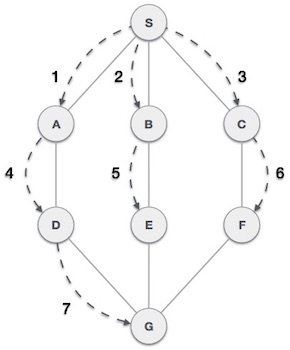
Breadth First Search (BFS) algorithm traverses a graph in a breadthward motion and uses a queue to remember to get the next vertex to start a search, when a dead end occurs in any iteration.



As in the example given above, BFS algorithm traverses from A to B to E to F first then to C and G lastly to D. It employs the following rules.

* **Rule 1** − Visit the adjacent unvisited vertex. Mark it as visited. Display it. Insert it in a queue.
* **Rule 2** − If no adjacent vertex is found, remove the first vertex from the queue.
* **Rule 3** − Repeat Rule 1 and Rule 2 until the queue is empty.

|  |  |  |
| --- | --- | --- |
| **Step** | **Traversal** | **Description** |
| 1. | Breadth First Search Step One | Initialize the queue. |
| 2. | Breadth First Search Step Two | We start from visiting **S** (starting node), and mark it as visited. |
| 3. | Breadth First Search Step Three | We then see an unvisited adjacent node from **S**. In this example, we have three nodes but alphabetically we choose **A**, mark it as visited and enqueue it. |
| 4. | Breadth First Search Step Four | Next, the unvisited adjacent node from **S** is **B**. We mark it as visited and enqueue it. |
| 5. | Breadth First Search Step Five | Next, the unvisited adjacent node from **S** is **C**. We mark it as visited and enqueue it. |
| 6. | Breadth First Search Step Six | Now, **S** is left with no unvisited adjacent nodes. So, we dequeue and find **A**. |
| 7. | Breadth First Search Step Seven | From **A** we have **D** as unvisited adjacent node. We mark it as visited and enqueue it. |

At this stage, we are left with no unmarked (unvisited) nodes. But as per the algorithm we keep on dequeuing in order to get all unvisited nodes. When the queue gets emptied, the program is over