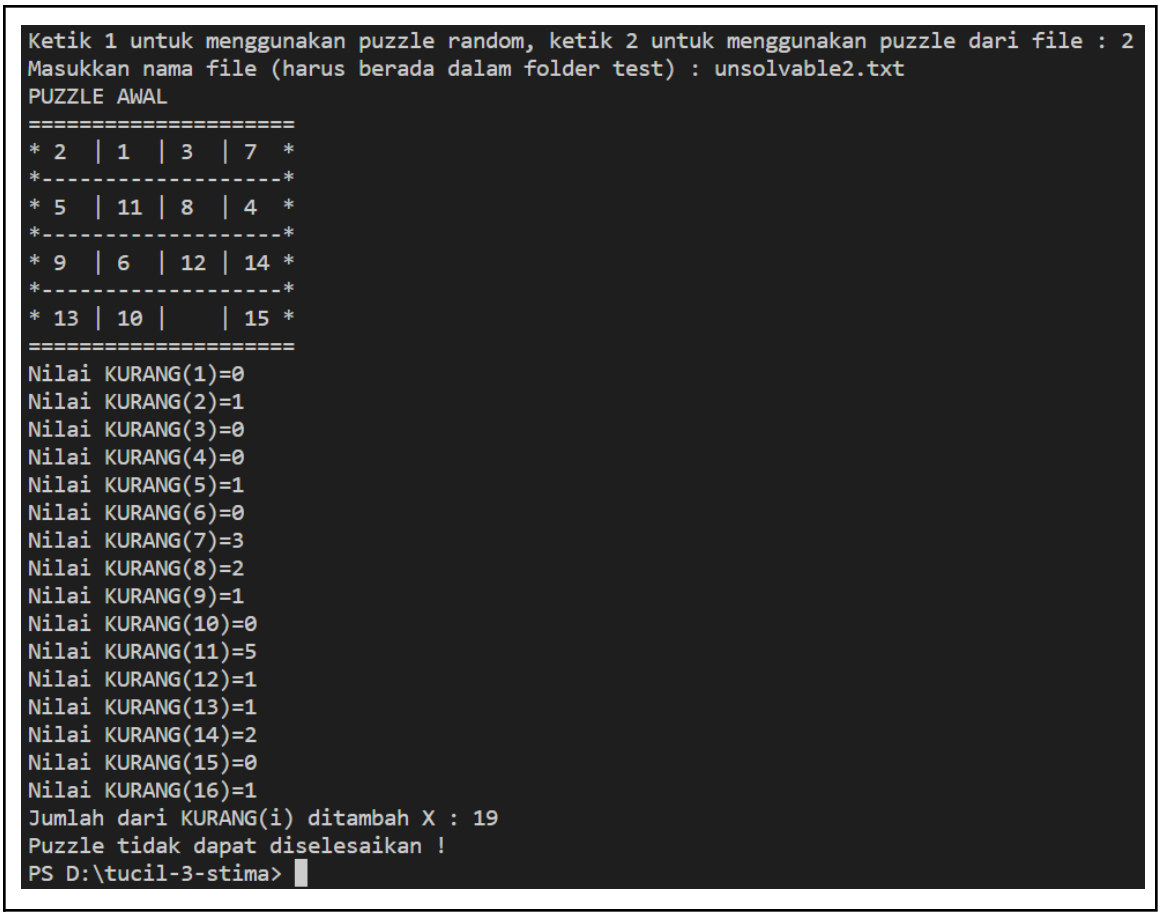

==========
* 1 | 2 | 3 | 4 *
----
* 5 | 6 | 7 | 8 *
----
* 9 | 12 | 14 | *
----
* 13 | 10 | 11 | 15 *
==========
==========
* 1 | 2 | 3 | 4 *
----
* 5 | 6 | 7 | 8 *
----
* 9 | 12 | 14 | 15 *
----
* 13 | 10 | 11 | *
==========
==========
* 1 | 2 | 3 | 4 *
----
* 5 | 6 | 7 | 8 *
----
* 9 | 12 | 14 | 15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



unsolvable1.txt



unsolvable2.txt



F. Lampiran

Poin	Ya	Tidak
1. Program berhasil dikompilasi	v	
2. Program berhasil *running*	v	
3. Program dapat menerima input dan menuliskan output.	v	
4. Luaran sudah benar untuk semua data uji	v	
5. Bonus dibuat	Hanya membuat animasi pergerakan ubin di terminal	

Link repository https://github.com/rizkyramadhana26/tucil-3-stima

Video simulasi https://youtu.be/5-vmJjZcmL8