Batch: IAI-2 **Experiment Number: 2**

Roll Number: 16010422234 **Name: Chandana Ramesh Galgali**

Aim of the Experiment: To implement BFS - Uninformed search algorithm in state space

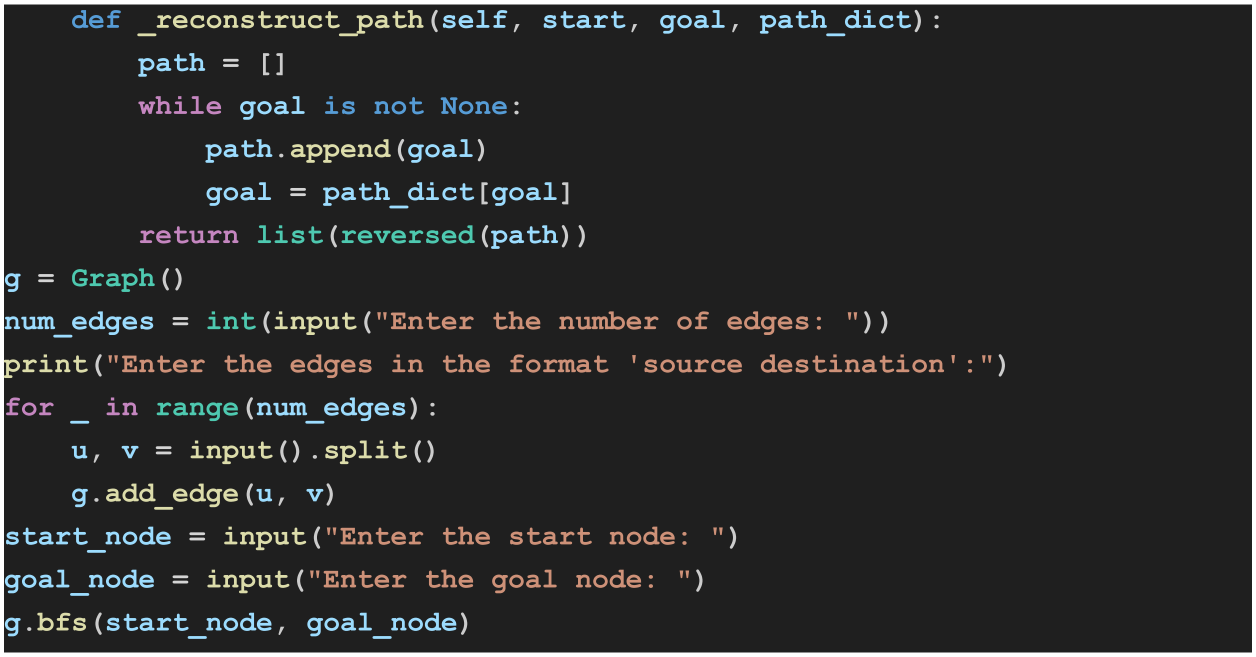
---

Program/Steps:

1. Implement the Mentioned algorithm for BFS for graph search.
2. Output must show the contents of Fringe and Visited nodes for each iteration of graph traversal. Also, finally it must print the path traversed.

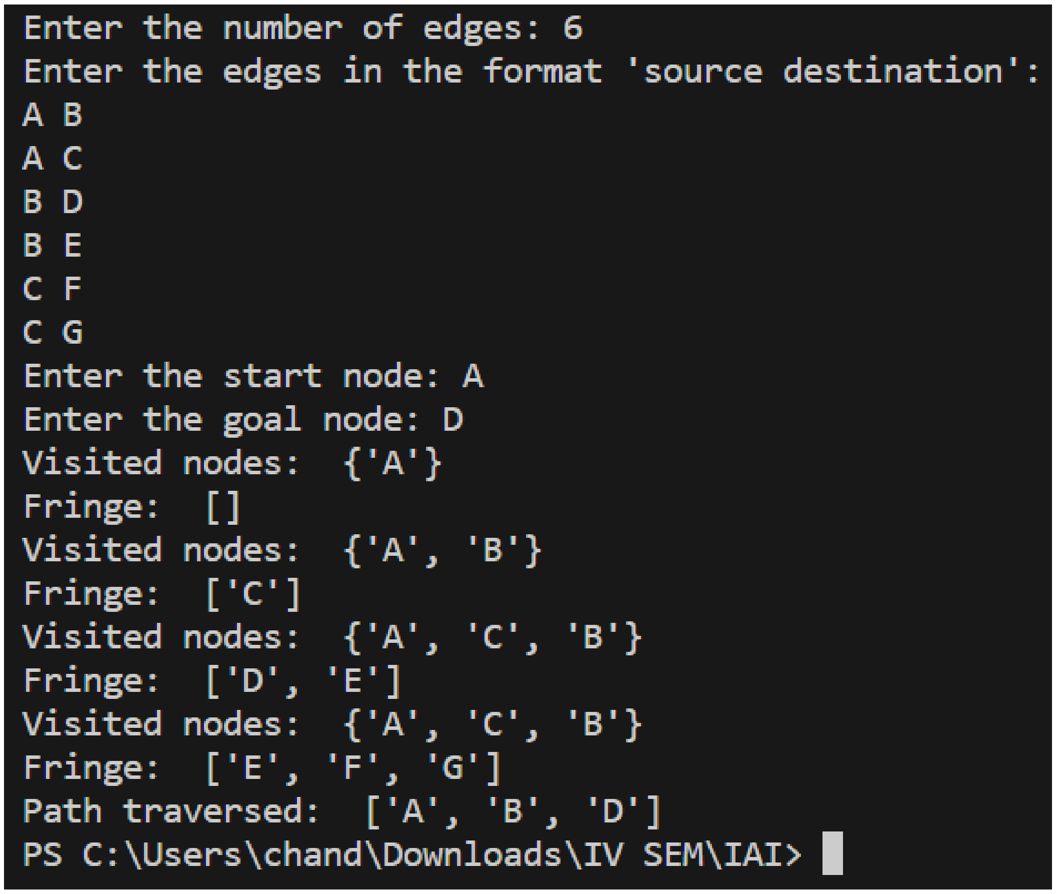
Code:

```python
from collections import defaultdict, deque
class Graph:
    def __init__(self):
        self.graph = defaultdict(list)
    def add_edge(self, u, v):
        self.graph[u].append(v)
    def bfs(self, start, goal):
        visited = set()
        queue = deque([start])
        path_dict = {start: None}
        while queue:
            node = queue.popleft()
            if node == goal:
                path = self._reconstruct_path(start, goal, path_dict)
                print("Visited nodes: ", visited)
                print("Fringe: ", [n for n in queue])
                print("Path traversed: ", path)
                return path
            if node not in visited:
                visited.add(node)
                print("Visited nodes: ", visited)
                print("Fringe: ", [n for n in queue])
                for neighbor in self.graph[node]:
                    if neighbor not in visited and neighbor not in queue:
                        queue.append(neighbor)
                        path_dict[neighbor] = node
        print("No path found!")
        return None
```



```python
    def _reconstruct_path(self, start, goal, path_dict):
        path = []
        while goal is not None:
            path.append(goal)
            goal = path_dict[goal]
        return list(reversed(path))
g = Graph()
num_edges = int(input("Enter the number of edges: "))
print("Enter the edges in the format 'source destination':")
for _ in range(num_edges):
    u, v = input().split()
    g.add_edge(u, v)
start_node = input("Enter the start node: ")
goal_node = input("Enter the goal node: ")
g.bfs(start_node, goal_node)
```

Output/Result:



```
Enter the number of edges: 6
Enter the edges in the format 'source destination':
A B
A C
B D
B E
C F
C G
Enter the start node: A
Enter the goal node: D
Visited nodes:  {'A'}
Fringe:  []
Visited nodes:  {'A', 'B'}
Fringe:  ['C']
Visited nodes:  {'A', 'C', 'B'}
Fringe:  ['D', 'E']
Visited nodes:  {'A', 'C', 'B'}
Fringe:  ['E', 'F', 'G']
Path traversed:  ['A', 'B', 'D']
PS C:\Users\chand\Downloads\IV SEM\IAI> 
```

Outcomes: Analyze and formalize the problem (as a state space, graph, etc.) and select the appropriate search method and write the algorithm

---

Conclusion (Based on the Results and outcomes achieved):

The implementation of BFS in the state space has proven to be effective in achieving the goal of systematically exploring and finding optimal solutions. Understanding the algorithm's strengths, such as completeness and optimality, as well as its limitations, including high time and memory complexity, is crucial for making informed decisions on its application in various problem domains.

References:

1. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Second Edition, Pearson Publication
2. Luger, George F. Artificial Intelligence : Structures and strategies for complex problem solving, 2009, 6th Edition, Pearson Education