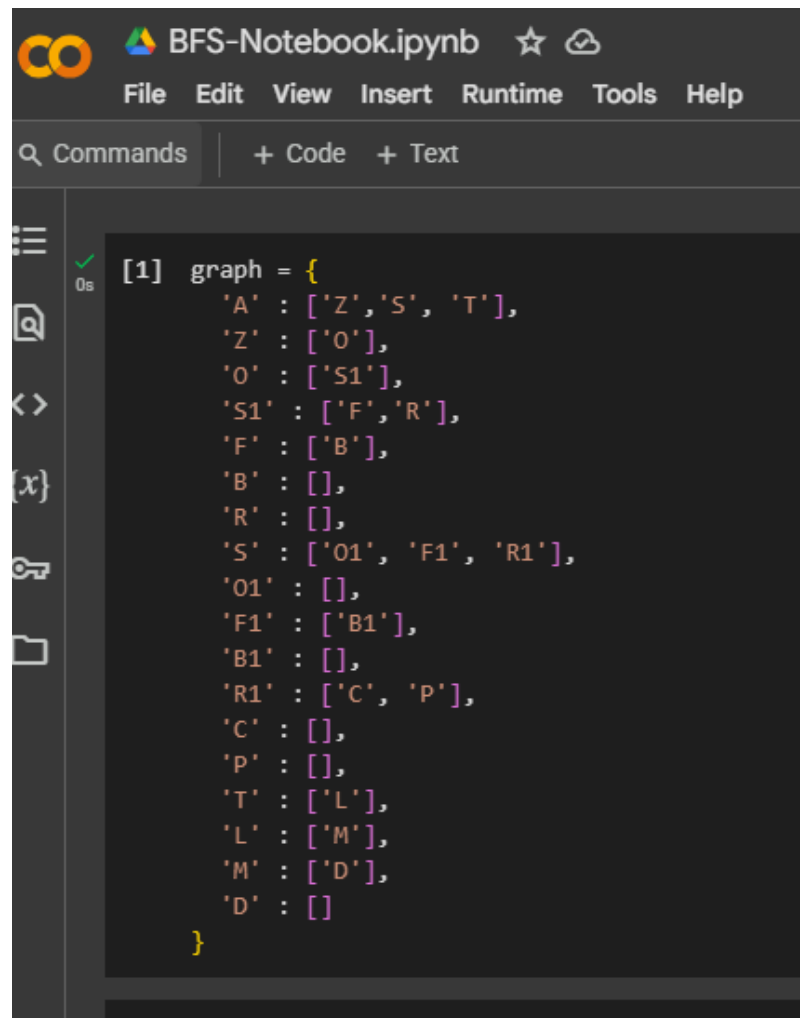


MUHAMMAD TARMIDZI BARIQ

51422161

3IA11

M4



The image shows a Jupyter Notebook interface with a dark theme. The title bar at the top reads "BFS-Notebook.ipynb" and includes icons for a star and a cloud. Below the title bar is a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". A "Commands" search bar is located below the menu bar, with "+ Code" and "+ Text" buttons. On the left side, there is a sidebar with icons for a menu, search, navigation, and file explorer. The main area displays a code cell with the following Python code:

```
[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' : []  
}
```

Mendefinisikan sebuah struktur data graf menggunakan dictionary di Python



```
[2] visited = [] # List for visited nodes.  
    queue = []    #Initialize a queue
```

Implementasi algoritma Breadth-First Search (BFS) pada graf

1. menyimpan node-node yang sudah dikunjungi selama proses traversal.
2. antrian (queue) yang menyimpan node-node yang akan dieksplorasi berikutnya.



```
def bfs(visited, graph, node): #function for BFS  
    visited.append(node)  
    queue.append(node)  
  
    while queue:                # Creating loop to visit each node  
        m = queue.pop(0)  
        print (m, end = " ")  
  
        for neighbour in graph[m]:  
            if neighbour not in visited:  
                visited.append(neighbour)  
                queue.append(neighbour)
```

- visited.append(node) dan queue.append(node)

Menambahkan node awal ke daftar visited dan juga queue.

Ini adalah titik awal dari penelusuran.

- while queue:

Loop ini akan terus berjalan selama masih ada node di antrian.

FIFO: Node pertama yang masuk queue akan diproses duluan.

- m = queue.pop(0)

Mengambil node pertama dari antrian.

Node inilah yang sedang "diproses" atau dikunjungi saat itu.

- print(m, end=" ")

Menampilkan node yang sedang dikunjungi.

- for neighbour in graph[m]:

Mengecek semua tetangga dari node saat ini (m).

Kalau tetangganya belum dikunjungi, maka:

Ditambahkan ke visited

Dimasukkan ke queue, supaya nanti juga diproses

```
[4] # Driver Code
    print("Following is the Breadth-First Search")
    bfs(visited, graph, 'A')
```

```
⇒ Following is the Breadth-First Search
A Z S T O O1 F1 R1 L S1 B1 C P M F R D B
```

Output ini menunjukkan urutan node yang dikunjungi oleh algoritma BFS, dimulai dari A, lalu node-node tetangganya, lalu tetangga dari tetangga, dan seterusnya secara melebar (level demi level).