Graphs: Basic Concepts, Breadth First Search
This is an undirected graph where there are 5 vertices $v_1, v_2, \ldots, v_5$, and 5 edges $e_1, e_2, \ldots, e_5$. 
An undirected graph is a pair of \((V, E)\) where

- \(V\) is a set of elements, each of which called a node
- \(E\) is a set of pairs \((u, v)\) such that
  - \(u\) and \(v\) are distinct nodes;
  - if \((u, v)\) is in \(E\), then \((v, u)\) is also in \(E\) – we say that there is an edge between \(u\) and \(v\).

A node may also be called a vertex. We will refer to \(V\) as the vertex set or the node set of the graph, and \(E\) the edge set.
This is a directed graph \((V, E)\) where there are 5 vertices \(v_1, v_2, \ldots, v_5\), and 7 edges \(e_1, e_2, \ldots, e_7\). Note that every edge has a direction. Edge \(e_6\), for instance, is an outgoing edge of \(v_5\), and an incoming edge of \(v_4\).
Directed Graphs

A directed graph is a pair of \((V, E)\) where

- \(V\) is a set of elements, each of which called a node
- \(E\) is a set of pairs \((u, v)\) where \(u\) and \(v\) are nodes in \(V\). We say that there is a (directed) edge from \(u\) to \(v\).

A node may also be called a vertex. We will refer to \(V\) as the vertex set or the node set of the graph, and \(E\) the edge set.

A (directed) edge \((u, v)\) is said to be an outgoing edge of \(u\), and an incoming edge of \(v\). Accordingly, \(v\) is an out-neighbor of \(u\), and \(u\) an in-neighbor of \(v\).
In an undirected graph, the degree of a vertex $u$ is the number of edges of $u$.

In a directed graph, the out-degree of a vertex $u$ is the number of outgoing edges of $u$, and its in-degree is the number of its incoming edges.

In the left graph, the degree of $v_5$ is 2. In the right graph, the out-degree of $v_3$ is 2, and its in-degree is 1.
Next, we discuss two common ways to store a graph: adjacency list and adjacency matrix. In both cases, we represent each vertex in \( V \) using a unique id in 1, 2, \ldots, \( |V| \).
Adjacency list - Undirected graphs

Each vertex $u \in V$ is associated with a linked list that enumerates all the vertices that are connected to $u$.

Space $= O(|V| + |E|)$. 
Each vertex $u \in V$ is associated with a linked list that enumerates all the vertices that are connected to $u$.

Example 2.

Space $= O(|V| + |E|)$. 
Adjacency matrix - Undirected graphs

A $|V| \times |V|$ matrix $A$ where $A[u, v] = 1$ if $(u, v) \in E$, or 0 otherwise.

- $A$ must be symmetric.
- Space = $O(|V|^2)$.

Think: How to store $A$ so that, for any vertices $u, v \in V$, we can find out if they have an edge in constant time?
Adjacency matrix - Directed graphs

Defined in the same way as in the undirected case.

Example 4.

- $A$ may not be symmetric.
- $\text{Space} = O(|V|^2)$. 
Next, we will discuss a simple algorithm – called breadth first search – to traverse all the nodes and edges in a graph once.

We will cast it in a concrete problem: single source shortest path (SSSP) with unit weights.
Shortest Path

Let $G = (V, E)$ be a directed graph.

A path in $G$ is a sequence of edges $(v_1, v_2), (v_2, v_3), \ldots, (v_l, v_{l+1})$, for some integer $l \geq 1$, which is called the length of the path. The path is said to be from $v_1$ to $v_{l+1}$.

- Sometimes, we will also denote the path as $v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_{l+1}$.

Given two vertices $u, v \in V$, a shortest path from $u$ to $v$ is a path from $u$ to $v$ that has the minimum length among all the paths from $u$ to $v$.

If there is no path from $u$ to $v$, then $v$ is said to be unreachable from $u$. 
There are several paths from $a$ to $g$:

- $a \rightarrow b \rightarrow c \rightarrow d \rightarrow g$ (length 4)
- $a \rightarrow b \rightarrow c \rightarrow e \rightarrow d \rightarrow g$ (length 5)
- $a \rightarrow d \rightarrow g$ (length 2)

The last one is a shortest path. In this case, the shortest path is unique. Note that $h$ is unreachable from $a$. 
Let $G = (V, E)$ be a directed graph, and $s$ be a vertex in $V$. The goal of the SSSP problem is to find, for every other vertex $t \in V \setminus \{s\}$, a shortest path from $s$ to $t$, unless $t$ is unreachable from $s$. 
Next, we will describe the breadth first search (BFS) algorithm to solve the problem in $O(|V| + |E|)$ time, which is clearly optimal (because any algorithm must at least see every vertex and every edge once in the worst case.)

At first glance, this may look surprising because the total length of all the shortest paths may reach $O(|V|^2)$, even when $|E| = O(|V|)$ (can you give such an example?)! So shouldn’t the algorithm need $O(|V|^2)$ time just to output all the shortest paths in the worst case?

The answer, interestingly, is no.
At the beginning, color all vertices in the graph white. And create an empty BFS tree $T$.

Create a queue $Q$. Insert the source vertex $s$ into $Q$, and color it gray (which means "in the queue").

Make $s$ the root of $T$. 
Example

Suppose that the source vertex is $a$.

$Q = (a)$.
Repeat the following until $Q$ is empty.

1. De-queue from $Q$ the first vertex $v$.
2. For every out-neighbor $u$ of $v$ that is still white:
   1. En-queue $u$ into $Q$, and color $u$ gray.
   2. Make $u$ a child of $v$ in the BFS tree $T$.
3. Color $v$ black (meaning $v$ is done).

BFS behaves like “spreading a virus”, as we will see from our running example.
Running example

After de-queueing $a$:

$Q = (b, d)$. 
After de-queueing $b$: 

\[ Q = (d, c). \]
Running example

After de-queueing $d$:

$Q = (c, g)$. 
Running example

After de-queueing c:

\[ Q = (g, e) \].

Note: \( d \) is not en-queued again because it is black.
Running example

After de-queueing $g$:

$Q = (e, f, i)$. 
Running example

After de-queueing $e, f, i$:

\[ Q = () \]

This is the end of BFS. Note that $h$ remains white – we can conclude that it is not reachable from $a$. 

BFS tree
Running example

Where are the shortest paths?

The shortest path from $a$ to any vertex, say $x$, is simply the path from $a$ to node $x$ in the BFS tree!

- The proof will be left as an exercise.
Time Analysis

When a vertex $v$ is de-queued, we spend $O(1 + d^+(v))$ time processing it, where $d^+(v)$ is the out-degree of $v$.

Clearly, every vertex enters the queue at most once.

The total running time of BFS is therefore

$$O\left(\sum_{v \in V} (1 + d^+(v))\right) = O(|V| + |E|)$$

.