

## **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**

<b>BAB I</b>	<b>3</b>
<b>DASAR TEORI</b>	<b>3</b>
1.1 ALGORITMA BRANCH AND BOUND	3
1.2 CARA PENYELESAIAN	3
<b>BAB II</b>	<b>6</b>
INPUT-OUTPUT PROGRAM	6
<b>BAB III</b>	<b>12</b>
SOURCE CODE PROGRAM	12
<b>LINK</b>	<b>18</b>
<b>DAFTAR PUSTAKA</b>	<b>19</b>

# 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

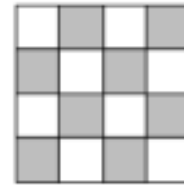
susunan awal jika  $\sum_{i=1}^{15} KURANG(i) + X$  bernilai genap.  $X = 1$  jika sel yang kosong pada state awal berada pada sel yang diarsir.

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

State awal

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

State akhir



Sumber : [rinaldimunir](#)

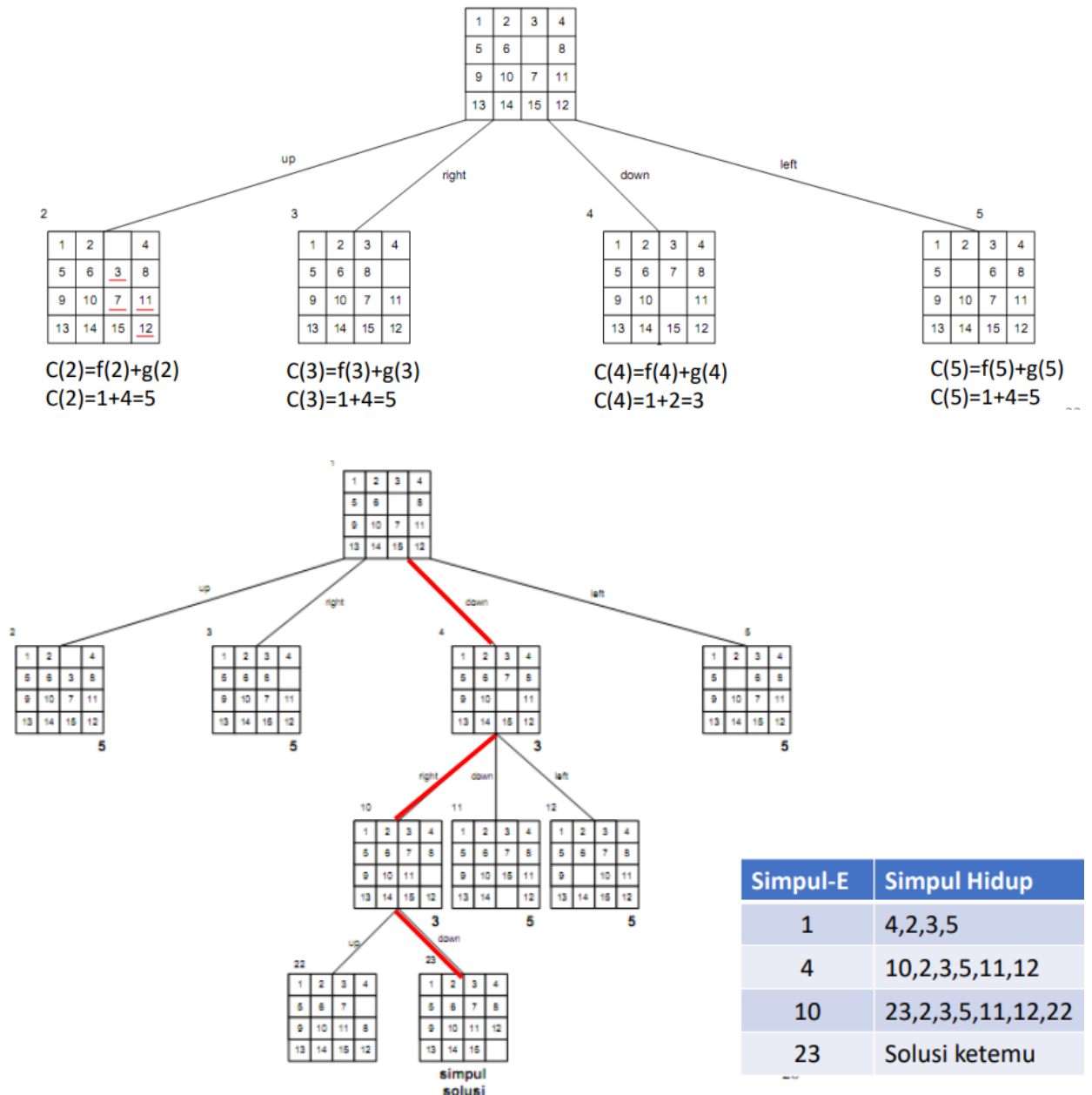
KURANG(i) merupakan banyaknya ubin bernomor  $j$  sedemikian sehingga  $j < i$  dan  $POSISI(j) > POSISI(i)$ .  $POSISI(i)$  = posisi ubin bernomor  $i$  pada susunan yang diperiksa. Contohnya sebagai berikut, KURANG(4) = 1 karena hanya ada 1 sel yang letaknya setelah posisi sel-4 yang nilainya kurang 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.



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 file 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 file 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 file 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 file 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 file 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 > ...
1  from puzzleSolver import *
2  import time
3
4  print("----- Selamat datang ke Permaiann 15-Puzzle -----")
5
6  print("\nMasukkan nama fle puzzle yang akan dijalankan: (contoh: puzzle.txt)")
7
8  fileName = input("Nama File : ")
9  path = "../test/" + fileName
10 puzzle = getPuzzle(path)
11 print("\nPuzzle : \n")
12 printPuzzle(puzzle)
13 print("\nNilai KURANG(i) dalam setiap sel yang tidak kosong:\n")
14 solvable = isCrackable(puzzle)
15 if solvable:
16     print("\nLangkah-langkah untuk menyelesaikan puzzle: ")
17     print("\nState awal : ")
18     printPuzzle(puzzle)
19     start = time.time()
20     solvePuzzle(puzzle)
21     stop = time.time()
22     print("Waktu yang dibutuhkan = ", stop - start, "detik")
23     print("-----")
24 else:
25     print("-----")
```

## 2. puzzleSolver.py

```
15puzzleSolver > src > puzzleSolver.py > getPuzzle
1  import numpy as np
2  from heapq import heappush, heappop
3
4  # --- INISIALISASI ---
5  goalState = [[1, 2, 3, 4],
6               [5, 6, 7, 8],
7               [9, 10, 11, 12],
8               [13, 14, 15, 16]]
9  # ^Tujuan/susunan akhir matriks
10
11  producedState_count = 0
12  # ^Jumlah state/susunan anak yang dihasilkan state parentnya
13
14  # --- FUNGSI FUNGSI ---
15  def searchElement(puzzle, val):
16      for i in range(len(puzzle)):
17          for j in range(len(puzzle[i])):
18              if puzzle[i][j] == val:
19                  return i, j
20  # ^Fungsi untuk mencari indeks elemen pada suatu matriks
21
22  def getKurangI(puzzle):
23      sumKurangI = 0
24      for tiles in range(1,17):
25          currRow, currCol = searchElement(puzzle, tiles)
26          currKurangI = 0
27          for inspectedRow in range(len(puzzle)):
28              for inspectedCol in range(len(puzzle[inspectedRow])):
29                  if puzzle[inspectedRow][inspectedCol] < tiles:
30                      if currRow == inspectedRow:
31                          if inspectedCol > currCol:
32                              currKurangI += 1
33                      elif currRow < inspectedRow:
34                          currKurangI += 1
35          sumKurangI += currKurangI
36          print("Kurang(",tiles,") = ", currKurangI)
37      return sumKurangI
38  # ^Fungsi untuk mendapatkan nilai KURANG(i) dan jumlah totalnya
39
40  def getX(puzzle):
41      n = 16
42      i_blank, j_blank = searchElement(puzzle, n)
43      temp = i_blank + j_blank
44      if temp % 2 == 0:
45          return 0
46      else:
47          return 1
48  # ^Fungsi untuk mendapatkan nilai X dari matriks
```

```

15puzzleSolver > src > puzzleSolver.py > getPuzzle
50 def getPuzzle(textFile):
51     with open(textFile, 'r') as test:
52         testLine = test.readlines()
53         testLine = [row.strip() for row in testLine]
54         testLine = [row.split(' ') for row in testLine]
55         testLine = [[int(x) for x in row] for row in testLine]
56         return np.array(testLine)
57 # ^Fungsi untuk mendapatkan matriks dari text file
58
59 def stateCost(puzzle):
60     currCost = 0
61     for inspectedRow in range(len(puzzle)):
62         for inspectedCol in range(len(puzzle[inspectedRow])):
63             if puzzle[inspectedRow][inspectedCol] != 16:
64                 if puzzle[inspectedRow][inspectedCol] != goalState[inspectedRow][inspectedCol]:
65                     currCost += 1
66     return currCost
67 # ^Fungsi untuk menentukan cost dari setiap kemungkinan pergerakan blank cell
68
69 def isCrackable(puzzle):
70     sumKurangi = getKurangi(puzzle)
71     xValue = getX(puzzle)
72     final = sumKurangi + xValue
73     print("\nNilai X = ", xValue)
74     print("\nJumlah(KURANG(i)) + X = ", final)
75     if (final) % 2 == 0: # kalau genap
76         print("Puzzle ini dapat diselesaikan :)")
77         return True
78     else: # kalau ganjil
79         print("Maaf :( Puzzle ini tidak dapat diselesaikan")
80         return False
81 # ^Fungsi untuk mencari tahu apakah puzzle dapat mencapai tujuan akhir
82
83 class priorityQueue:
84     def __init__(own):
85         own.heap = []
86
87     def push(own, value):
88         heappush(own.heap, value)
89
90     def pop(own):
91         return heappop(own.heap)
92
93     def isEmpty(own):
94         if not own.heap:
95             return True
96         else:
97             return False
98 # ^Pembuatan kelas priority queue untuk menyimpan state/susunan dan costnya

```

```

100 class State:
101     def __init__(own, root, puzzle, blankRow, blankCol, stateCost, depthLevel, prevMove):
102         own.root = root
103         own.puzzle = puzzle
104         own.stateCost = stateCost
105         own.depthLevel = depthLevel
106         own.blankRow = blankRow
107         own.blankCol = blankCol
108         own.prevMove = prevMove
109
110     def __lt__(own, other):
111         return own.stateCost < other.stateCost
112 # ^Pembuatan kelas node
113
114 def getValidMove(blankRow, blankCol, prevMove):
115     validMove = ["Up", "Down", "Left", "Right"]
116     if blankRow == 3 or prevMove == "Up":
117         validMove.remove("Down")
118     if blankCol == 0 or prevMove == "Right":
119         validMove.remove("Left")
120     if blankRow == 0 or prevMove == "Down":
121         validMove.remove("Up")
122     if blankCol == 3 or prevMove == "Left":
123         validMove.remove("Right")
124     return validMove
125 # ^Fungsi untuk mendapatkan daftar valid move dari suatu blank cell
126
127 def createChildState(state):
128     childState = []
129     validMove = []
130     validMove = getValidMove(state.blankRow, state.blankCol, state.prevMove)
131     for move in validMove:
132         if move == "Right":
133             newRow = state.blankRow
134             newCol = state.blankCol + 1
135         elif move == "Down":
136             newRow = state.blankRow + 1
137             newCol = state.blankCol
138         elif move == "Left":
139             newRow = state.blankRow
140             newCol = state.blankCol - 1
141         elif move == "Up":
142             newRow = state.blankRow - 1
143             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/15puzzleSolver>

## **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](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)