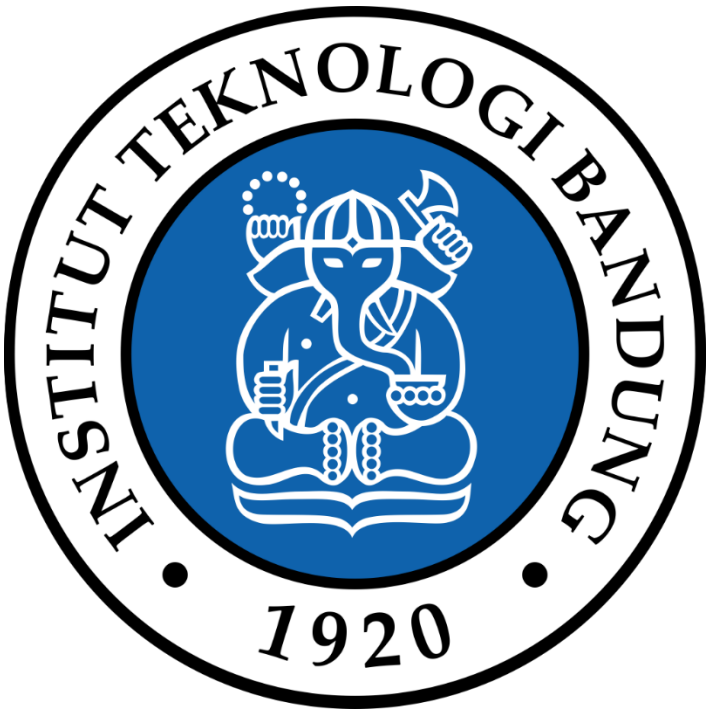
LAPORAN TUGAS KECIL 3
IF2211 – STRATEGI ALGORITMA
SEMESTER II TAHUN 2021/2022



Oleh

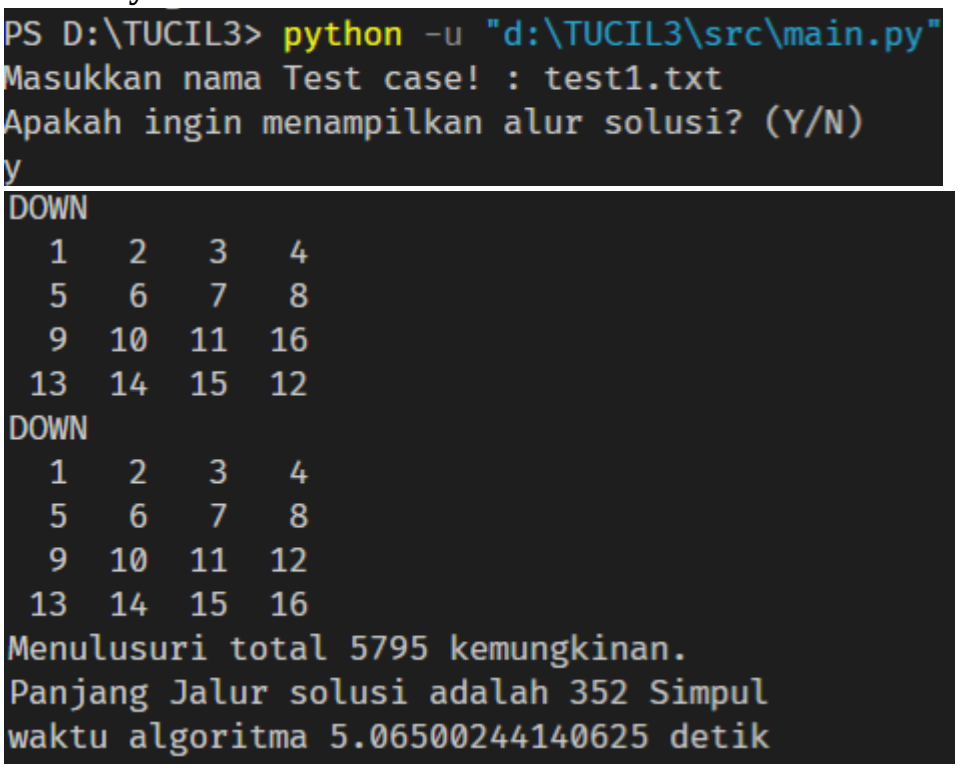
Fadil Fauzani 13520032

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

1. **Algoritma Branch n Bound**
   Algoritma Branch n Bound yang saya gunakan bekerja dengan, membangkitkan simpul-simpul baru yang dibuat berdasarkan pergerakan angka ke-16 dari puzzle dan memasukkannya ke list yang bekerja bagaikan *priority-Queue* yang di urutkan berdasarkan cost dari simpul tersebut, fungsi cost sendiri dihitung dari semua angka kecuali 16 yang tidak berada di tempat seharunsya (*Goal State*). Dalam algoritma ini untuk memangkas pengecekan, saya membuat list untuk kemungkinan yang telah ditelusuri sehingga untuk kemungkinan yang telah ditelusuri tidak ditelusuri 2 kali dan pengecekan menjadi lebih sedikit dilakukan.
2. **Skrinshyut**



Gambar 1. Hasil output Testcase 1

```
PS D:\TUCIL3> python -u "d:\TUCIL3\src\main.py"
Masukkan nama Test case! : test2.txt
Apakah ingin menampilkan alur solusi? (Y/N)
y
  1   2   3   4
  5   6  16   8
  9  10   7  11
 13  14  15  12
  1   2   3   4
  5   6   7   8
  9  10  16  11
 13  14  15  12
RIGHT
  1   2   3   4
  5   6   7   8
  9  10  11  16
 13  14  15  12
DOWN
  1   2   3   4
  5   6   7   8
  9  10  11  12
 13  14  15  16
Menulusuri total 4 kemungkinan.
Panjang Jalur solusi adalah 4 Simpul
waktu algoritma 0.0 detik
```

Gambar 2. Hasil output Testcase 2

```
PS D:\TUCIL3> python -u "d:\TUCIL3\src\main.py"
Masukkan nama Test case! : test3.txt
Apakah ingin menampilkan alur solusi? (Y/N)
n
Menulusuri total 16502 kemungkinan.
Panjang Jalur solusi adalah 493 Simpul
waktu algoritma 80.95309901237488 detik
```

Gambar 3. Hasil output Testcase 3

```
Masukkan nama Test case! : test4.txt
Apakah ingin menampilkan alur solusi? (Y/N)
y
Puzzle tidak dapat di selesaikan karena Jumlah Kurang(i) + X nya ganjil (15).
waktu algoritma 0.0 detik
```

Gambar 4. Hasil output Testcase 4

```
PS D:\TUCIL3> python -u "d:\TUCIL3\src\main.py"
Masukkan nama Test case! : test5.txt
Apakah ingin menampilkan alur solusi? (Y/N)
y
Puzzle tidak dapat di selesaikan karena Jumlah Kurang(i) + X nya ganjil (1).
waktu algoritma 0.0 detik
```

Gambar 5. Hasil output Testcase 5

3. Checklist

Poin	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	V	
5. Bonus dibuat		V

4. Source code

boardUtils.py:

```python
def cost(list):
    count = 0
    for i in range(15):
        if list[i]!=i+1:
            count+=1
    return count
def getindex(list, x):
    for i in range(len(list)):
        # print(i,x)
        if list[i] == x:
            return i

def kurang(list,i):
    count = 0
    for j in range(getindex(list, i), 16):
        if (list[j] < i):
            count+=1
    return count
def X(list):
    arsir = [1,3, 4,6, 9, 11, 12,14]
    idx = getindex(list, 16)
    if (idx in arsir):
        return 1
    else:
        return 0
def reachable(list):
    sum = 0
```

```
    for i in range(1,17):
        sum += kurang(list, i)
    sum += X(list)
    return sum
def goUp(list):
    idx = getindex(list,16)
    i = idx // 4
    j = idx % 4
    if (i != 0):
        swap(list, (i - 1) * 4 + j, idx)
def goDown(list):
    idx = getindex(list,16)
    i = idx // 4
    j = idx % 4
    if (i != 3):
        swap(list, (i + 1) * 4 + j, idx)
def goLeft(list):
    idx = getindex(list,16)
    i = idx // 4
    j = idx % 4
    if (j != 0):
        swap(list, i * 4 + j-1, idx)
def goRight(list):
    idx = getindex(list, 16)
    i = idx // 4
    j = idx % 4
    if (j != 3):
        swap(list, i * 4 + j+1, idx)
def swap(list, i ,j):
    temp = list[i]
    list[i] = list[j]
    list[j] = temp
def printBoard(list):
    for i in range(4):
        for j in range(4):
            print(str(list[i*4 + j]).rjust(3," "), end=' ')
        print()
def append(q, board, prev, step):
    c = cost(board)
    step.append(prev)
    q.append((c, board, prev, step))
    q.sort()
def isGoal(board):
    for i in range(16):
        if board[i]!=i+1:
            return False
    return True
def printStep(board, step):
```

```
    for i in range(len(step)):
        if (step[i] == 0):
            goUp(board)
            print("UP")
        if (step[i] == 1):
            goDown(board)
            print("DOWN")
        if (step[i] == 2):
            goLeft(board)
            print("LEFT")
        if (step[i] == 3):
            goRight(board)
            print("RIGHT")
        printBoard(board)
```

main.py:

```
import time
from boardUtils import *
print("Masukkan nama Test case! : ", end='')
filestr = input()
print("Apakah ingin menampilkan alur solusi? (Y/N)")
ans = input()
verbose = False
if (ans.lower() == "y"):
    verbose = True
f = open("./test/" + filestr)
inp = list(map(int, f.read().split(" ")))

QueuePenelusuran = []
ditelusuri = []

test = 0
start = time.time()
if (reachable(inp) % 2 == 0):
    found = False
    step = []
    append(QueuePenelusuran, inp.copy(), -1, step.copy())
    while QueuePenelusuran and not found:
        test += 1
        cc, cb, prev, steps = QueuePenelusuran.pop(0)
        if (cb in ditelusuri):
            #kalo udah ditelusuri g dicek lagi
            continue
        ditelusuri.append(cb.copy())

        if cc == 0:
            found = True
            end = time.time()
            break
```

```
        idx = getindex(cb, 16)
        i = idx // 4
        j = idx % 4
        # up = 0, down = 1, left = 2, right = 3
        if (i != 0 and prev != 1): #biar ga balik lagi
            goUp(cb)
            if (cb not in ditelusuri):
                append(QueuePenelusuran, cb.copy(), 0, steps.copy())
            goDown(cb)
        if (i != 3 and prev != 0):
            goDown(cb)
            if (cb not in ditelusuri):
                append(QueuePenelusuran, cb.copy(), 1, steps.copy())
            goUp(cb)
        if (j != 0 and prev != 3):
            goLeft(cb)
            if (cb not in ditelusuri):
                append(QueuePenelusuran, cb.copy(), 2, steps.copy())
            goRight(cb)
        if (j != 3 and prev != 2):
            goRight(cb)
            if (cb not in ditelusuri):
                append(QueuePenelusuran, cb.copy(), 3, steps.copy())
            goLeft(cb)
    if found:
        if verbose:
            printStep(inp, steps)
        print("Menulusuri total " + str(len(ditelusuri)) + "
kemungkinan." )
        print("Panjang Jalur solusi adalah " + str(len(steps)) + "
Simpul")
else:
    end = time.time()
    print("Puzzle tidak dapat di selesaikan karena Jumlah Kurang(i) +
X nya ganjil (" + str(reachable(inp))+").")
print("waktu algoritma ", end='')
print(end-start, end=' detik\n')
```

5. **Berkas testcase**

   test1.txt:

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

   test2.txt:

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

   test3.txt:

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

   test4.txt:

1 2 3 4 5 6 16 8 9 10 7 11 13 12 15 14
test5.txt:
2 1 3 4 5 6 7 8 9 10 11 12 13 14 15 16

6. **Alamat github**
   https://github.com/fadilfauzani/Tucil3_Stima