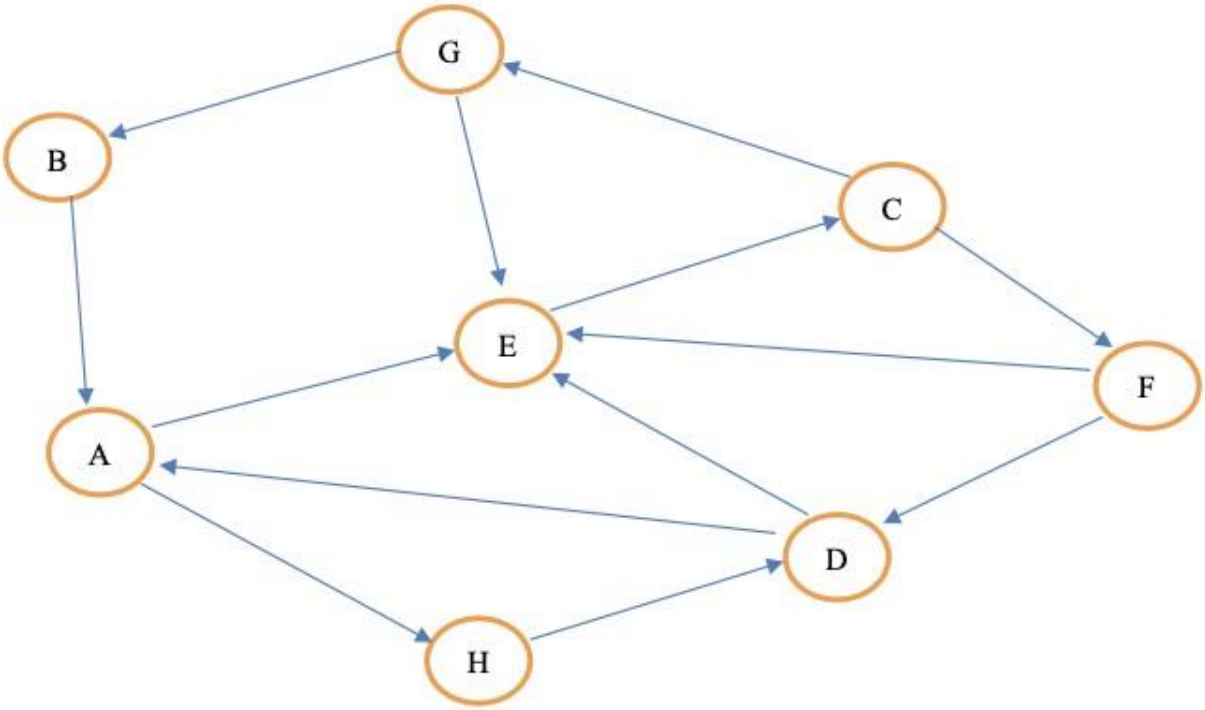
CSC6013 - Worksheet for Week 3

BFS - Breadth First Search using the brute force algorithm as seem in class

Consider the graph below:



1) Represent this graph using adjacency lists. Arrange the neighbors of each vertex in alphabetical order.

- list the triplets for this graph in the form (A, B, 1), where there is a edge from vertex A to vertex B;
- Note that this graph is directed, unlike the one presented in class.

- **(A, E, 1), (A, H, 1)**
- **(B, A, 1)**
- **(C, F, 1), (C, G, 1)**
- **(D, A, 1), (D, E, 1)**
- **(E, C, 1)**
- **(F, D, 1), (F, E, 1)**
- **(G, B, 1), (G, E, 1)**
- **(H, D, 1)**

2) Trace the BFS execution by adapting the code to deal with a directed graph (remove lines 14, 15, and 16) and instrumenting it to print every time a vertex is visited and everytime a vertex is enqueued or dequeued.

- Each time a vertex A is visited print: "Vertex A visited" and the current array V;
- Each time a vertex B is enqueued print: "Vertex B enqueued" and the current queue Q;
- Each time a vertex C is dequeued print: "Vertex C enqueued" and the current queue Q.

```
Vertex A enqueued, Queue: ['A']
Vertex A visited,  Visited: ['A']
Vertex E enqueued, Queue: ['A', 'E']
Vertex E visited,  Visited: ['A', 'E']
Vertex H enqueued, Queue: ['A', 'E', 'H']
Vertex H visited,  Visited: ['A', 'E', 'H']
Vertex A dequeued, Queue: ['E', 'H']
Vertex C enqueued, Queue: ['E', 'H', 'C']
Vertex C visited,  Visited: ['A', 'E', 'H', 'C']
Vertex E dequeued, Queue: ['H', 'C']
Vertex D enqueued, Queue: ['H', 'C', 'D']
Vertex D visited,  Visited: ['A', 'E', 'H', 'C', 'D']
Vertex H dequeued, Queue: ['C', 'D']
Vertex F enqueued, Queue: ['C', 'D', 'F']
Vertex F visited,  Visited: ['A', 'E', 'H', 'C', 'D', 'F']
Vertex G enqueued, Queue: ['C', 'D', 'F', 'G']
Vertex G visited,  Visited: ['A', 'E', 'H', 'C', 'D', 'F', 'G']
Vertex C dequeued, Queue: ['D', 'F', 'G']
Vertex D dequeued, Queue: ['F', 'G']
Vertex F dequeued, Queue: ['G']
Vertex B enqueued, Queue: ['G', 'B']
Vertex B visited,  Visited: ['A', 'E', 'H', 'C', 'D', 'F', 'G', 'B']
Vertex G dequeued, Queue: ['B']
Vertex B dequeued, Queue: []
```