

**LAPORAN TUGAS KECIL  
IF2211 – STRATEGI ALGORITMA**

**ALGORITMA PENYELESAIAN PERSOALAN 15-PUZZLE DENGAN  
METODE BRANCH AND BOUND**



Dipersiapkan oleh:

Dzaky Fattan Rizqullah

13520003

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
JL. GANESHA 10, BANDUNG 40132**

## Daftar Isi

A.	Algoritma <i>Divide-and-Conquer</i> .....	3
B.	Source Program .....	4
C.	Check List .....	13
D.	Kode Program .....	13
E.	Berkas Persoalan .....	19
F.	Alamat Drive dan Github .....	19

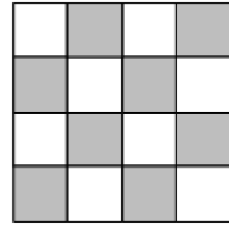
### A. Algoritma Branch and Bound

Program ini dibuat untuk dapat menyelesaikan persoalan 15-*puzzle* yang diberikan melalui file .txt yang diletakkan pada folder test. Algoritma yang digunakan adalah *Branch and Bound*. Secara garis besar, program akan meminta input file .txt yang sudah terletak pada folder test, kemudian akan menampilkan kondisi awal matriks persoalan, lalu menampilkan nilai dari fungsi Kurang(i) yang dapat menentukan apakah persoalan dapat diselesaikan atau tidak. Bila dapat diselesaikan, program akan berusaha menyelesaikan *puzzle*. Bila persoalan telah diselesaikan, program akan menampilkan langkah-langkah penyelesaian *puzzle* dari kondisi awal hingga kondisi terselesaikannya *puzzle*.

Fungsi Kurang(i) adalah fungsi yang menentukan apakah suatu *state* awal matriks dapat diselesaikan. Teoremanya, status tujuan hanya dapat dicapai dari status awal jika

$$\sum_{i=1}^{16} KURANG(i) + X$$

Bernilai genap. Nilai X sama dengan satu jika sel kosong pada posisi awal berada pada sel yang diarsir pada matriks di samping. Fungsi Kurang(i) sama dengan banyaknya ubin bernomor j sedemikian sehingga  $j < i$  dan  $POSISI(j) > POSISI(i)$ .  $POSISI(i)$  = posisi ubin bernomor i pada susunan yang diperiksa. Fungsi ini diimplementasikan dalam program sebagai  $KURANGFunc(matrix)$ . Nilai fungsi Kurang untuk masing-masing i ditampilkan beserta nilai dari  $\sum_{i=1}^{16} KURANG(i) + X$ .



Nilai *bound* tiap simpul adalah penjumlahan *cost* yang diperlukan untuk sampai suatu simpul x dari akar, dengan taksiran *cost* simpul x untuk sampai ke goal. Taksiran *cost* yang digunakan adalah jumlah ubin tidak kosong yang tidak berada pada tempat sesuai susunan akhir (*goal state*). Fungsi ini diimplementasikan dalam program sebagai  $gFunc(matrix)$ .

Setelah dipastikan bahwa matriks awal dapat diselesaikan, selanjutnya dilakukan penyelesaian *puzzle* menggunakan algoritma *Branch and Bound*. *State* matriks merupakan *node* dari *Tree* untuk *Branch and Bound*, dengan *state* awal matriks merupakan *node* akar. Fungsi ini diimplementasikan dalam program sebagai fungsi  $solve()$ . Ada beberapa objek yang diinisialisasi, yaitu sebagai berikut.

1. **startState**, bernilai matriks *state* awal,
2. **goalState**, bernilai matriks *state* akhir,
3. **Timer**, mengukur waktu eksekusi program,
4. Sebuah *Priority Queue* untuk menyimpan *node* matriks yang akan dibangkitkan anaknya dinisialisasi dengan elemen awal yaitu *state* awal matriks,
5. Sebuah *list visited* yang berisi *node* matriks yang sudah pernah dicapai,
6. Sebuah *list matrixStep* yang berisi *list* tahap penyelesaian matriks,
7. Sebuah *list of dictionary* dengan *key* berupa suatu *state* matriks dan *value*-nya adalah *state* matriks sebelum dikenai suatu gerakan yang menghasilkan *state* matriks *key*.

Satu per satu matriks pada *Priority Queue* di-*pop* dan dibangkitkan keempat kemungkinan perubahan yang dapat dilakukan, yaitu menggerakkan sel kosong ke atas, bawah, kiri, dan kanan, dengan mempertimbangkan validitas gerakan (tidak dapat menggeser sel kosong ke bawah bila sel kosong terletak di baris paling bawah matriks). Keempat matriks ini dicari nilai *cost*-nya


menggunakan *gFunc*. Selanjutnya keempat matriks ini di-*push* ke *Priority Queue* dengan urutan matriks bernilai *cost* terkecil terlebih dahulu. Keempat matriks ini dibuat *dictionary*-nya dengan properti seperti yang dijelaskan sebelumnya. Setelah itu semua, matriks selanjutnya pada *Priority Queue* di-*pop* dan dibangkitkan lagi anak-anaknya seperti matriks sebelumnya.

Proses di atas di-*loop* hingga jawaban didapatkan atau waktu eksekusi melebihi batas. Waktu eksekusi sengaja dibuat karena fungsi heuristik memiliki kemungkinan dapat membuat program terjebak dalam *infinite loop*, dikarenakan fungsi heuristik yang digunakan sebenarnya tidak cukup baik untuk kasus dengan jumlah gerakan yang cukup banyak (20 ke-atas). Waktu eksekusi dibatasi selama 300 detik.

Bila jawaban didapatkan, maka *list matrixStep* mulai diisi dengan meng-*push* matriks *goal state*, selanjutnya merunut *parent*-nya dengan menggunakan *dictionary* yang sudah dijelaskan semuanya, hingga ke matriks awal. Elemen-elemen pada *matrixStep* ditampilkan sebagai urutan langkah penyelesaian. Waktu eksekusi juga ditampilkan setelahnya.

## B. Screenshot Input dan Output Program

Berikut dilampirkan tangkapan layar *input* serta *output* program. Untuk contoh kasusnya dijelaskan lebih lanjut pada subbab E. Perlu diperhatikan bahwa *bonus* (pembuatan *GUI* yang menampilkan animasi urutan penyelesaian *puzzle*) tidak diimplementasikan untuk tugas kecil ini.



```
C:\Windows\System32\cmd.exe - python -u "src/15PuzzleSolverMain.py"
Microsoft Windows [Version 10.0.22581.100]
(c) Microsoft Corporation. All rights reserved.

D:\KULIAH\SMT 4\Stima\Tucil 3\bnb-15puzzle>python -u "src/15PuzzleSolverMain.py"

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

Dibuat oleh: Dzaky Fattan Rizqullah - 13520003

Input nama file, pastikan sudah terletak dalam folder test (contoh: input.txt): test1.txt_
```

**Gambar 2.** Antarmuka awal, dengan masukan file test1.txt



```
C:\Windows\System32\cmd.exe - python -u "src\15PuzzleSolverMain.py"
KURANG(14) = 3
KURANG(13) = 2
KURANG(11) = 1
KURANG(10) = 0
Jumlah Kurang(i) + X: 14
Status tujuan dapat dicapai, mencari solusi...

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

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

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

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

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

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

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

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

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

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

Langkah ke-10:
| 9 | 15 | 11 | 12 |
| 13 |   | 10 | 14 |

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

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

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

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

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

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

Langkah ke-16:
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 15 | 11 |
| 13 | 14 |   | 12 |

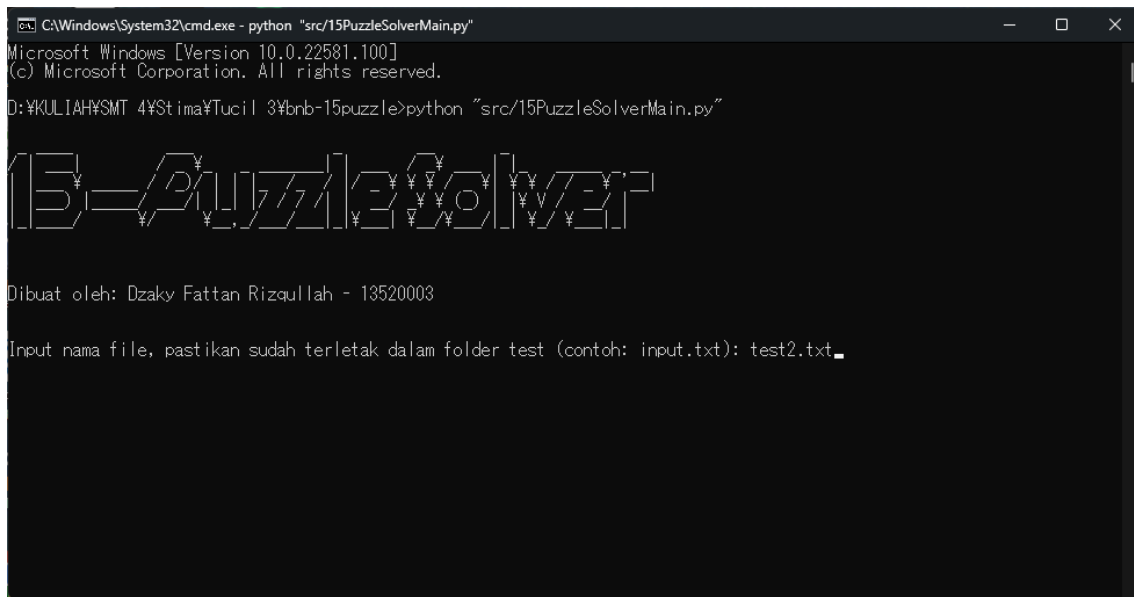
Langkah ke-17:
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 |   | 11 |
| 13 | 14 | 15 | 12 |

Langkah ke-18:
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 |   |
| 13 | 14 | 15 | 12 |

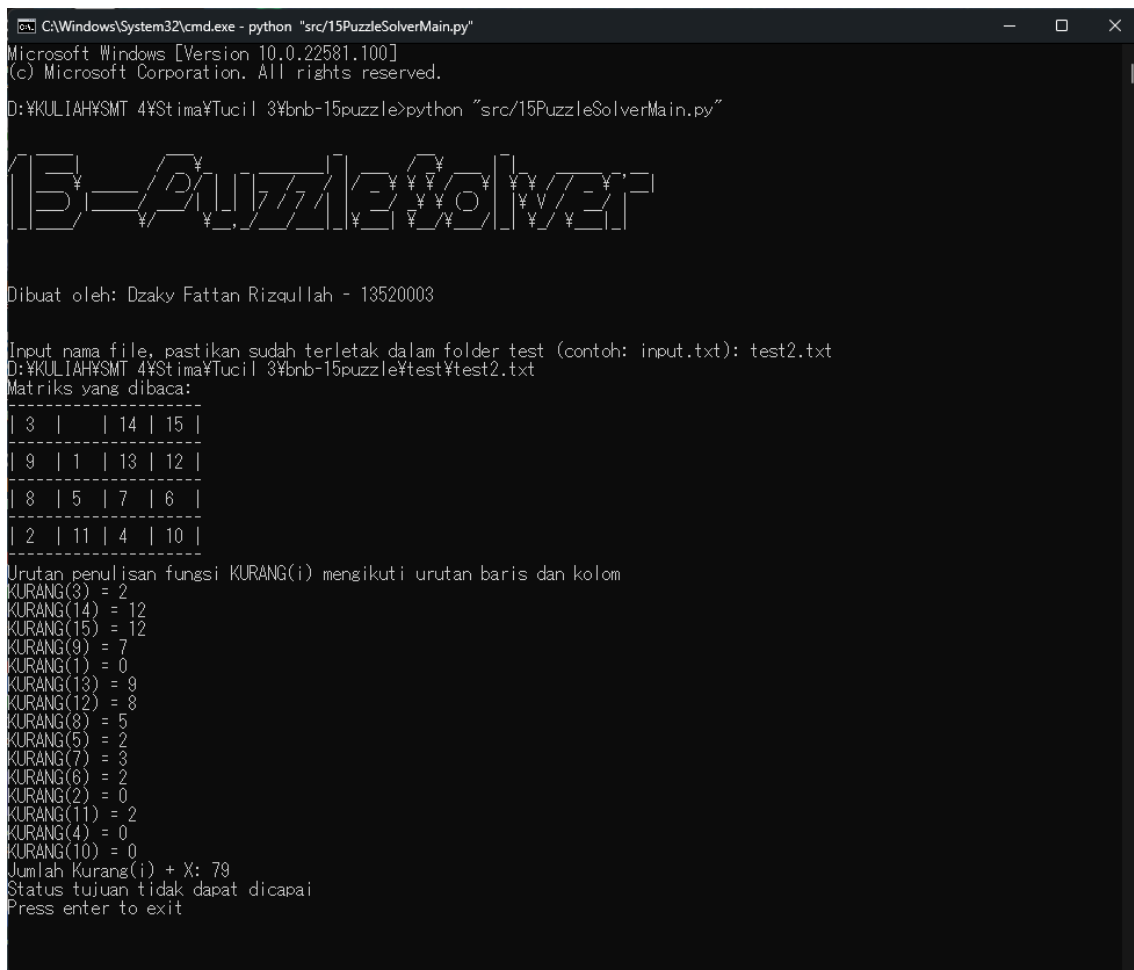
Langkah ke-19:
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 |   |

Waktu eksekusi: 8.823234 detik
Press enter to exit
```

**Gambar 4.** Hasil pencarian solusi untuk file test1.txt



**Gambar 5.** Antarmuka awal, dengan masukan file test2.txt



**Gambar 6.** Proses pencarian solusi untuk file test2.txt yang berujung gagal karena Status tujuan tidak dapat dicapai

```
C:\Windows\System32\cmd.exe - python "src/15PuzzleSolverMain.py"
Dibuat oleh: Dzaky Fattan Rizqullah - 13520003

Input nama file, pastikan sudah terletak dalam folder test (contoh: input.txt): test2.txt
D:\KULIAH\SMT 4\Stima\Tucil 3\bnb-15puzzle\test\test2.txt
Matriks yang dibaca:
-----
| 3 |   | 14 | 15 |
-----
| 9 | 1 | 13 | 12 |
-----
| 8 | 5 | 7 | 6 |
-----
| 2 | 11 | 4 | 10 |
-----

Urutan penulisan fungsi KURANG(i) mengikuti urutan baris dan kolom
KURANG(3) = 2
KURANG(14) = 12
KURANG(15) = 12
KURANG(9) = 7
KURANG(1) = 0
KURANG(13) = 9
KURANG(12) = 8
KURANG(8) = 5
KURANG(5) = 2
KURANG(7) = 3
KURANG(6) = 2
KURANG(2) = 0
KURANG(11) = 2
KURANG(4) = 0
KURANG(10) = 0
Jumlah Kurang(i) + X: 79
Status tujuan tidak dapat dicapai
Press enter to exit

D:\KULIAH\SMT 4\Stima\Tucil 3\bnb-15puzzle>python "src/15PuzzleSolverMain.py"

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

Urutan penulisan fungsi KURANG(i) mengikuti urutan baris dan kolom
KURANG(5) = 4
KURANG(1) = 0
KURANG(3) = 1
KURANG(4) = 1
KURANG(2) = 0
KURANG(10) = 4
KURANG(8) = 2
KURANG(9) = 2
KURANG(14) = 5
KURANG(6) = 0
KURANG(11) = 1
KURANG(13) = 2
KURANG(15) = 2
KURANG(7) = 0
KURANG(12) = 0
Jumlah Kurang(i) + X: 34
Status tujuan dapat dicapai, mencari solusi...
```

**Gambar 7.** Proses pencarian solusi untuk file test3.txt



```

C:\Windows\System32\cmd.exe - python "src\13PuzzleSolveMain.py"
KURANG(11) = 1
KURANG(13) = 2
KURANG(15) = 2
KURANG(7) = 0
KURANG(12) = 0
Jumlah Kurang(i) + X: 34
Status tujuan dapat dicapai, mencari solusi...

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

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

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

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

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

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

Langkah ke-6:
5 | 1 | 3 | 4 |
2 | 10 | 6 | 8 |
8 | 11 | 12 |
13 | 14 | 15 | 7 |

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

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

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

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

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

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

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

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

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

Langkah ke-16:
1 | 2 | 3 | 4 |
5 | 6 | 8 | 12 |
9 | 10 | 11 | 7 |
13 | 14 | 15 |

Langkah ke-17:
1 | 2 | 3 | 4 |
5 | 6 | 8 | 12 |
9 | 10 | 17 |
13 | 14 | 11 | 15 |

Langkah ke-18:
1 | 2 | 3 | 4 |
5 | 6 | 8 | 12 |
9 | 10 | 7 |
13 | 14 | 11 | 15 |

Langkah ke-19:
1 | 2 | 3 | 4 |
5 | 6 | 8 |
9 | 10 | 7 | 12 |
13 | 14 | 11 | 15 |

Langkah ke-20:
1 | 2 | 3 | 4 |
5 | 6 | 8 |
9 | 10 | 7 | 12 |
13 | 14 | 11 | 15 |

Langkah ke-21:
1 | 2 | 3 | 4 |
5 | 6 | 7 | 8 |
9 | 10 | 12 |
13 | 14 | 11 | 15 |

Langkah ke-22:
1 | 2 | 3 | 4 |
5 | 6 | 7 | 8 |
9 | 10 | 11 | 12 |
13 | 14 | 15 |

Langkah ke-23:
1 | 2 | 3 | 4 |
5 | 6 | 7 | 8 |
9 | 10 | 11 | 12 |
13 | 14 | 15 |

Waktu eksekusi: 2,404503 detik
Press enter to exit

```

**Gambar 8.** Hasil pencarian solusi untuk file test3.txt

```
C:\Windows\System32\cmd.exe - python "src/15PuzzleSolverMain.py"
D:\KULIAH\SMT 4\Stima\Tucil 3\bnb-15puzzle>python "src/15PuzzleSolverMain.py"

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

Dibuat oleh: Dzaky Fattan Rizqullah - 13520003

Input nama file, pastikan sudah terletak dalam folder test (contoh: input.txt):
test4.txt
D:\KULIAH\SMT 4\Stima\Tucil 3\bnb-15puzzle\test\test4.txt
Matriks yang dibaca:
-----
| 10 | 13 | 4 | 1 | 15 | 8 |
| 1 | 14 | 15 | 5 | 3 | 7 |
| 11 | 9 | 3 | 12 | 6 | 2 |
| 8 | 6 | 7 | 2 | 13 | 10 |
-----
Urutan penulisan fungsi KURANG(i) mengikuti urutan baris dan kolom
KURANG(10) = 9
KURANG(13) = 11
KURANG(4) = 3
KURANG(1) = 0
KURANG(14) = 9
KURANG(15) = 9
KURANG(5) = 2
KURANG(11) = 6
KURANG(9) = 5
KURANG(3) = 1
KURANG(12) = 4
KURANG(8) = 3
KURANG(6) = 1
KURANG(7) = 1
KURANG(2) = 0
Jumlah Kurang(i) + X: 77
Status tujuan tidak dapat dicapai
Press enter to exit
```

**Gambar 9.** Proses pencarian solusi untuk file test4.txt yang berujung gagal karena Status tujuan tidak dapat dicapai

```
C:\Windows\System32\cmd.exe - python "src/15PuzzleSolverMain.py"
D:\KULIAH\SMT 4\Stima\Tucil 3\bnb-15puzzle>python "src/15PuzzleSolverMain.py"

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

Urutan penulisan fungsi KURANG(i) mengikuti urutan baris dan kolom
KURANG(1) = 0
KURANG(2) = 0
KURANG(6) = 3
KURANG(3) = 0
KURANG(5) = 1
KURANG(7) = 1
KURANG(4) = 0
KURANG(9) = 1
KURANG(11) = 2
KURANG(10) = 1
KURANG(15) = 4
KURANG(13) = 2
KURANG(14) = 2
KURANG(12) = 1
KURANG(8) = 0
Jumlah Kurang(i) + X: 30
Status tujuan dapat dicapai, mencari solusi...
```

**Gambar 10.** Proses pencarian solusi untuk file test5.txt

```
C:\Windows\System32\cmd.exe - python "src\15PuzzleSolverMain.py"
KURANG(14) = 2
KURANG(12) = 1
KURANG(8) = 0
Jumlah Kurang(i) + X: 30
Status tujuan dapat dicapai, mencari solusi...

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

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

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

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

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

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

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

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

Langkah ke-8:
-----
| 1 | 2 | 6 | 3 |
| 5 | 11 | 14 |
| 9 | 10 | 7 | 8 |
| 13 | 14 | 15 | 12 |
-----

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

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

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

C:\Windows\System32\cmd.exe - python "src\15PuzzleSolverMain.py"
| 13 | 14 | 15 | 12 |
-----
Langkah ke-12:
-----
| 1 | 2 | 3 | 4 |
| 5 | 11 | 6 | 8 |
| 9 | 10 | 7 |
| 13 | 14 | 15 | 12 |
-----
Langkah ke-13:
-----
| 1 | 2 | 3 | 4 |
| 5 | 11 | 6 | 8 |
| 9 | 10 | 7 | 12 |
| 13 | 14 | 15 |
-----
Langkah ke-14:
-----
| 1 | 2 | 3 | 4 |
| 5 | 11 | 6 | 8 |
| 9 | 10 | 7 | 12 |
| 13 | 14 | 15 |
-----
Langkah ke-15:
-----
| 1 | 2 | 3 | 4 |
| 5 | 11 | 6 | 8 |
| 9 | 10 | 7 | 12 |
| 13 | 14 | 15 |
-----
Langkah ke-16:
-----
| 1 | 2 | 3 | 4 |
| 5 | 11 | 6 | 8 |
| 9 | 7 | 12 |
| 13 | 10 | 14 | 15 |
-----
Langkah ke-17:
-----
| 1 | 2 | 3 | 4 |
| 5 | 6 | 8 |
| 9 | 11 | 7 | 12 |
| 13 | 10 | 14 | 15 |
-----
Langkah ke-18:
-----
| 1 | 2 | 3 | 4 |
| 5 | 6 | 8 |
| 9 | 11 | 7 | 12 |
| 13 | 10 | 14 | 15 |
-----
Langkah ke-19:
-----
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 11 | 12 |
| 13 | 10 | 14 | 15 |
-----
Langkah ke-20:
-----
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 11 | 12 |
| 13 | 10 | 14 | 15 |
-----
Langkah ke-21:
-----
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 |
-----
Langkah ke-22:
-----
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 |
-----
Langkah ke-23:
-----
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 |
-----
Waktu eksekusi: 34.9369775 detik
Press enter to exit
```

**Gambar 11.** Hasil pencarian solusi untuk file test5.txt

### C. Check List

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

### D. Kode Program

Berikut dilampirkan kode program yang ditulis dalam bahasa Python 3. Program dibagi menjadi tiga file, yang akan dijabarkan masing-masing di bawah.

`MatrixController.py`, merupakan file yang berisi fungsi manipulasi matriks yang merupakan representasi 15-puzzle.

```
import os

# read matrix from .txt file from specified input
def readMatrix(filename):
    matrix = []
    # get parent folder path
    fileDir = os.path.join(os.getcwd(), 'test', filename)
    print(fileDir)
    try:
        with open(fileDir, 'r') as f:
            for line in f:
                tempList = list(line.strip().split())
                matrix.append([int(x) for x in tempList])
        return matrix
    except FileNotFoundError:
        print("File tidak ditemukan")
        return None

# print matrix properly
def printMatrix(matrix):
    if (matrix != None):
        print("-----")
        for i in range(len(matrix)):
            for j in range(len(matrix[i])):
                strToPrint = str(matrix[i][j]) + " " if matrix[i][j] < 10 else
str(matrix[i][j])
                if strToPrint == "16": strToPrint = " "
                print("| " + strToPrint, end=" ")
            if j == len(matrix[i])-1:
                print("|")
```

```

        print("-----")
    else:
        print("Matrix kosong")

# find 16 position in matrix, return -1, -1 if not found
def find16(matrix):
    for i in range(len(matrix)):
        for j in range(len(matrix[i])):
            if (matrix[i][j] == 16):
                return i, j
    return -1, -1

# copy matrix
def copyMatrix(matrix):
    tempMatrix = []
    for i in range(len(matrix)):
        tempMatrix.append([])
        for j in range(len(matrix[i])):
            tempMatrix[i].append(matrix[i][j])
    return tempMatrix

# move 16 to specified position
def move16(matrix, direction):
    i, j = find16(matrix)
    tempMatrix = copyMatrix(matrix)
    if direction == 0:
        if i > 0:
            tempMatrix[i][j], tempMatrix[i-1][j] = matrix[i-1][j], matrix[i][j]
    elif direction == 2:
        if i < len(matrix)-1:
            tempMatrix[i][j], tempMatrix[i+1][j] = matrix[i+1][j], matrix[i][j]
    elif direction == 3:
        if j > 0:
            tempMatrix[i][j], tempMatrix[i][j-1] = matrix[i][j-1], matrix[i][j]
    elif direction == 1:
        if j < len(matrix[i])-1:
            tempMatrix[i][j], tempMatrix[i][j+1] = matrix[i][j+1], matrix[i][j]
    return tempMatrix

```

Solver.py, merupakan file yang berisi fungsi-fungsi tempat diimplementasikannya algoritma *Branch and Bound*.

```

import time
import MatrixController as m

# goal state

```

```

goalState = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12], [13, 14, 15, 16]]
# counter to create a key for each matrixState
counter = 0
tempDict = {}
visited = []
startState = []
timelimit = 300

# Function Kurang(i)
def KURANGFunc(matrix):
    kurang = 0
    emptyPos = -1
    print("Urutan penulisan fungsi KURANG(i) mengikuti urutan baris dan kolom")
    for i in range(len(matrix)):
        for j in range(len(matrix[i])):
            if matrix[i][j] == 16:
                emptyPos = i + j
            tempKurang = 0
            for k in range(i * len(matrix) + j + 1, len(matrix)**2):
                if matrix[k//len(matrix)][k % len(matrix)] < matrix[i][j]:
                    tempKurang += 1
            if (matrix[i][j] != 16):
                print("KURANG(" + str(matrix[i][j]) + ") = " + str(tempKurang))
            kurang += tempKurang
    return kurang if emptyPos % 2 == 0 else kurang+1

# our heuristic function, the number of misplaced tiles
def gFunc(matrix):
    g = 0
    for i in range(len(matrix)):
        for j in range(len(matrix[i])):
            if matrix[i][j] != goalState[i][j] and matrix[i][j] != 16:
                g += 1
    return g

# get the key of the matrixState
def getKey(matrixAsVal):
    for key, value in tempDict.items():
        if matrixAsVal == value:
            return key

# create matrixStep from the goal state to the start state
def connectParent(newMatrix, matrixStep, parent):
    matrixStep.append(newMatrix)
    tempPar = parent[getKey(newMatrix)]
    while tempPar != startState:

```

```

        if tempPar not in matrixStep:
            matrixStep.append(tempPar)
            tempPar = parent[getKey(tempPar)]
        matrixStep.append(startState)
        matrixStep.reverse()

# solver
def solve(matrix, matrixStep, queue, parent):
    # matrix stores the current matrixState
    # matrixStep stores the steps of solving the matrix
    # queue stores the list of matrixState that are not yet visited, is a PrioQueue
    # to make sure we visited the matrixState with the lowest gFunc
    # parent stores the parent of each matrixState

    # a timer, to prevent a very long execution time
    start = time.time()

    # append first matrix state to queue
    queue.append(matrix)
    count = 0
    while queue and time.time() - start < timelimit:
        matrix = queue.pop(0)

        # check if matrix already visited, this part can be improved
        if matrix in visited:
            continue
        visited.append(matrix)

        # if start state is already a goal state, return the matrixStep
        if matrix == goalState:
            matrixStep.append(matrix)
            return

        # iterate all 4 possible direction
        for i in range(4):
            newMatrix = m.move16(matrix, i)

            # queue the newMatrix, could also be improved
            queue.append(newMatrix)

            if len(queue) > 1:

                # gFunc check
                pos = queue.index(newMatrix)
                while (pos > 0 and gFunc(queue[pos-1]) > gFunc(queue[pos])):
                    queue[pos-1], queue[pos] = queue[pos], queue[pos-1]

```



```

        pos -= 1

        # create a key for the newMatrix, to be used as a key in the parent
dictionary
        global counter
        tempDict[counter] = newMatrix
        parent[counter] = matrix
        counter += 1

        # finally, check if newMatrix is the goal state
        if newMatrix == goalState:
            connectParent(newMatrix, matrixStep, parent)
            return

        # if time limit exceeded
        if queue:
            connectParent(matrix, matrixStep, parent)

```

15PuzzleSolverMain.py, merupakan program utama yang menjalankan fungsi interaksi dengan pengguna. Masukan dan keluaran program di-handle oleh file ini.

```

import time
import sys
import MatrixController as m
import Solver as s

# variables for BnB
matrixStep = []
liveQueue = []
f = 1
parent = {}

def printTitle():
    print("""
    _ _ _ _ _      _ _ _ _ _      _ _ _ _ _      _ _ _ _ _      _ _ _ _ _
 / | | | |      / | | | | _ _ _ _ _ | | | | | / | | | | _ _ _ _ _
 | | | | | _ _ _ / | | | | / | | | | / | | | | _ _ _ _ _ / | | | | _ _ _
 | | | | | _ _ _ / | | | | / | | | | / | | | | _ _ _ _ _ / | | | | _ _ _
 | | | | | _ _ _ / | | | | / | | | | / | | | | _ _ _ _ _ / | | | | _ _ _

    """)
    print("\nDibuat oleh: Dzaky Fattan Rizqullah - 13520003\n")

# The Main Program
sys.setrecursionlimit((10*3))

```

```

printTitle()
ipt = input("\nInput nama file, pastikan sudah terletak dalam folder test (contoh:
input.txt): ")
matrix = m.readMatrix(ipt)
if (matrix != None):
    print("Matriks yang dibaca:")
    m.printMatrix(matrix)
    kurang = s.KURANGFunc(matrix)
    print("Total kurang:", kurang)
    if kurang % 2 == 1:
        print("Status tujuan tidak dapat dicapai")
    else:
        start = time.time_ns()
        print("Status tujuan dapat dicapai, mencari solusi...")
        s.startState = matrix
        s.solve(matrix, matrixStep, liveQueue, parent)
        end = time.time_ns()
        if (matrixStep != []):
            print("\nPosisi awal:")
            m.printMatrix(matrixStep.pop(0))
            for i in range(len(matrixStep)):
                print("\nLangkah ke-" + str(i+1) + ":")
                m.printMatrix(matrixStep[i])
                print("gFunc: " + str(s.gFunc(matrixStep[i])))
        duration = (end - start)
        if duration > 1000000000:
            duration = duration / 1000000000
            print("\nWaktu eksekusi:", duration, "detik")
        elif duration > 1000000:
            duration = duration / 1000000
            print("\nWaktu eksekusi:", duration, "ms")
        else:
            print("\nWaktu eksekusi:", duration, "ns")
        # prompt before exiting
        input("Press enter to exit\n")

```

### E. Berkas Persoalan

Berikut dilampirkan lima buat persoalan 15-puzzle yang digunakan sebagai contoh kasus untuk tugas kecil ini. Tiga di antaranya (test1.txt, test3.txt, test5.txt) dapat diselesaikan, sementara dua lainnya (test2.txt dan test4.txt) tidak dapat diselesaikan. Keseluruhan file persoalan diletakkan dalam folder test

File test	test1.txt	test2.txt	test3.txt	test4.txt	test5.txt
Matriks	1 2 3 4 5 6 7 8 9 12 15 14 13 11 16 10	3 16 14 15 9 1 13 12 8 5 7 6 2 11 4 10	5 1 3 4 2 10 16 8 9 14 6 11 13 15 7 12	10 13 4 16 1 14 15 5 11 9 3 12 8 6 7 2	1 2 6 3 16 5 7 4 9 11 10 15 13 14 12 8
Waktu eksekusi	±8.75 detik	-	±2.4 detik	-	±35 detik

### F. Alamat Drive dan Github

Drive:	<a href="https://drive.google.com/drive/folders/1DQ0uRJXf9d5Kjcs_h8G9EMOLFURw5wt4?usp=sharing">https://drive.google.com/drive/folders/1DQ0uRJXf9d5Kjcs_h8G9EMOLFURw5wt4?usp=sharing</a>  (Link di atas akan menampilkan drive berisi kode program, executable, testcase, serta file laporan ini. Pastikan diakses dengan akun fakultas. )
Github:	<a href="https://github.com/DzakyFattan/bnb-15puzzle">https://github.com/DzakyFattan/bnb-15puzzle</a>