DAA LAB-6 REPORT

**AIM:

1) To Implement BFS & DFS (Inorder Traversal) for a given Graph.
2) To check if a graph is strongly connected.
3) To print pre & post visited times.

**THEORY:

Traversing the graph means examining all the nodes and vertices of the graph. There are two standard methods by using which, we can traverse the graphs.

- **Breadth First Search**
- **Depth First Search**

**Breadth First Search (BFS) Algorithm:

Breadth first search is a graph traversal algorithm that starts traversing the graph from root node and explores all the neighboring nodes. Then, it selects the nearest node and explore all the unexplored nodes. The algorithm follows the same process for each of the nearest node until it finds the goal.

The algorithm of breadth first search is given below. The algorithm starts with examining the node A and all of its neighbours. In the next step, the neighbours of the nearest node of A are explored and process continues in the further steps.

The algorithm explores all neighbours of all the nodes and ensures that each node is visited exactly once and no node is visited twice.

**Depth First Search (DFS) Algorithm:

Depth first search (DFS) algorithm starts with the initial node of the graph G, and then goes to deeper and deeper until we find the goal node or the node which has no children. The algorithm, then backtracks from the dead end towards the most recent node that is yet to be completely unexplored.

The data structure which is being used in DFS is stack. The process is similar to BFS algorithm. In DFS, the edges that lead to an unvisited node are called discovery edges while the edges that lead to an already visited node are called block edges.

Difference between BFS and DFS

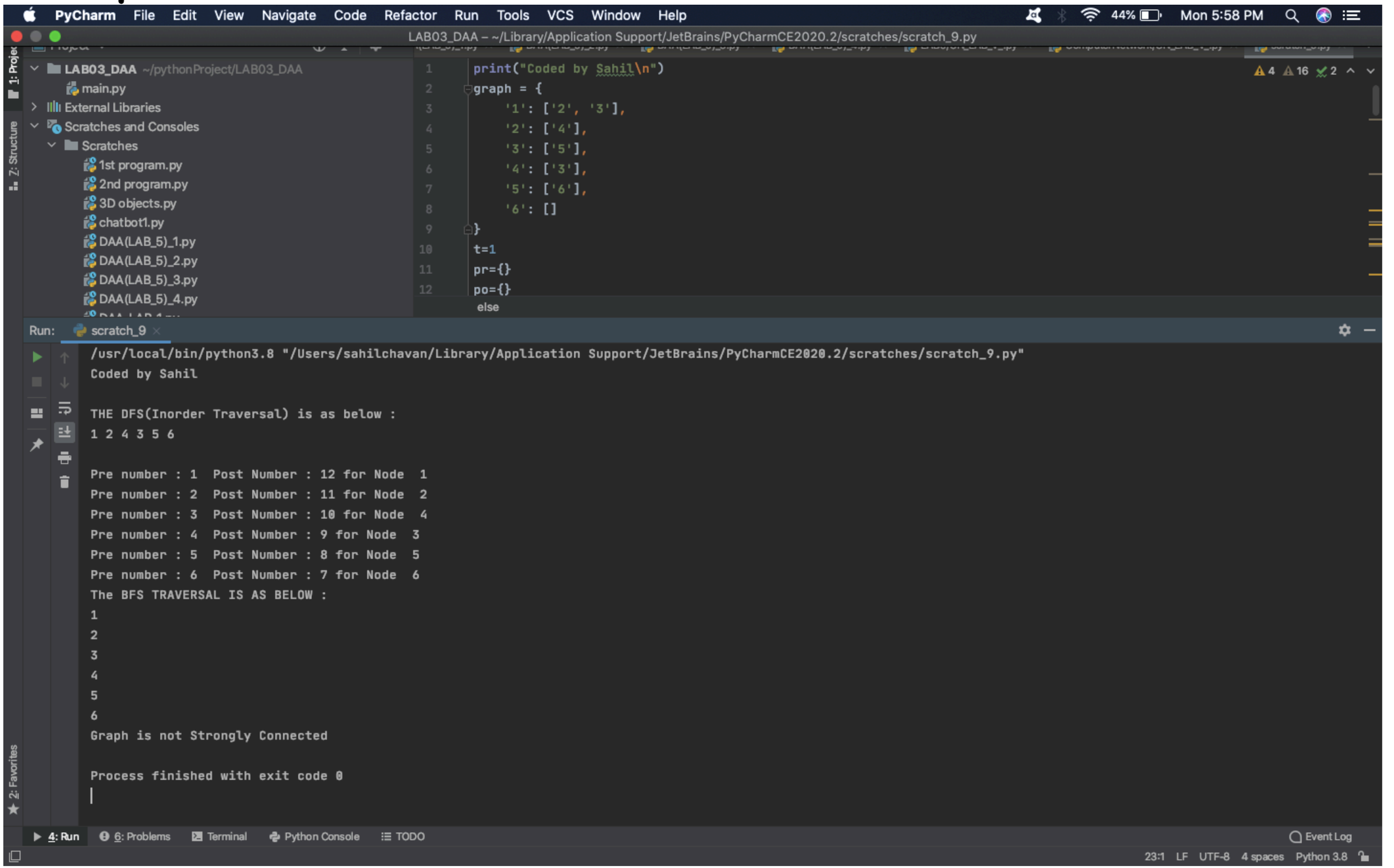
BFS

- It uses the data structure queue.
- BFS is complete because it finds the solution if one exists.
- BFS takes more space i.e. equivalent to o (b0) where b is the maximum breath exist in a search
- tree and d is the maximum depth exit in a search tree.
- In case of several goals, it finds the best one.

DFS

- It uses the data structure stack.
- It is not complete because it may take infinite loop to reach at the goal node.
- The space complexity is O (d).
- In case of several goals, it will terminate the solution in any order.

**Output :


PyCharm File Edit View Navigate Code Refactor Run Tools VCS Window Help
Mon 5:58 PM
LAB03_DAA – ~/Library/Application Support/JetBrains/PyCharmCE2020.2/scratches/scratch_9.py
LAB03_DAA ~/pythonProject/LAB03_DAA
main.py
External Libraries
Scratches and Consoles
Scratches
1st program.py
2nd program.py
3D objects.py
chatbot1.py
DAA(LAB_5)_1.py
DAA(LAB_5)_2.py
DAA(LAB_5)_3.py
DAA(LAB_5)_4.py
print("Coded by Sahil\n")
graph = {
'1': ['2', '3'],
'2': ['4'],
'3': ['5'],
'4': ['3'],
'5': ['6'],
'6': []
}
t=1
pr={}
po={}
else
Run: scratch_9
/usr/local/bin/python3.8 "/Users/sahilchavan/Library/Application Support/JetBrains/PyCharmCE2020.2/scratches/scratch_9.py"
Coded by Sahil
THE DFS(Inorder Traversal) is as below :
1 2 4 3 5 6
Pre number : 1 Post Number : 12 for Node 1
Pre number : 2 Post Number : 11 for Node 2
Pre number : 3 Post Number : 10 for Node 4
Pre number : 4 Post Number : 9 for Node 3
Pre number : 5 Post Number : 8 for Node 5
Pre number : 6 Post Number : 7 for Node 6
The BFS TRAVERSAL IS AS BELOW :
1
2
3
4
5
6
Graph is not Strongly Connected
Process finished with exit code 0
4: Run
6: Problems
Terminal
Python Console
TODO
Event Log
23:1 LF UTF-8 4 spaces Python 3.8

**Conclusion :

We Successfully implemented BFS & DFS for the given graph.

NAME : Sahil Chavan

PRN : 20190802042