

# Laporan Tugas Kecil Strategi Algoritma 2021/2022 15-Puzzle Solver



Disusun Oleh:

Andreas Indra Kurniawan 13520091/K-1

# Contents

BAB 1 Penjelasan Teori .....	3
a. Cara Kerja Program .....	3
b. Penjelasan Fungsi Solvable .....	3
c. Penjelasan <i>Cost</i> dari Setiap Simpul .....	4
BAB II Kode Program .....	5
1. File Puzzle.py .....	5
2. File puzzleSolver.py .....	10
3. File puzzleSolverGUI.py .....	11
4. File prioQueue.py .....	17
5. File Utility.py .....	18
BAB III Pengujian .....	20
a. Screen-shot Input-Output Program .....	20
b. Berkas Teks Data Uji .....	26
1. Data uji 1(15 <i>move</i> ) .....	26
2. Data uji 2(10 <i>move</i> ) .....	26
3. Data uji 3(tidak bisa di solve) .....	27
4. Data uji 4(tidak bisa di solve) .....	28
5. Data uji 5(20 <i>move</i> ) .....	28
LAMPIRAN .....	29
a. Checklist .....	29
b. Link Github .....	29

# BAB 1

## Penjelasan Teori

### a. Cara Kerja Program

Program menggunakan algoritma branch and bound yang membangkitkan simpul sesuai dengan cost yang dihasilkan.

Langkah-langkah program:

1. Program menerima input puzzle
2. Cek state awal *puzzle* menggunakan fungsi  $\sum_{i=1}^{16} \text{kurang}(i) + X$  dimana  $X$  merupakan posisi dari *space* kosong. Jika hasil dari  $\sum_{i=1}^{16} \text{kurang}(i) + X$  bernilai ganjil maka puzzle tidak dapat diselesaikan, jika bernilai genap maka dapat diselesaikan.
3. Bangkitkan *child node* dari simpul awal berdasarkan *possible move* dari puzzle
4. Cek jika puzzle pernah berada di state yang sama dengan menggunakan hashtable
5. Hitung cost dari masing-masing *move*
6. Masukkan setiap simpul yang dibangkitkan ke priorityQueue berdasarkan costnya
7. Bangkitkan simpul selanjutnya dari queue hingga menemukan puzzle yang terselesaikan

### b. Penjelasan Fungsi Solvable

$\sum_{i=1}^{16} \text{kurang}(i) + X$  adalah fungsi yang digunakan untuk mengecek apakah sebuah puzzle solvable atau tidak. *kurang(i)* merupakan jumlah angka lebih kecil dari  $i$  yang muncul setelah angka  $i$ . Sebagai contoh pada puzzle berikut.

Puzzle:				Kurang(i):	
1	2	3	4	Kurang(1): 0	Kurang(9): 0
6	10	7	8	Kurang(2): 0	Kurang(10): 4
5	13		12	Kurang(3): 0	Kurang(11): 0
9	14	11	15	Kurang(4): 0	Kurang(12): 2
				Kurang(5): 0	Kurang(13): 3
				Kurang(6): 1	Kurang(14): 1
				Kurang(7): 1	Kurang(15): 0
				Kurang(8): 1	Kurang(16): 5
<div>Up</div> <div>LeftRight</div> <div>Down</div> <div>Solve</div>				Kurang(i) + X: 18	
				Generated Node: 56	
				Time: 0.0020034313201904297	
				Total Moves: 10	

kurang(6) memiliki nilai 1 karena angka 5 muncul setelah angka 6. Pembacaan muncul setelah suatu angka disini dibaca dari atas ke bawah dan kiri ke kanan.

c. Penjelasan *Cost* dari Setiap Simpul

Berdasarkan slide perkuliahan dan spek tugas kecil Strategi Algoritma, cost yang digunakan disini merupakan misplaced tiles + kedalaman simpul. Misplaced tiles disini berarti angka yang tidak sesuai dengan tempatnya. Dengan mengambil contoh pada gambar di atas terdapat 8 misplaced tiles yaitu 6 10 5 13 12 9 11 dan 15.

## BAB II

### Kode Program

#### 1. File Puzzle.py

```
1. import copy
2. from Utility import *
3.
4. Moves = {"U" : 0, "L" : 1, "R" : 2, "D" : 3}
5. Slot = {1 : [0,0], 2 : [0,1], 3 : [0,2], 4 : [0,3], 5 : [1,0], 6 :
    [1,1], 7 : [1,2], 8 : [1,3], 9 : [2,0], 10 : [2,1], 11 : [2,2], 12 :
    [2,3], 13 : [3,0], 14 : [3,1], 15 : [3,2], 16 : [3,3]}
6.
7. class Puzzle:
8.     def __init__(self, puzzle, indexKosong, moves, depth):
9.         self.puzzle = copy.deepcopy(puzzle)
10.        self.indexKosong = [indexKosong[0], indexKosong[1]]
11.        self.moves = copy.deepcopy(moves)
12.        # self.loopMove = copy.deepcopy(loopMove)
13.        self.possibleMoves = []
14.        self.depth = depth
15.        self.lastMove = ""
16.        self.stage = 1
17.
18.        #HEURISTIK YANG TIDAK JADI DIGUNAKAN
19.        def checkStage(self):
20.            if(self.stage >= 1 and self.puzzle[0][0] == 1 and
    self.puzzle[0][1] == 2):
21.                self.stage = 2
22.            if(self.stage >= 2 and self.puzzle[0][2] == 3 and
    self.puzzle[0][3] == 4):
23.                self.stage = 3
24.            if(self.stage >= 3 and self.puzzle[1][0] == 5 and
    self.puzzle[1][1] == 6):
25.                self.stage = 4
26.            if(self.stage >= 4 and self.puzzle[1][2] == 7 and
    self.puzzle[1][3] == 8):
27.                self.stage = 5
28.            if(self.stage >= 5 and self.puzzle[2][0] == 9 and
    self.puzzle[3][0] == 13):
29.                self.stage = 6
30.            if(self.stage >= 6 and self.puzzle[2][1] == 10 and
    self.puzzle[2][2] == 11 and self.puzzle[2][3] == 12):
31.                self.stage = 7
32.
33.        #MEMINDAHKAN SPACE KOSONG PADA PUZZLE
34.        def moveSpace(self, move):
35.            if(move == "U"):
```

```

36.         self.puzzle[self.indexKosong[0]][self.indexKosong[1]] =
            self.puzzle[self.indexKosong[0]-1][self.indexKosong[1]]
37.         self.puzzle[self.indexKosong[0]-1][self.indexKosong[1]] =
16
38.         self.indexKosong[0] -= 1
39.         # self.LoopMove.append("U")
40.         self.moves.append("U")
41.         self.lastMove = "D"
42.         # self.clearLoop("U")
43.     elif(move == "L"):
44.         self.puzzle[self.indexKosong[0]][self.indexKosong[1]] =
            self.puzzle[self.indexKosong[0]][self.indexKosong[1]-1]
45.         self.puzzle[self.indexKosong[0]][self.indexKosong[1]-1] =
16
46.         self.indexKosong[1] -= 1
47.         # self.LoopMove.append("L")
48.         self.moves.append("L")
49.         self.lastMove = "R"
50.         # self.clearLoop("L")
51.     elif(move == "R"):
52.         self.puzzle[self.indexKosong[0]][self.indexKosong[1]] =
            self.puzzle[self.indexKosong[0]][self.indexKosong[1]+1]
53.         self.puzzle[self.indexKosong[0]][self.indexKosong[1]+1] =
16
54.         self.indexKosong[1] += 1
55.         # self.LoopMove.append("R")
56.         self.moves.append("R")
57.         self.lastMove = "L"
58.         # self.clearLoop("R")
59.     elif(move == "D"):
60.         self.puzzle[self.indexKosong[0]][self.indexKosong[1]] =
            self.puzzle[self.indexKosong[0]+1][self.indexKosong[1]]
61.         self.puzzle[self.indexKosong[0]+1][self.indexKosong[1]] =
16
62.         self.indexKosong[0] += 1
63.         # self.LoopMove.append("D")
64.         self.moves.append("D")
65.         self.lastMove = "U"
66.         # self.clearLoop("D")
67.
68.         self.depth += 1
69.
70.     #MENUNJUKAN SEQUENCE PUZZLE HINGGA PENYELESAIAN
71.     def moveAndPrint(self):
72.         for i in range(len(self.moves)):
73.             self.moveSpace(self.moves[i])
74.             print("Move ke-",i+1," : ",self.moves[i])
75.             printPuzzle(self.puzzle)

```

```

76.         print()
77.
78.     #MENGECEK SELURUH POSSIBLE MOVE
79.     def generatePossibleMove(self):
80.         possibleMove = []
81.         up = False
82.         down = False
83.         left = False
84.         right = False
85.         if(self.indexKosong[0] != 0):
86.             up = True
87.         if(self.indexKosong[1] != 0):
88.             left = True
89.         if(self.indexKosong[0] != 3):
90.             down = True
91.         if(self.indexKosong[1] != 3):
92.             right = True
93.         if(up):
94.             self.possibleMoves.append("U")
95.         if(left):
96.             self.possibleMoves.append("L")
97.         if(right):
98.             self.possibleMoves.append("R")
99.         if(down):
100.             self.possibleMoves.append("D")
101.             if(self.lastMove in self.possibleMoves):
102.                 self.possibleMoves.remove(self.lastMove)
103.
104.     def clearPossibleMoves(self):
105.         self.possibleMoves = []
106.
107.     #MENGECEK JIKA PUZZLE SUDAH SELESAI
108.     def isFinalState(self):
109.         for i in range(4):
110.             for j in range(4):
111.                 if(self.puzzle[i][j] != i*4+j+1):
112.                     return False
113.         return True
114.
115.     #MENCARI NILAI DARI K
116.     def KValue(self):
117.         if((self.indexKosong[0]+self.indexKosong[1])% 2 == 0):
118.             return 0
119.         else:
120.             return 1
121.
122.     #MENGECEK KURANG(i)
123.     def checkKurang(self, val):

```

```

124.         checking = False
125.         counter = 0
126.         for i in range(4):
127.             for j in range(4):
128.                 if(self.puzzle[i][j] == val):
129.                     checking = True
130.                 elif(checking):
131.                     if(self.puzzle[i][j]<val):
132.                         counter += 1
133.         return counter
134.
135.         #MENGECEK JIKA SUATU PUZZLE REACHABLE
136.         def checkReachable(self,arrKurang):
137.             k = self.KValue()
138.             reachable = 0
139.             for i in range(1,17): #MENGECEK FUNGSI KURANG(i) DARI 2
                HINGGA 15 KARENA 1 SUDAH PASTI 0 NILAINYA
140.                 kurang = self.checkKurang(i)
141.                 arrKurang.append(kurang)
142.                 reachable += kurang
143.                 arrKurang.append(reachable+k)
144.                 # print("K = ", k, "reachable = ", reachable)
145.                 return (reachable+k)%2 == 0
146.
147.         #MERESET LOOPMOVE
148.         def clearLoop(self,move):
149.             if(len(self.loopMove) < 4 and move in self.loopMove):
150.                 self.loopMove = []
151.             elif(len(self.loopMove) == 4 and move !=
                self.loopMove[0]):
152.                 self.loopMove = []
153.             elif(len(self.loopMove) == 5):
154.                 self.loopMove = []
155.
156.         def printInfo(self):
157.             print("Puzzle: ")
158.             printPuzzle(self.puzzle)
159.             print("Possible Move: ", self.possibleMoves)
160.             # print("Loop Move: ", self.loopMove)
161.             print("Index Kosong: ", self.indexKosong)
162.             # print("Moves: ", self.moves)
163.             print("Last move: ", self.lastMove)
164.             print()
165.
166.         def getNumIndex(self,num):
167.             for i in range(4):
168.                 for j in range(4):
169.                     if(self.puzzle[i][j] == num):

```



```

170.             return [i,j]
171.
172.         def isNear(indexNum,indexKosong):
173.             if(indexNum[0] == indexKosong[0]+1 or indexNum[0] ==
indexKosong[0]-1 or indexNum[1] == indexKosong[1]+1 or indexNum[1] ==
indexKosong[1]-1):
174.                 return True
175.                 return False
176.
177.         def isInPosition(index,num):
178.             return index[0] + index[1]+1 != num
179.
180.         #MENGHASILKAN COST DARI MOVE
181.         def getMoveCost(self):
182.
183.
184.
185.             # CARA PPT
186.             cost = 0
187.             for i in range(4):
188.                 for j in range(4):
189.                     if self.puzzle[i][j] != i*4+j+1 and
self.puzzle[i][j] != 16:
190.                         cost += 1
191.             return cost + self.depth
192.
193.             # MANHATTAN DISTANCE
194.             # cost = 0
195.             # for i in range(1,16):
196.             #     index = self.getNumIndex(i)
197.             #     cost += abs(index[0] - Slot[i][0]) + abs(index[1] -
Slot[i][1])
198.             # return cost + self.depth
199.
200.         #MENCOBA BERGERAK KE SEGALA ARAH DAN JIKA PUZZLE BELUM PERNAH
BERADA PADA SATE TERTENTU, MAKA AKAN DIHITUNG COSTNYA DAN
DIPERHITUNGKAN KE QUEUE
201.         def generateChildNode(self,PrioQueue,exist):
202.             for i in range(len(self.possibleMoves)):
203.                 puzzleCheck =
Puzzle(self.puzzle,self.indexKosong,self.moves, self.depth)
204.                 puzzleCheck.moveSpace(self.possibleMoves[i])
205.                 string1D = puzzleToString(puzzleCheck.puzzle)
206.                 if(not string1D in exist):
207.                     cost = puzzleCheck.getMoveCost()
208.                     PrioQueue.enqueue((cost,puzzleCheck))
209.
210.                 exist[string1D] = puzzleCheck.depth

```

211.

212.

## 2. File puzzleSolver.py

```
1. import copy
2. from Utility import *
3. import Puzzle as Pz
4. import PriorityQueue as PQ
5. import time
6.
7. #IMPLEMENTASI ALGORITMA BRANCH AND BOUND
8. def BranchAndBound(puzzle):
9.     Pq = PQ.PriorityQueue()
10.    exist = {puzzleToString(puzzle.puzzle) : puzzle.depth}
11.
12.    while(not puzzle.isFinalState()):
13.        puzzle.generatePossibleMove()
14.        puzzle.generateChildNode(Pq,exist)
15.        puzzle = Pq.dequeue()[1]
16.    return puzzle, len(exist)
17.
18. #PRINT PENGECEKAN PUZZLE SOLVABLE ATAU TIDAK DENGAN RUMUS Kurang(i) + X
19. def printKurang(arr):
20.     print("Kurang(i):")
21.     total = 0
22.     for i in range(len(arr)-1):
23.         print("Kurang("+ str(i+1) +") =", arr[i])
24.         total += arr[i]
25.     print("Total Kurang(i) + X = ", arr[len(arr)-1])
26.     print()
27.
28. if(__name__ == "__main__"):
29.     name = input("Masukkan nama file(tanpa ekstensi): ")
30.     name = "test\\" + name + ".txt"
31.     matPuzzle, indexKosong = parsePuzzle(name)
32.     moveSeq = []
33.     arrKurang = []
34.     Puzzle = Pz.Puzzle(matPuzzle,indexKosong,moveSeq, 0)
35.     PuzzleCheck =
        Pz.Puzzle(Puzzle.puzzle,Puzzle.indexKosong,Puzzle.moves, Puzzle.depth)
36.     PuzzleCheck.generatePossibleMove()
37.     PuzzleCheck.printInfo()
38.
39.     if(PuzzleCheck.checkReachable(arrKurang)):
40.         printKurang(arrKurang)
41.         start = time.time()
42.         solvedPuzzle,generatedNode = BranchAndBound(PuzzleCheck)
43.         end = time.time()
```

```

44.         print("Waktu pencarian: ", end-start)
45.         print("Node yang di generate: ", generatedNode)
46.         Puzzle.moves = copy.deepcopy(solvedPuzzle.moves)
47.         Puzzle.moveAndPrint()
48.     else:
49.         print("Puzzle Unsolvable")

```

### 3. File puzzleSolverGUI.py

```

1. import tkinter as tk
2. from tkinter import filedialog as fd
3. from tkinter.messagebox import showinfo
4. import copy
5. import PuzzleSolver as PS
6. import Puzzle as P
7. import time
8. from Utility import *
9.
10.puzzle = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15, ' ']
11.puzzleCheck = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]
12.indexKosong = [3,3]
13.kurang = ['', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '']
14.moves = []
15.generated = ''
16.waktu = 0
17.totalMoves = 0
18.
19.# IMPORT FILE
20.def openFile():
21.    filetypes = [
22.        ('Text files', '*.txt')
23.    ]
24.    global indexKosong, puzzle
25.    try:
26.        filename = fd.askopenfilename(title='Open a
        file',filetypes=filetypes)
27.
28.        with open(filename,"r") as f:
29.            datas = f.readlines()
30.            count = 0
31.            for line in datas:
32.                num = line.split()
33.                for i in num:
34.                    if(int(i) == 16):
35.                        puzzle[count] = 16
36.                        indexKosong = [count//4,count%4]
37.                    else:
38.                        puzzle[count] = int(i)
39.                count += 1

```

```

40.         updatePuzzle()
41.
42.     except:
43.         showinfo('Error', 'Invalid puzzle')
44.
45. #UPDATE PUZZLE PADA GUI
46. def updatePuzzle():
47.     global puzzle, puzzleCheck
48.     puzzleCheck = copy.deepcopy(puzzle)
49.     puzzle[indexKosong[0]*4 + indexKosong[1]] = ' '
50.     l1.config(text=puzzle[0])
51.     l2.config(text=puzzle[1])
52.     l3.config(text=puzzle[2])
53.     l4.config(text=puzzle[3])
54.     l5.config(text=puzzle[4])
55.     l6.config(text=puzzle[5])
56.     l7.config(text=puzzle[6])
57.     l8.config(text=puzzle[7])
58.     l9.config(text=puzzle[8])
59.     l10.config(text=puzzle[9])
60.     l11.config(text=puzzle[10])
61.     l12.config(text=puzzle[11])
62.     l13.config(text=puzzle[12])
63.     l14.config(text=puzzle[13])
64.     l15.config(text=puzzle[14])
65.     l16.config(text=puzzle[15])
66.
67. #UPDATE LABEL KURANG
68. def updateKurang():
69.     global kurang
70.     lKurang1.config(text="Kurang(1): " + str(kurang[0]))
71.     lKurang2.config(text="Kurang(2): " + str(kurang[1]))
72.     lKurang3.config(text="Kurang(3): " + str(kurang[2]))
73.     lKurang4.config(text="Kurang(4): " + str(kurang[3]))
74.     lKurang5.config(text="Kurang(5): " + str(kurang[4]))
75.     lKurang6.config(text="Kurang(6): " + str(kurang[5]))
76.     lKurang7.config(text="Kurang(7): " + str(kurang[6]))
77.     lKurang8.config(text="Kurang(8): " + str(kurang[7]))
78.     lKurang9.config(text="Kurang(9): " + str(kurang[8]))
79.     lKurang10.config(text="Kurang(10): " + str(kurang[9]))
80.     lKurang11.config(text="Kurang(11): " + str(kurang[10]))
81.     lKurang12.config(text="Kurang(12): " + str(kurang[11]))
82.     lKurang13.config(text="Kurang(13): " + str(kurang[12]))
83.     lKurang14.config(text="Kurang(14): " + str(kurang[13]))
84.     lKurang15.config(text="Kurang(15): " + str(kurang[14]))
85.     lKurang16.config(text="Kurang(16): " + str(kurang[15]))
86.     lkurangx.config(text="Kurang(i) + X: " + str(kurang[16]))
87.

```

```

88. # COMMAND PERGERAKAN
89. def moveUp():
90.     global puzzle, indexKosong
91.     if(indexKosong[0] == 0):
92.         showinfo('Error', 'Move out of bound')
93.     else:
94.         puzzle[indexKosong[0]*4 + indexKosong[1]] =
puzzle[indexKosong[0]*4 + indexKosong[1] - 4]
95.         puzzle[indexKosong[0]*4 + indexKosong[1] - 4] = 16
96.         indexKosong[0] -= 1
97.         updatePuzzle()
98.
99. def moveDown():
100.    global puzzle, indexKosong
101.    if(indexKosong[0] == 3):
102.        showinfo('Error', 'Move out of bound')
103.    else:
104.        puzzle[indexKosong[0]*4 + indexKosong[1]] =
puzzle[indexKosong[0]*4 + indexKosong[1] + 4]
105.        puzzle[indexKosong[0]*4 + indexKosong[1] + 4] = 16
106.        indexKosong[0] += 1
107.        updatePuzzle()
108.
109.    def moveLeft():
110.        global puzzle, indexKosong
111.        if(indexKosong[1] == 0):
112.            showinfo('Error', 'Move out of bound')
113.        else:
114.            puzzle[indexKosong[0]*4 + indexKosong[1]] =
puzzle[indexKosong[0]*4 + indexKosong[1] - 1]
115.            puzzle[indexKosong[0]*4 + indexKosong[1] - 1] = 16
116.            indexKosong[1] -= 1
117.            updatePuzzle()
118.
119.    def moveRight():
120.        global puzzle, indexKosong
121.        if(indexKosong[1] == 3):
122.            showinfo('Error', 'Move out of bound')
123.        else:
124.            puzzle[indexKosong[0]*4 + indexKosong[1]] =
puzzle[indexKosong[0]*4 + indexKosong[1] + 1]
125.            puzzle[indexKosong[0]*4 + indexKosong[1] + 1] = 16
126.            indexKosong[1] += 1
127.            updatePuzzle()
128.
129.    # SOLVE PUZZLE DENGAN BRANCH AND BOUND
130.    def solvePuzzle():

```

```

131.         global puzzleCheck, indexKosong, moves, kurang, generated,
           waktu, totalMoves
132.         objPuzzle = P.Puzzle(puzzleCheck, indexKosong, [], 0)
133.         objPuzzle.puzzle = make2DPuzzle(objPuzzle.puzzle)
134.         kurang = []
135.         start = time.time()
136.         if(objPuzzle.checkReachable(kurang)):
137.             updateKurang()
138.             objPuzzle, generated = PS.BranchAndBound(objPuzzle)
139.             end = time.time()
140.             waktu = end - start
141.             moves = objPuzzle.moves
142.             lGeneratedNode.config(text="Generated Node: "+
           str(generated))
143.             totalMoves = len(moves)
144.             lTotalMoves.config(text="Total Moves: "+ str(totalMoves))
145.             lTime.config(text="Time: "+ str(waktu))
146.             autoSolve()
147.         else:
148.             updateKurang()
149.             showinfo('Check Result', 'Puzzle is not solvable')
150.
151.         #BERGERAK OTOMATIS UNTUK MENYELESAIKAN PUZZLE
152.         def autoSolve():
153.             global moves
154.             for i in moves:
155.                 if(i == "U"):
156.                     moveUp()
157.                 elif(i=="D"):
158.                     moveDown()
159.                 elif(i=="L"):
160.                     moveLeft()
161.                 elif(i=="R"):
162.                     moveRight()
163.             frame.update()
164.             time.sleep(0.5)
165.
166.         def saveFile():
167.             global puzzleCheck
168.             f = fd.asksaveasfile(initialfile = 'Untitled.txt',
           defaultextension=".txt", filetypes=[("All Files", "*.*"), ("Text
           Documents", "*.txt")])
169.             for i in range(16):
170.                 f.write(str(puzzleCheck[i])+" ")
171.                 if(i%4==3):
172.                     f.write("\n")
173.
174.         def resetPuzzle():

```

```
175.         global puzzle, indexKosong
176.         puzzle = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]
177.         indexKosong = [3,3]
178.         updatePuzzle()
179.
180.     frame = tk.Tk()
181.
182.     menu = tk.Menu(frame)
183.     frame.config(menu=menu)
184.     fileMenu = tk.Menu(menu, tearoff = 0)
185.     menu.add_cascade(Label='Import/Export Puzzle', menu=fileMenu)
186.     fileMenu.add_command(Label='Reset Puzzle', command=resetPuzzle)
187.     fileMenu.add_command(Label='Save Puzzle', command=saveFile)
188.     fileMenu.add_command(Label='Load Puzzle', command=openFile)
189.
190.     lPuzzle = tk.Label(frame, text='Puzzle:')
191.
192.     l1 = tk.Label(frame, text=puzzle[0], font="Helvetica 16 bold",
193.         width=5)
194.     l2 = tk.Label(frame, text=puzzle[1], font="Helvetica 16 bold",
195.         width=5)
196.     l3 = tk.Label(frame, text=puzzle[2], font="Helvetica 16 bold",
197.         width=5)
198.     l4 = tk.Label(frame, text=puzzle[3], font="Helvetica 16 bold",
199.         width=5)
200.     l5 = tk.Label(frame, text=puzzle[4], font="Helvetica 16 bold",
201.         width=5)
202.     l6 = tk.Label(frame, text=puzzle[5], font="Helvetica 16 bold",
203.         width=5)
204.     l7 = tk.Label(frame, text=puzzle[6], font="Helvetica 16 bold",
205.         width=5)
206.     l8 = tk.Label(frame, text=puzzle[7], font="Helvetica 16 bold",
207.         width=5)
208.     l9 = tk.Label(frame, text=puzzle[8], font="Helvetica 16 bold",
209.         width=5)
210.     l10 = tk.Label(frame, text=puzzle[9], font="Helvetica 16 bold",
211.         width=5)
212.     l11 = tk.Label(frame, text=puzzle[10], font="Helvetica 16 bold",
213.         width=5)
214.     l12 = tk.Label(frame, text=puzzle[11], font="Helvetica 16 bold",
215.         width=5)
216.     l13 = tk.Label(frame, text=puzzle[12], font="Helvetica 16 bold",
217.         width=5)
218.     l14 = tk.Label(frame, text=puzzle[13], font="Helvetica 16 bold",
219.         width=5)
220.     l15 = tk.Label(frame, text=puzzle[14], font="Helvetica 16 bold",
221.         width=5)
```

```

207.     l16 = tk.Label(frame, text=puzzle[15], font="Helvetica 16 bold",
208.         width=5)
209.     lKurang = tk.Label(frame, text="Kurang(i):")
210.     lKurang.grid(row=0, column=7)
211.     lKurang1 = tk.Label(frame, text="Kurang(1): " + kurang[0])
212.     lKurang1.grid(row=1, column=7)
213.     lKurang2 = tk.Label(frame, text="Kurang(2): " + kurang[1])
214.     lKurang2.grid(row=2, column=7)
215.     lKurang3 = tk.Label(frame, text="Kurang(3): " + kurang[2])
216.     lKurang3.grid(row=3, column=7)
217.     lKurang4 = tk.Label(frame, text="Kurang(4): " + kurang[3])
218.     lKurang4.grid(row=4, column=7)
219.     lKurang5 = tk.Label(frame, text="Kurang(5): " + kurang[4])
220.     lKurang5.grid(row=5, column=7)
221.     lKurang6 = tk.Label(frame, text="Kurang(6): " + kurang[5])
222.     lKurang6.grid(row=6, column=7)
223.     lKurang7 = tk.Label(frame, text="Kurang(7): " + kurang[6])
224.     lKurang7.grid(row=7, column=7)
225.     lKurang8 = tk.Label(frame, text="Kurang(8): " + kurang[7])
226.     lKurang8.grid(row=8, column=7)
227.     lKurang9 = tk.Label(frame, text="Kurang(9): " + kurang[8])
228.     lKurang9.grid(row=1, column=8)
229.     lKurang10 = tk.Label(frame, text="Kurang(10): " + kurang[9])
230.     lKurang10.grid(row=2, column=8)
231.     lKurang11 = tk.Label(frame, text="Kurang(11): " + kurang[10])
232.     lKurang11.grid(row=3, column=8)
233.     lKurang12 = tk.Label(frame, text="Kurang(12): " + kurang[11])
234.     lKurang12.grid(row=4, column=8)
235.     lKurang13 = tk.Label(frame, text="Kurang(13): " + kurang[12])
236.     lKurang13.grid(row=5, column=8)
237.     lKurang14 = tk.Label(frame, text="Kurang(14): " + kurang[13])
238.     lKurang14.grid(row=6, column=8)
239.     lKurang15 = tk.Label(frame, text="Kurang(15): " + kurang[14])
240.     lKurang15.grid(row=7, column=8)
241.     lKurang16 = tk.Label(frame, text="Kurang(16): " + kurang[15])
242.     lKurang16.grid(row=8, column=8)
243.     lkurangx = tk.Label(frame, text="Kurang(i) + X: " + kurang[16])
244.     lkurangx.grid(row=9, column=7, columnspan=2)
245.
246.     lPuzzle.grid(row=0, column=0, columnspan=4)
247.
248.     lGeneratedNode = tk.Label(frame, text="Generated Node: " +
249.         generated)
250.
251.     lTime = tk.Label(frame, text="Time: " + str(waktu))
252.     lTime.grid(row=11, column=7, columnspan=2)

```



```

253.
254.     lTotalMoves = tk.Label(frame, text="Total Moves: " +
    str(totalMoves))
255.     lTotalMoves.grid(row=12, column=7, columnspan=2)
256.
257.     l1.grid(row=1, column=3)
258.     l2.grid(row=1, column=4)
259.     l3.grid(row=1, column=5)
260.     l4.grid(row=1, column=6)
261.     l5.grid(row=2, column=3)
262.     l6.grid(row=2, column=4)
263.     l7.grid(row=2, column=5)
264.     l8.grid(row=2, column=6)
265.     l9.grid(row=3, column=3)
266.     l10.grid(row=3, column=4)
267.     l11.grid(row=3, column=5)
268.     l12.grid(row=3, column=6)
269.     l13.grid(row=4, column=3)
270.     l14.grid(row=4, column=4)
271.     l15.grid(row=4, column=5)
272.     l16.grid(row=4, column=6)
273.
274.     bUp = tk.Button(frame, text='Up', command=moveUp, width=10)
275.     bDown = tk.Button(frame, text='Down', command=moveDown, width=10)
276.     bLeft = tk.Button(frame, text='Left', command=moveLeft, width=10)
277.     bRight = tk.Button(frame, text='Right', command=moveRight,
    width=10)
278.
279.     bUp.grid(row = 6, column = 4, columnspan=2)
280.     bLeft.grid(row = 7, column = 4)
281.     bRight.grid(row = 7, column = 5)
282.     bDown.grid(row = 8, column = 4, columnspan=2)
283.
284.     bSolve = tk.Button(frame, text='Solve', command=solvePuzzle,
    width=20)
285.     bSolve.grid(row = 9, column = 3, columnspan=4)
286.
287.     frame.mainloop()

```

#### 4. File prioQueue.py

```

1. class PrioQueue:
2.     def __init__(self):
3.         self.Queue = []
4.         self.length = 0
5.
6.     def enqueue(self, Puzzle):
7.         if(self.length == 0):
8.             self.Queue.insert(0,Puzzle)

```

```

9.         else:
10.             i = 0
11.             while(i < self.length and Puzzle[0] > self.Queue[i][0]):
12.                 i += 1
13.                 self.Queue.insert(i, Puzzle)
14.             self.length += 1
15.
16.     def dequeue(self):
17.         if(self.length == 0):
18.             print("Queue kosong")
19.             self.length -= 1
20.             return self.Queue.pop(0)
21.
22.

```

## 5. File Utility.py

```

1. import random as rd
2.
3. #IMPORT PUZZLE DARI FILE TXT
4. def parsePuzzle(filename):
5.     puzzle = []
6.     with open(filename) as f:
7.         for line in f:
8.             puzzle.append([int(x) for x in line.split()])
9.
10.    for i in range(4):
11.        for j in range(4):
12.            if puzzle[i][j] == 16:
13.                indexKosong = [i,j]
14.                break
15.    return puzzle, indexKosong
16.
17. #DIGUNAKAN UNTUK MENGGENERATE PUZZLE
18. def generatePuzzle():
19.     puzzle = []
20.     i = 0
21.     while(i != 16):
22.         rand = rd.randint(1,16)
23.         while(rand in puzzle): #cek apakah sudah ada di puzzle
24.             rand = rd.randint(1,16)
25.         puzzle.append(rand)
26.         i += 1
27.     matPuzzle = []
28.     k = 0
29.     indexKosong = 0
30.     for i in range(4): #membuat puzzle
31.         matPuzzle.append([])
32.         for j in range(4):

```

```

33.         if puzzle[j+k] == 16: #mencari space kosong
34.             indexKosong = [int(k/4),j] #menyimpan index kosong
35.             matPuzzle[i].append(puzzle[j+k])
36.             # print(matPuzzle, i, j)
37.         k += 4
38.     return matPuzzle, indexKosong
39.
40. #PRINT STATE PUZZLE
41. def printPuzzle(puzzle):
42.     for i in range(4):
43.         for j in range(4):
44.             if(puzzle[i][j] == 16):
45.                 print('space'.ljust(5), end=' ')
46.             else:
47.                 print(str(puzzle[i][j]).ljust(5), end=" ")
48.         print()
49.
50. #MENGUBAH PUZZLE MENJADI STRING
51. def puzzleToString(puzzle):
52.     string = ""
53.     for i in range(4):
54.         for j in range(4):
55.             string += str(puzzle[i][j])
56.     return string
57.
58. #MENGUBAH 2D PUZZLE MENJADI 1D PUZZLE
59. def make1DPuzzle(Puzzle):
60.     puzzle = [0 for i in range(16)]
61.     for i in range(4):
62.         for j in range(4):
63.             puzzle[i*4+j] = Puzzle[i][j]
64.     return puzzle
65.
66. #MEMBUAT 1D PUZZLE MENJADI 2D PUZZLE
67. def make2DPuzzle(Puzzle):
68.     puzzle = [[0 for i in range(4)] for j in range(4)]
69.     for i in range(4):
70.         for j in range(4):
71.             puzzle[i][j] = Puzzle[i*4+j]
72.     return puzzle
73.

```

## BAB III

### Pengujian

a. Screen-shot Input-Output Program

a. Contoh memilih input file

```
Masukkan nama file(tanpa ekstensi): test2
Puzzle:
1      2      3      4
5      6      space 8
9      10     7      11
13     14     15     12
Possible Move: ['U', 'L', 'R', 'D']
Index Kosong: [1, 2]
Last move:
```

b. Contoh Input dari file

```
test2.txt X
test2.txt
1      1 2 3 4
2      5 6 16 8
3      9 10 7 11
4      13 14 15 12
```

c. Output program

```
Puzzle:
1      2      3      4
5      6      space 8
9      10     7      11
13     14     15     12
Possible Move: ['U', 'L', 'R', 'D']
Index Kosong: [1, 2]
Last move:
```

```
Less move 1

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

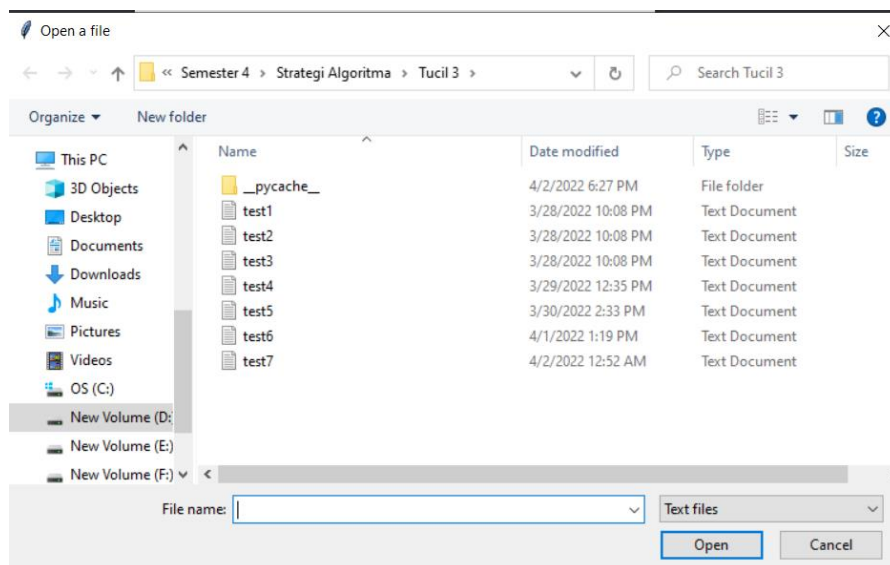
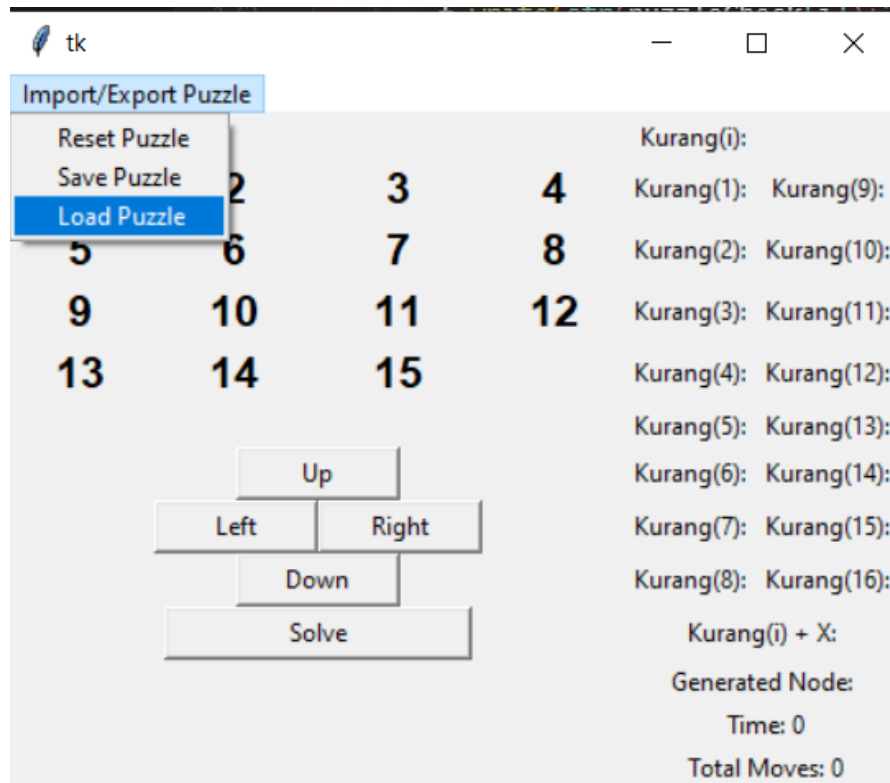
Waktu pencarian: 0.0
Node yang di generate: 10
Move ke- 1 : D
1    2    3    4
5    6    7    8
9    10   space 11
13   14   15   12
```


```
Waktu pencarian: 0.0
Node yang di generate: 10
Move ke- 1 : D
1    2    3    4
5    6    7    8
9    10   space 11
13   14   15   12

Move ke- 2 : R
1    2    3    4
5    6    7    8
9    10   11   space
13   14   15   12

Move ke- 3 : D
1    2    3    4
5    6    7    8
9    10   11   12
13   14   15   space
```

d. Input dari GUI



 tk

Import/Export Puzzle

Puzzle:

1234

56

910711

13141512

Up

LeftRight

Down

Solve

Kurang(i):

Kurang(1):Kurang(9):

Kurang(2):Kurang(10):

Kurang(3):Kurang(11):

Kurang(4):Kurang(12):

Kurang(5):Kurang(13):

Kurang(6):Kurang(14):

Kurang(7):Kurang(15):

Kurang(8):Kurang(16):

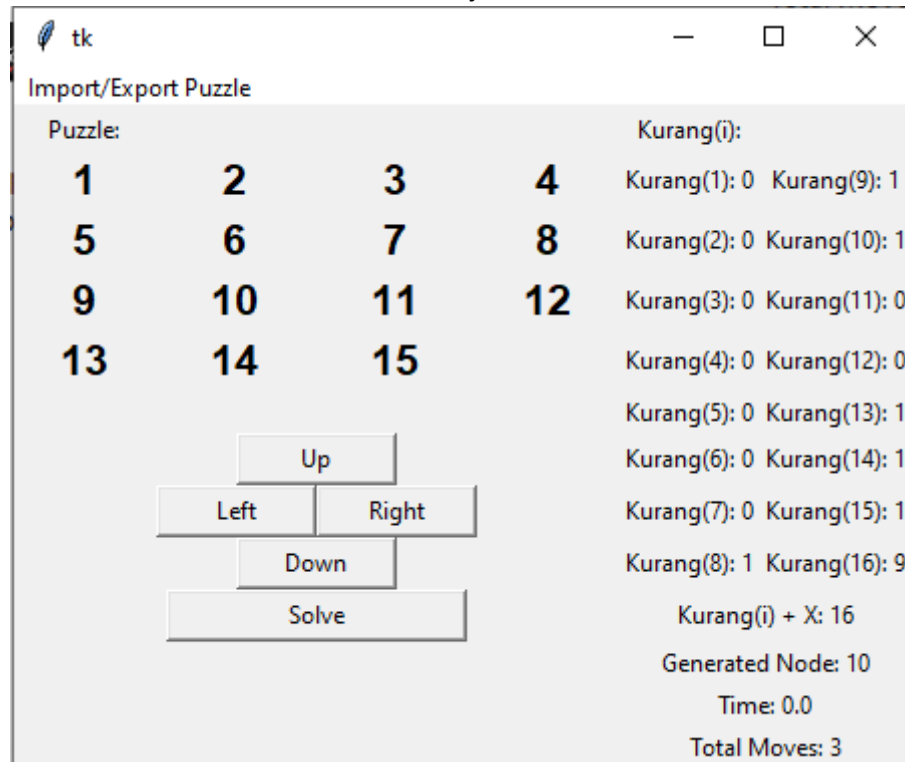
Kurang(i) + X:

Generated Node:

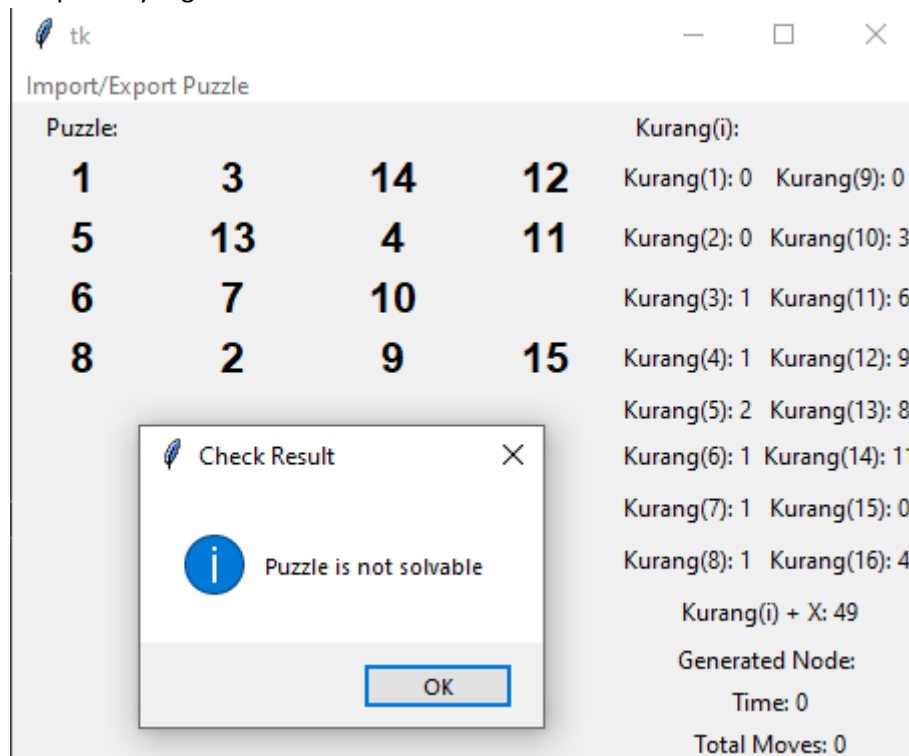
Time: 0

Total Moves: 0

e. Output setelah menekan Solve dan animasi berjalan

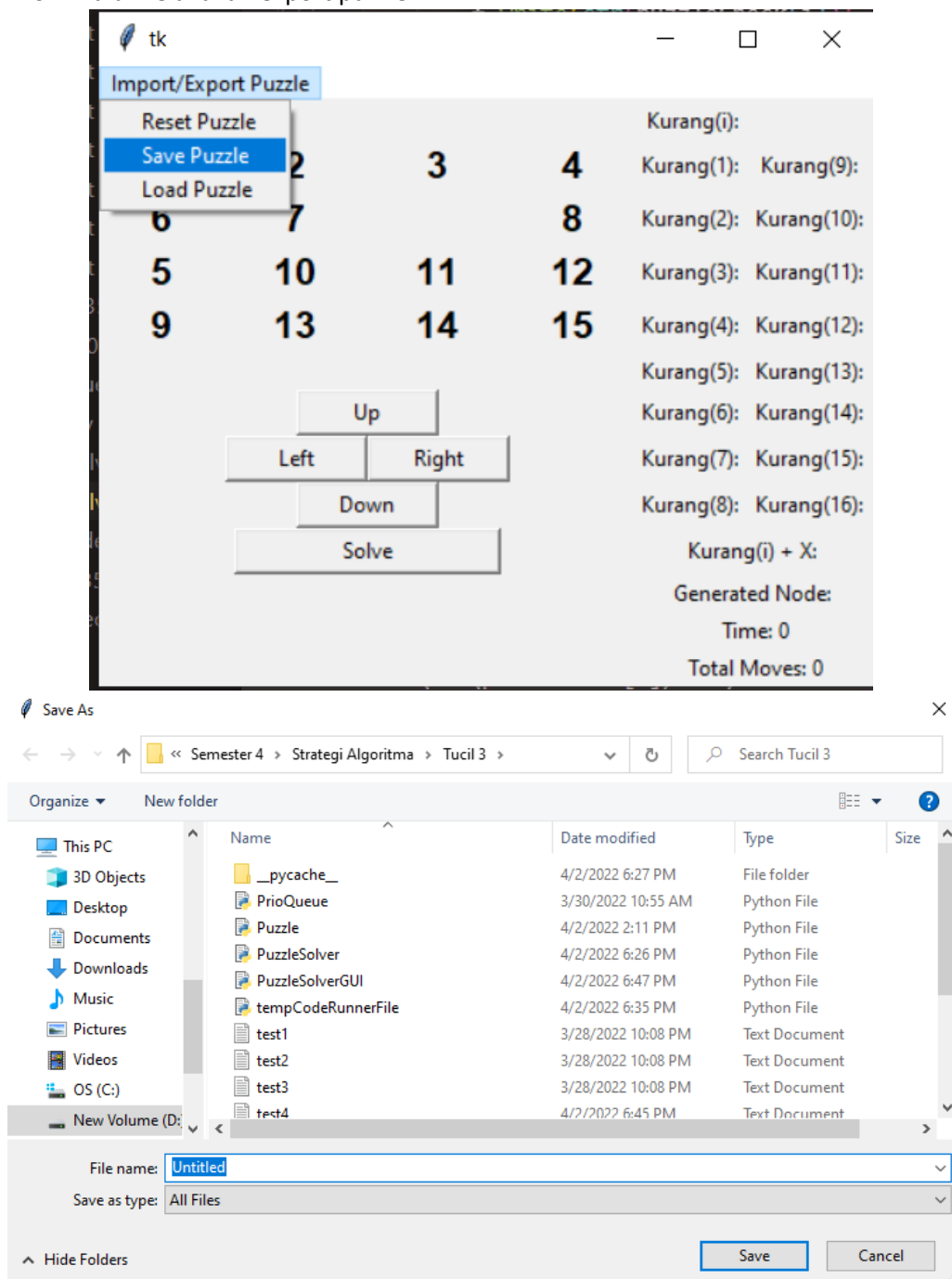


f. Output dari puzzle yang tidak bisa di solve





- g. Pengguna juga bisa membuat test case nya sendiri dengan menekan Button Up Left Right dan Down lalu melakukan export puzzle



b. Berkas Teks Data Uji

1. Data uji 1(15 move)

Puzzle:			
<b>5</b>	<b>1</b>	<b>2</b>	<b>4</b>
<b>6</b>	<b>3</b>	<b>8</b>	<b>11</b>
<b>9</b>	<b>14</b>	<b>7</b>	
<b>13</b>	<b>15</b>	<b>10</b>	<b>12</b>

Puzzle:		Kurang(i):
<b>1</b>	<b>2</b>	Kurang(1): 0 Kurang(9): 1
<b>5</b>	<b>6</b>	Kurang(2): 0 Kurang(10): 0
<b>9</b>	<b>10</b>	Kurang(3): 0 Kurang(11): 3
<b>13</b>	<b>14</b>	Kurang(4): 1 Kurang(12): 0
	<b>15</b>	Kurang(5): 4 Kurang(13): 2
		Kurang(6): 1 Kurang(14): 4
		Kurang(7): 0 Kurang(15): 2
		Kurang(8): 1 Kurang(16): 4
		Kurang(i) + X: 24
		Generated Node: 194
		Time: 0.005995750427246094
		Total Moves: 15

Up

Left

Right

Down

Solve

2. Data uji 2(10 move)

Puzzle:			
<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>6</b>	<b>10</b>	<b>7</b>	<b>8</b>
<b>5</b>	<b>13</b>		<b>12</b>
<b>9</b>	<b>14</b>	<b>11</b>	<b>15</b>

Puzzle:

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

Up

LeftRight

Down

Solve

Kurang(i):

Kurang(1): 0

Kurang(9): 0

Kurang(2): 0

Kurang(10): 4

Kurang(3): 0

Kurang(11): 0

Kurang(4): 0

Kurang(12): 2

Kurang(5): 0

Kurang(13): 3

Kurang(6): 1

Kurang(14): 1

Kurang(7): 1

Kurang(15): 0

Kurang(8): 1

Kurang(16): 5

Kurang(i) + X: 18

Generated Node: 56

Time: 0.0019960403442382812

Total Moves: 10

3. Data uji 3(tidak bisa di solve)

Puzzle:

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

Kurang(i):

Kurang(1): 0

Kurang(9): 0

Kurang(2): 0

Kurang(10): 0

Kurang(3): 1

Kurang(11): 3

Kurang(4): 1

Kurang(12): 6

Kurang(5): 0

Kurang(13): 0

Kurang(6): 0

Kurang(14): 4

Kurang(7): 1

Kurang(15): 11

Kurang(8): 0

Kurang(16): 10

Kurang(i) + X: 37

Generated Node: 0

Time: 0

Total Moves: 0

Check Result

i

Puzzle is not solvable

OK

4. Data uji 4(tidak bisa di solve)

Puzzle:

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

Kurang(i):

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

Kurang(i) + X: 49

Generated Node: 0

Time: 0

Total Moves: 0

Check Result

i Puzzle is not solvable

OK

5. Data uji 5(20 move)

Puzzle:

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

Kurang(i):

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

Kurang(i) + X: 38

Generated Node: 3907

Time: 0.3383042812347412

Total Moves: 20

Up

Down

Left

Right

Solve

## LAMPIRAN

### a. Checklist

Poin	Ya	Tidak
1. <u>Program berhasil dikompilasi</u>	V	
2. Program berhasil <i>running</i>	V	
3. Program dapat menerima input dan menuliskan output.	V	
4. Luaran sudah benar untuk semua data uji	V	
5. Bonus dibuat	V	

### b. Link Github

<https://github.com/IMYELI/15-Puzzle-Solver.git>