Queues

William Fiset

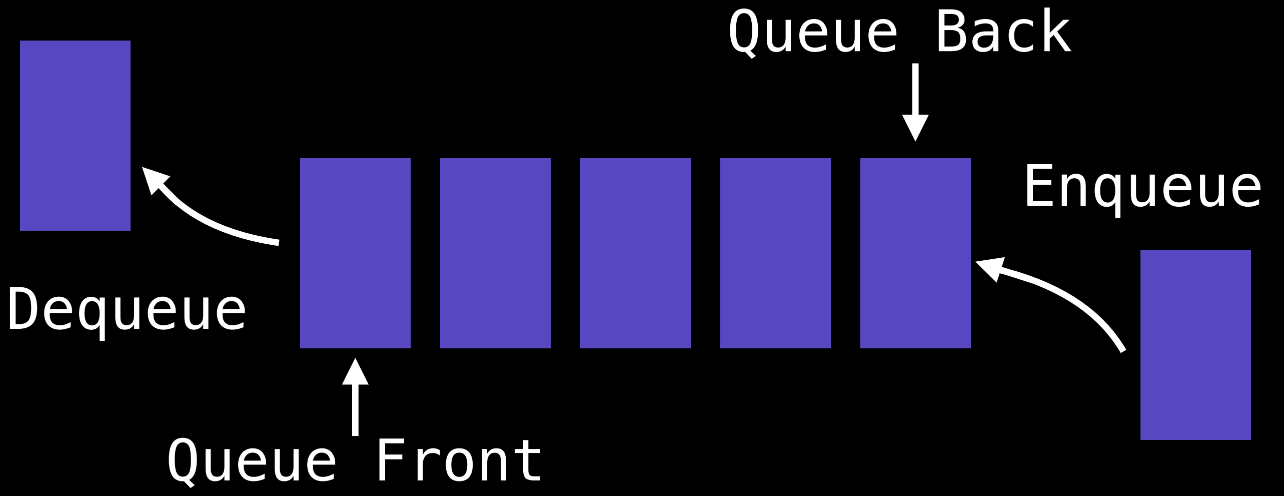
Outline

- Discussion About Queues
  - What is a queue?
  - Terminology
  - When and where is a queue used?
  - Complexity Analysis
  - Queue Breadth First Search (BFS) example
- Implementation Details
  - How to enqueue (add) elements to a queue
  - How to dequeue (remove) elements from a queue
- Code Implementation

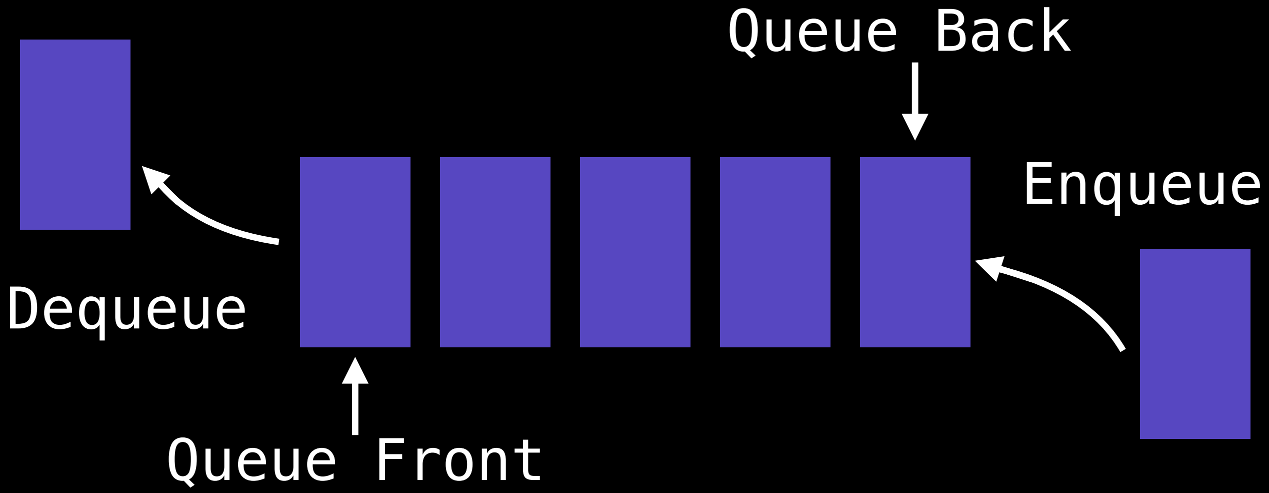
Discussion

What is a Queue?

A queue is a linear data structure which models real world queues by having two primary operations, namely **enqueue** and **dequeue**.



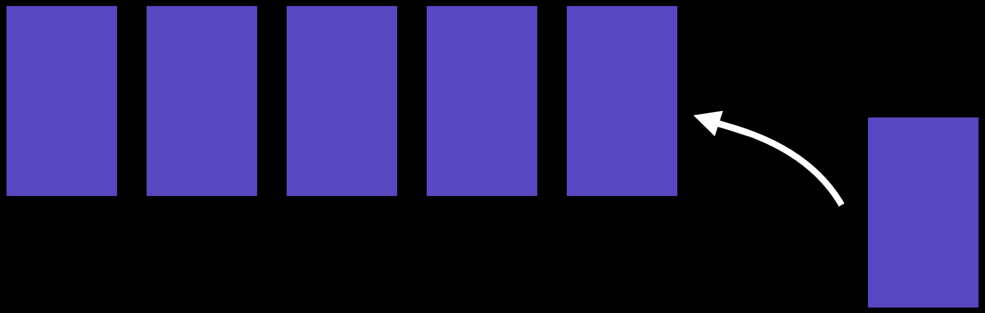
Queue Terminology



Queue Terminology

There does not seem to be consistent terminology for inserting and removing elements from queues.

Enqueue = Adding = Offering



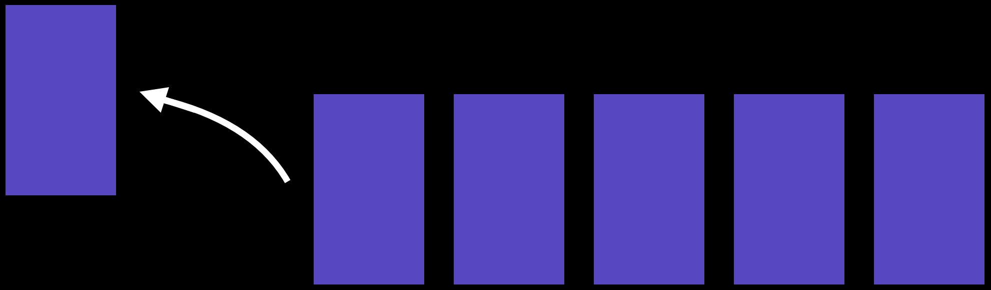
Queue Terminology

There does not seem to be consistent terminology for inserting and removing elements from queues.

Dequeue = Polling

(These are also sometimes called *removing*, but I find this ambiguous)

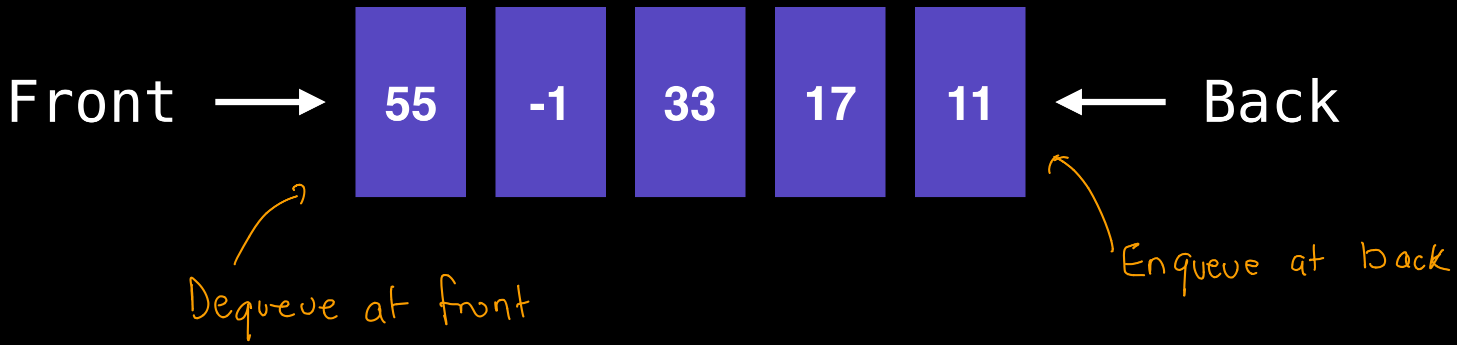
(front or back?)



Queue Example

Instructions:

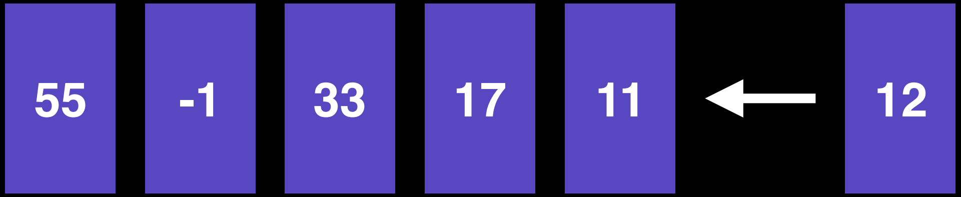
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

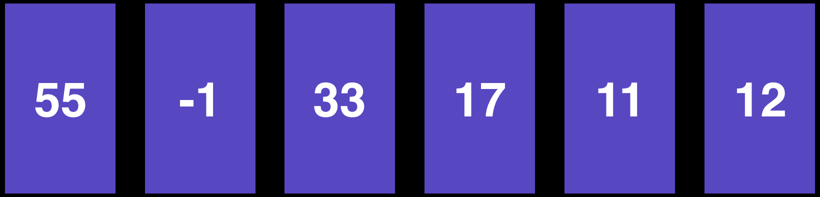
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

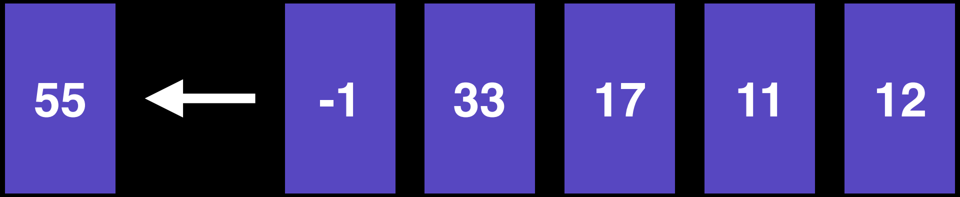
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

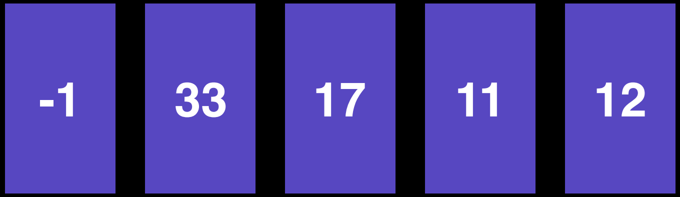
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(−6)
```



Queue Example

Instructions:

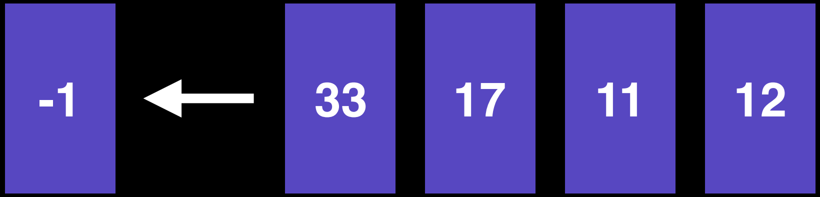
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

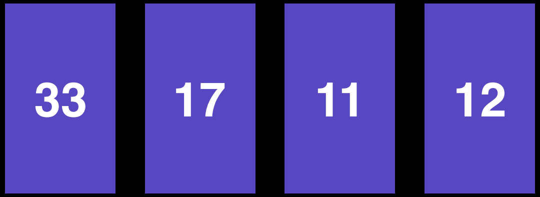
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

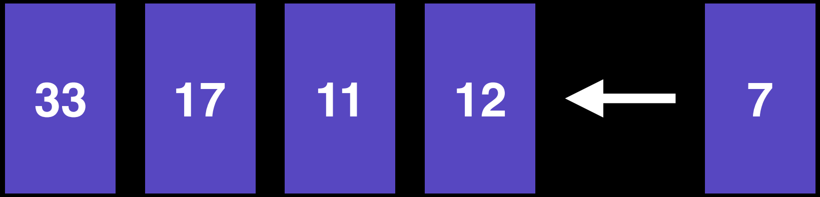
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

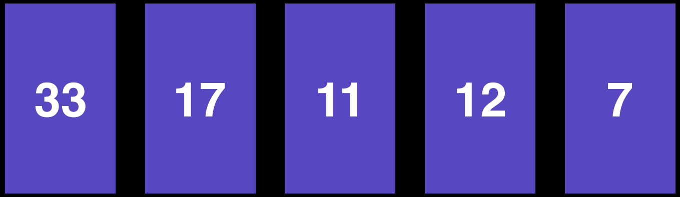
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

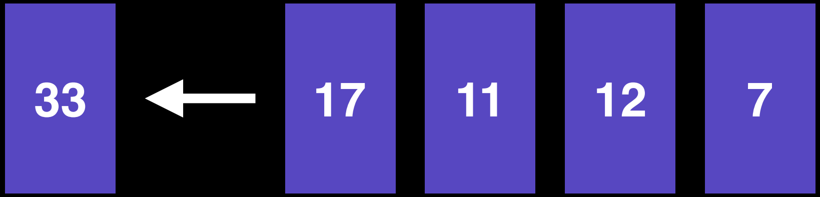
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

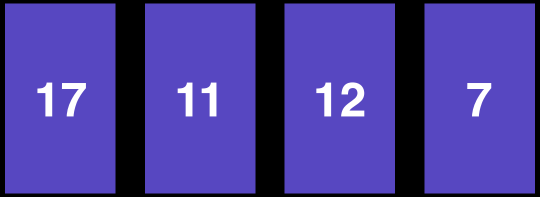
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

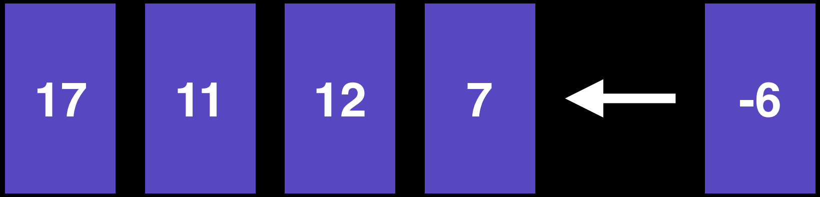
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

Instructions:

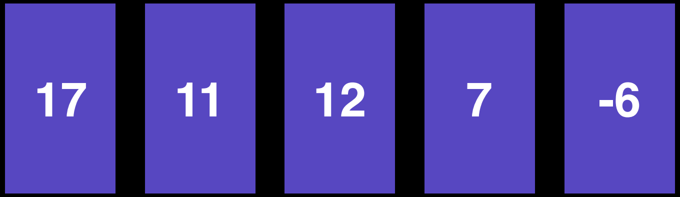
```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



Queue Example

<u>Instructions</u>:

```
Enqueue(12)
Dequeue()
Dequeue()
Enqueue(7)
Dequeue()
Enqueue(-6)
```



When and where is a Queue used?

- Any waiting line models a queue, for example a lineup at a movie theatre.
- Can be used to efficiently keep track of the ***x*** most recently added elements.
- Web server request management where you want first come first serve.
- Breadth first search (BFS) graph traversal.

Complexity Analysis

Complexity

Enqueue	O(1)
Dequeue	O(1)
Peeking Value at the front of the queue, without removing it	O(1)
Contains	O(n) Potentially have to scan all of the elements
Removal	O(n) //
Is Empty	O(1)

Enqueuing & Dequeuing

Enqueuing

Instructions:

```
Enqueue(5)
Enqueue(1)
Enqueue(6)
Enqueue(17)
Enqueue(8)
```

Can implement the queue abstract data type in mutiple ways,

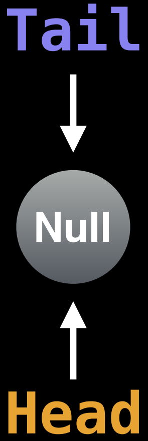
↳ Most popular are : arrays, singly/doubly linked lists

Enqueuing

Instructions:

```
Enqueue(5)
Enqueue(1)
Enqueue(6)
Enqueue(17)
Enqueue(8)
```

Initially both null

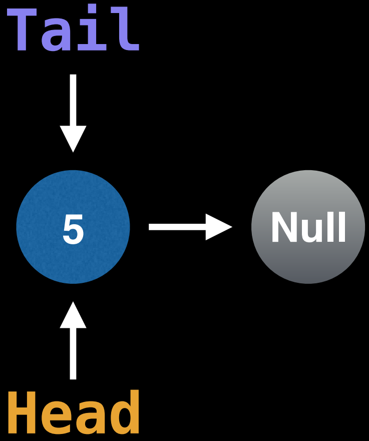


Enqueuing

Instructions:

Enqueue(5)
Enqueue(1)
Enqueue(6)
Enqueue(17)
Enqueue(8)

Both point at node if of size 1

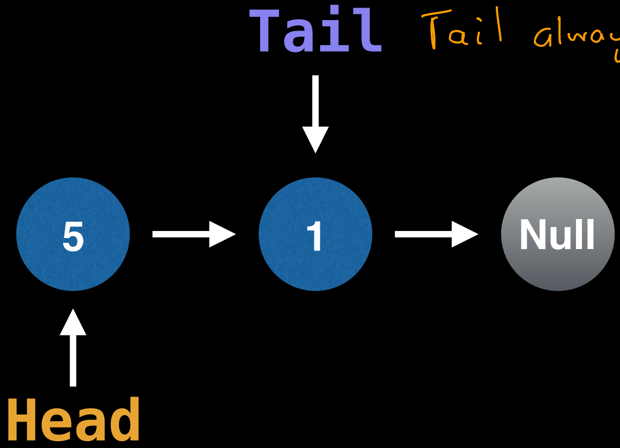


Enqueuing

Instructions:

```
Enqueue(5)
Enqueue(1)
Enqueue(6)
Enqueue(17)
Enqueue(8)
```

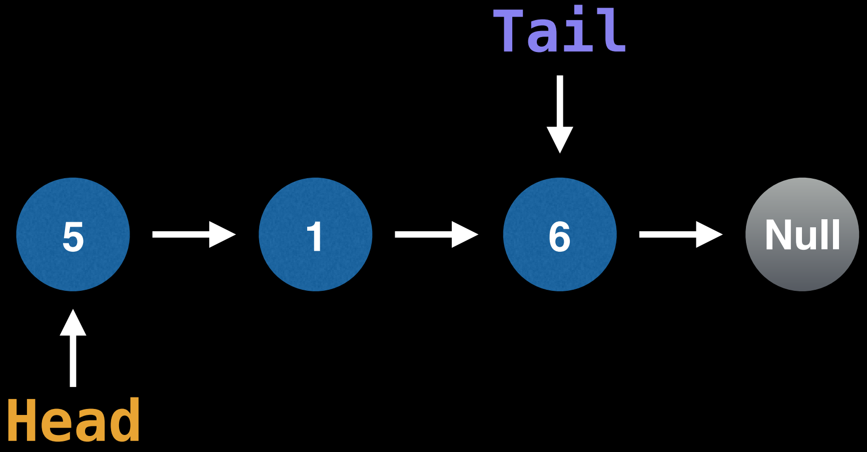
Tail always points to the newest node



Enqueuing

<u>Instructions</u>:

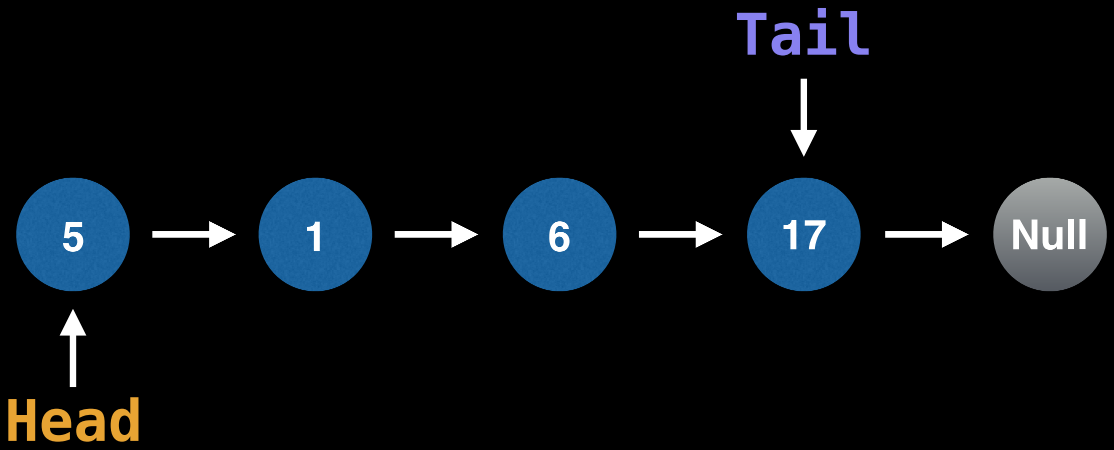
```
Enqueue(5)
Enqueue(1)
Enqueue(6)
Enqueue(17)
Enqueue(8)
```



Enqueuing

Instructions:

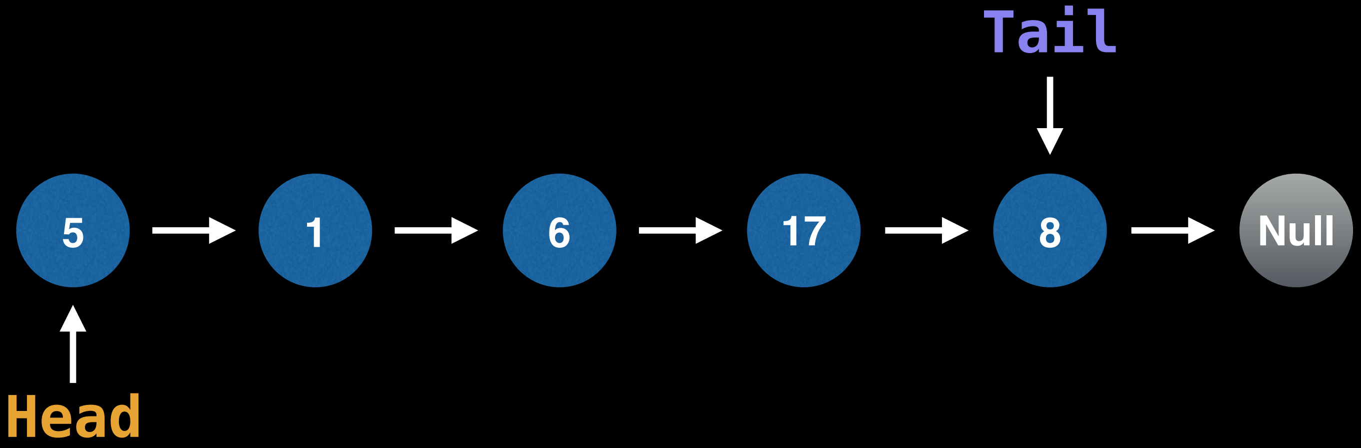
```
Enqueue(5)
Enqueue(1)
Enqueue(6)
Enqueue(17)
Enqueue(8)
```



Enqueuing

Instructions:

```
Enqueue(5)
Enqueue(1)
Enqueue(6)
Enqueue(17)
Enqueue(8)
```

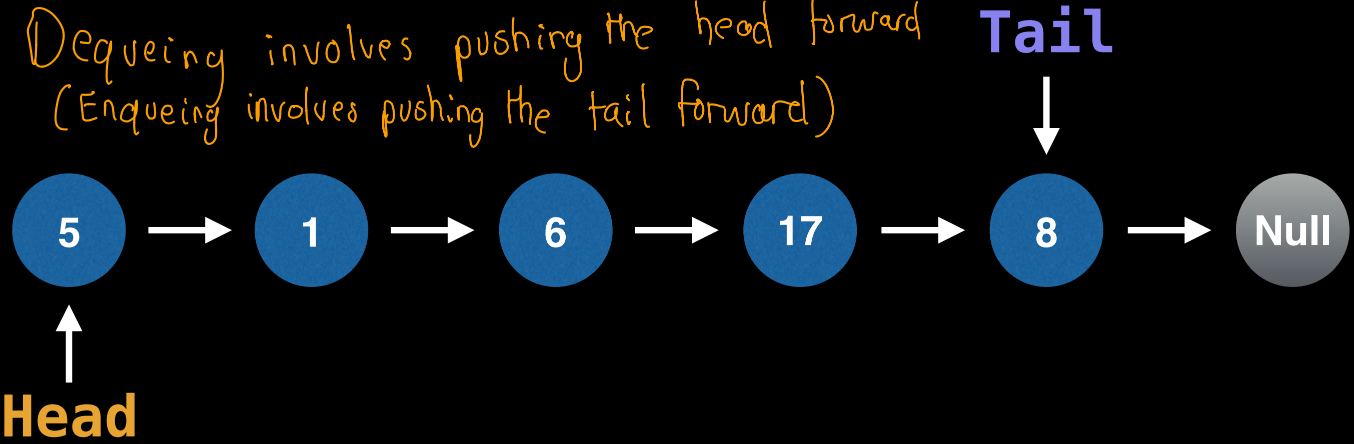


Dequeuing

Instructions:

```
Dequeue()
Dequeue()
Dequeue()
Dequeue()
Dequeue()
```

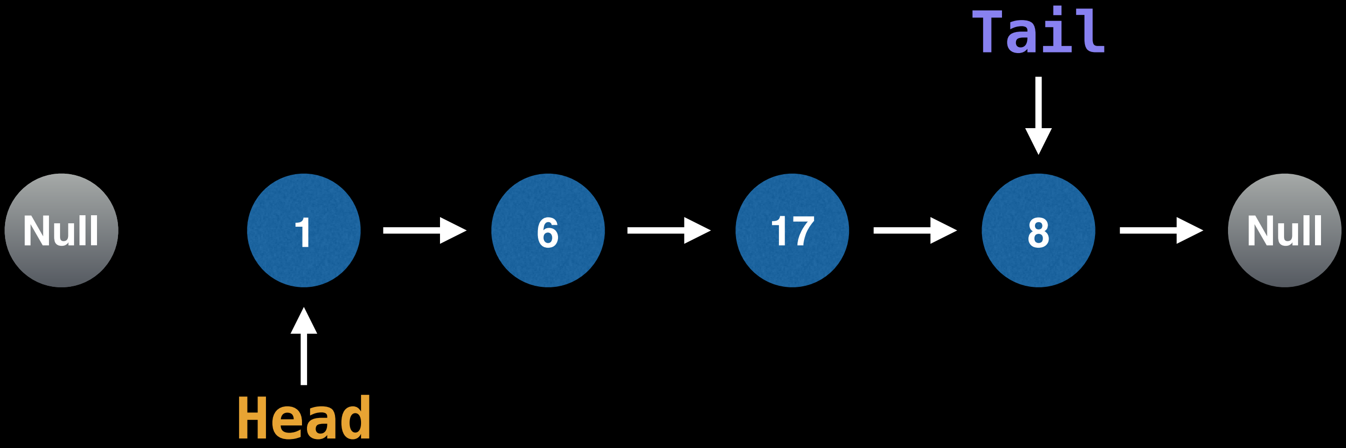
Dequeing involves pushing the head forward
(Enqueing involves pushing the tail forward)



Dequeuing

Instructions:

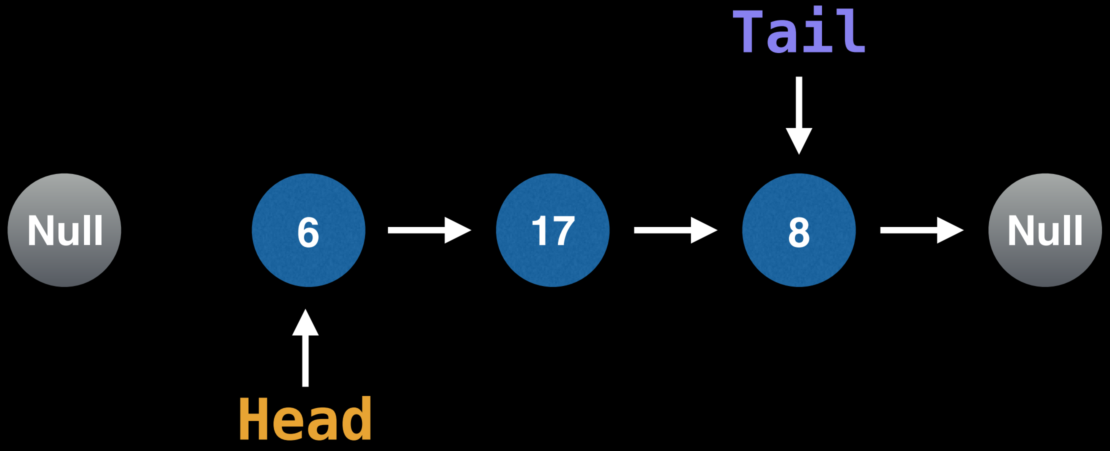
```
Dequeue()
Dequeue()
Dequeue()
Dequeue()
Dequeue()
```



Dequeuing

Instructions:

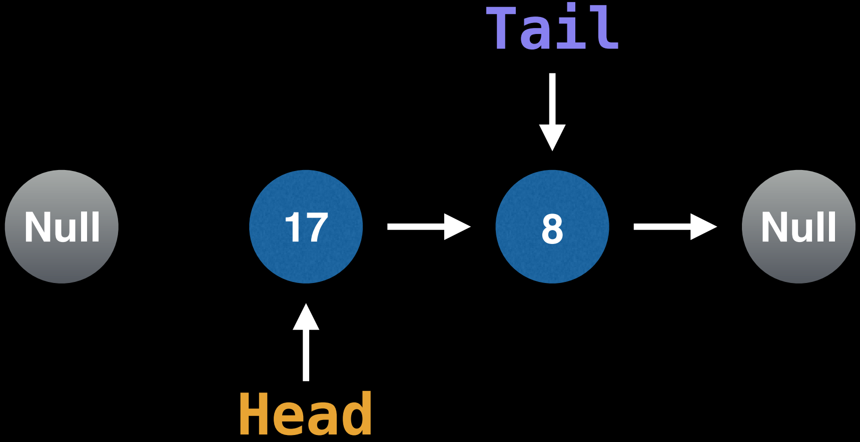
```
Dequeue()
Dequeue()
Dequeue()
Dequeue()
Dequeue()
```



Dequeuing

Instructions:

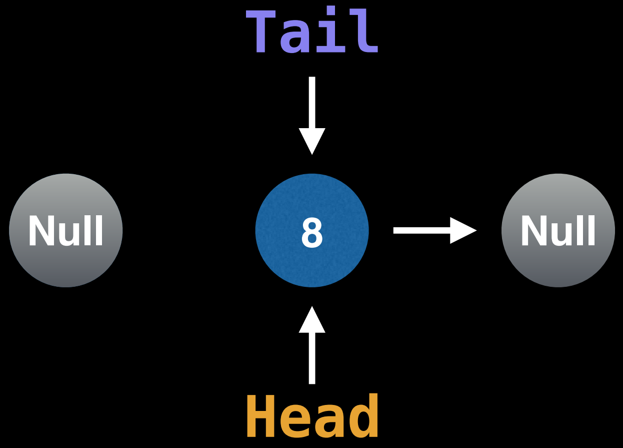
```
Dequeue()
Dequeue()
Dequeue()
Dequeue()
Dequeue()
```



Dequeuing

Instructions:

```
Dequeue()
Dequeue()
Dequeue()
Dequeue()
Dequeue()
```

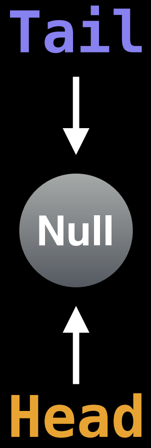


Dequeuing

Instructions:

```
Dequeue()
Dequeue()
Dequeue()
Dequeue()
Dequeue()
```

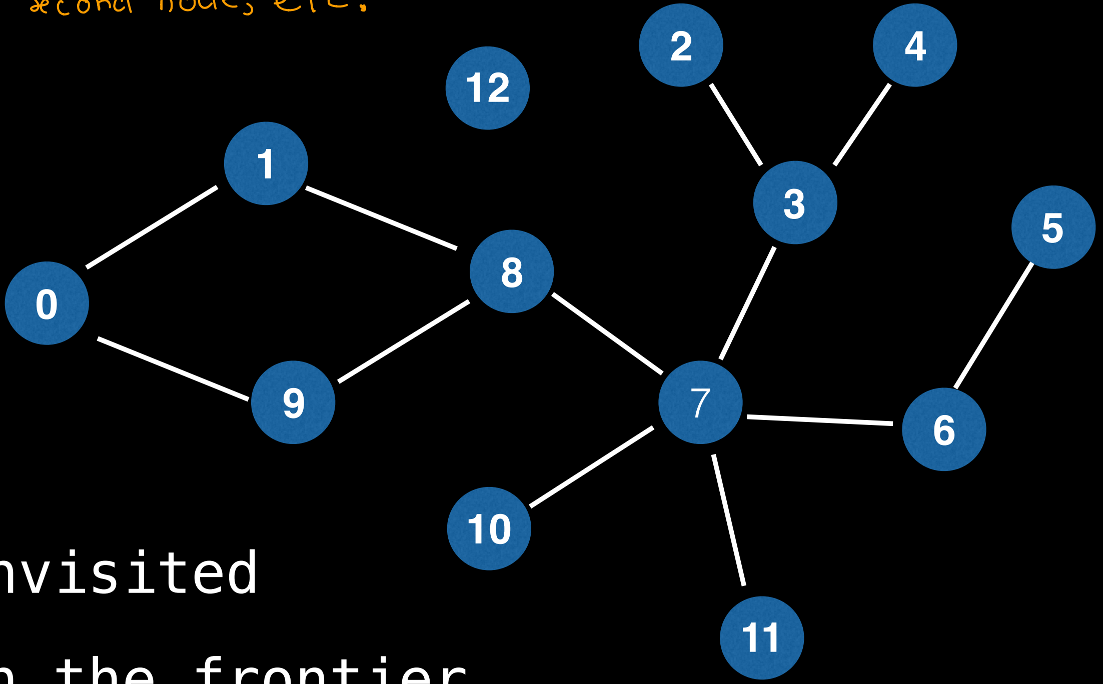
If no elements, head and tail point to null



Queue Example – BFS

BFS: Objective is to start at a node and traverse the entire graph

First visit the neighbours of all the starting node, the neighbours of the second node, etc.

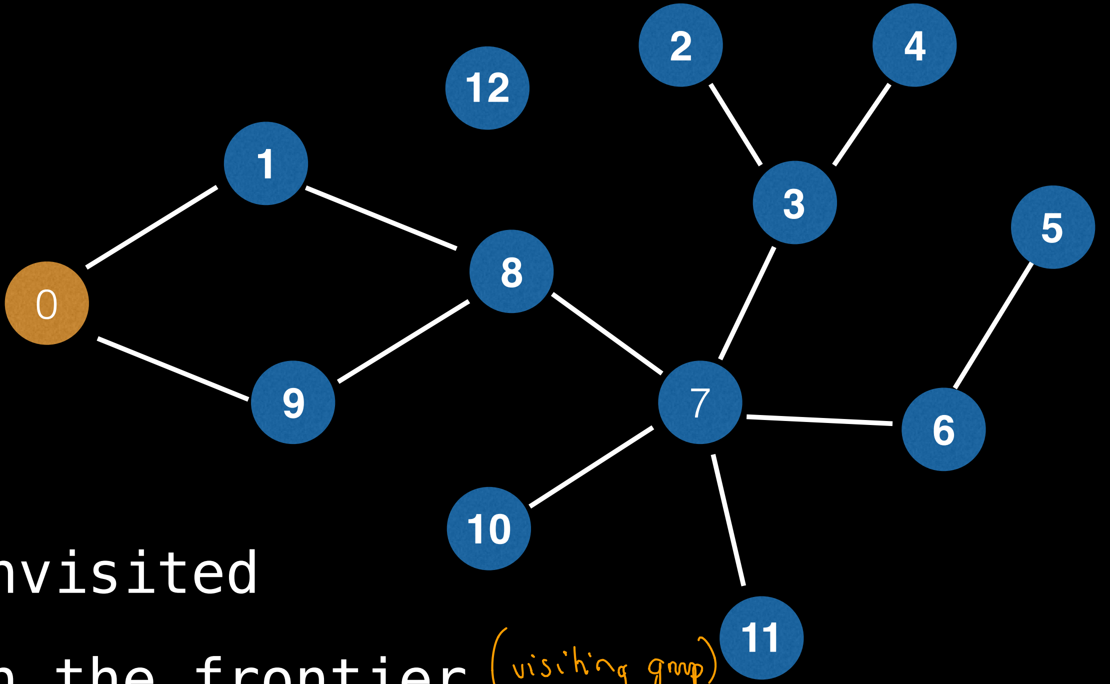


🔵 Unvisited

🟠 On the frontier

🟢 Visited

Queue Example – BFS

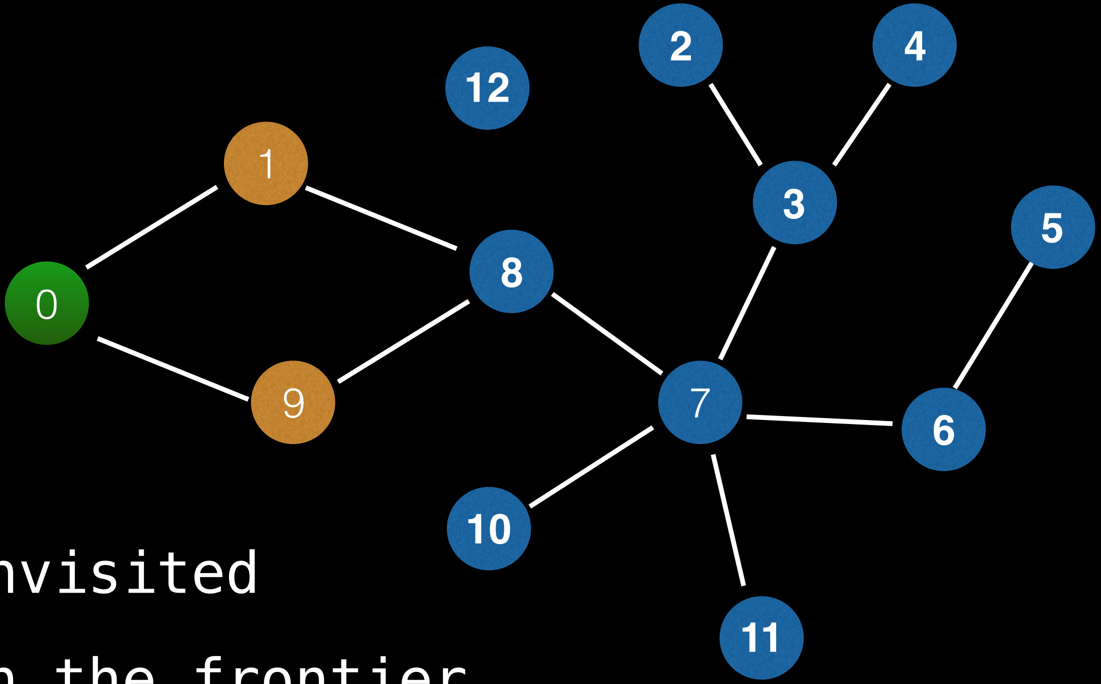


● Unvisited

● On the frontier (visiting group)

● Visited

Queue Example - BFS

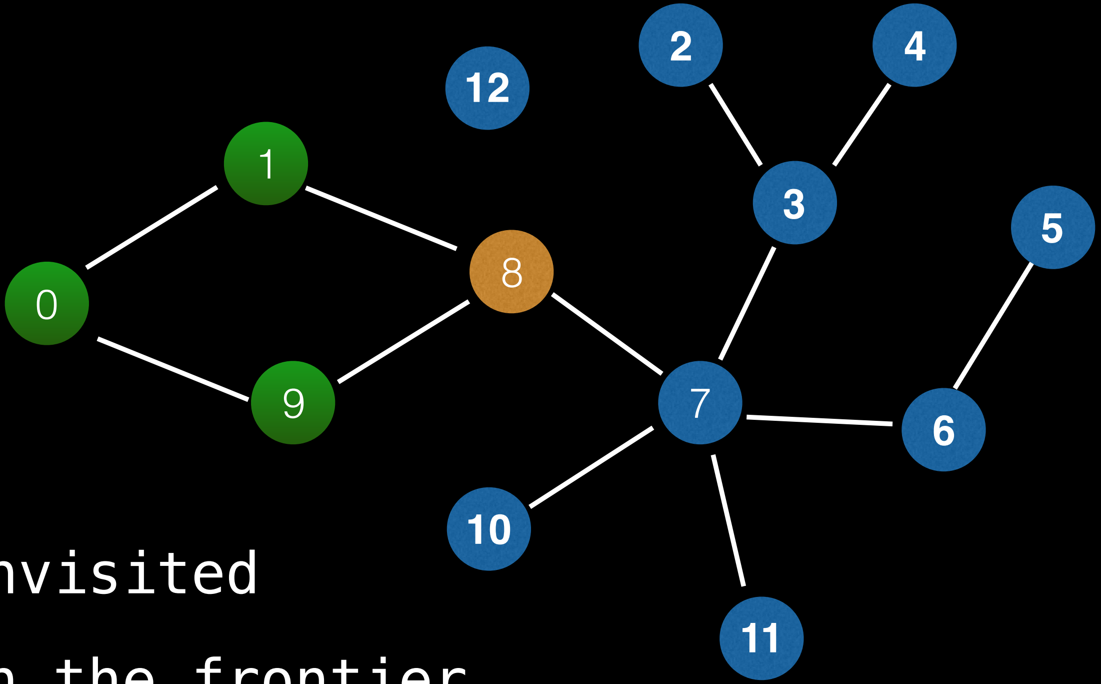


🟦 Unvisited

🟧 On the frontier

🟩 Visited

Queue Example – BFS

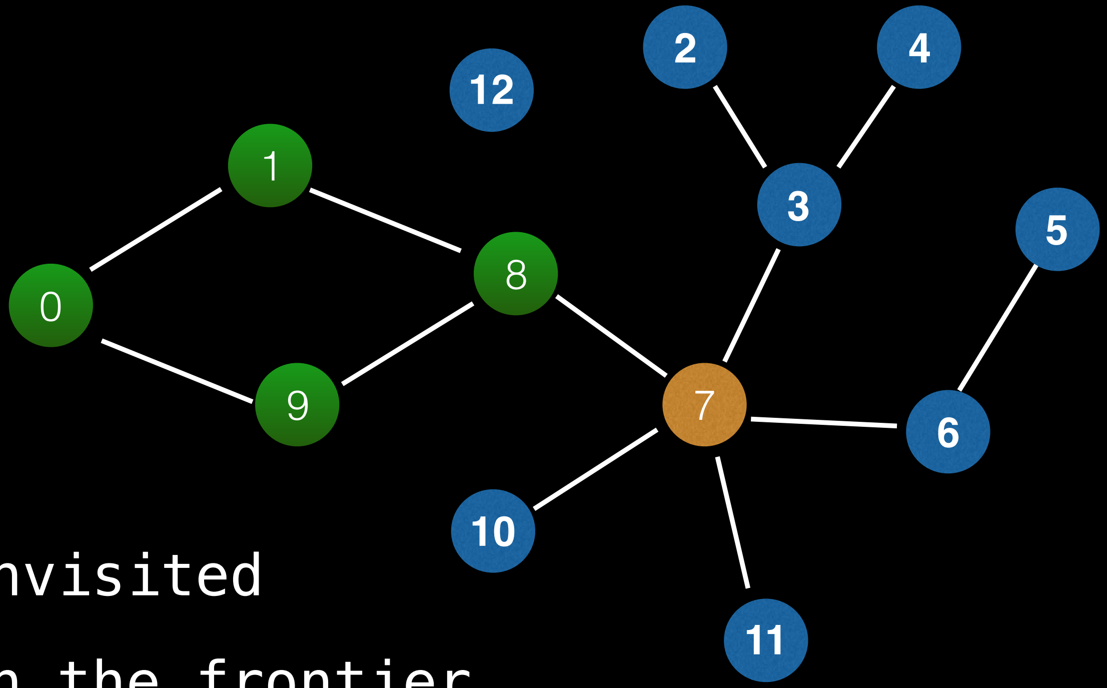


🟦 Unvisited

🟧 On the frontier

🟩 Visited

Queue Example - BFS

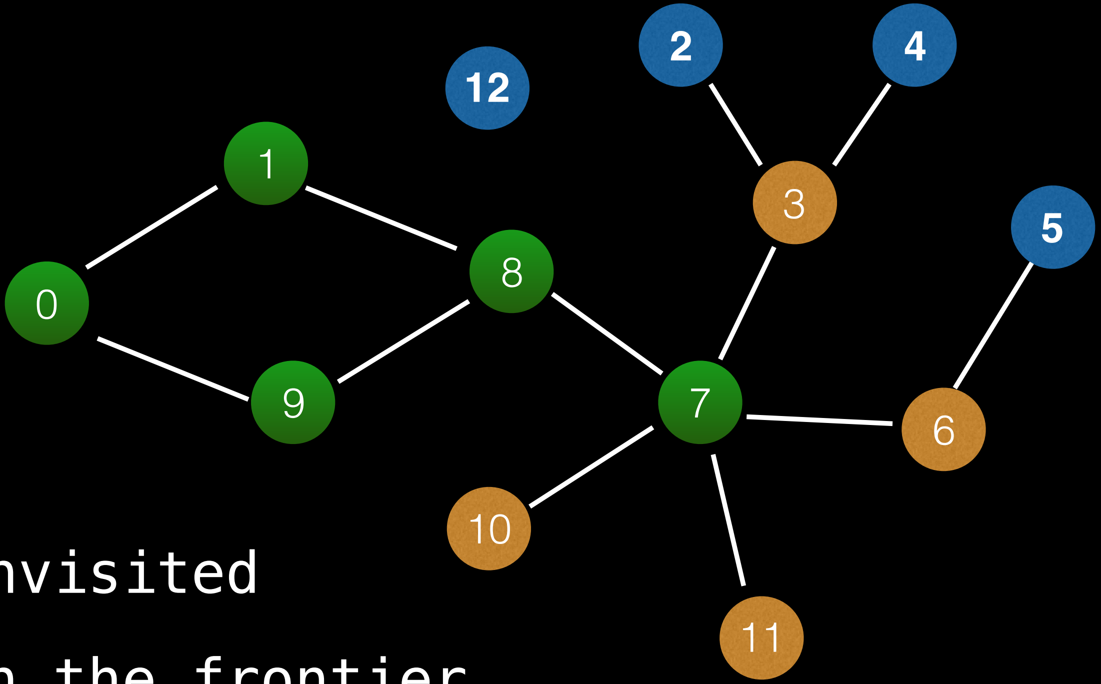


🟦 Unvisited

🟧 On the frontier

🟩 Visited

Queue Example – BFS

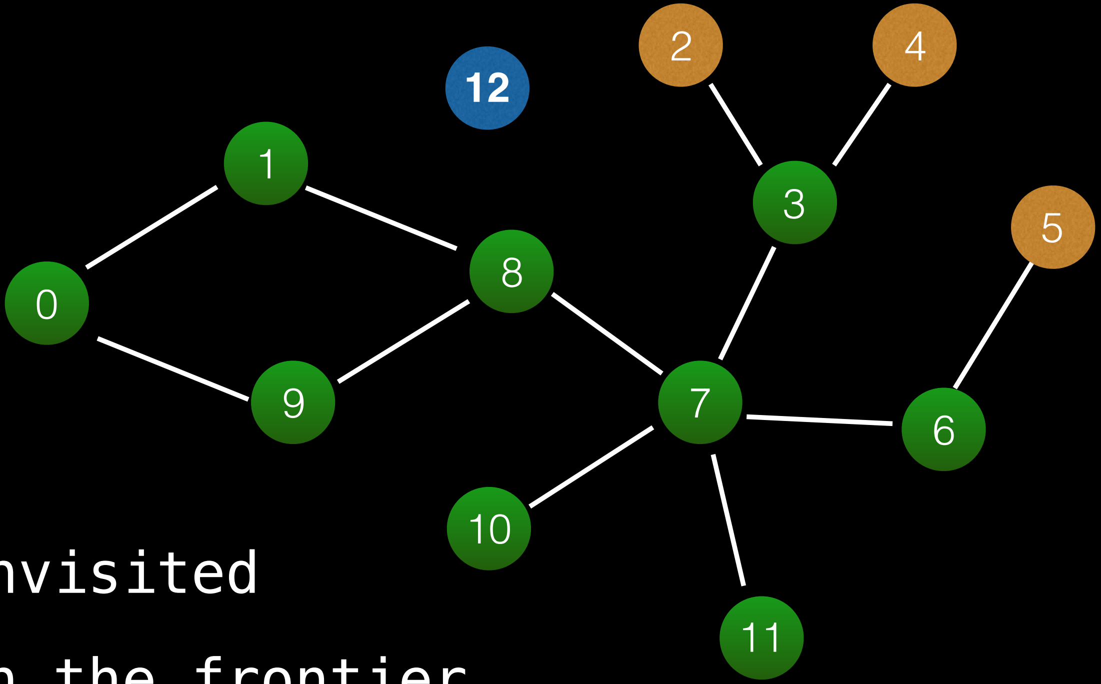


🟦 Unvisited

🟧 On the frontier

🟩 Visited

Queue Example - BFS

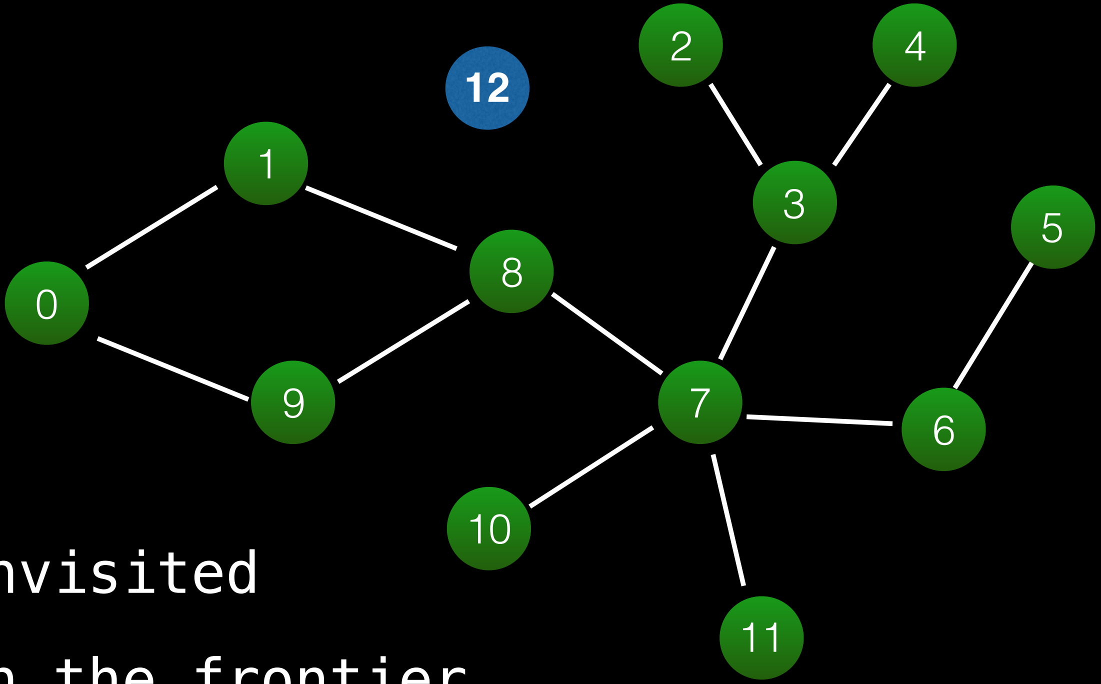


🔵 Unvisited

🟠 On the frontier

🟢 Visited

Queue Example - BFS



🔵 Unvisited

🟠 On the frontier

🟢 Visited

Queue Example - BFS

```
Let Q be a Queue
Q.enqueue(starting_node)          Add starting node to our queue
starting_node.visited = true      Mark starting node as visited

While Q is not empty Do

    node = Q.dequeue()     remove element from start

    For neighbour in neighbours(node):
        If neighbour has not been visited:
            neighbour.visited = true
            Q.enqueue(neighbour)
```