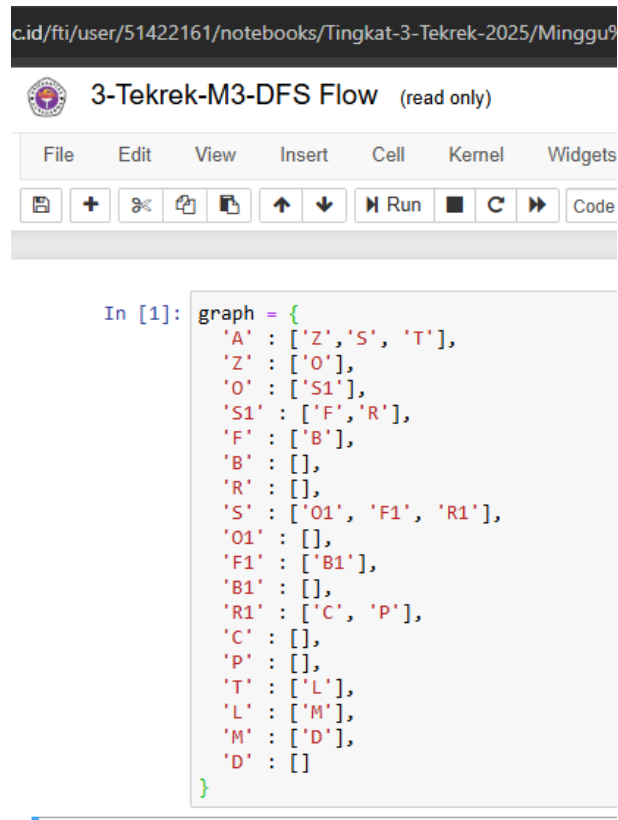


MUHAMMAD TARMIDZI BARIQ

51422161

3IA11

M3



The screenshot shows a Jupyter Notebook titled "3-Tekrek-M3-DFS Flow (read only)". The notebook interface includes a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", and "Widgets". Below the menu bar is a toolbar with icons for saving, adding cells, zooming, copying, pasting, undo, redo, running, and code execution. The main content area displays a Python code cell with the following code:

```
In [1]: graph = {  
    'A' : ['Z', 'S', 'T'],  
    'Z' : ['O'],  
    'O' : ['S1'],  
    'S1' : ['F', 'R'],  
    'F' : ['B'],  
    'B' : [],  
    'R' : [],  
    'S' : ['O1', 'F1', 'R1'],  
    'O1' : [],  
    'F1' : ['B1'],  
    'B1' : [],  
    'R1' : ['C', 'P'],  
    'C' : [],  
    'P' : [],  
    'T' : ['L'],  
    'L' : ['M'],  
    'M' : ['D'],  
    'D' : []  
}
```

Representasi graf dalam bentuk adjacency list (daftar ketetanggaan). Setiap kunci dalam dictionary adalah sebuah node/vertex, dan nilai yang terkait dengan kunci tersebut adalah list yang berisi semua node yang terhubung langsung (adjacent) dengan node tersebut.

Misalnya:

- Node 'A' terhubung langsung ke node 'Z', 'S', dan 'T'
- Node 'Z' terhubung ke 'O'
- Node 'B', 'R', 'O1', 'C', 'P', dan 'D' tidak terhubung ke node lain (list kosong)

Graf ini bersifat directed (terarah), karena hubungan antar node hanya satu arah.

```
In [2]: visited = set() # Set to keep track of visited nodes of graph.
```

```
In [3]: def dfs(visited, graph, node): #function for dfs
        if node not in visited:
            print (node)
            visited.add(node)
            for neighbour in graph[node]:
                dfs(visited, graph, neighbour)
```

- `visited = set()` - Membuat sebuah set kosong untuk menyimpan node-node yang sudah dikunjungi. Set digunakan karena operasi pengecekan keanggotaan (`in`) sangat efisien pada struktur data ini.
- Fungsi `dfs(visited, graph, node)` melakukan penelusuran graf dengan strategi depth-first, artinya:

Jika node belum pernah dikunjungi, maka:

- Cetak node tersebut
- Tambahkan node ke dalam set `visited`
- Untuk setiap tetangga dari node tersebut, panggil rekursif fungsi `dfs` dengan tetangga sebagai node baru

```
In [4]: print("Following is the Depth-First Search")
```

```
Following is the Depth-First Search
```

```
In [5]: dfs(visited, graph, 'A')
```

```
A
Z
O
S1
F
B
R
S
O1
F1
B1
R1
C
P
T
L
M
D
```

```
In [ ]:
```

- `print("Following is the Depth-First Search")` akan mencetak pesan tersebut ke layar.
- `dfs(visited, graph, 'A')` memanggil fungsi DFS dengan parameter:
 - `visited`: set kosong yang sudah dideklarasikan sebelumnya
 - `graph`: struktur graf yang telah didefinisikan
 - `'A'`: node awal untuk memulai penelusuran