Breadth first search

In this tutorial, you will learn about breadth first search algorithm. Also, you will find working examples of bfs algorithm in C, C++, Java and Python.

Traversal means visiting all the nodes of a graph. Breadth First Traversal or Breadth First Search is a recursive algorithm for searching all the vertices of a graph or tree data structure.

BFS algorithm

A standard BFS implementation puts each vertex of the graph into one of two categories:

1. Visited
2. Not Visited

The purpose of the algorithm is to mark each vertex as visited while avoiding cycles.

The algorithm works as follows:

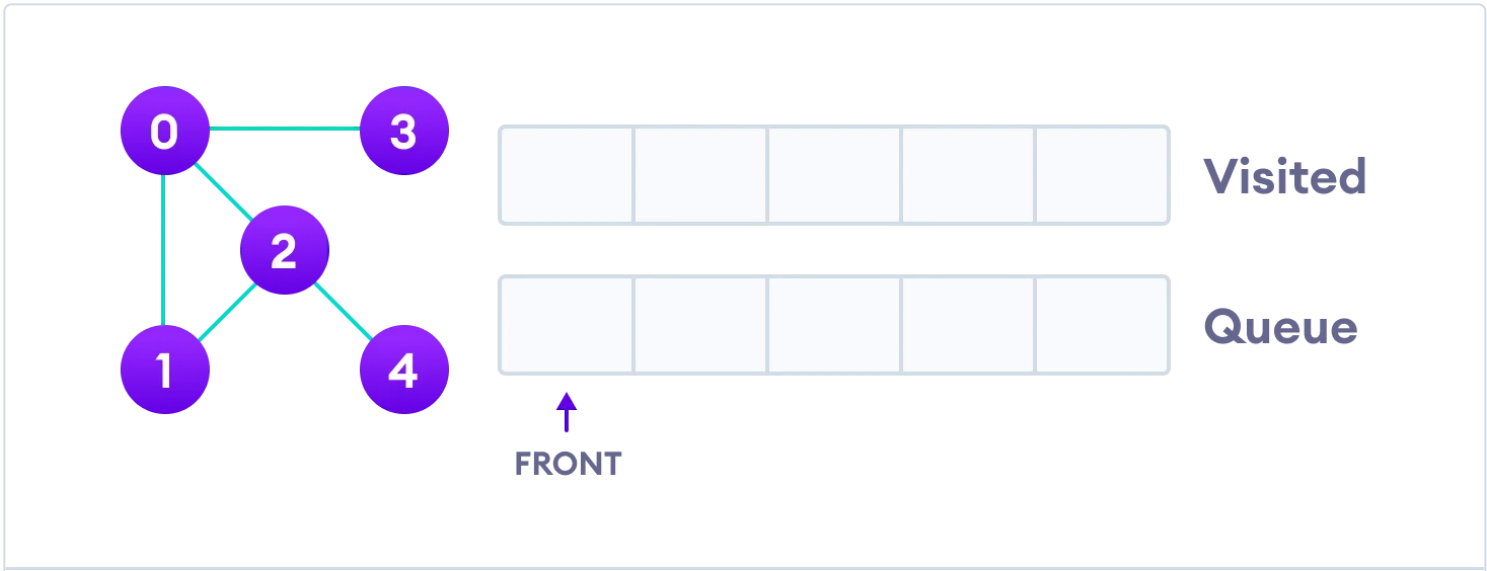
1. Start by putting any one of the graph's vertices at the back of a queue.
2. Take the front item of the queue and add it to the visited list.
3. Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the back of the queue.
4. Keep repeating steps 2 and 3 until the queue is empty.

The graph might have two different disconnected parts so to make sure that we cover every vertex, we can also run the BFS algorithm on every node

---

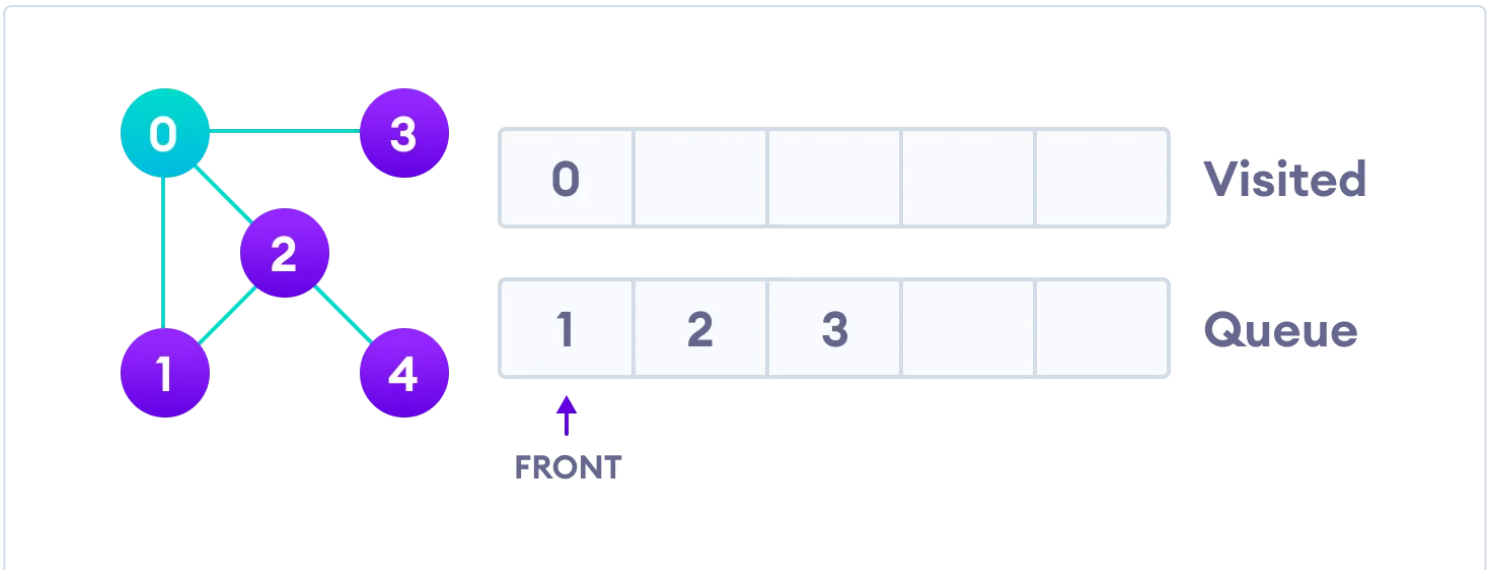
BFS example

Let's see how the Breadth First Search algorithm works with an example. We use an undirected graph with 5 vertices.



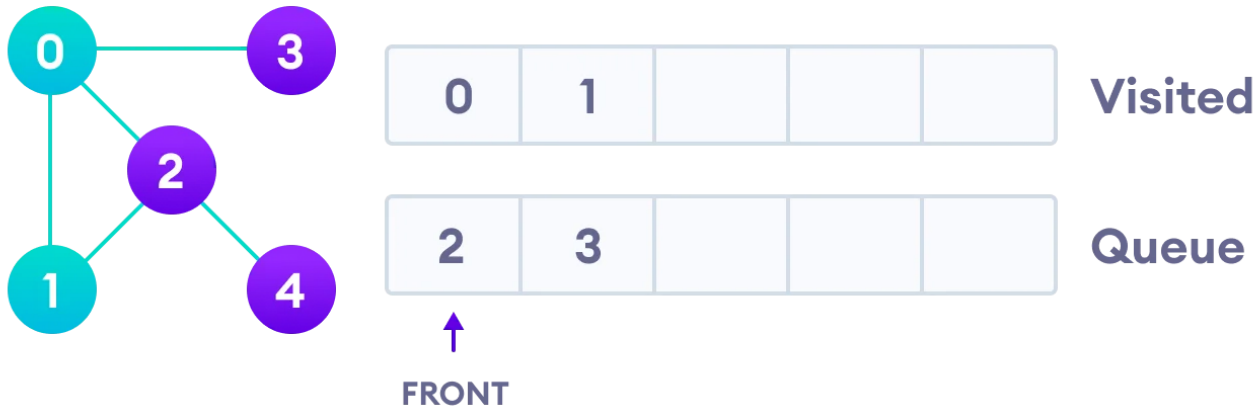
Undirected graph with 5 vertices

We start from vertex 0, the BFS algorithm starts by putting it in the Visited list and putting all its adjacent vertices in the stack.



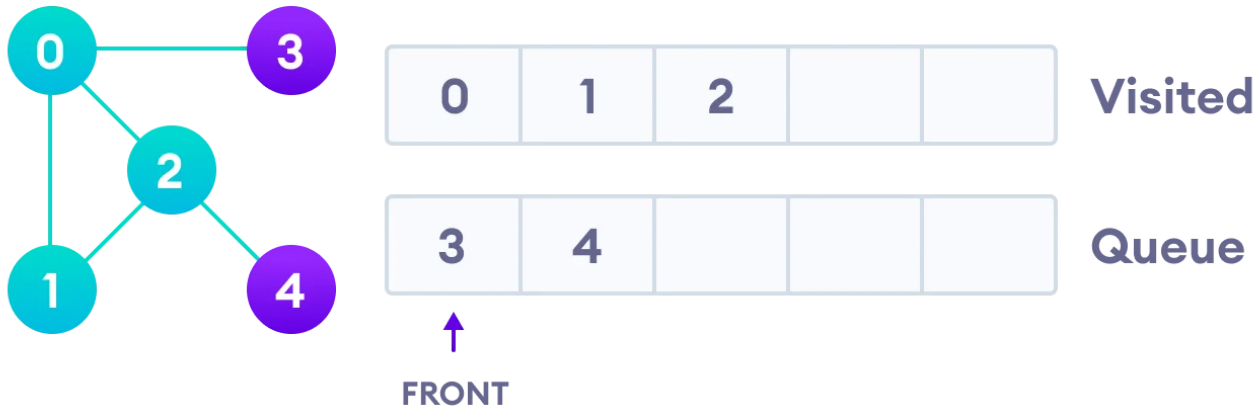
Visit start vertex and add its adjacent vertices to queue

Next, we visit the element at the front of queue i.e. 1 and go to its adjacent nodes. Since 0 has already been visited, we visit 2 instead.



Visit the first neighbour of start node 0, which is 1

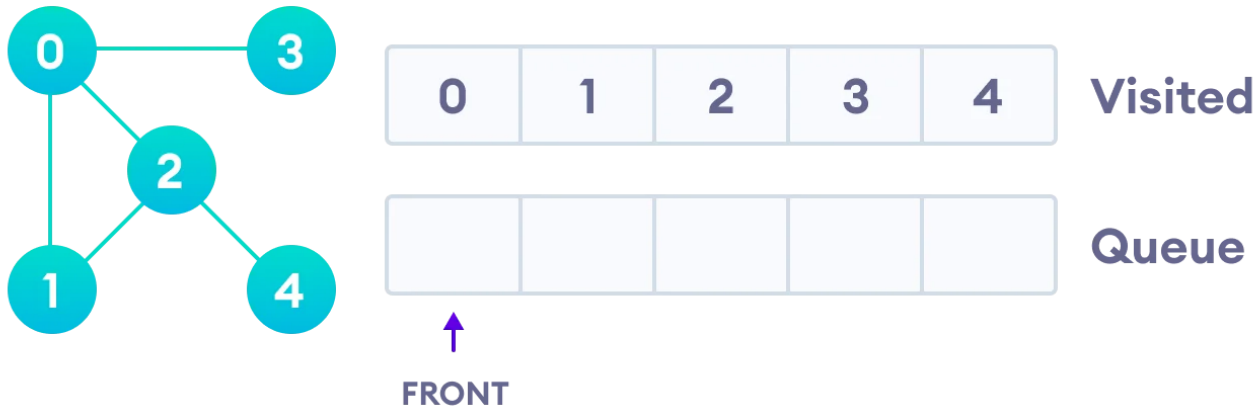
Vertex 2 has an unvisited adjacent vertex in 4, so we add that to the back of the queue and visit 3, which is at the front of the queue.



Visit 2 which was added to queue earlier to add its neighbours

4 remains in the queue

Only 4 remains in the queue since the only adjacent node of 3 i.e. 0 is already visited. We visit it.



Visit last remaining item in the queue to check if it has unvisited neighbors

Since the queue is empty, we have completed the Breadth First Traversal of the graph.

BFS pseudocode

```
create a queue Q
mark v as visited and put v into Q
while Q is non-empty
    remove the head u of Q
    mark and enqueue all (unvisited) neighbours of u
```

Python, Java and C/C++ Examples

The code for the Breadth First Search Algorithm with an example is shown below. The code has been simplified so that we can focus on the algorithm rather than other details.

Python Java C C++

```c
// BFS algorithm in C

#include <stdio.h>
#include <stdlib.h>
#define SIZE 40

struct queue {
  int items[SIZE];
  int front;
  int rear;
};

struct queue* createQueue();
void enqueue(struct queue* q, int);
int dequeue(struct queue* q);
void display(struct queue* q);
int isEmpty(struct queue* q);
void printQueue(struct queue* q);

struct node {
  int vertex;
  struct node* next;
};

struct node* createNode(int);

struct Graph {
  int numVertices;
```

BFS Algorithm Complexity

The time complexity of the BFS algorithm is represented in the form of `O(V + E)`, where `V` is the number of nodes and E is the number of edges.

The space complexity of the algorithm is `O(V)`.

BFS Algorithm Applications

1. To build index by search index
2. For GPS navigation
3. Path finding algorithms

4. In Ford-Fulkerson algorithm to find maximum flow in a network
5. Cycle detection in an undirected graph
6. In minimum spanning tree