LAPORAN TUGAS KECIL 3

IF2211 STRATEGI ALGORITMA SEMESTER II TAHUN 2021/2022

PENYELESAIAN PERSOALAN 15-PUZZLE DENGAN ALGORITMA BRANCH AND BOUND

DANIEL SALIM
13520008
K02

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

DAFTAR ISI

BAB I
DASAR TEORI

1.1 ALGORITMA BRANCH AND BOUND

Algoritma Branch and Bound sering digunakan dalam persoalan optimasi, yaitu dengan meminimumkan suatu fungsi objektif, yang tidak melanggar batasan (*constraints)* persoalan. Algoritma Branch and Bound secara umum merupakan algoritma BFS yang diintegrasikan dengan *least cost search.* Dalam Algoritma Branch and Bound setiap simpul diberi sebuah nilai cost. Simpul berikutnya yang akan di-expand tidak lagi berdasarkan urutan pembangkitannya, tetapi simpul yang memiliki *cost* yang paling kecil (*least cost search)* - pada kasus minimasi.

1.2 CARA PENYELESAIAN

Dalam permainan 15-puzzle, cara untuk memenangkan atau memecahkan puzzle tersebut adalah dengan mengubah susunan awal puzzle yang diketahui menjadi susunan akhir seperti gambar dibawah, dengan state berdasarkan ubin kosong dan aksi untuk mencapai susunan akhir adalah *up, down, left,* dan *right.*

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

(a) Susunan awal

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

(b) Susunan akhir

Sumber : rinaldimunir

Karena terdapat 16 sel maka ada sebanyak 16! Susunan ubin yang berbeda dan hanya setengahnya yang dapat dicapai dengan susunan awal sembarang. Terdapat teorema yang dipakai yaitu status tujuan atau susunan akhir hanya dapat dicapai dari

susunan awal jika $\sum_{i=1}^{16} KURANG(i) + X$ bernilai genap. X = 1 jika sel yang kosong pada state awal berada pada sel yang diarsir.

Sumber : rinaldimunir

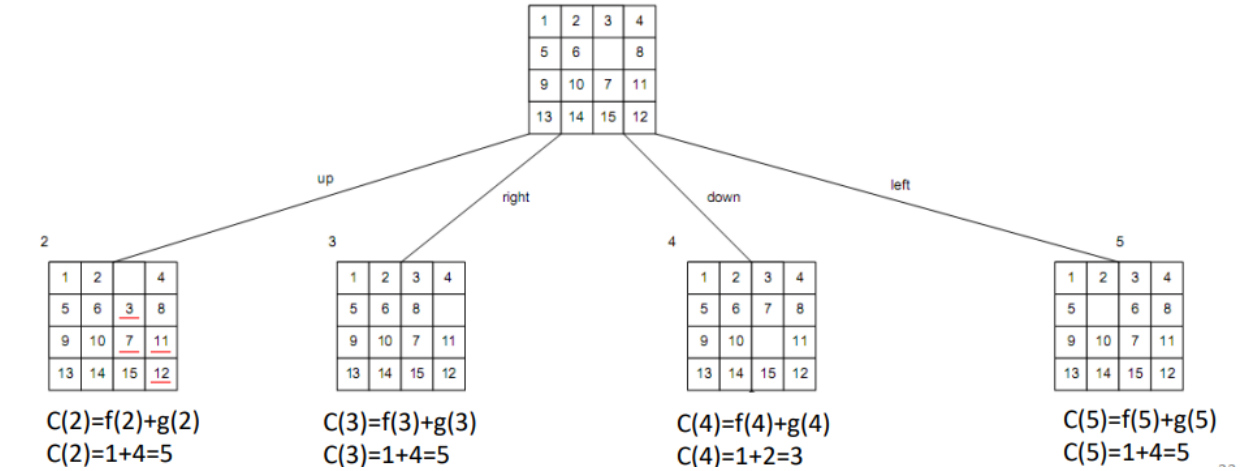
KURANG(i) merupakan banyaknya ubin bernomor j sedemikian sehingga j < i dan POSISI(j) > POSISI(i). POSISI(i) = posisi ubin bernomor i pada susunan yang diperiksa. Contohnya sebagai berikut, KURANG(4) = 1 karena hanya ada 1 sel yang letaknya setelah posisi sel-4 yang nilainya kurang dari 4 yaitu sel-2, sehingga KURANG(4) = 1. Pada contoh dibawah, jumlah semua KURANG(i) adalah 37. Karena 37 merupakan bilangan ganjil, maka dari susunan awal dibawah tidak dapat mencapai tujuan akhir.

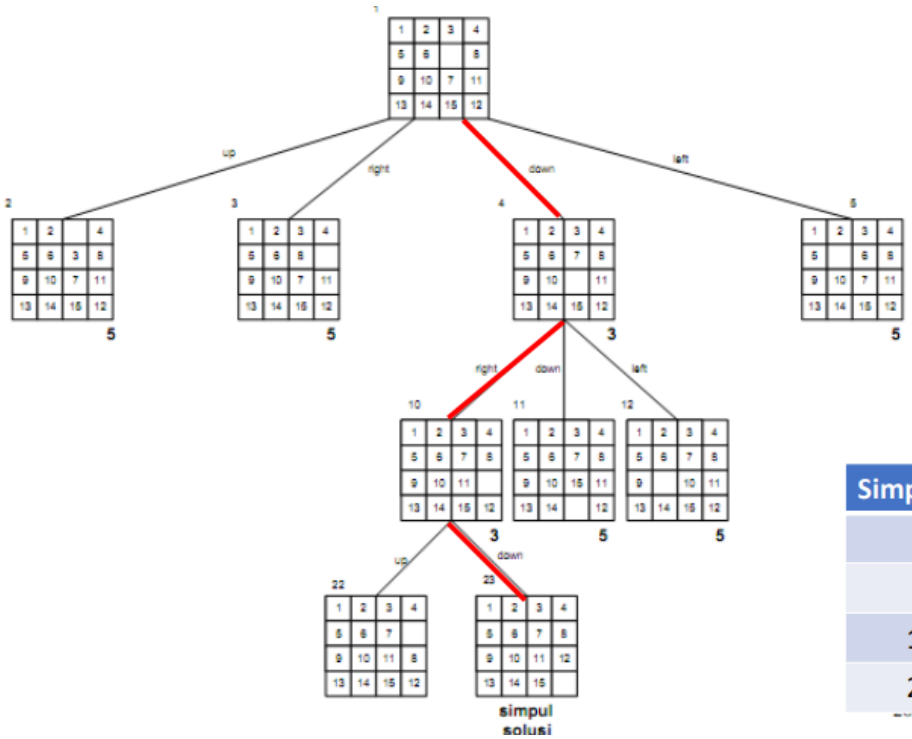
i	Kurang (i)
1	0
2	0
3	1
4	1
5	0
6	0
7	1
8	0
9	0
10	0
11	3
12	6
13	0
14	4
15	11
16	10

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

Sumber : rinaldimuni

Dalam penentuan cost dari simpul hidupnya adalah jumlah ubin tidak kosong yang tidak terdapat pada susunan akhir. Seperti ilustrasi di bawah.





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

Pembentukan Pohon Ruang Status 15-Puzzle dengan Algoritma Branch & Bound
Sumber : [rinaldimunir]

BAB II
INPUT-OUTPUT PROGRAM

1. Input Tc1.txt

```
PS D:\USER\IF Semester 4\Stima\tucil3\15puzzlesolver\src> py main.py
------ Selamat datang ke Permaiann 15-Puzzle ------

Masukkan nama fle puzzle yang akan dijalankan: (contoh: puzzle.txt)
Nama File : tc1.txt

Puzzle :

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



Nilai KURANG(i) dalam setiap sel yang tidak kosong:

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

Nilai X =  1

Jumlah(KURANG(i)) + X =  15
Maaf :( Puzzle ini tidak dapat diselesaikan
```

2. Input Tc2.txt

```
------ Selamat datang ke Permaiann 15-Puzzle ------

Masukkan nama fle puzzle yang akan dijalankan: (contoh: puzzle.txt)
Nama File : tc2.txt

Puzzle :

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



Nilai KURANG(i) dalam setiap sel yang tidak kosong:

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

Nilai X =  0

Jumlah(KURANG(i)) + X =  37
Maaf :( Puzzle ini tidak dapat diselesaikan
---------------------------------------
```

3. Input tc3.txt

```
PS D:\USER\IF Semester 4\Stima\tucil3\15puzzlesolver\src> py main.py
------ Selamat datang ke Permaiann 15-Puzzle ------

Masukkan nama fle puzzle yang akan dijalankan: (contoh: puzzle.txt)
Nama File : tc3.txt

Puzzle :

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



Nilai KURANG(i) dalam setiap sel yang tidak kosong:

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

Nilai X =  0

Jumlah(KURANG(i)) + X =  2
Puzzle ini dapat diselesaikan :)

Langkah-langkah untuk menyelesaikan puzzle:

State awal :
1       2       3       4
5       6       7       8
9       10      11      12
13      -       14      15
```

```
Nilai X =  0

Jumlah(KURANG(i)) + X =  2
Puzzle ini dapat diselesaikan :)

Langkah-langkah untuk menyelesaikan puzzle:

State awal :
1       2       3       4
5       6       7       8
9       10      11      12
13      -       14      15



Berpindah :  Right
1       2       3       4
5       6       7       8
9       10      11      12
13      14      -       15


Berpindah :  Right
1       2       3       4
5       6       7       8
9       10      11      12
13      14      15      -


State anak yang dihasilkan =  5
Waktu yang dibutuhkan =  0.005991697311401367 detik
----------------------------------------
```

4. Input tc4.txt

```
PS D:\USER\IF Semester 4\Stima\tucil3\15puzzlesolver\src> py main.py
------ Selamat datang ke Permaiann 15-Puzzle ------

Masukkan nama fle puzzle yang akan dijalankan: (contoh: puzzle.txt)
Nama File : tc4.txt

Puzzle :

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



Nilai KURANG(i) dalam setiap sel yang tidak kosong:

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

Nilai X =  0

Jumlah(KURANG(i)) + X =  12
Puzzle ini dapat diselesaikan :)

Langkah-langkah untuk menyelesaikan puzzle:

State awal :
1       2       3       4
5       6       7       -
9       10      12      8
11      13      14      15
```

```
State awal :
1       2       3       4
5       6       7       -
9       10      12      8
11      13      14      15


Berpindah :  Down
1       2       3       4
5       6       7       8
9       10      12      -
11      13      14      15


Berpindah :  Left
1       2       3       4
5       6       7       8
9       10      -       12
11      13      14      15


Berpindah :  Down
1       2       3       4
5       6       7       8
9       10      14      12
11      13      -       15


Berpindah :  Left
1       2       3       4
5       6       7       8
9       10      14      12
11      -       13      15


Berpindah :  Left
1       2       3       4
5       6       7       8
9       10      14      12
-       11      13      15
```

```
Berpindah :  Up
1       2       3       4
5       6       7       8
-       10      14      12
9       11      13      15


Berpindah :  Right
1       2       3       4
5       6       7       8
10      -       14      12
9       11      13      15


Berpindah :  Down
1       2       3       4
5       6       7       8
10      11      14      12
9       -       13      15


Berpindah :  Right
1       2       3       4
5       6       7       8
10      11      14      12
9       13      -       15


Berpindah :  Up
1       2       3       4
5       6       7       8
10      11      -       12
9       13      14      15


Berpindah :  Left
1       2       3       4
5       6       7       8
10      -       11      12
9       13      14      15


Berpindah :  Left
1       2       3       4
5       6       7       8
-       10      11      12
9       13      14      15
```

```
Berpindah :  Down
1       2       3       4
5       6       7       8
9       10      11      12
-       13      14      15


Berpindah :  Right
1       2       3       4
5       6       7       8
9       10      11      12
13      -       14      15


Berpindah :  Right
1       2       3       4
5       6       7       8
9       10      11      12
13      14      -       15


Berpindah :  Right
1       2       3       4
5       6       7       8
9       10      11      12
13      14      15      -


State anak yang dihasilkan =  1724
Waktu yang dibutuhkan =  0.1120140552520752 detik
---------------------------------------
```

5. Input tc5.txt

```
PS D:\USER\IF Semester 4\Stima\tucil3\15puzzlesolver\src> py main.py
------ Selamat datang ke Permaiann 15-Puzzle ------

Masukkan nama fle puzzle yang akan dijalankan: (contoh: puzzle.txt)
Nama File : tc5.txt

Puzzle :

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



Nilai KURANG(i) dalam setiap sel yang tidak kosong:

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

Nilai X =  1

Jumlah(KURANG(i)) + X =  24
Puzzle ini dapat diselesaikan :)

Langkah-langkah untuk menyelesaikan puzzle:

State awal :
1       -       2       4
5       6       3       7
9       10      11      8
13      14      15      12
```

```
Berpindah :  Right
1       2       -       4
5       6       3       7
9       10      11      8
13      14      15      12


Berpindah :  Down
1       2       3       4
5       6       -       7
9       10      11      8
13      14      15      12


Berpindah :  Right
1       2       3       4
5       6       7       -
9       10      11      8
13      14      15      12


Berpindah :  Down
1       2       3       4
5       6       7       8
9       10      11      -
13      14      15      12


Berpindah :  Down
1       2       3       4
5       6       7       8
9       10      11      12
13      14      15      -


State anak yang dihasilkan =  12
Waktu yang dibutuhkan =  0.015052318572998047 detik
---------------------------------------
```

BAB III
SOURCE CODE PROGRAM

1. Main.py

15puzzleSolver > src > main.py > ...

```python
from puzzleSolver import *
import time

print("------ Selamat datang ke Permaiann 15-Puzzle ------")

print("\nMasukkan nama fle puzzle yang akan dijalankan: (contoh: puzzle.txt)")

fileName = input("Nama File : ")
path = "../test/" + fileName
puzzle = getPuzzle(path)
print("\nPuzzle : \n")
printPuzzle(puzzle)
print("\nNilai KURANG(i) dalam setiap sel yang tidak kosong:\n")
solvable = isCrackable(puzzle)
if solvable:
    print("\nLangkah-langkah untuk menyelesaikan puzzle: ")
    print("\nState awal : ")
    printPuzzle(puzzle)
    start = time.time()
    solvePuzzle(puzzle)
    stop = time.time()
    print("Waktu yang dibutuhkan = ", stop - start, "detik")
    print("---------------------------------------")
else:
    print("---------------------------------------")
```

2. puzzleSolver.py

15puzzleSolver > src > puzzleSolver.py > getPuzzle

```python
import numpy as np
from heapq import heappush, heappop

# --- INISIALISASI ---
goalState = [[1, 2, 3, 4],
             [5, 6, 7, 8],
             [9, 10, 11, 12],
             [13, 14, 15, 16]]
# ^Tujuan/susunan akhir matriks

producedState_count = 0
# ^Jumlah state/susunan anak yang dihasilkan state parentnnya

# --- FUNGSI FUNGSI ---
def searchElement(puzzle, val):
    for i in range(len(puzzle)):
        for j in range(len(puzzle[i])):
            if puzzle[i][j] == val:
                return i, j
# ^Fungsi untuk mencari indeks elemen pada suatu matriks

def getKurangI(puzzle):
    sumKurangI = 0
    for tiles in range(1,17):
        currRow, currCol = searchElement(puzzle, tiles)
        currKurangI = 0
        for inspectedRow in range(len(puzzle)):
            for inspectedCol in range(len(puzzle[inspectedRow])):
                if puzzle[inspectedRow][inspectedCol] < tiles:
                    if currRow == inspectedRow:
                        if inspectedCol > currCol:
                            currKurangI += 1
                    elif currRow < inspectedRow:
                        currKurangI += 1
        sumKurangI += currKurangI
        print("Kurang(",tiles,") = ", currKurangI)
    return sumKurangI
# ^Fungsi unutuk mendapatkan nilai KURANG(i) dan jumlah totalnya

def getX(puzzle):
    n = 16
    i_blank, j_blank = searchElement(puzzle, n)
    temp = i_blank + j_blank
    if temp % 2 == 0:
        return 0
    else:
        return 1
# ^Fungsi untuk mendapatkan nilai X dari matriks
```

15puzzleSolver > src > puzzleSolver.py > getPuzzle

```python
def getPuzzle(textFile):
    with open(textFile, 'r') as test:
        testLine = test.readlines()
    testLine = [row.strip() for row in testLine]
    testLine = [row.split(' ') for row in testLine]
    testLine = [[int(x) for x in row] for row in testLine]
    return np.array(testLine)
# ^Fungsi untuk mendapatkan matriks dari text file

def stateCost(puzzle):
    currCost = 0
    for inspectedRow in range(len(puzzle)):
        for inspectedCol in range(len(puzzle[inspectedRow])):
            if puzzle[inspectedRow][inspectedCol] != 16:
                if puzzle[inspectedRow][inspectedCol] != goalState[inspectedRow][inspectedCol]:
                    currCost += 1
    return currCost
# ^Fungsi untuk menentukan cost dari setiap kemungkinan pergerakan blank cell

def isCrackable(puzzle):
    sumKurangI = getKurangI(puzzle)
    xValue = getX(puzzle)
    final = sumKurangI + xValue
    print("\nNilai X = ", xValue)
    print("\nJumlah(KURANG(i)) + X = ", final)
    if (final) % 2 == 0: # kalau genap
        print("Puzzle ini dapat diselesaikan :)")
        return True
    else: # kalau ganjil
        print("Maaf :( Puzzle ini tidak dapat diselesaikan")
        return False
# ^Fungsi untuk mencari tahu apakah puzzle dapat mencapai tujuan akhir

class priorityQueue:
    def __init__(own):
        own.heap = []

    def push(own, value):
        heappush(own.heap, value)

    def pop(own):
        return heappop(own.heap)

    def isEmpty(own):
        if not own.heap:
            return True
        else:
            return False
# ^Pembuatan kelas priority queue untuk menyimpan state/susunan dan costnya
```

```python
class State:
    def __init__(own, root, puzzle, blankRow, blankCol, stateCost, depthLevel, prevMove):
        own.root = root
        own.puzzle = puzzle
        own.stateCost = stateCost
        own.depthLevel = depthLevel
        own.blankRow = blankRow
        own.blankCol = blankCol
        own.prevMove = prevMove

    def __lt__(own, other):
        return own.stateCost < other.stateCost
# ^Pembuatan kelas node

def getValidMove(blankRow, blankCol, prevMove):
    validMove = ["Up", "Down", "Left", "Right"]
    if blankRow == 3 or prevMove == "Up":
        validMove.remove("Down")
    if blankCol == 0 or prevMove == "Right":
        validMove.remove("Left")
    if blankRow == 0 or prevMove == "Down":
        validMove.remove("Up")
    if blankCol == 3 or prevMove == "Left":
        validMove.remove("Right")
    return validMove
# ^Fungsi untuk mendapatkan daftar valid move dari suatu blank cell

def createChildState(state):
    childState =[]
    validMove = []
    validMove = getValidMove(state.blankRow,state.blankCol ,state.prevMove)
    for move in validMove:
        if move == "Right":
            newRow = state.blankRow
            newCol = state.blankCol + 1
        elif move == "Down":
            newRow = state.blankRow + 1
            newCol = state.blankCol
        elif move == "Left":
            newRow = state.blankRow
            newCol = state.blankCol - 1
        elif  move == "Up":
            newRow = state.blankRow - 1
            newCol = state.blankCol
```

```
144
145         global producedState_count
146         puzzle = state.puzzle
147         blankRow = state.blankRow
148         blankCol = state.blankCol
149         depthLevel = state.depthLevel
150
151         newPuzzle = np.copy(puzzle)
152         save = newPuzzle[blankRow][blankCol]
153         newPuzzle[blankRow][blankCol] = newPuzzle[newRow][newCol]
154         newPuzzle[newRow][newCol] = save
155
156         childState.append(State(state, newPuzzle, newRow, newCol, stateCost(newPuzzle) + depthLevel + 1, depthLevel+1, move))
157         producedState_count += 1
158     return childState
159 # ^Fungsi untuk membuat node/state child dari suatu node/state parent
160
161 def printStates(state):
162     if state.root == None:
163         print("")
164     else:
165         printStates(state.root)
166         print("Berpindah : ", state.prevMove)
167         printPuzzle(state.puzzle)
168 # ^Fungsi untuk mencetak langkah-langkah yang dilakukan untuk mencapai tujuan akhir
169
170 def printPuzzle(puzzle):
171     for inspectedRow in range(len(puzzle)):
172         for inspectedCol in range(len(puzzle[inspectedRow])):
173             if puzzle[inspectedRow][inspectedCol] == 16:
174                 print("-", end="\t")
175             else:
176                 print(puzzle[inspectedRow][inspectedCol], end="\t")
177         print("")
178     print("\n")
179 # ^Fungsi untuk mencetak puzzle
180
```

```
180
181 def solvePuzzle(puzzle):
182     prioQ = priorityQueue()
183     prioQ.push(State(None, puzzle, searchElement(puzzle, 16)[0], searchElement(puzzle,16)[1], stateCost(puzzle), 0, "None"))
184     while not prioQ.isEmpty():
185         node = prioQ.pop()
186         if node.stateCost == 0 or node.stateCost == node.depthLevel:
187
188             printStates(node)
189             print("State anak yang dihasilkan = ", producedState_count)
190             return
191         else:
192             childState = createChildState(node)
193             for child in childState:
194                 prioQ.push(child)
195 # ^Fungsi untuk mencari solusi dari puzzle
```

Point	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		**V**

LINK

https://github.com/danielsalim/Tucil3_13520008

DAFTAR PUSTAKA

Munir, Rinaldi. (2021). Algoritma Branch and Bound
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Branch-and-Bound-2021-Bagian1.pdf (diakses tanggal 2 April 2022).
(Materi tentang Algoritma Branch and Bound)

Munir, Rinaldi. (2022). Spesifikasi Tugas Besar.
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Kecil-3-(2022).pdf
(diakses tanggal 2 April 2022).
(Spesifikasi Tugas Kecil 3)

(15 November 2021). Algoritma Branch and Bound
https://www.geeksforgeeks.org/branch-and-bound-algorithm/ (diakses tanggal 2 April 2022).
(Materi tentang Algoritma Branch and Bound)