

Code Python :

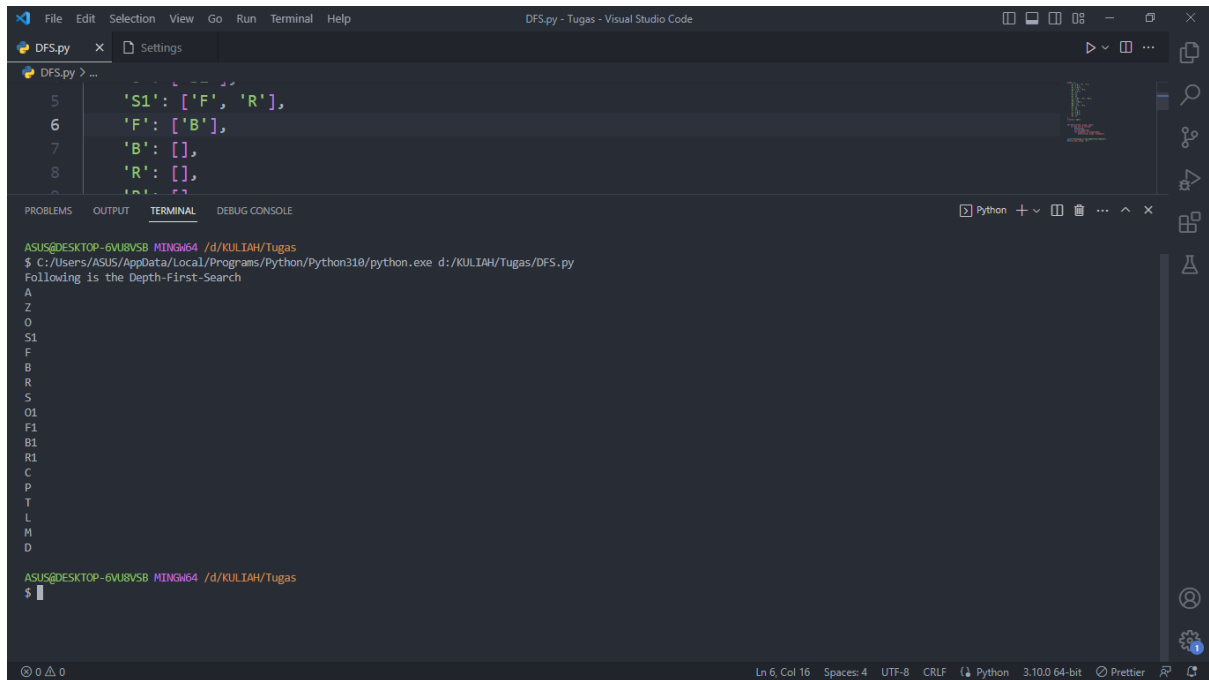
```
graph = {
    'A': ['Z', 'S', 'T'],
    'Z': ['O'],
    'O': ['S1'],
    'S1': ['F', 'R'],
    'F': ['B'],
    'B': [],
    'R': [],
    'R': [],
    'S': ['O1', 'F1', 'R1'],
    'O1': [],
    'F1': ['B1'],
    'B1': [],
    'R1': ['C', 'P'],
    'C': [],
    'P': [],
    'T': ['L'],
    'L': ['M'],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

visited = set()

def dfs(visited, graph, node):
    if node not in visited:
        print(node)
        visited.add(node)
        for neighbour in graph[node]:
            dfs(visited, graph, neighbour)

print("Following is the Depth-First-Search")
dfs(visited, graph, 'A')
```

Hasil algoritma DFS :



```
DFS.py
5 'S1': ['F', 'R'],
6 'F': ['B'],
7 'B': [],
8 'R': [],
9

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
ASUS@DESKTOP-6VU8V5B MINGW64 /d/KULIAH/Tugas
$ C:/Users/ASUS/AppData/Local/Programs/Python/Python310/python.exe d:/KULIAH/Tugas/DFS.py
Following is the Depth-First-Search
A
Z
O
S1
F
B
R
S
O1
F1
B1
R1
C
P
T
L
M
D
ASUS@DESKTOP-6VU8V5B MINGW64 /d/KULIAH/Tugas
$
```

Penjelasan algoritma DFS di atas :

Code python di atas merupakan implementasi dari algoritma Depth First Search (DFS) pada sebuah graph. Graph adalah sebuah dictionary dengan key berupa nama node, dan memiliki value yang berupa list node.

Setelah itu, sebuah variabel set visited dideklarasikan untuk menyimpan node node yang telah dikunjungi oleh DFS. Lalu function dfs diimplementasikan dengan menerima 3 parameter, yaitu visited, graph, dan node. Fungsi ini dapat mengeksekusi DFS pada graph dimulai dari node yang diberikan.

Function dfs memiliki 3 bagian utama. Pertama function akan melakukan pengecekan apakah node yang diinputkan telah dikunjungi atau belum. Jika belum, maka node tersebut akan dicetak (print()) dan dimasukkan pada variabel visited.

Selanjutnya, function melakukan iterasi / pengulangan kode pada node node yang lainnya pada node yang satu level, untuk setiap node yang lain function dfs akan dijalankan secara rekursif atau berulang kali. Maksudnya algoritma DFS akan terus menjelajahi graph secara mendalam, melewati setiap node yang sedang dikunjungi, sebelum kembali ke node asalnya untuk mengecek node node yang lainnya.

Setelah seluruh node telah dikunjungi, algoritma akan kembali ke node asal yang kemudian akan mengecek node level selanjutnya, begitu seterusnya hingga seluruh node pada graph diinputkan dalam variabel visited. Pada akhirnya setiap node yang telah dikunjungi akan dicetak (print()) sebagai hasil dari algoritma DFS.