**Name: Muzamil Mushtaq**

**Roll no: SU92-BSAIM-F24-037**

**Section: BSAI-3A**

**Submitted to: Sir Rasikh Ali**

**BFS without queue and without node.**

**Introduction:**

This is simplified and procedural implementation of BFS. The graph is represented as adjacency list. Nodes are identified by string and numbers. A list is used instead of queue to proceed nodes level by level.

**How it actually works:**

It starts with the root node. And use a list instead of queue to track the current level of the node. It visits each node and add its neighbor to a new list. It keeps track of visited nodes using a separate list.

**Advantages:**

1. It is simple to understand and also in implementation.
2. There is no need of advanced data structure.
3. It is good for basic learning of BFS.

**Disadvantages:**

* Memory is not efficient for large graphs.
* There is no clear separation of data and logics
* It is hard to maintain for complex features
* Its is not able to store extra information like parents , depth etc.

**BFS with queue and nodes.**

**Introduction:**

This is standard object oriented BFS implementation. Each node is represented by a class that contains its value. list of neighbor, and a visited flag. A queue is used to manage the traversal order.

**How it actually works:**

Each node is an object. a queue is used to explore the nodes in the correct order. The neighbor of each visited node are added to the queue. a visited node prevents revisiting nodes.

**Advantages:**

* It is efficient for reliable and large graphs.
* It is easy to extend, modify and reuse.
* It can store additional information like parents, depth etc.
* It is better for real world applications.

**Disadvantages:**

* It is slightly more complex for implementation.
* Requires the understanding of OOP.
* More code and memory needed due to nodes.

**Comparison:**

**Programming:**

**In without queue:** we need to study procedural programming.

**In queue:** we need to study OOP.

**Node representation:**

**In without queue**: we use simple value**.**

**In queue:** we use node objects**.**

**Data handling:**

**In without queue:** we use basic tracking method.

In queue: we use structural OOP.

**Best use case:**

**In without queue:** There are simple graphs and learning.

**In queue**: helpful for large graphs and real time applications.

**Thanks for viewing my work…….**