Shortest Route

Given an unweighted, undirected graph which represents a metro map as follows

* vertices are stations
* edges are the path between two stations

Given a start station and ending station, determine the minimum number of stops that must be made to get to the destination.

Input: A Graph (unweighted/undirected), a starting Vertex, and an ending Vertex  
Output: Integer

# Example

Input: The graph represented below, Vertex A, Vertex F



Output: 2 Stops (From A stop at C, and then stop at F)

# Constraints

Time Complexity: O(V + E) where V is the number of Vertices and E is the number of Edges  
Auxiliary Space Complexity: O(V)  
  
A graph Vertex instance has the following properties:

value : a string  
 edges : a list which contains references to other vertices in the Graph

The graph has a list of all the vertices: Graph.vertices

The Vertex values are all unique

# Solution

Perform a BFS from the starting node.

1. Create a ‘distance’ hashtable.
   1. Store each vertex value as the key, and a distance of INFINITY
   2. Set the ‘start’ vertex value to 0
2. Create a ‘visited’ hashtable
   1. Store each vertex value as the key and a visited to false
3. Instantiate a queue and place the starting vertex in the queue
4. While there is something in the queue
   1. Dequeue the ‘current’ vertex and mark it as visited.
   2. For each edge, if the associated ‘neighbor’ has not been visited
      1. Update the neighbor’s ‘distance’ as the minimum of:
         1. the distance of ‘neighbor’
         2. the distance of the ‘current’ + 1
      2. Add the neighbor to the queue
5. Return the distance of ‘end’ vertex

# Resources

<https://www.hackerrank.com/challenges/bfsshortreach>

<http://www.geeksforgeeks.org/breadth-first-traversal-for-a-graph/>