TUGAS KECIL 3

PENYELESAIAN PERSOALAN 15-PUZZLE DENGAN ALGORITMA *BRANCH AND BOUND*

LAPORAN

Diajukan sebagai salah satu tugas mata kuliah IF2211 Strategi Algoritma pada Semester II

Tahun Akademik 2021-2022

Oleh

Fachry Dennis Heraldi 13520139

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2022

DAFTAR ISI

Cara Kerja Program *Branch and Bound* yang Dibuat 3
Source Code Program 7
Screenshot Input dan *Output* 17
Alamat *Repository* Kode Program 32

Cara Kerja Program *Branch and Bound* yang Dibuat

Diberikan suatu masukan matriks yang merepresentasikan posisi awal suatu instansiasi persoalan 15-puzzle sebagai berikut.

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

Gambar 1 Ilustrasi Posisi Awal Persoalan 15-puzzle

Mula-mula, program akan menentukan terlebih dahulu apakah persoalan 15-puzzle tersebut dapat diselesaikan atau tidak. Untuk menentukannya akan dilakukan perhitungan sesuai dengan teorema berikut.

Status tujuan hanya dapat dicapai dari status awal jika $\sum_{i=1}^{16} KURANG(i) + X$ *bernilai genap.*

KURANG(*i*) adalah banyaknya puzzle bernomor *j* sedemikian sehingga *j* < *i* dan *POSISI*(*j*) > *POSISI*(*i*). Nilai *X* adalah nilai dari kolom dan baris dari posisi ubin kosong (didefinisikan sebagai ubin bernomor 16) kemudian dimodulo dengan 2 (*X* = (*BARIS*(16) + *KOLOM*(16)) mod 2).

Berdasarkan masukan diatas, akan didapatkan informasi sebagai berikut.

Tabel 1 Nilai KURANG(i) untuk Masing-masing Nomor Ubin

i	*KURANG(i)*	*i*	*KURANG(i)*	*i*	*KURANG(i)*	*i*	*KURANG(i)*
1	0	*5*	0	9	1	13	1
2	0	*6*	0	10	1	14	1
3	0	*7*	0	11	0	15	1
4	0	*8*	1	12	0	16	9

Sehingga didapatkan,

$$\sum_{i=1}^{16} KURANG(i) + X = 15 + 1 = 16$$

Karena 16 adalah angka genap maka persoalan dapat diselesaikan.

Berikutnya, akan dilakukan pembangkitan pohon status pencarian dengan algoritma *branch and bound*. Pada program, akan digunakan struktur data priority queue untuk menyimpan simpul hidup. Prioritas antrian akan ditentukan oleh nilai cost. Cost simpul P pada 15-puzzle didefinisikan sebagai berikut.

$$\hat{c}(P) = f(P) + \hat{g}(P)$$

$f(P)$ adalah panjang lintasan dari simpul akar ke P sedangkan $\hat{g}(P)$ adalah taksiran panjang lintasan terpendek dari *P* ke simpulan solusi pada upapohon yang akarnya *P*. Dengan kata lain, $f(P)$ adalah level dari simpul dan $\hat{g}(P)$ adalah jumlah ubin tidak kosong yang tidak terdapat pada susunan akhir.

Susunan akhir dari persoalan 15-puzzle adalah sebagai berikut.

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

Gambar 2 Ilustrasi Susunan Akhir Persoalan 15-puzzle

Pembangkitan simpul didasarkan oleh aksi yang dapat dilakukan pada puzzle. Terdapat 4 aksi, diantaranya:

1. *up*, ubin kosong ditukar posisinya dengan ubin di sebalah atasnya.
2. *right*, ubin kosong ditukar posisinya dengan ubin di sebelah kanannya.
3. *down*, ubin kosong ditukar posisinya dengan ubin di sebelah bawahnya.
4. *left*, ubin kosong ditukar posisinya dengan ubin di sebelah kirinya.

Adapun batasan aksi diberikan agar setiap langkah selalu valid sebagai berikut.

1. Jika ubin kosong terletak dibaris paling atas atau simpul induknya melakukan aksi *down* maka aksi *up* tidak dapat dilakukan.

2. Jika ubin kosong terletak dikolom paling kanan atau simpul induknya melakukan aksi *left* maka aksi *right* tidak dapat dilakukan.
3. Jika ubin kosong terletak dibaris paling bawah atau simpul induknya melakukan aksi *up* maka aksi *down* tidak dapat dilakukan.
4. Jika ubin kosong terletak dikolom paling kiri atau simpul induknya melakukan aksi *right* maka aksi *left* tidak dapat dilakukan.

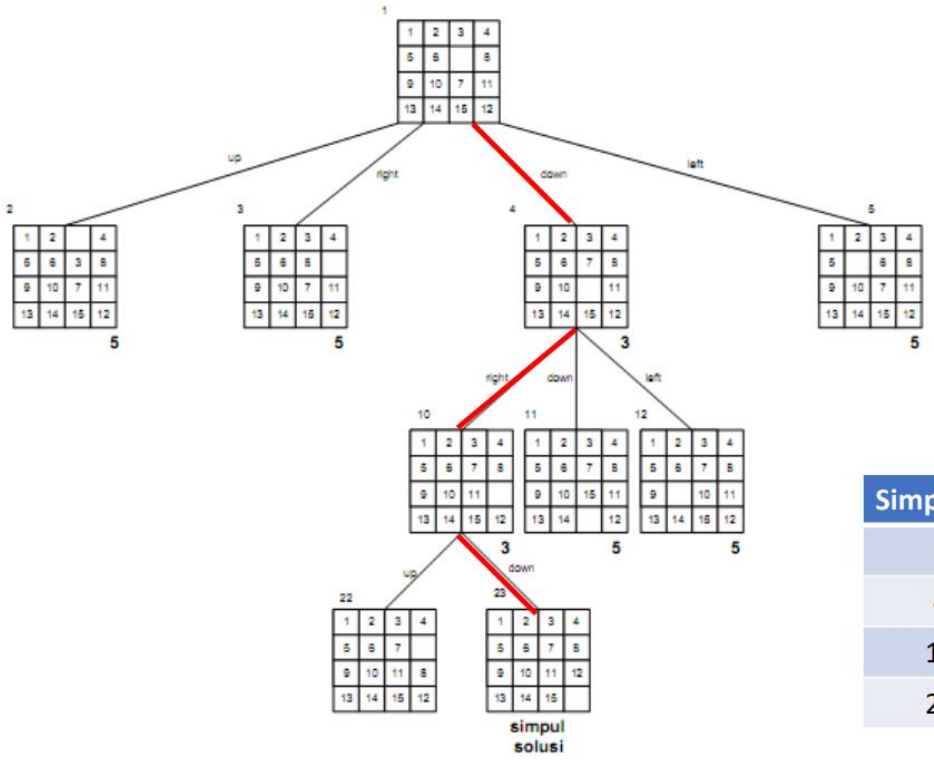
Berdasarkan penjelasan diatas, setiap simpul akan menyimpan informasi sebagai berikut.

1. Simpul induknya
2. Status dari puzzle yang telah dibangkitkan pada simpul tersebut
3. Level dari simpul tersebut
4. Cost dari simpul tersebut

Selanjutnya akan dijelaskan algoritma *branch* and *bound* yang diimplementasikan sebagai berikut.

1. Simpul akar diinisiasi dengan menyimpan status puzzle awal. Simpul akar dimasukkan kedalam antrian simpul hidup.
2. Akan dilakukan perulangan pada antrian simpul hidup. Perulangan akan berhenti ketika antrian simpul hidup kosong atau simpul goal ditemukan.
3. Pada setiap perulangan, satu simpul di-*dequeue* dari antrian simpul hidup. Simpul tersebut menjadi simpul yang akan diekspansi (simpul ekspan). Pengambilan simpul ekspan diprioritaskan oleh nilai cost yang paling kecil.
4. Dilakukan iterasi sebanyak 4 kali pada simpul ekspan tersebut sesuai dengan aksi yang dapat dilakukan. Jika aksi yang diberikan pada status puzzle pada simpul ekspan tersebut valid, maka akan dibangkitkan simpul baru. Simpul baru tersebut akan menyimpan informasi simpul ekspan sebagai simpul induknya, status puzzle sesuai dengan aksi yang diberikan, level dari simpul (level simpul ekspan + 1), dan cost dari simpul tersebut.
5. Simpul goal ditemukan maka perulangan berhenti. Selanjutnya akan dipanggil fungsi yang mencetak langkah penyelesaian puzzle.

Ilustrasi pohon pencarian yang dibangkitkan adalah sebagai berikut.



Simpul-E	Simpul Hidup
1	4,2,3,5
4	10,2,3,5,11,12
10	23,2,3,5,11,12,22
23	Solusi ketemu

Gambar 3 Ilustrasi Pohon Status Pencarian
(Sumber: Slide Algoritma Branch and Bound Bagian 1 Bahan Kuliah IF2211 Strategi Algoritma oleh Rinaldi Munir, Nur Ulfa Maulidevi, Masayu Leylia Khodra)

Source Code Program

Agar program menjadi modular, program dipecah menjadi 5 file berikut.

1. main.py

   File main.py bertanggung jawab sebagai program utama dan menjadi file yang dijalankan pertama kali.

```
from PuzzleRoot import PuzzleRoot
from Solver import SolvePuzzle

if __name__ == "__main__":
    print("\n15-Puzzle Solver")
    print("--------------------------")
    filename = input("\nMasukkan nama file puzzle yang akan diselesaikan (.txt):
")
    filename = "test/" + filename

    puzzle_root = PuzzleRoot(filename)

    if not puzzle_root.is_puzzle_solvable():
        print("\nPuzzle tidak dapat diselesaikan.\n")
    else:
        print("\nPuzzle dapat diselesaikan. \n")
        solve = SolvePuzzle(puzzle_root)
```

2. Puzzle.py

   File Puzzle.py bertanggung jawab untuk mendefinisikan kelas Puzzle. Kelas Puzzle adalah kelas yang membentuk objek Puzzle dan informasi dari Puzzle sesuai statusnya.

```
from copy import deepcopy

class Puzzle:
    # Konstruktor untuk kelas Puzzle
    # Default = puzzle : array of array of int kosong []
    # Default = move_index : -1 (perpindahan belum dilakukan atau tidak valid)
    def __init__(self, puzzle=[], move_index=-1):
        self.puzzle = deepcopy(puzzle)
        self.pos_16 = self.find_ubin_position(16)
        self.row_16, self.col_16 = self.pos_to_rowcol(self.pos_16)
        self.move_index = self.move(move_index)
        self.difference = self.calc_difference()

    # Method untuk mencetak status puzzle
    def print_puzzle(self):
        for i in range(16):
            if (i==0):
                print(" ________________________")
```

```
            if (i in [1,4,7,10]):
                print("|     |     |     |     |")
            if (i in [2,5,8,11]):
                print("|",end="")
                for j in range(4):
                    print("  ",end="")
                    print((self.puzzle[(i-2)//3][j],"  ")[self.puzzle[(i-
2)//3][j]==16],end="")
                    print(("  "," ")[self.puzzle[(i-2)//3][j]>9],end="|")
                print("")
            if (i in [3,6,9,12]):
                print("|_____|_____|_____|_____|")

    # Method untuk mencari posisi ubin
    # Return: posisi ubin dalam puzzle {1..16}
    def find_ubin_position(self, number):
        i = 0
        found = False
        while (not found and i < 4):
            j = 0
            while(not found and j < 4):
                if self.puzzle[i][j] == number:
                    found = True
                j += 1
            i += 1
        return self.rowcol_to_pos(i-1,j-1)

    # Method untuk mengkonversi posisi row dan col ke posisi puzzle
    # Return: posisi dalam puzzle {1..16}
    def rowcol_to_pos(self,row,col):
        return (row*4)+col+1

    # Method untuk mengkonversi posisi puzzle ke posisi row dan col
    # Return: row {0..3} dan col {0..3}
    def pos_to_rowcol(self,pos):
        row = pos//4 if pos%4!=0 else pos//4-1
        col = pos%4-1 if pos%4!=0 else pos%4+3
        return row,col

    # Method untuk mengembalikan nilai angka ubin sesuai dengan masukan
posisinya
    # Return: angka ubin {1..16}
    def pos_to_number(self, pos):
        row, col = self.pos_to_rowcol(pos)
        return self.puzzle[row][col]

    # Method untuk mencari row dan col dari suatu angka ubin
    # Return: row {0..3} dan col {0..3}
    def number_to_rowcol(self,number):
        for i in range(4):
            for j in range(4):
                if self.puzzle[i][j] == number:
                    return i,j

    # Method untuk menggerakkan ubin bernomor 16 sesuai dengan arahnya
    # Return: move_index {-1..3} <- mengindikasikan arah perpindahan
    def move(self, move_index):
```

```
        # move_index:
        # 0 : up, 1 : right, 2 : down, 3 : left
        row_move = [-1,0,1,0]
        col_move = [0,1,0,-1]
        is_moving = False

        if move_index!= -1: # move indexnya valid
            moving_row = self.row_16 + row_move[move_index]
            moving_col = self.col_16 + col_move[move_index]

            if (moving_row >= 0 and moving_row < 4
                and moving_col >=0 and moving_col < 4): # ubin setelah move
valid
                is_moving = True
                swap = self.puzzle[moving_row][moving_col] # proses swapping
                self.puzzle[self.row_16][self.col_16]= swap
                self.puzzle[moving_row][moving_col] = 16
                self.row_16 = moving_row
                self.col_16 = moving_col

        if not is_moving: # if not move
            move_index = -1

        return move_index

    # method untuk menghitung perbedaan antara status puzzle saat ini dengan
kondisi akhir
    # return : nilai perbedaan antara status puzzle saat ini dan kondisi akhir
    def calc_difference(self):
        count = 0
        for i in range(15):
            if (self.pos_to_number(i+1) != i+1):
                count+=1
        return count
```

3. PuzzleRoot.py

   File PuzzleRoot.py bertanggung jawab untuk mendefinisikan kelas PuzzleRoot yang merupakan kelas anak dari Puzzle.py. Kelas PuzzleRoot ditujukan untuk menyimpan informasi status awal persoalan 15-puzzle.

```
from Puzzle import Puzzle

class PuzzleRoot(Puzzle):
    def __init__(self, filename):
        self.puzzle = []
        self.load_puzzle(filename)
        self.kurang_i=[]
        self.value_X = 0
        self.move_index = -1
        self.difference = self.calc_difference()
        self.calc_kurang_i() # Menghitung nilai kurang_i
```

```
        self.find_value_X() # Mencari nilai X
        self.puzzle_info = ""
        self.puzzle_info_gui() # Mencetak informasi puzzle untuk gui
        self.print_puzzle_info_cli() # Mencetak informasi puzzle untuk CLI

    # Method untuk menerima input puzzle dari file
    def load_puzzle(self, filename):
        with open(filename) as f:
            for line in f:
                self.puzzle.append(list(map(int, line.split())))
        self.pos_16 = self.find_ubin_position(16)
        self.row_16 , self.col_16 = self.pos_to_rowcol(self.pos_16)
        print("\nPuzzle berhasil dimuat.")

    # Method untuk mencari nilai kurang i untuk setiap ubin
    def calc_kurang_i(self):
        for i in range(16): # Untuk ubin bernomor i {1(0)..16(15)}
            count = 0
            pos_ubin_i = self.find_ubin_position(i+1)
            for j in range(pos_ubin_i+1,17): # POSISI(j) > POSISI(i)
                if (self.pos_to_number(j) < i+1): # j < i
                    count+=1
            self.kurang_i.append(count)

    # Method untuk mencari nilai X
    def find_value_X(self):
        if ((self.row_16+self.col_16) % 2 == 1):
            self.value_X = 1

    # Method untuk mencetak kurang_i
    def print_kurang_i(self):
        print("\nNilai Kurang(i) untuk setiap ubin tersebut: \n")
        for j in range(4):
            for i in range(4):
                print(f"Kurang({self.rowcol_to_pos(i,j)})={self.kurang_i[self.ro
wcol_to_pos(i,j)-1]}", end="\t")
            print()

    # Method untuk mencetak sum of kurang_i + value X
    def print_kurang_i_plus_x(self):
        result = sum(self.kurang_i) + self.value_X
        print("\nNilai dari Σ(i=1,16) Kurang(i) + X adalah", result)

    # Method untung mengecek apakah puzzle dapat diselesaikan atau tidak
    # berdasarkan nilai sum of kurang_i + value X
    def is_puzzle_solvable(self):
        return (sum(self.kurang_i) + self.value_X)%2 == 0

    # Method untuk mencetak informasi puzzle di CLI
    def print_puzzle_info_cli(self):
        self.print_puzzle()
        self.print_kurang_i()
        self.print_kurang_i_plus_x()

    # Method untuk GUI
    def puzzle_info_gui(self):
        self.puzzle_info+="\nNilai Kurang(i) untuk setiap ubin tersebut: \n"
```

```
        for i in range(8):
            self.puzzle_info+="Kurang("+str(i+1)+") =
"+str(self.kurang_i[i])+"\t\t"
            self.puzzle_info+="Kurang("+str(i+9)+") =
"+str(self.kurang_i[i+8])+"\n"


        self.puzzle_info+= f"\nNilai dari Σ(i=1,16) Kurang(i) + X adalah
{sum(self.kurang_i) + self.value_X}\n"
        if self.is_puzzle_solvable():
            self.puzzle_info+= "\nPuzzle dapat diselesaikan. \n\n"
        else:
            self.puzzle_info+= "\nPuzzle tidak dapat diselesaikan. \n\n"
```

4. Solver.py

   File Solver.py bertanggung jawab untuk mendefinisikan kelas Node dan kelas SolvePuzzle. Kelas Node merupakan kelas yang merepresentasikan simpul dan menampung informasi-informasi sesuai dengan status puzzle. Kelas SolvePuzzle berisi algoritma *branch and bound* untuk menyelesaikan persoalan 15-puzzle.

```
from queue import PriorityQueue
from copy import deepcopy
from Puzzle import Puzzle
import time

class Node:
    def __init__(self, parent, puzzle_obj, level):
        self.parent = deepcopy(parent) # parent node
        self.puzzle_obj = deepcopy(puzzle_obj)
        self.level = level
        self.cost = self.puzzle_obj.difference + self.level

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


class SolvePuzzle:
    # initial_puzzle telah terdefinisi sebagai objek puzzle state awal
    def __init__(self,initial_puzzle):
        # Mencatat waktu awal eksekusi algoritma
        self.start_timer = time.time()
        # Counter simpul yang dibangkitkan
        self.generated_node = 0
        # Inisialisasi flag
        self.is_found = False
        # Inisialisasi steps untuk menyimpan langkah penyelesaian
        self.steps = []
        # Inisialisasi priority queue pada simpul hidup
        live_node = PriorityQueue()
        # Inisialisasi akar yaitu state awal puzzle
```

```
        # parent = None, puzzle = initial_puzzle, level = 0
        root = Node(None, initial_puzzle, 0)
        self.generated_node+=1 # Akar dibangkitkan
        # Enqueue akar
        live_node.put(root)

        while (not (live_node.empty()) and not self.is_found):

            e_node = live_node.get()

            if (e_node.puzzle_obj.difference == 0): # Solusi ditemukan ketika
semua ubin berada pada posisi sesuai nomor (difference = 0)
                self.print_solution(e_node)
                self.end_timer = time.time() # mencatat waktu akhir eksekusi
algoritma
                self.exc_time = self.end_timer - self.start_timer # menghitung
waktu eksekusi algoritma
                self.is_found = True
                print("\nJumlah simpul yang dibangkitkan:", self.generated_node)
                print(f"Waktu eksekusi program: {self.exc_time:.6f} s")

            else:
                for i in range(4):
                    if ( (e_node.puzzle_obj.move_index%2 != i%2) or
(e_node.puzzle_obj.move_index == i) or (e_node.parent == None) ): #
                        moved_puzzle = Puzzle(e_node.puzzle_obj.puzzle, i)
                        if moved_puzzle.move_index!=-1: #perpindahan valid maka
tambahkan simpul
                            child = Node(e_node, moved_puzzle, e_node.level+1)
                            self.generated_node+=1
                            live_node.put(child)


    def print_solution(self, node):
        if(node.parent is None):
            print("Langkah penyelesaian:")
        else:
            self.print_solution(node.parent)
            print()
            # move_index:
            # 0 : up, 1 : right, 2 : down, 3 : left
            direction = ["UP", "RIGHT", "DOWN", "LEFT"]
            if node.puzzle_obj.move_index!=-1:
                self.steps.append(node.puzzle_obj.move_index)
                print(f"Langkah {len(self.steps)}:
{direction[node.puzzle_obj.move_index]}")
                node.puzzle_obj.print_puzzle()
```

5. GUI.py

File GUI.py bertanggung jawab dalam tampilan antarmuka program untuk menampilkan animasi langkah-langkah penyelesaian puzzle.

```
from cgitb import text
from tkinter import *
from tkinter import filedialog as fd
from tkinter.messagebox import showinfo
from time import *
from PuzzleRoot import PuzzleRoot
from Solver import SolvePuzzle

# Inisialisasi GUI
class GUI:
    def __init__(self, window):
        self.window = window
        self.window.title("15 Puzzle Solver")
        self.window.resizable(width=False, height=False)
        self.window.configure(background='#fbf8f3')

        self.img = []
        for i in range(16):
            self.img.append(PhotoImage(file="src/assets/"+str(i+1)+".png"))

        self.label_title = Label(text="15-Puzzle Solver", font=("Courier New",
18, "bold"),fg="#827972", background='#fbf8f3')
        self.label_title.pack(ipadx=10, ipady=10, fill='x',side="top")

        self.filename = ""
        self.load_file_button = Button(
            text="Load Puzzle",
            command=lambda: self.load_file(),
            font=("Courier New", 10, "bold"),
            fg="#827972",
            background='#fbf8f3'
            )
        self.load_file_button.pack(
            ipadx=5, ipady=5
        )

        self.solve_button_clicked = False
        self.solve_button = Button(
            text="Solve Puzzle",
            command=lambda: self.solve_puzzle(),
            font=("Courier New", 10, "bold"),
            fg="#827972",
            background='#fbf8f3',
            state="disabled"
        )
        self.solve_button.pack(
            ipadx=5, ipady=5
        )
        self.solve_button_reload()

        self.label_dir = Label(text="Selected path:\n", font=("Courier New", 9,
"bold"), fg="#827972", background='#fbf8f3')
        self.label_dir.pack(ipadx=10, ipady=10, fill='x',side="top")
```

```
        self.label_dir_reload()

        self.number=[]
        self.canvas = Canvas(self.window, width=400, height=400,
background="#b8aea2")
        self.canvas.pack(side="left")

        self.scrollbar = Scrollbar(self.window)
        self.text_result = Text(self.window, width=50, height = 25,
font=("Courier New", 9, "bold"), fg="#827972", background='#fbf8f3')
        self.scrollbar.pack(side=RIGHT, fill=Y)
        self.text_result.pack(side = "right")
        self.scrollbar.config(command=self.text_result.yview)
        self.text_result.config(yscrollcommand=self.scrollbar.set)
        self.text_result.insert(END,"Result:\n\n")
        self.is_info_printed = False


    def load_file(self):
        self.solve_button_clicked = False
        self.text_result.delete("1.0",END)

        filetypes = (
            ("Text Files", "*.txt"),
        )

        self.filename = fd.askopenfilename(
            title = 'Select puzzle file',
            initialdir='test/',
            filetypes = filetypes)

        self.puzzle_root = PuzzleRoot(self.filename)

        self.text_result.insert(END,"Result: \n\n"
+self.puzzle_root.puzzle_info)

        self.canvas.delete("all")
        for i in range(16):
                self.number.append(
                    PuzzleSquare(self.canvas,
                    self.puzzle_root.number_to_rowcol(i+1)[1]*100,
                    self.puzzle_root.number_to_rowcol(i+1)[0]*100,
                    self.img[i]))

        showinfo(
            title = 'File Selected',
            message = 'You selected: ' + self.filename
        )

        if self.puzzle_root.is_puzzle_solvable():
            showinfo(title = 'Solvable Puzzle', message = 'Puzzle is solvable')
        else:
            showinfo(title = 'Unsolvable Puzzle', message = 'Puzzle is
unsolvable')

    def label_dir_reload(self):
        self.label_dir.configure(text="Selected path:\n" + self.filename)
```

```
        self.label_dir.after(400, self.label_dir_reload)

    def solve_button_reload(self):
        if self.filename != "":
            if self.puzzle_root.is_puzzle_solvable():
                self.solve_button.configure(state="normal")
            else:
                self.solve_button.configure(state="disabled")
        self.solve_button.after(400, self.solve_button_reload)

    def move_16(self,move_index):
        move_x = [0,1,0,-1]
        move_y = [-1,0,1,0]
        moving = [2,3,0,1]
        # move_index:
        # 0 :#  up, 1 : right, 2 : down, 3 : left
        for i in range(16):
            if (self.number[i].y == self.number[15].y+100*move_y[move_index] and
self.number[i].x == self.number[15].x+100*move_x[move_index]):
                self.number[i].move(moving[move_index])
                self.number[15].move(move_index)
                break

    def solve_puzzle(self):
        self.solve_button_clicked = True
        self.text_result.insert(END, "Pencarian solusi dilakukan...\n")
        self.solved = SolvePuzzle(self.puzzle_root)
        self.text_result.insert(END, f"Waktu eksekusi pencarian:
{self.solved.exc_time:.6f} s\n")
        self.text_result.insert(END, f"Jumlah simpul yang dibangkitkan:
{self.solved.generated_node} \n")
        self.text_result.insert(END, "\nLangkah penyelesaian: \n")
        for i in range(len(self.solved.steps)):
            langkah = ["UP","RIGHT","DOWN","LEFT"]
            self.text_result.insert(END,"Langkah ke-"+str(i+1)+":
"+langkah[self.solved.steps[i]]+"\n")
            self.move_16(self.solved.steps[i])
        self.text_result.insert(END, "\nPuzzle berhasil diselesaikan!\n")


class PuzzleSquare:
    def __init__(self, canvas, x, y,image):
        self.canvas = canvas
        self.x = x #column
        self.y = y #row
        self.width = 100
        self.height = 100
        self.image = self.canvas.create_image(self.x+self.width/2,
self.y+self.height/2, image=image)

    def move(self, direction):
        move_x = [0,1,0,-1]
        move_y = [-1,0,1,0]
        x_i = self.x
        y_i = self.y
        while self.x!=x_i+move_x[direction]*100 or
self.y!=y_i+move_y[direction]*100:
```

```
            self.canvas.move(self.image, move_x[direction]*4,
move_y[direction]*4)
            self.x += move_x[direction]*4
            self.y += move_y[direction]*4
            sleep(0.005)
            self.canvas.update()

gui = GUI(Tk())
gui.window.mainloop()
```

Screenshot Input dan *Output*

Untuk menguji kebenaran program, diberikan 5 buah instansiasi persoalan 15-puzzle, dengan 2 kasus tidak dapat diselesaikan dan 3 kasus yang dapat diselesaikan.

1. Persoalan Kasus Dapat Diselesaikan 1

Input: file reachable1.txt

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

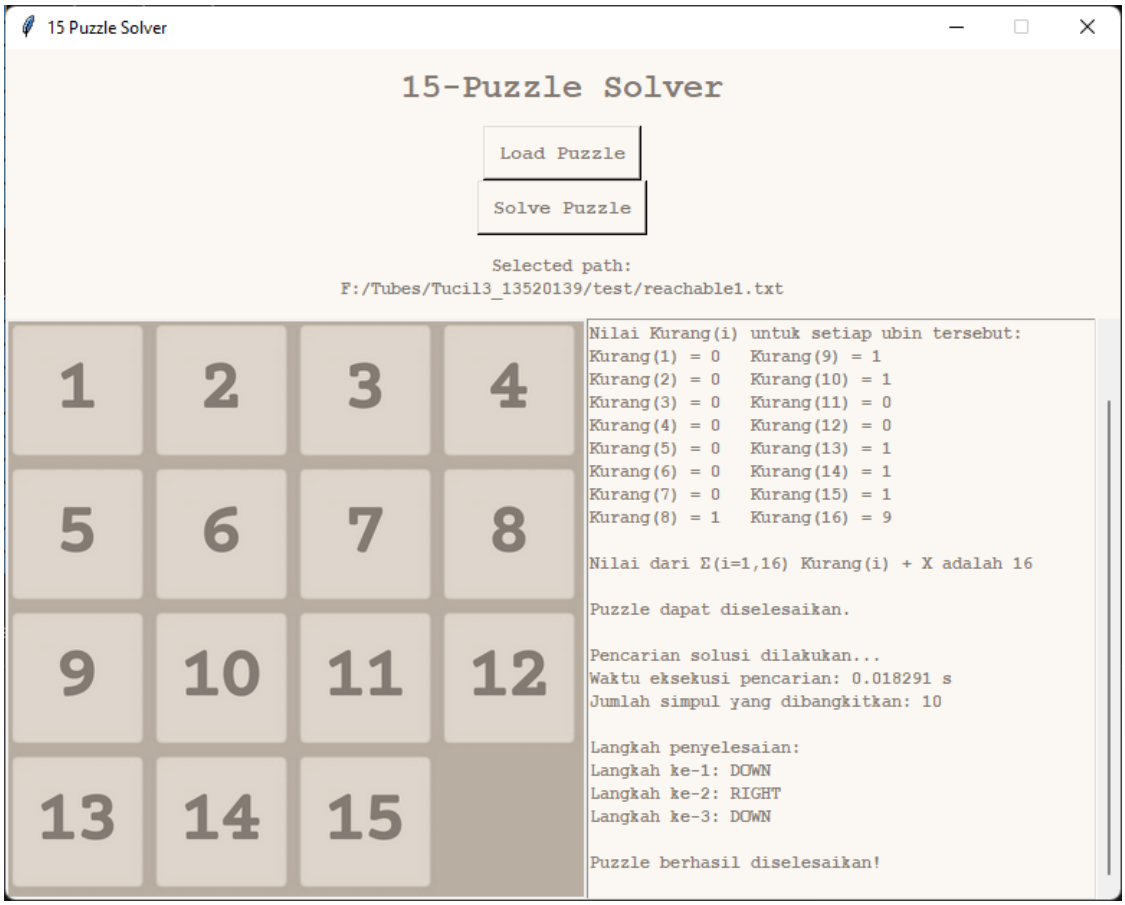
Output:

```
Dijalankan dengan CLI
```

```
15-Puzzle Solver
----------------------------

Masukkan nama file puzzle yang akan diselesaikan (.txt): reachable1.txt

Puzzle berhasil dimuat.
 _____ _____ _____ _____
|     |     |     |     |
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  6  |     |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  10 |  7  |  11 |
|_____|_____|_____|_____|
|     |     |     |     |
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|

Nilai Kurang(i) untuk setiap ubin tersebut:

Kurang(1)=0     Kurang(5)=0     Kurang(9)=1     Kurang(13)=1
Kurang(2)=0     Kurang(6)=0     Kurang(10)=1    Kurang(14)=1
Kurang(3)=0     Kurang(7)=0     Kurang(11)=0    Kurang(15)=1
Kurang(4)=0     Kurang(8)=1     Kurang(12)=0    Kurang(16)=9

Nilai dari Σ(i=1,16) Kurang(i) + X adalah 16

Puzzle dapat diselesaikan.

Langkah penyelesaian:

Langkah 1: DOWN
 _____ _____ _____ _____
|     |     |     |     |
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  6  |  7  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  10 |     |  11 |
|_____|_____|_____|_____|
|     |     |     |     |
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|

Langkah 2: RIGHT
 _____ _____ _____ _____
|     |     |     |     |
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  6  |  7  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  10 |  11 |     |
|_____|_____|_____|_____|
|     |     |     |     |
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|
```

```
Langkah 3: DOWN
 _____ _____ _____ _____
|     |     |     |     |
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  6  |  7  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  10 |  11 |  12 |
|_____|_____|_____|_____|
|     |     |     |     |
|  13 |  14 |  15 |     |
|_____|_____|_____|_____|

Jumlah simpul yang dibangkitkan: 10
Waktu eksekusi program: 0.028739 s
Press any key to continue . . .
```

Dijalankan dengan GUI



2. Persoalan Kasus Dapat Diselesaikan 2

Input: file reachable2.txt

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

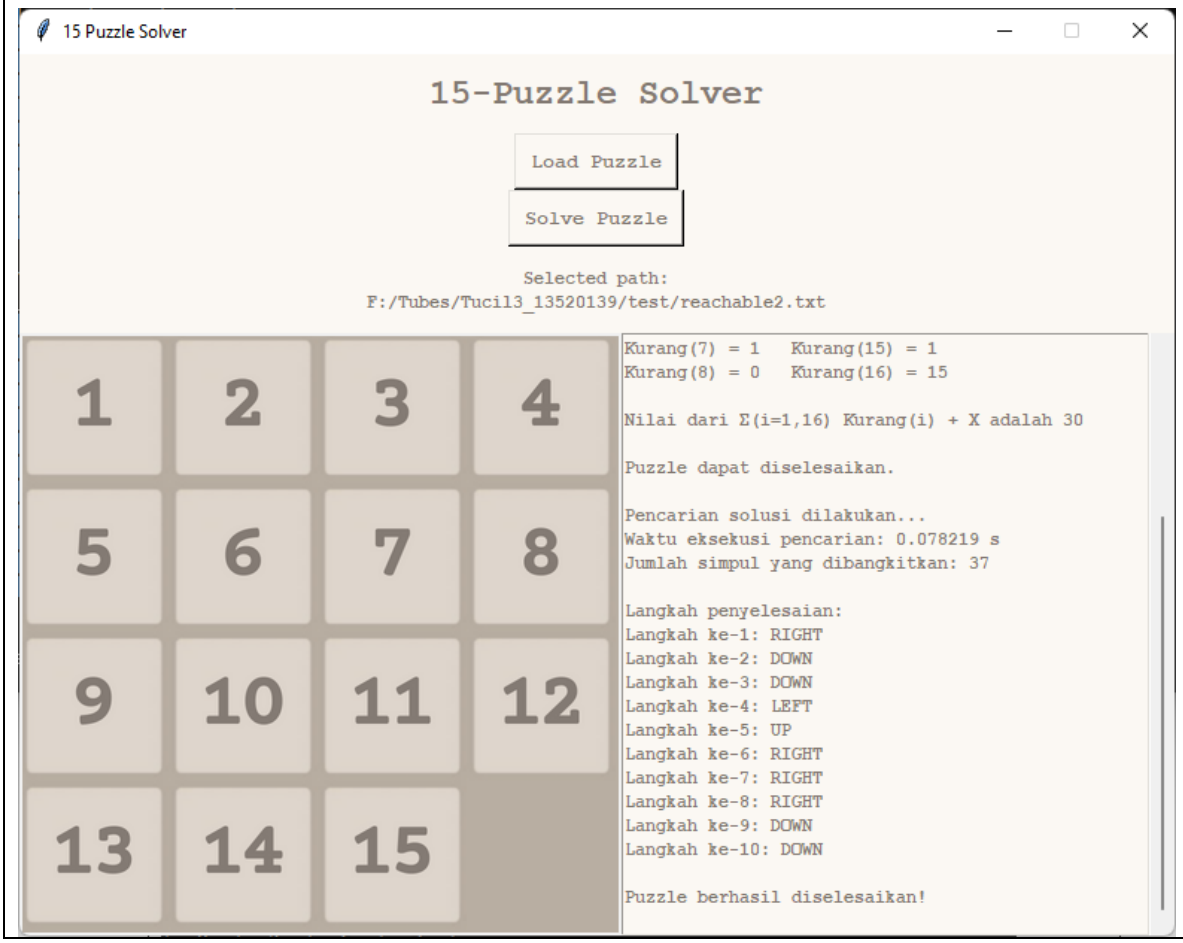
Output:

```
Dijalankan dengan CLI
```

```
15-Puzzle Solver
--------------------------

Masukkan nama file puzzle yang akan diselesaikan (.txt): reachable2.txt

Puzzle berhasil dimuat.
 _____ _____ _____ _____
|     |  1  |  3  |  4  |
|_____|_____|_____|_____|
|  9  |  2  |  6  |  7  |
|_____|_____|_____|_____|
|  10 |  5  |  11 |  8  |
|_____|_____|_____|_____|
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|

Nilai Kurang(i) untuk setiap ubin tersebut:

Kurang(1)=0     Kurang(5)=0     Kurang(9)=5     Kurang(13)=1
Kurang(2)=0     Kurang(6)=1     Kurang(10)=2    Kurang(14)=1
Kurang(3)=1     Kurang(7)=1     Kurang(11)=1    Kurang(15)=1
Kurang(4)=1     Kurang(8)=0     Kurang(12)=0    Kurang(16)=15

Nilai dari Σ(i=1,16) Kurang(i) + X adalah 30

Puzzle dapat diselesaikan.

Langkah penyelesaian:

Langkah 1: RIGHT
 _____ _____ _____ _____
|  1  |     |  3  |  4  |
|_____|_____|_____|_____|
|  9  |  2  |  6  |  7  |
|_____|_____|_____|_____|
|  10 |  5  |  11 |  8  |
|_____|_____|_____|_____|
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|

Langkah 2: DOWN
 _____ _____ _____ _____
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|  9  |     |  6  |  7  |
|_____|_____|_____|_____|
|  10 |  5  |  11 |  8  |
|_____|_____|_____|_____|
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|
```

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

Langkah 8: RIGHT
 _____________________
|     |     |     |     |
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  6  |  7  |     |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  10 |  11 |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|

Langkah 9: DOWN
 _____________________
|     |     |     |     |
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  6  |  7  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  10 |  11 |     |
|_____|_____|_____|_____|
|     |     |     |     |
|  13 |  14 |  15 |  12 |
|_____|_____|_____|_____|

Langkah 10: DOWN
 _____________________
|     |     |     |     |
|  1  |  2  |  3  |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  6  |  7  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  10 |  11 |  12 |
|_____|_____|_____|_____|
|     |     |     |     |
|  13 |  14 |  15 |     |
|_____|_____|_____|_____|

Jumlah simpul yang dibangkitkan: 37
Waktu eksekusi program: 0.083709 s
Press any key to continue . . .
```

Dijalankan dengan GUI



3. Persoalan Kasus Dapat Diselesaikan 3

Input: file reachable3.txt

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

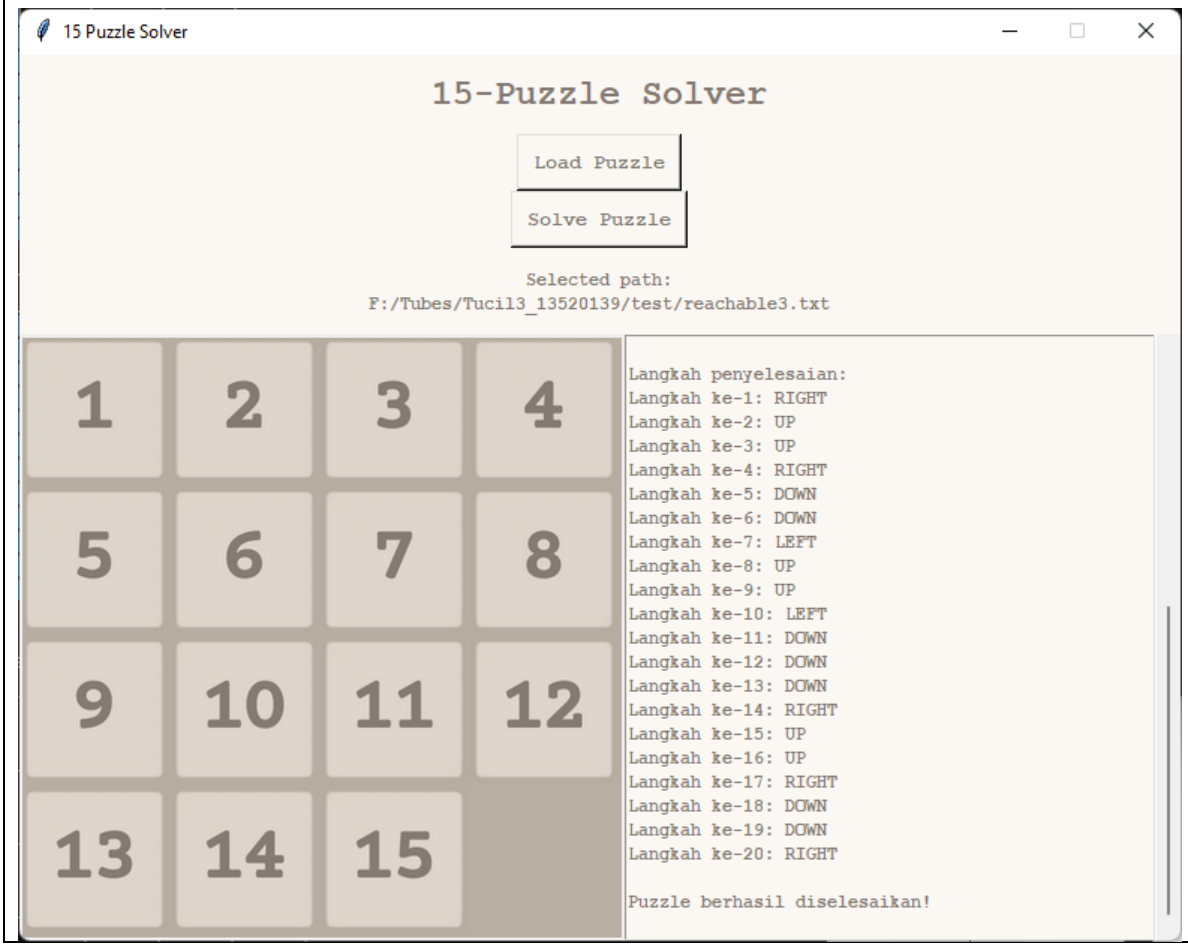
Output:

Dijalankan dengan CLI

```
15-Puzzle Solver
--------------------------

Masukkan nama file puzzle yang akan diselesaikan (.txt): reachable3.txt

Puzzle berhasil dimuat.
 _____ _____ _____ _____
|     |     |     |     |
|  2  |  14 |  10 |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  1  |  7  |  3  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|     |  5  |  6  |  12 |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  13 |  11 |  15 |
|_____|_____|_____|_____|

Nilai Kurang(i) untuk setiap ubin tersebut:

Kurang(1)=0     Kurang(5)=0     Kurang(9)=0     Kurang(13)=1
Kurang(2)=1     Kurang(6)=0     Kurang(10)=8    Kurang(14)=12
Kurang(3)=0     Kurang(7)=3     Kurang(11)=0    Kurang(15)=0
Kurang(4)=2     Kurang(8)=2     Kurang(12)=2    Kurang(16)=7

Nilai dari Σ(i=1,16) Kurang(i) + X adalah 38

Puzzle dapat diselesaikan.

Langkah penyelesaian:

Langkah 1: RIGHT
 _____ _____ _____ _____
|     |     |     |     |
|  2  |  14 |  10 |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  1  |  7  |  3  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |     |  6  |  12 |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  13 |  11 |  15 |
|_____|_____|_____|_____|

Langkah 2: UP
 _____ _____ _____ _____
|     |     |     |     |
|  2  |  14 |  10 |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  1  |     |  3  |  8  |
|_____|_____|_____|_____|
|     |     |     |     |
|  5  |  7  |  6  |  12 |
|_____|_____|_____|_____|
|     |     |     |     |
|  9  |  13 |  11 |  15 |
|_____|_____|_____|_____|
```

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

Langkah 18: DOWN
 _____________________
|      |      |      |      |
|  1   |  2   |  3   |  4   |
|______|______|______|______|
|      |      |      |      |
|  5   |  6   |  7   |  8   |
|______|______|______|______|
|      |      |      |      |
|  9   |  10  |      |  12  |
|______|______|______|______|
|      |      |      |      |
|  13  |  14  |  11  |  15  |
|______|______|______|______|

Langkah 19: DOWN
 _____________________
|      |      |      |      |
|  1   |  2   |  3   |  4   |
|______|______|______|______|
|      |      |      |      |
|  5   |  6   |  7   |  8   |
|______|______|______|______|
|      |      |      |      |
|  9   |  10  |  11  |  12  |
|______|______|______|______|
|      |      |      |      |
|  13  |  14  |      |  15  |
|______|______|______|______|

Langkah 20: RIGHT
 _____________________
|      |      |      |      |
|  1   |  2   |  3   |  4   |
|______|______|______|______|
|      |      |      |      |
|  5   |  6   |  7   |  8   |
|______|______|______|______|
|      |      |      |      |
|  9   |  10  |  11  |  12  |
|______|______|______|______|
|      |      |      |      |
|  13  |  14  |  15  |      |
|______|______|______|______|

Jumlah simpul yang dibangkitkan: 6067
Waktu eksekusi program: 3.458603 s
Press any key to continue . . .
```

Dijalankan dengan GUI



4. Persoalan Kasus Tidak Dapat Diselesaikan 1

Input: file unreachable1.txt

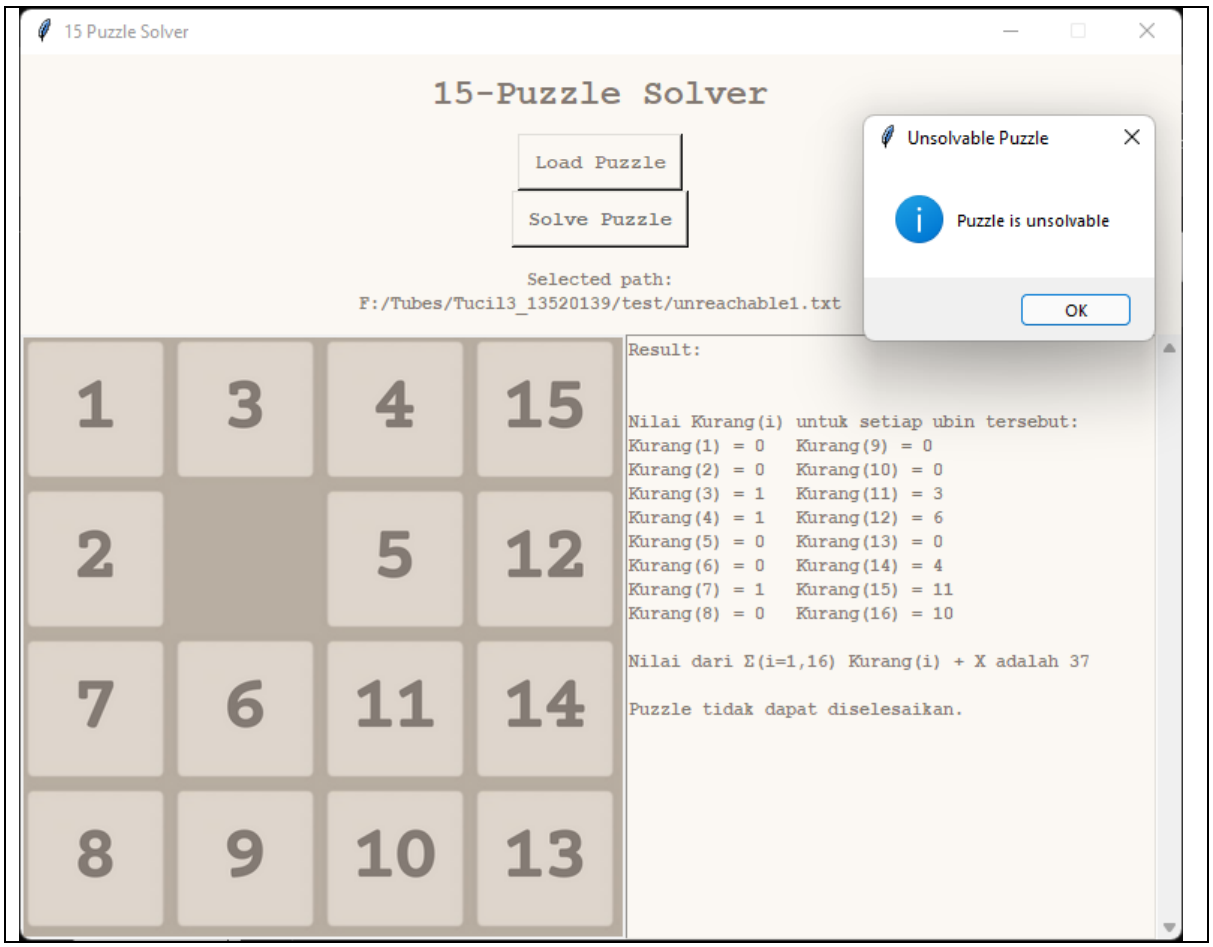
```
1 3 4 15
2 16 5 12
7 6 11 14
8 9 10 13
```

Output:

Dijalankan dengan CLI

```
15-Puzzle Solver
--------------------------

Masukkan nama file puzzle yang akan diselesaikan (.txt): unreachable1.txt

Puzzle berhasil dimuat.
 _____ _____ _____ _____
|     |     |     |     |
|  1  |  3  |  4  |  15 |
|_____|_____|_____|_____|
|     |     |     |     |
|  2  |     |  5  |  12 |
|_____|_____|_____|_____|
|     |     |     |     |
|  7  |  6  |  11 |  14 |
|_____|_____|_____|_____|
|     |     |     |     |
|  8  |  9  |  10 |  13 |
|_____|_____|_____|_____|

Nilai Kurang(i) untuk setiap ubin tersebut:

Kurang(1)=0     Kurang(5)=0     Kurang(9)=0     Kurang(13)=0
Kurang(2)=0     Kurang(6)=0     Kurang(10)=0    Kurang(14)=4
Kurang(3)=1     Kurang(7)=1     Kurang(11)=3    Kurang(15)=11
Kurang(4)=1     Kurang(8)=0     Kurang(12)=6    Kurang(16)=10

Nilai dari Σ(i=1,16) Kurang(i) + X adalah 37

Puzzle tidak dapat diselesaikan.

Press any key to continue . . .
```

Dijalankan dengan GUI



5. Persoalan Kasus Tidak Dapat Diselesaikan 2

Input: file unreachable2.txt

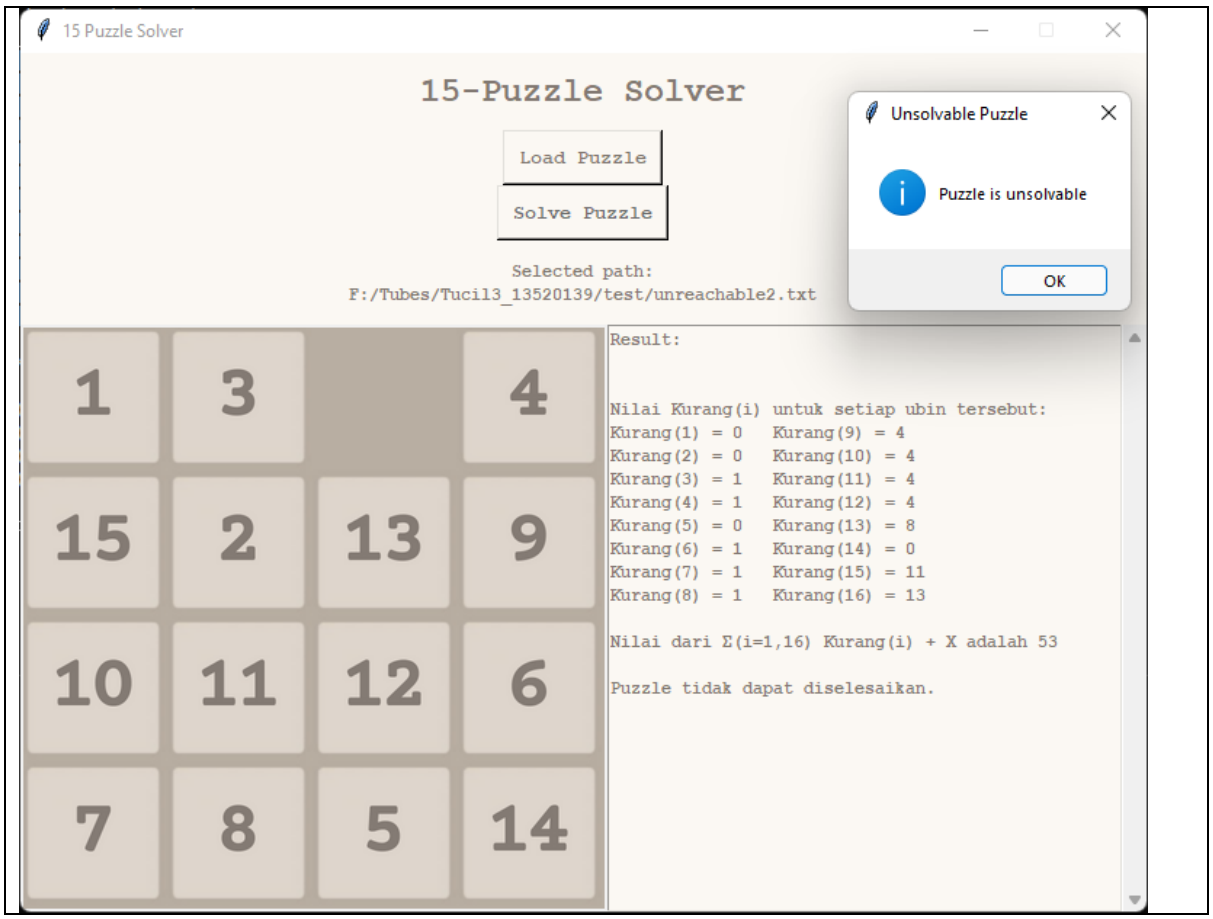
```
1 3 16 4
15 2 13 9
10 11 12 6
7 8 5 14
```

Output:

Dijalankan dengan CLI

```
15-Puzzle Solver
--------------------------

Masukkan nama file puzzle yang akan diselesaikan (.txt): unreachable2.txt

Puzzle berhasil dimuat.
 _____ _____ _____ _____
|     |     |     |     |
|  1  |  3  |     |  4  |
|_____|_____|_____|_____|
|     |     |     |     |
|  15 |  2  |  13 |  9  |
|_____|_____|_____|_____|
|     |     |     |     |
|  10 |  11 |  12 |  6  |
|_____|_____|_____|_____|
|     |     |     |     |
|  7  |  8  |  5  |  14 |
|_____|_____|_____|_____|

Nilai Kurang(i) untuk setiap ubin tersebut:

Kurang(1)=0     Kurang(5)=0     Kurang(9)=4     Kurang(13)=8
Kurang(2)=0     Kurang(6)=1     Kurang(10)=4    Kurang(14)=0
Kurang(3)=1     Kurang(7)=1     Kurang(11)=4    Kurang(15)=11
Kurang(4)=1     Kurang(8)=1     Kurang(12)=4    Kurang(16)=13

Nilai dari Σ(i=1,16) Kurang(i) + X adalah 53

Puzzle tidak dapat diselesaikan.

Press any key to continue . . .
```

Dijalankan dengan GUI


15 Puzzle Solver
15-Puzzle Solver
Load Puzzle
Solve Puzzle
Selected path:
F:/Tubes/Tucil3_13520139/test/unreachable2.txt
Unsolvable Puzzle
Puzzle is unsolvable
OK
1 3 4
15 2 13 9
10 11 12 6
7 8 5 14
Result:
Nilai Kurang(i) untuk setiap ubin tersebut:
Kurang(1) = 0 Kurang(9) = 4
Kurang(2) = 0 Kurang(10) = 4
Kurang(3) = 1 Kurang(11) = 4
Kurang(4) = 1 Kurang(12) = 4
Kurang(5) = 0 Kurang(13) = 8
Kurang(6) = 1 Kurang(14) = 0
Kurang(7) = 1 Kurang(15) = 11
Kurang(8) = 1 Kurang(16) = 13
Nilai dari Σ(i=1,16) Kurang(i) + X adalah 53
Puzzle tidak dapat diselesaikan.

Alamat *Repository* Kode Program

Github: https://github.com/dennisheraldi/Tucil3_13520139

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	✓	