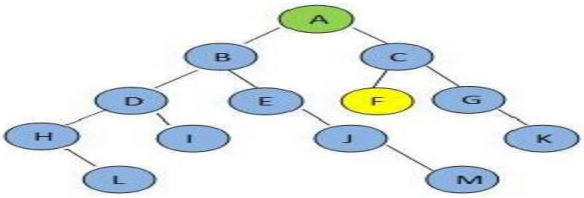
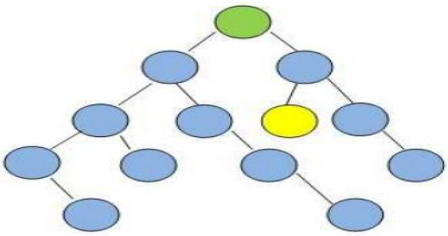
Sheet- 4 Int to AI

Q1: list the sequence of the visiting node in BFS, DFS



Q2:

- In the following graphs, assume that if there is ever a choice amongst multiple nodes, both the BFS and DFS algorithms will choose the left-most node first.

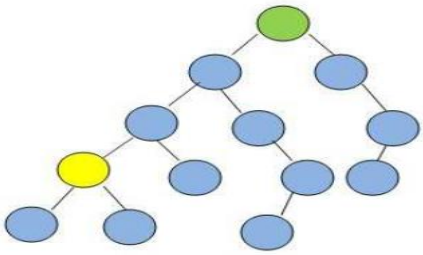


Starting from the green node at the top, which algorithm will visit the least number of nodes before visiting the yellow goal node?

- A: BFS
- B: DFS
- C: Neither BFS nor DFS will ever encounter the goal node in this graph.
- D: BFS and DFS encounter same number of nodes before encounter the goal node

Q3:

- In the following graphs, assume that if there is ever a choice amongst multiple nodes, both the BFS and DFS algorithms will choose the left-most node first.



Starting from the green node at the top, which algorithm will visit the least number of nodes before visiting the yellow goal node?

- A: BFS
- B: DFS
- C: Neither BFS nor DFS will ever encounter the goal node in this graph.
- D: BFS and DFS encounter same number of nodes before encounter the goal node

Q4 :

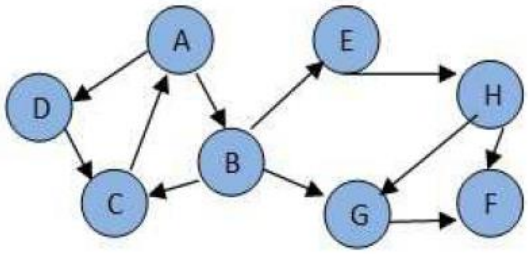
1) Suppose you are searching for a girl's name written using only the letters D, N and A. You have the letters ordered alphabetically (A, D, N) and you start writing down possibilities:

A, D, N, AA, AD, AN, DA, DD, DN, NA, ND, NN, ...

1a) How many strings of four or fewer letters are there where the letters are D, N or A?
1b) In the above possibilities, are you searching in a depth first or breadth first way?
1c) What are the next three possible names you would write down?
1d) How many possibilities will you write down before getting to the name ANNA? [difficult]
1e) Are you guaranteed to find all girls names with letters D, N and A in this manner?

Q5:

- Consider the following graph. If there is ever a decision between multiple neighbor nodes in the BFS or DFS algorithms, assume we always choose the letter closest to the beginning of the alphabet first.



Before countering goal node F:
BFS algorithm encounters nodes: ABCDE
DFS algorithm encounters nodes: ABDHLIEJMC

How can we get ?

A->B->D

B->C->E->G

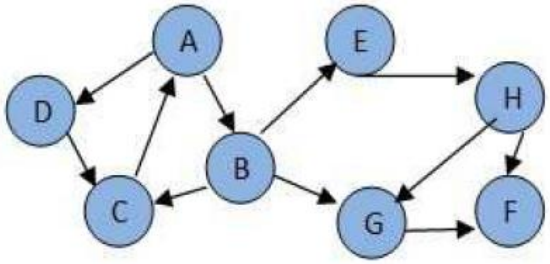
D->C

C->A

E->H

G->F

H->F->G



So for BFS, the answer is ABDCEGHF

for DFS, the answer is ABCEHFGD

In what order will the nodes be visited using a Breadth First Search? The answer is: ABDCEGHF
In what order will the nodes be visited using a Depth First Search? The answer is: ABCEHFGD

Adding all the possibilities up, we get: 81 + 27 + 9 + 3 = 120 possible names.

1b) This is a breadth first search (BFS), because all possibilities for two letter names are exhausted before moving on to three letter names, and so on. It's also a *uniform* search, because three nodes are expanded each time – one for A, one for D and one for N.

1c) The next names would be: AAA, AAD, AAN, because you expand the AA node (using the graph analogy for search). Similarly, the agenda items (AA, add A); (AA, add D); (AA, add N) would be at the top of the agenda by this stage.

1d) All three-or-fewer-letter names will have been exhausted by this stage, so there will be 3 + 9 + 27 = 39 possibilities written down before ANNA. All four-letter possibilities starting with AAA, AAD, AAN, ADA, ADD, ADN, ANA and AND (no pun intended) will have also been written down. There are three four-letter possibilities starting AAA, namely AAAA, AAAD and AAAN, similarly three for AAD and so on. So, there will be 3+3+3+3+3+3+3+3 = 8*3 = 24 possibilities with four letters in already written down. There are no more before ANNA, hence the total is 39 + 24 = 63. Hence ANNA is found after roughly half the search (of 120 nodes). [Extra question for the really keen among you: how many would be written down if the search was a Russell & Norvig style depth first search? How many for an IDS?]

1e) Yes, all names will eventually be enumerated, because breadth first search is *complete*, i.e., guaranteed to cover all possibilities. It's worth bearing in mind that if you wanted to find a long name, say, up to 10 or 12 letters, then the Universe might end before you had exhausted your BFS.