# LAB #3:

## BREADTH FIRST SEARCH FOR GRAPH AND TREE TRAVERSAL

**Objectives:**

* To implement breadth first search (BFS) algorithm for graphand tree traversals using python

**Hardware/Software Required:**

Hardware: Desktop/ Notebook Computer

Software Tool: Python 2.7/ 3.6.2

**Introduction:**

Breadth first search (BFS) is an algorithm for traversing or searching tree or graph data structures. It starts at the tree root or some arbitrary node of a graph and explores the neighbor nodes first, before moving to the next level neighbors.

In this lab, we will implement BFS for both tree and graph traversals. Since BFS searches the sibling or neighboring nodes first before moving to the next child nodes so we will employ First In First Out (FIFO) structure in our implementation where siblings nodes will be added first before child nodes and since they are added first so they will be searched earlier as well.

**Lab Tasks:**

1. Implement a FIFO data structure in python.

2. Use the implemented FIFO data structure to implementBFS for Graph 1 and 2 in python. The starting node is ‘6’ for Graph 1 while the starting node is ‘C’ for Graph 2.

3. Implement BFS for Tree 1 and 2 in python. The starting node is ‘1’ for Tree 1 while the starting node is ‘Frankfurt’ for Tree 2.

|  |  |
| --- | --- |
| C:\Users\Taimur Hassan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\250px-6n-graf.svg.png  **Graph 1** | C:\Users\Taimur Hassan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\YA7NX.PNG  **Graph 2** |
| C:\Users\Taimur Hassan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\BFS tree.png  **Tree 1** | C:\Users\Taimur Hassan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\BFS2.png  **Tree 2** |

**LabEvaluation:**

**Q:**Decompose the following 3x3 3-bit grayscale image into an undirected graph using 8 connectivity and search all pixels using BFS algorithm starting with node ‘5’.Also tell the total time taken by your algorithm.

|  |  |  |
| --- | --- | --- |
| 150 | 2 | 5 |
| 80 | 145 | 45 |
| 74 | 102 | 165 |

**Bonus Question (Uniform Cost Search):**

**Q:** Decompose it the above image into an undirected graph where each pixel represents a node and the edge cost between adjacent nodes is computed by taking the absolute difference. Now use Uniform Cost Search (UCS) algorithm to traverse the graph starting with node ‘150’. UCS requires priority queues for its implementation and the pseudo-code is given below:

Insert the root into the queue

While the queue is not empty

Dequeue the maximum priority element from the queue

(If priorities are same, alphabetically smaller path is chosen)

If the path is ending in the goal state, print the path and exit

Else

Insert all the children of the dequeued element, with the cumulative costs as priority

***Hint: Use nested dictionaries to represent graph with edge costs***

**Conclusion:**

Write the conclusion about this lab

**NOTE:** A lab journal is expected to be submitted for this lab.