

LAPORAN TUGAS KECIL II

Penyelesaian Persoalan 15-Puzzle dengan Algoritma Branch and Bound

Laporan dibuat untuk memenuhi salah satu tugas mata kuliah

IF2211 Strategi Algoritma



Disusun oleh:

Dimas Shidqi Parikesit 13520087

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG**

2022

DAFTAR ISI

DAFTAR ISI	1
Algoritma Branch and Bound	2
Kode Program	4
Screenshot Input - Output Program	13
TC 1	13
TC 2	15
TC 3	21
TC 4	23
TC 5	26
Link Kode Program	28
Checklist	28
Daftar Referensi	28

Algoritma Branch and Bound

Pada kasus ini digunakan algoritma branch and bound yang berbasis BFS dengan struktur data penyimpan berupa priority queue. Priority queue yang dibuat adalah berdasarkan nilai cost function terkecil. Cost function didefinisikan sebagai berikut

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

$\hat{c}(i)$ = ongkos untuk simpul i

$\hat{f}(i)$ = ongkos mencapai simpul i dari akar

$\hat{g}(i)$ = ongkos mencapai simpul tujuan dari simpul i

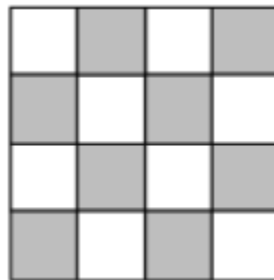
Algoritma ini terdiri dari beberapa bagian

1. Pembuatan Priority Queue

Pada program ini, priority queue diimplementasikan sendiri tanpa menggunakan fitur bawaan python. Implementasi priority queue ini dilakukan dengan memodifikasi struktur data List pada python. Pada program dibuat sebuah class PriorityQueue yang memiliki atribut List queue dan Integer count. Kemudian priority queue diimplementasikan pada member function insert, dimana setelah data dimasukkan ke List queue, List tersebut kemudian di sort sesuai prioritas. Prioritas pada priority queue ini adalah nilai cost terendah diikuti nilai jarak dari akar terendah. Selain itu terdapat pula member function pop, isEmpty, length, dan simpulCount.

2. Menghitung jumlah hasil fungsi Kurang(i) tiap ubin+X

Nilai jumlah Kurang(i) tiap ubin+X ini dapat digunakan untuk mengecek apakah puzzle mungkin untuk diselesaikan atau tidak tanpa mencari kemungkinan yang ada. Kurang(i) didefinisikan sebagai banyaknya ubin bernomor j sedemikian sehingga $j < i$ dan Posisi(j) $>$ Posisi(i), dimana Posisi(i) = posisi ubin bernomor i pada susunan yang diperiksa. Sementara itu X bernilai satu apabila sel kosong terletak pada ubin yang diarsir pada gambar berikut.



Apabila sel kosong tidak pada ubin yang diarsir, maka X bernilai nol.

Setelah didapatkan jumlah $Kurang(i)+X$, maka dapat diambil kesimpulan puzzle dapat diselesaikan apabila nilai nya genap. Apabila ganjil maka ubin tidak dapat diselesaikan.

3. Pembangkitan simpul

Dari state saat ini, simpul dapat dibangkitkan dengan membuat kemungkinan state apabila puzzle digerakkan ke atas, bawah, kanan, dan kiri ditambah aturan simpul yang baru dibangkitkan ini belum pernah dicek atau dibangkitkan sebelumnya. Dengan tambahan aturan ini dapat dipastikan bahwa puzzle tidak akan kembali ke state yang pernah dialami sebelumnya.

4. Menghitung cost simpul

Tiap simpul yang akan dibangkitkan ini harus dicari nilai cost function nya. Cost function dapat dihitung dengan menjumlahkan jarak simpul dari akar dengan jumlah ubin yang saat ini belum berada pada posisi akhir. Nilai cost function ini dimasukkan pada sebuah tuple yang sama dengan simpul untuk kemudian dimasukkan ke dalam priority queue.

5. Menjalankan algoritma untuk simpul berikutnya

Algoritma ini akan terus berjalan sampai priority kosong atau kondisi akhir telah terpenuhi. Penentuan simpul berikutnya yang akan dicek adalah dengan berdasarkan urutan pada priority queue, dimana simpul yang memiliki cost function terkecil akan diprioritaskan.

Kode Program

```
main.py 2 prioqueue.py X test.txt bnb.py
prioqueue.py > PriorityQueue > insert
1  def takeCost(elem):
2      return elem[2],elem[1]
3
4
5  class PriorityQueue(object):
6      def __init__(self):
7          self.queue = []
8          self.count = 0
9
10     def __str__(self):
11         return ' '.join([str(i) for i in self.queue])
12
13     # for checking if the queue is empty
14     def isEmpty(self):
15         return len(self.queue) == 0
16
17     def length(self):
18         return len(self.queue)
19
20     def simpulCount(self):
21         return self.count
22
23     # for inserting an element in the queue
24     def insert(self, data):
25         self.queue.append(data)
26         self.queue.sort(key=takeCost)
27         self.count += 1
28
29
30     # for popping an element based on Priority
31     def pop(self):
32         item = self.queue[0]
33         del self.queue[0]
34         return item
```

```
main.py 2  prioqueue.py  test.txt  bnb.py X
bnb.py > solve
1  import prioqueue
2  import copy
3  import time
4
5  # state adalah tuple dengan urutan (array 15 puzzle, int distFromRoot, int cost, int idx parent di array done)
6
7  def kurang(num, state):
8      idx = -1
9      for i in range(16):
10         if state[0][i] == num:
11             idx = i
12
13         count = 0
14         for i in range(idx, 16):
15             if state[0][i] != -1 and state[0][i] < num:
16                 count += 1
17
18         return count
19
20
21 def sumKurang(state):
22     count = 0
23     for i in range(1, 17):
24         count += kurang(i, state)
25     return count
26
27
28 def findX(state):
29     shadowed = [1, 3, 4, 6, 9, 11, 12, 14]
30     for i in range(1, 16):
31         if state[0][i] == 16 and i in shadowed:
32             return 1
33     return 0
34
35
36 def kurangPlusX(state):
37     return sumKurang(state) + findX(state)
```

```
main.py 2 prioqueue.py test.txt bnb.py X
bnb.py > solve
37     return kurangAng(state) + fIdx(state)
38
39
40 def solvable(state):
41     return kurangPlusX(state) % 2 == 0
42
43
44 def costFuncG(matrix):
45     count = 0
46     for i in range(16):
47         if matrix[i] != i + 1 and matrix[i] != 16:
48             count += 1
49     return count
50
51
52 def move(state, direction):
53     newMatrix = copy.deepcopy(state[0])
54
55     idx = -1
56     for i in range(16):
57         if newMatrix[i] == 16:
58             idx = i
59
60     idxTukar = -1
61     if direction == "top":
62         if idx < 4:
63             return newMatrix
64         idxTukar = idx - 4
65     elif direction == "bottom":
66         if idx > 11:
67             return newMatrix
68         idxTukar = idx + 4
69     elif direction == "left":
70         if idx % 4 == 0:
71             return newMatrix
72         idxTukar = idx - 1
73     elif direction == "right":
74         if (idx + 1) % 4 == 0:
```

```
main.py 2 prioqueue.py test.txt bnb.py X
bnb.py > solve
74         if (idx + 1) % 4 == 0:
75             return newMatrix
76         idxTukar = idx + 1
77
78     newMatrix[idx] = newMatrix[idxTukar]
79     newMatrix[idxTukar] = 16
80
81     return newMatrix
82
83 def solve(initState, path, simpulTime):
84     start = time.perf_counter()
85
86     # Nilai fungsi Kurang(i) tiap ubin awal
87     for i in range(1, 17):
88         print("Kurang",i, kurang(i, initState))
89
90     # Nilai sumKurang(i)+X
91     print()
92     print("sumKurang(i)+X =", kurangPlusX(initState))
93     print()
94
95     prioQueue = prioqueue.PriorityQueue()
96
97     if solvable(initState) == False:
98         print("Jumlah simpul dibangkitkan =", prioQueue.simpulCount())
99         return False
100
101     prioQueue.insert(initState)
102     allPath = []
103     done = []
104     inQueue = []
105
106     while prioQueue.isEmpty() == False:
107         currState = prioQueue.pop()
108         done.append(currState[0])
109         allPath.append(currState)
110
```



```
main.py 2 prioqueue.py test.txt bnb.py X
bnb.py > solve
110
111     if solvable(currState):
112         if (currState[1]==currState[2]):
113             path.insert(0, currState)
114             while path[0][3]!=-1:
115                 path.insert(0, allPath[path[0][3]])
116
117             print("Here")
118             for i in range(len(path)):
119                 print(path[i][0])
120
121             print()
122             print("Jumlah simpul dibangkitkan =", prioQueue.simpulCount())
123             print("Waktu =", "{:.5f}".format(time.perf_counter()-start), "detik")
124
125             simpulTime.append(prioQueue.simpulCount())
126             simpulTime.append(time.perf_counter()-start)
127             return True
128
129         print(currState)
130         print(prioQueue.length())
131         newTop = move(currState, "top")
132         newBot = move(currState, "bottom")
133         newLeft = move(currState, "left")
134         newRight = move(currState, "right")
135
136         f = currState[1] + 1
137         gTop = costFuncG(newTop) if (newTop not in done) and (newTop not in inQueue) else -1
138         gBot = costFuncG(newBot) if (newBot not in done) and (newBot not in inQueue) else -1
139         gLeft = costFuncG(newLeft) if (newLeft not in done) and (newLeft not in inQueue) else -1
140         gRight = costFuncG(newRight) if (newRight not in done) and (newRight not in inQueue) else -1
141
142         if gTop != -1:
143             stateTop = (newTop, f, f + gTop, len(done)-1)
144             prioQueue.insert(stateTop)
145             inQueue.append(newTop)
146         if gBot != -1:
147             stateBot = (newBot, f, f + gBot, len(done)-1)
```

```
main.py 2 prioqueue.py test.txt bnb.py X
bnb.py > solve
147         stateBot = (newBot, f, f + gBot, len(done)-1)
148         prioQueue.insert(stateBot)
149         inQueue.append(newBot)
150         if gLeft != -1:
151             stateLeft = (newLeft, f, f + gLeft, len(done)-1)
152             prioQueue.insert(stateLeft)
153             inQueue.append(newLeft)
154         if gRight != -1:
155             stateRight = (newRight, f, f + gRight, len(done)-1)
156             prioQueue.insert(stateRight)
157             inQueue.append(newRight)
158     print("Unknown Error")
```

```
main.py 2 x  prioqueue.py  test.txt  bnb.py
main.py > MyWidget > getTable
1  from bnb import *
2  import sys
3  import os
4  import random
5  from PySide6 import (QtCore, QtWidgets, QtGui)
6
7  class MyWidget(QtWidgets.QWidget):
8      def __init__(self):
9          super().__init__()
10
11          self.path = []
12          self.isSolvable = False
13          self.arrSimpulTime = []
14          self.initState = ()
15
16          self.button = QtWidgets.QPushButton("Start!")
17          self.text = QtWidgets.QLabel("Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar.
18          Apabila solvable, pencet button start lagi untuk menunjukkan step")
19          self.table = QtWidgets.QTableWidget()
20          self.status = QtWidgets.QLabel("Status: Not Started")
21          self.result = QtWidgets.QTextEdit("")
22          self.rand = QtWidgets.QPushButton("Randomize!")
23          self.file = QtWidgets.QFileDialog(filter="Text files (*.txt)")
24
25          self.table.setRowCount(4)
26          self.table.setColumnCount(4)
27          self.result.setReadOnly(True)
28
29          layout = QtWidgets.QVBoxLayout(self)
30          layout.addWidget(self.text)
31          layout.addWidget(self.button)
32          layout.addWidget(self.status)
33          layout.addWidget(self.result)
34          layout.addWidget(self.table)
35          layout.addWidget(self.rand)
36          layout.addWidget(self.file)
```

```
main.py x prioqueue.py test.txt bnb.py
main.py > MyWidget
37 self.button.clicked.connect(self.magic) # type: ignore
38 self.rand.clicked.connect(self.randomizer) # type: ignore
39
40 def getTable(self):
41     matrix = []
42     matrix.append(int(self.table.item(0,0).text()))
43     matrix.append(int(self.table.item(0,1).text()))
44     matrix.append(int(self.table.item(0,2).text()))
45     matrix.append(int(self.table.item(0,3).text()))
46     matrix.append(int(self.table.item(1,0).text()))
47     matrix.append(int(self.table.item(1,1).text()))
48     matrix.append(int(self.table.item(1,2).text()))
49     matrix.append(int(self.table.item(1,3).text()))
50     matrix.append(int(self.table.item(2,0).text()))
51     matrix.append(int(self.table.item(2,1).text()))
52     matrix.append(int(self.table.item(2,2).text()))
53     matrix.append(int(self.table.item(2,3).text()))
54     matrix.append(int(self.table.item(3,0).text()))
55     matrix.append(int(self.table.item(3,1).text()))
56     matrix.append(int(self.table.item(3,2).text()))
57     matrix.append(int(self.table.item(3,3).text()))
58     return matrix
59
60 def setTable(self, matrix):
61     print(matrix)
62     self.table.setItem(0,0,QtWidgets.QTableWidgetItem(str(matrix[0])))
63     self.table.setItem(0,1,QtWidgets.QTableWidgetItem(str(matrix[1])))
64     self.table.setItem(0,2,QtWidgets.QTableWidgetItem(str(matrix[2])))
65     self.table.setItem(0,3,QtWidgets.QTableWidgetItem(str(matrix[3])))
66     self.table.setItem(1,0,QtWidgets.QTableWidgetItem(str(matrix[4])))
67     self.table.setItem(1,1,QtWidgets.QTableWidgetItem(str(matrix[5])))
68     self.table.setItem(1,2,QtWidgets.QTableWidgetItem(str(matrix[6])))
69     self.table.setItem(1,3,QtWidgets.QTableWidgetItem(str(matrix[7])))
70     self.table.setItem(2,0,QtWidgets.QTableWidgetItem(str(matrix[8])))
71     self.table.setItem(2,1,QtWidgets.QTableWidgetItem(str(matrix[9])))
72     self.table.setItem(2,2,QtWidgets.QTableWidgetItem(str(matrix[10])))
73     self.table.setItem(2,3,QtWidgets.QTableWidgetItem(str(matrix[11])))
```

```
main.py x prioqueue.py test.txt bnb.py
main.py > MyWidget
73 self.table.setItem(2,3,QtWidgets.QTableWidgetItem(str(matrix[11])))
74 self.table.setItem(3,0,QtWidgets.QTableWidgetItem(str(matrix[12])))
75 self.table.setItem(3,1,QtWidgets.QTableWidgetItem(str(matrix[13])))
76 self.table.setItem(3,2,QtWidgets.QTableWidgetItem(str(matrix[14])))
77 self.table.setItem(3,3,QtWidgets.QTableWidgetItem(str(matrix[15])))
78
79 def setTextBox(self):
80     for i in range(1,17):
81         self.result.append("Kurang "+str(i)+ " = " +str(kurang(i, self.initState)))
82
83     self.result.append("sumKurang(i)+X = "+ str(kurangPlusX(self.initState)))
84
85 def openFile(self, fileName):
86     matrix = []
87     f = open(fileName, "r")
88     for line in f:
89         matrix.append(int(line.strip('\n').strip()))
90     f.close()
91     return matrix
92
93 @QtCore.Slot()
94 def randomizer(self):
95     matrix = random.sample(range(1,17), 16)
96     self.setTable(matrix)
97     self.initState = (matrix, 0, 0+costFuncG(matrix), -1)
98
99 @QtCore.Slot()
100 def magic(self):
101     if self.isSolvable==False:
102         if self.initState == ():
103             matrix = []
104             if self.file.selectedFiles()[0]!=os.getcwd().replace('\\', '/'):
105                 matrix = self.openFile(self.file.selectedFiles()[0])
106                 self.setTable(matrix)
107             else:
108                 matrix = self.getTable()
109             self.status.setText("Status: Processing")
```

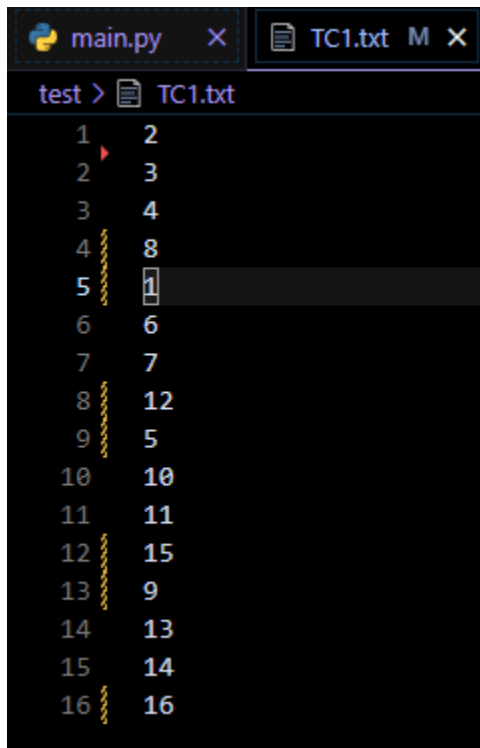
```
main.py x prioqueue.py test.txt bnb.py
main.py > MyWidget
106         self.setTable(matrix)
107     else:
108         matrix = self.getTable()
109         self.status.setText("Status: Processing")
110         self.initState = (matrix, 0, 0+costFuncG(matrix), -1)
111
112     self.setTextBox()
113     self.isSolvable = solve(self.initState, self.path, self.arrSimpulTime)
114     self.status.setText("Status: " + ("Solvable with time "+ "{:.5f}".format(self.arrSimpulTime[1]) + " seconds and simpul count " +
115 str(self.arrSimpulTime[0])) if self.isSolvable==True else "Unsolvable" )
116     if self.isSolvable==True:
117         print("Time =", self.arrSimpulTime[1])
118         print("Simpul count =", self.arrSimpulTime[0])
119         self.result.append("\n Urutan path")
120         for i in range(len(self.path)):
121             self.result.append(str(self.path[i][0]))
122             print(self.path[i][0])
123         del self.path[0]
124     elif len(self.path)>0:
125         self.setTable(self.path[0][0])
126         del self.path[0]
127
128 if __name__ == '__main__':
129     app = QtWidgets.QApplication([])
130
131     widget = MyWidget()
132     widget.resize(800, 600)
133     widget.show()
134
135     sys.exit(app.exec())
```

Screenshot Input - Output Program

Perhatian: Dalam melakukan input, slot kosong direpresentasikan dengan angka 16

1. TC 1

Input



```
test > TC1.txt
1 2
2 3
3 4
4 8
5 1
6 6
7 7
8 12
9 5
10 10
11 11
12 15
13 9
14 13
15 14
16 16
```

Output

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51

Kurang 1 = 0

Kurang 2 = 1

Kurang 3 = 1

Kurang 4 = 1

Kurang 5 = 0

Kurang 6 = 1

Kurang 7 = 1

Kurang 8 = 4

Kurang 9 = 0

Kurang 10 = 1

Kurang 11 = 1

Kurang 12 = 4

Kurang 13 = 0

Kurang 14 = 0

Kurang 15 = 3

Kurang 16 = 0

sumKurang(i)+X = 18

Urutan path

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

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

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

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

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

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

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

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

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

Randomize!

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51

Kurang 6 = 1
Kurang 7 = 1
Kurang 8 = 4
Kurang 9 = 0
Kurang 10 = 1
Kurang 11 = 1
Kurang 12 = 4
Kurang 13 = 0
Kurang 14 = 0
Kurang 15 = 3
Kurang 16 = 0
sumKurang(i)+X = 18

Urutan path
[2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 15, 9, 13, 14, 16]
[2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 16, 9, 13, 14, 15]
[2, 3, 4, 8, 1, 6, 7, 16, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 3, 4, 16, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 3, 16, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 16, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[16, 2, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 16, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 16, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 16, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

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

Randomize!

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path
 [2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 15, 9, 13, 14, 16]
 [2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 16, 9, 13, 14, 15]
 [2, 3, 4, 8, 1, 6, 7, 16, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 3, 4, 16, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 3, 16, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 16, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [16, 2, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 16, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 16, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 16, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

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

Randomize!

python3

— □ ×

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51

```

Kurang 6 = 1
Kurang 7 = 1
Kurang 8 = 4
Kurang 9 = 0
Kurang 10 = 1
Kurang 11 = 1
Kurang 12 = 4
Kurang 13 = 0
Kurang 14 = 0
Kurang 15 = 3
Kurang 16 = 0
sumKurang(i)+X = 18

Urutan path
[2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 15, 9, 13, 14, 16]
[2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 16, 9, 13, 14, 15]
[2, 3, 4, 8, 1, 6, 7, 16, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 3, 4, 16, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 3, 16, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 16, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[16, 2, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 16, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 16, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 16, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

```

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

Randomize!

python3

— □ ×

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path

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

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

Randomize!

python3

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51

Kurang 6 = 1
Kurang 7 = 1
Kurang 8 = 4
Kurang 9 = 0
Kurang 10 = 1
Kurang 11 = 1
Kurang 12 = 4
Kurang 13 = 0
Kurang 14 = 0
Kurang 15 = 3
Kurang 16 = 0
sumKurang(i)+X = 18

Urutan path

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

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

Randomize!

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51
Kurang 6 = 1
Kurang 7 = 1
Kurang 8 = 4
Kurang 9 = 0
Kurang 10 = 1
Kurang 11 = 1
Kurang 12 = 4
Kurang 13 = 0
Kurang 14 = 0
Kurang 15 = 3
Kurang 16 = 0
sumKurang(i)+X = 18

Urutan path
[2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 15, 9, 13, 14, 16]
[2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 16, 9, 13, 14, 15]
[2, 3, 4, 8, 1, 6, 7, 16, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 3, 4, 16, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 3, 16, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[2, 16, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[16, 2, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 16, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 16, 10, 11, 12, 9, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 13, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 16, 14, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 15]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

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

Randomize!

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02532 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path
 [2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 15, 9, 13, 14, 16]
 [2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 16, 9, 13, 14, 15]
 [2, 3, 4, 8, 1, 6, 7, 16, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 3, 4, 16, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 3, 16, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 16, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [16, 2, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 16, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 16, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 16, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

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

Randomize!

python3

— □ ×

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02825 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path

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

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

Randomize!

python3

— □ ×

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02825 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path

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

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

Randomize!

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02825 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path
 [2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 15, 9, 13, 14, 16]
 [2, 3, 4, 8, 1, 6, 7, 12, 5, 10, 11, 16, 9, 13, 14, 15]
 [2, 3, 4, 8, 1, 6, 7, 16, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 3, 4, 16, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 3, 16, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [2, 16, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [16, 2, 3, 4, 1, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 16, 6, 7, 8, 5, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 16, 10, 11, 12, 9, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 13, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 16, 14, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 15]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

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

Randomize!

python3

— □ ×

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02825 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path

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

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

Randomize!

python3

— □ ×

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.02825 seconds and simpul count 51

Kurang 6 = 1
 Kurang 7 = 1
 Kurang 8 = 4
 Kurang 9 = 0
 Kurang 10 = 1
 Kurang 11 = 1
 Kurang 12 = 4
 Kurang 13 = 0
 Kurang 14 = 0
 Kurang 15 = 3
 Kurang 16 = 0
 sumKurang(i)+X = 18

Urutan path

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

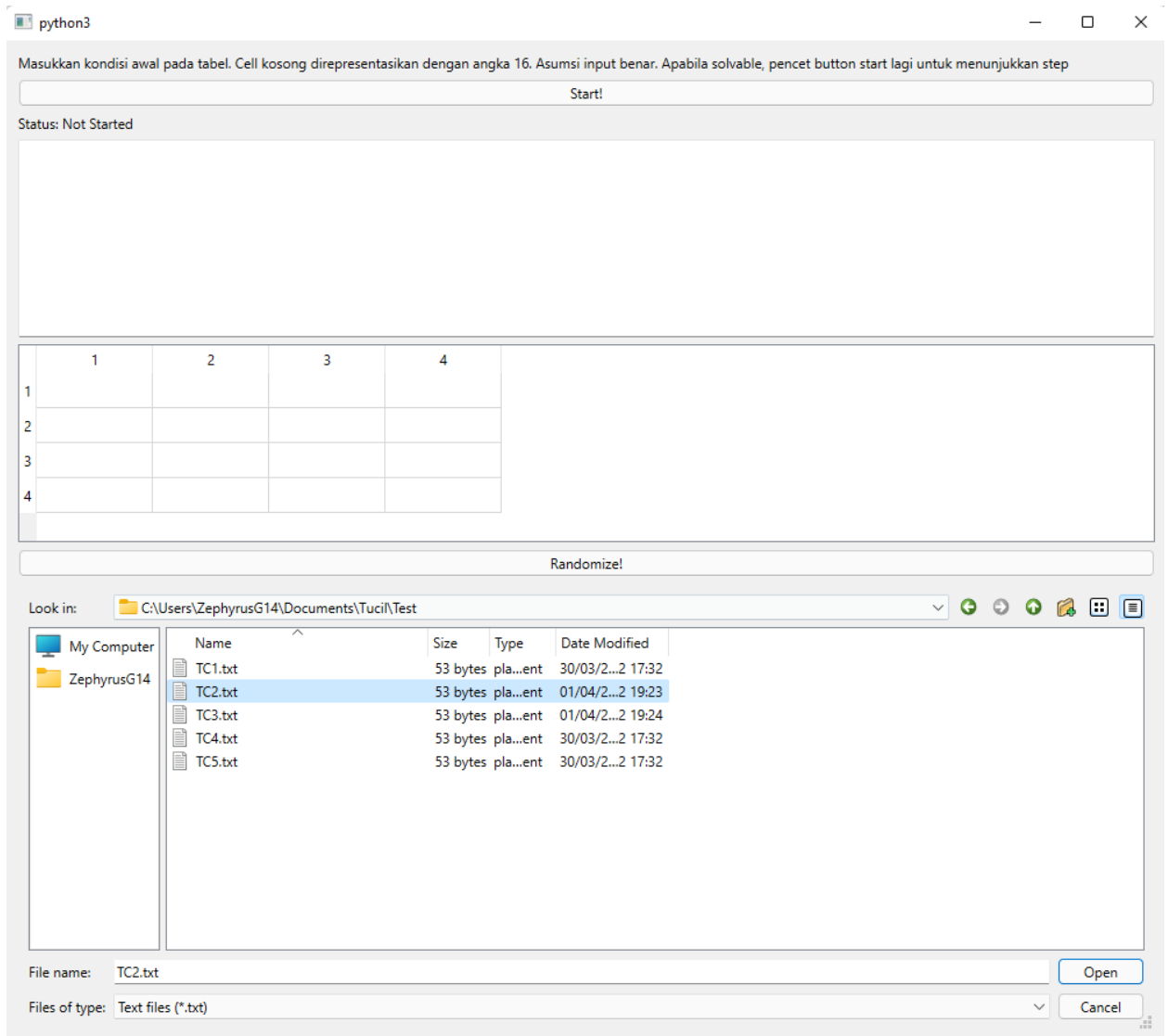
	1	2	3	4
1	1	2	3	4
2	5	6	7	8
3	9	10	11	12
4	13	14	15	16

Randomize!

2. TC 2

Input

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



Output

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.00759 seconds and simpul count 6

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 = 1

Kurang 12 = 0

Kurang 13 = 1

Kurang 14 = 1

Kurang 15 = 1

Kurang 16 = 8

sumKurang(i)+X = 14

Urutan path

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

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

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

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

Randomize!

30 - IF2211 - Strategi Algoritma

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.00759 seconds and simpl count 6

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 = 1

Kurang 12 = 0

Kurang 13 = 1

Kurang 14 = 1

Kurang 15 = 1

Kurang 16 = 8

sumKurang(i)+X = 14

Urutan path

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

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

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

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

Randomize!

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.00759 seconds and simpul count 6

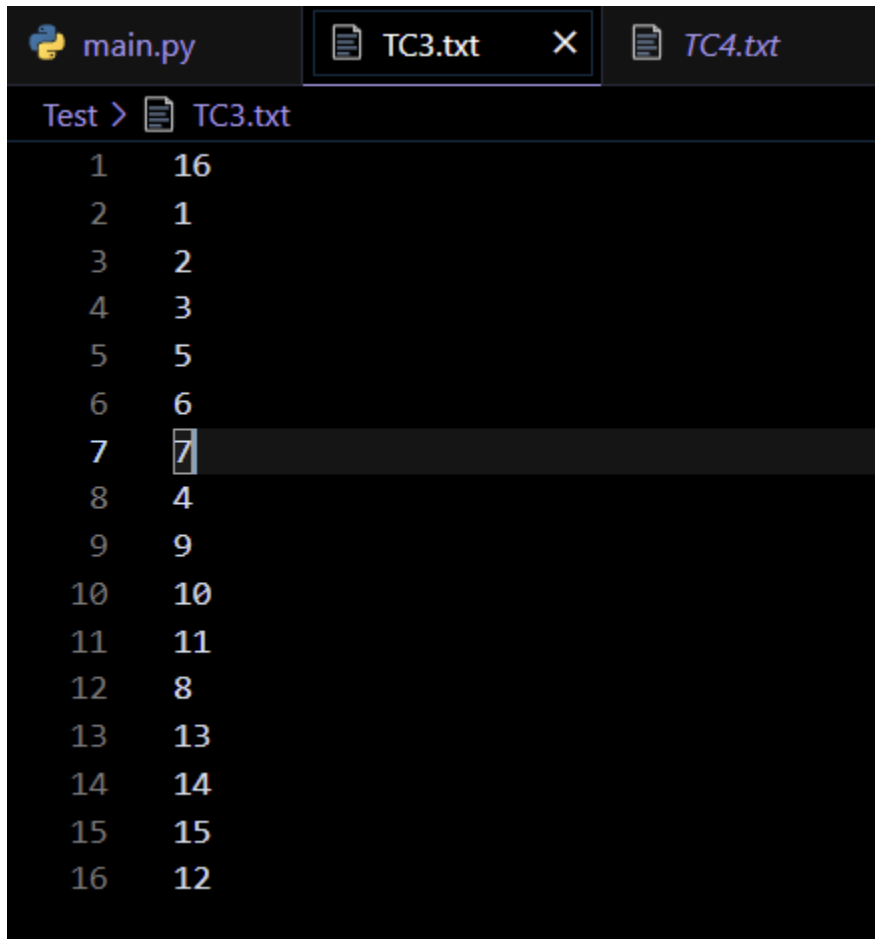
Kurang 1 = 0
Kurang 2 = 0
Kurang 3 = 0
Kurang 4 = 0
Kurang 5 = 0
Kurang 6 = 0
Kurang 7 = 0
Kurang 8 = 0
Kurang 9 = 1
Kurang 10 = 1
Kurang 11 = 1
Kurang 12 = 0
Kurang 13 = 1
Kurang 14 = 1
Kurang 15 = 1
Kurang 16 = 8
sumKurang(i)+X = 14

Urutan path
[1, 2, 3, 4, 5, 6, 7, 16, 9, 10, 11, 8, 13, 14, 15, 12]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 16, 13, 14, 15, 12]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

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

Randomize!

3. TC 3



The image shows a code editor window with three tabs: 'main.py', 'TC3.txt', and 'TC4.txt'. The 'TC3.txt' tab is active, displaying a table with 16 rows. The first column contains numbers from 1 to 16, and the second column contains corresponding values. The row with index 7 and value 7 is highlighted.

Test >	TC3.txt
1	16
2	1
3	2
4	3
5	5
6	6
7	7
8	4
9	9
10	10
11	11
12	8
13	13
14	14
15	15
16	12

Output

State awal

python3

— □ ×

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.01871 seconds and simpul count 34

Kurang 1 = 0

Kurang 2 = 0

Kurang 3 = 0

Kurang 4 = 0

Kurang 5 = 1

Kurang 6 = 1

Kurang 7 = 1

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 = 15

sumKurang(i)+X = 24

Urutan path

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

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

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

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

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

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

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

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

Randomize!

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Status: Solvable with time 0.01871 seconds and simpul count 34

Kurang 1 = 0
 Kurang 2 = 0
 Kurang 3 = 0
 Kurang 4 = 0
 Kurang 5 = 1
 Kurang 6 = 1
 Kurang 7 = 1
 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 = 15
 sumKurang(i)+X = 24

Urutan path
 [16, 1, 2, 3, 5, 6, 7, 4, 9, 10, 11, 8, 13, 14, 15, 12]
 [1, 16, 2, 3, 5, 6, 7, 4, 9, 10, 11, 8, 13, 14, 15, 12]
 [1, 2, 16, 3, 5, 6, 7, 4, 9, 10, 11, 8, 13, 14, 15, 12]
 [1, 2, 3, 16, 5, 6, 7, 4, 9, 10, 11, 8, 13, 14, 15, 12]
 [1, 2, 3, 4, 5, 6, 7, 16, 9, 10, 11, 8, 13, 14, 15, 12]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 16, 13, 14, 15, 12]
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]

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

Randomize!

Mohon maaf tidak di screenshot semua karena cukup banyak

4. TC 4

Input

```
main.py TC4.txt X
```

Test > TC4.txt

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

Output

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Unsolvable

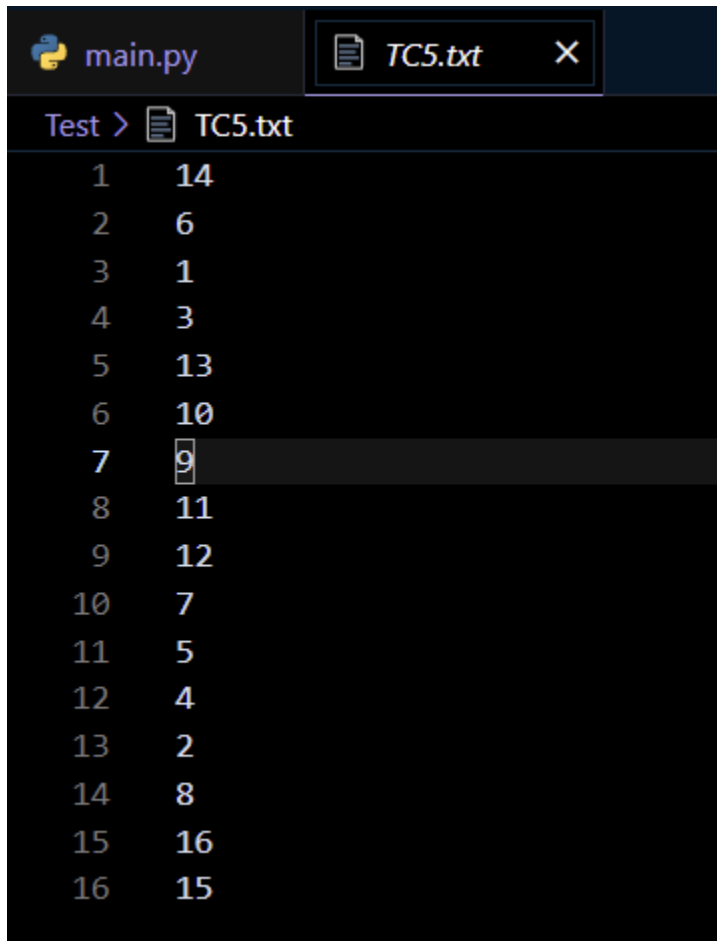
Kurang 1 = 0
Kurang 2 = 1
Kurang 3 = 2
Kurang 4 = 1
Kurang 5 = 0
Kurang 6 = 0
Kurang 7 = 4
Kurang 8 = 6
Kurang 9 = 6
Kurang 10 = 0
Kurang 11 = 10
Kurang 12 = 6
Kurang 13 = 9
Kurang 14 = 0
Kurang 15 = 5
Kurang 16 = 13
sumKurang(i)+X = 63

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

Randomize!

5. TC 5

Input



Test >	TC5.txt
1	14
2	6
3	1
4	3
5	13
6	10
7	9
8	11
9	12
10	7
11	5
12	4
13	2
14	8
15	16
16	15

Output

python3

Masukkan kondisi awal pada tabel. Cell kosong direpresentasikan dengan angka 16. Asumsi input benar. Apabila solvable, pencet button start lagi untuk menunjukkan step

Start!

Unsolvable

Kurang 1 = 0
Kurang 2 = 0
Kurang 3 = 1
Kurang 4 = 1
Kurang 5 = 2
Kurang 6 = 5
Kurang 7 = 3
Kurang 8 = 0
Kurang 9 = 5
Kurang 10 = 6
Kurang 11 = 5
Kurang 12 = 5
Kurang 13 = 9
Kurang 14 = 13
Kurang 15 = 0
Kurang 16 = 1
sumKurang(i)+X = 57

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

Randomize!

Link Kode Program

Repository Github : https://github.com/dParikesit/Tucil3_13520087

Checklist

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	✓	

Daftar Referensi

1. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Branch-and-Bound-2021-Bagian1.pdf>