BFS

Main Idea

- traverse the graph in a "ripple" like pattern, from a source vertex, traverse OUTWARDS.
- For each vertex being visited, mark it as visited (and do something else). Then, enqueue all its neighbors. Repeat this process until the queue is empty

Pseudo-code

- Requires a **queue**
- Although this looks awfully similar to iterative DFS, it requires a **queue**, which results in traversal difference.

```
// bool visited[vertex] already initialized for all vertices in G
function BFS(vertex src) {
    Queue Q = new();
    Q.enqueue(src);
    while !Q.empty() {
        vertex u = Q.dequeue();
        if !visited[u] {
            visited[u] = true;
            for each v in u.adj_list() {
                Q.enqueue(v);
            }
        }
    }
}
```