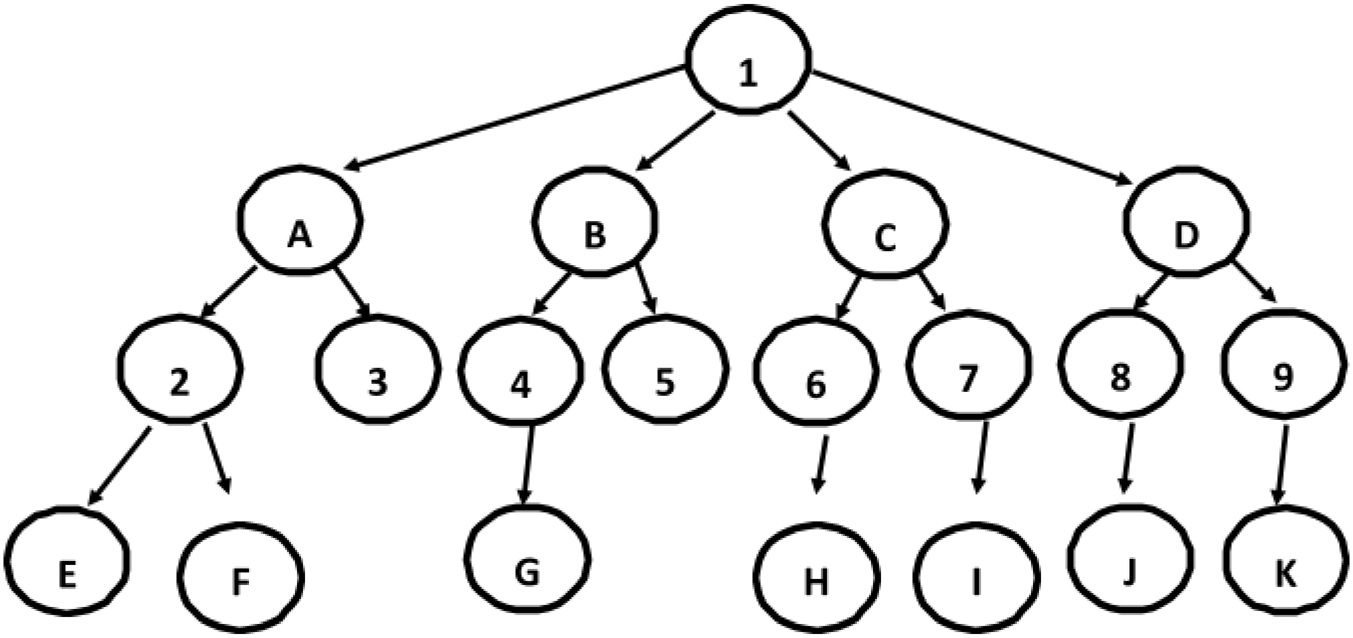
COP4601 – Assignment - Search

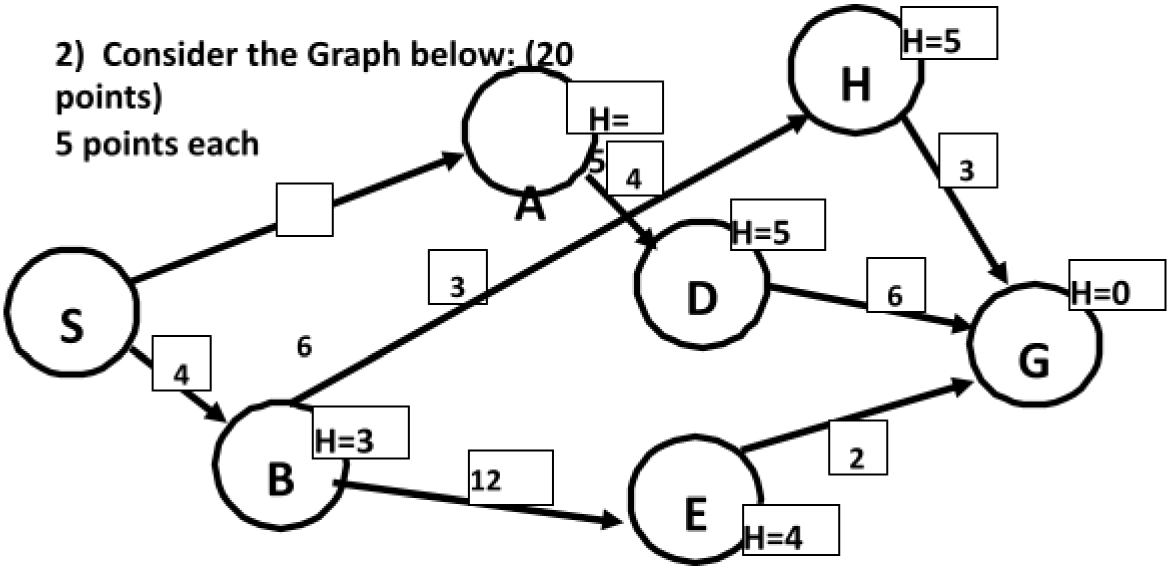
1) **Consider the Tree below: (20 points)**
   2 points each



- List the order in which the nodes are visited to depth first search for Node 5
- List the order in which the nodes are visited to breadth first search for Node 5
- Which method looked at fewer nodes? Why?
- List the order in which the nodes are visited to depth first search for Node D
- List the order in which the nodes are visited to breadth first search for Node D
- Which method looked at fewer nodes? Why?
- List the order in which the nodes are visited to depth first search for Node K
- List the order in which the nodes are visited to breadth first search for Node K
- Which method looked at fewer nodes? Why?
- Which node is the worst node for breadth first search to locate? Why?

COP4601 – Assignment - Search

2) Consider the Graph below: (20 points)
5 points each



Using the graph above

- Solve using greedy search starting at Node S and ending at Node G
- Solve using A* search starting at Node S and ending at Node G
- What is the optimal path starting at Node S and ending at Node G?
- Does A* locate the optimal path starting at Node S and ending at Node G? Why or why not?

COP4601 – Assignment - Search

1. **Tree Problem**
   a. DFS to node 5: 1, A, 2, E, F, 3, B, 4, G, 5
   b. BFS to node 5: 1, A, B, C, D, 2, 3, 4, 5
   c. The DFS took 10 steps, while the BFS took 9 steps. It is faster because node 5 was not excessively deep into the tree, therefore BFS is marginally faster.
   d. DFS to node D: 1, A, 2, E, F, 3, B, 4, G, 5, C, 6, H, 7, I, D
   e. BFS to node D: 1, A, B, C, D
   f. DFS took 16 steps, while BFS took 5 steps. This is because node D is on a high level, which means that the BFS will reach it very early in its runtime (much faster than DFS can). BFS is significantly faster in this case because DFS got stuck searching irrelevant branches on the left hand side of the graph.
   g. DFS to node K: 1, A, 2, E, F, 3, B, 4, G, 5, C, 6, H, 7, I, D, 8, J, 9, K
   h. BFS to node K: 1, A, B, C, D, 2, 3, 4, 5, 6, 7, 8, 9, E, F, G, H, I, J, K
   i. They both take the maximum number of steps to reach node K because it is the bottom right node in the tree, which is last node searched for both BFS and DFS.
   j. The bottom right node is the worst node for BFS to locate because it will always be last in the search order.
2. **Graph Problem**
   a. Greedy Search from S -> G: S, B, E, G
   b. A* Search from S -> G: S, B, H, G
   c. The optimal path is S, B, H, G.
   d. The A* search did successfully locate the most optimal path from node S to node G. The alternative paths are S,B,E,G (25), S,A,D,G (26), while S,B,H,G has a weight of 20.