

- **BFS Graph Search**

- **Code:**

```
# TC1_LAB2
# SIA VASHIST_ 20190802107
# BFS Search & DFS Search

graph = {
    'A': ['B', 'C'],
    'B': ['D', 'E'],
    'C': ['F', 'G'],
    'D': ['H'],
    'E': ['I', 'J'],
    'F': ['K'],
    'G': [],
    'H': [],
    'I': [],
    'J': [],
    'K': []
}

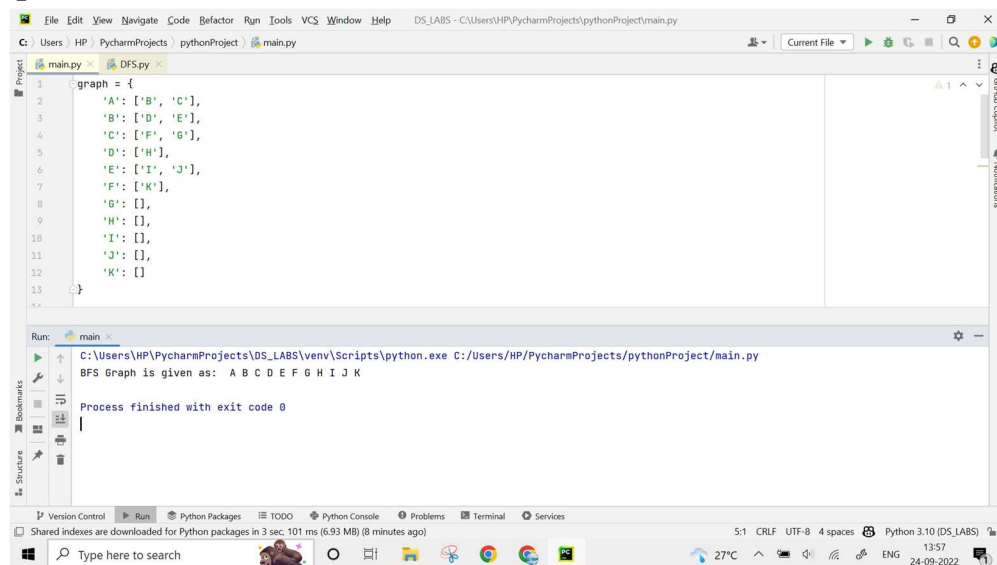
def bfs_connected_component(graph, start):
    # keep track of all visited nodes
    explored = []
    # keep track of nodes to be checked
    queue = [start]

    # keep looping until there are nodes still to be checked
    while queue:
        # pop shallowest node (first node) from queue
        node = queue.pop(0)
        if node not in explored:
            # add node to list of checked nodes
            explored.append(node)
            neighbours = graph[node]

            # add neighbours of node to queue
            for neighbour in neighbours:
                queue.append(neighbour)
    return explored

# drivers code
path = bfs_connected_component(graph, 'A')
print("BFS Graph is given as: ", " ".join(path))
```

- **Output:**



The screenshot shows the PyCharm IDE with the same code as above. The 'Run' window at the bottom displays the output: 'BFS Graph is given as: A B C D E F G H I J K'. The status bar at the bottom indicates the file encoding is UTF-8, the line length is 4 spaces, and the Python version is 3.10 (DS_LABS).

Lab report: TC1- AI/ML

- **DFS Graph Search**

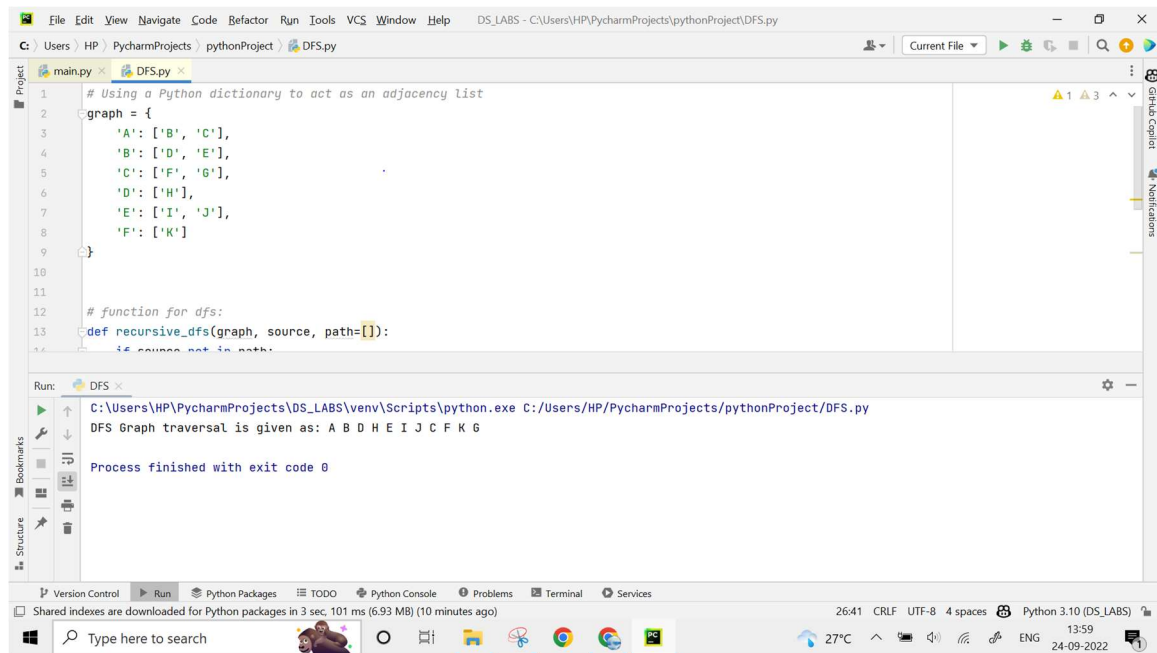
- **Code:**

```
# Using a Python dictionary to act as an adjacency list
graph = {
    'A': ['B', 'C'],
    'B': ['D', 'E'],
    'C': ['F', 'G'],
    'D': ['H'],
    'E': ['I', 'J'],
    'F': ['K']
}

# function for dfs:
def recursive_dfs(graph, source, path=[]):
    if source not in path:
        path.append(source)
        if source not in graph:
            # leaf node, backtrack
            return path
        for neighbour in graph[source]:
            path = recursive_dfs(graph, neighbour, path)
    return path

# Driver Code
path = recursive_dfs(graph, 'A')
print("DFS Graph traversal is given as:", " ".join(path))
```

- **Output:**



The screenshot shows the PyCharm IDE interface. The main editor window displays the Python code for DFS graph search. The code defines a graph as a dictionary and a recursive function to traverse it. The output window at the bottom shows the execution result: "DFS Graph traversal is given as: A B D H E I J C F K G". The status bar at the bottom indicates the file encoding (UTF-8), line length (4 spaces), and the Python version (3.10).

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help DS_LABS - C:\Users\HP\PycharmProjects\pythonProject\DFS.py
C:\Users\HP\PycharmProjects\pythonProject\DFS.py
main.py DFS.py
1 # Using a Python dictionary to act as an adjacency list
2 graph = {
3     'A': ['B', 'C'],
4     'B': ['D', 'E'],
5     'C': ['F', 'G'],
6     'D': ['H'],
7     'E': ['I', 'J'],
8     'F': ['K']
9 }
10
11
12 # function for dfs:
13 def recursive_dfs(graph, source, path=[]):
14     if source not in path:
15         path.append(source)
16         if source not in graph:
17             # leaf node, backtrack
18             return path
19         for neighbour in graph[source]:
20             path = recursive_dfs(graph, neighbour, path)
21     return path
22
23 # Driver Code
24 path = recursive_dfs(graph, 'A')
25 print("DFS Graph traversal is given as:", " ".join(path))
Run DFS
C:\Users\HP\PycharmProjects\DS_LABS\venv\Scripts\python.exe C:/Users/HP/PycharmProjects/pythonProject/DFS.py
DFS Graph traversal is given as: A B D H E I J C F K G
Process finished with exit code 0
Shared indexes are downloaded for Python packages in 3 sec, 101 ms (6.93 MB) (10 minutes ago)
26:41 CRLF UTF-8 4 spaces Python 3.10 (DS_LABS)
Type here to search 27°C 13:59 24-09-2022
```