CS2230A

Edit Quiz

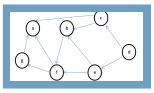
SAVE & EXIT

Enable Sharing
SOC-28285274

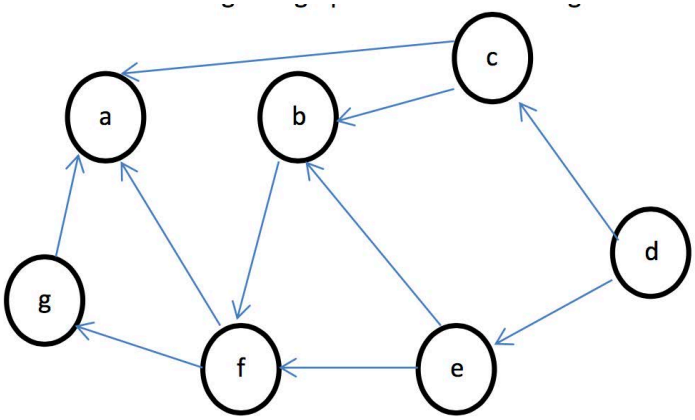
Graph traversals

Align entire quiz to a standard

#1



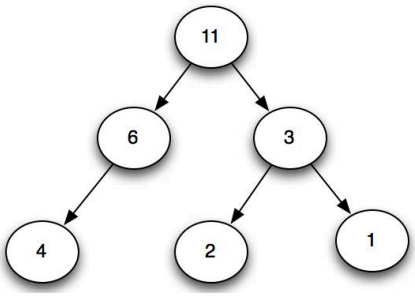
Find a path from d to a.

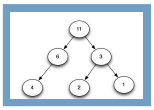


Explanation:

Your answer could be any sequence of nodes following directed edges that starts with d and ends with a
For example, d, c, a
or d,e,b,f,a
or several others!

#2





Remember depth-first search of trees? What is the pre-order DFS from the root of this tree? Look at left before right.

ANSWER CHOICE

A 4, 2, 1, 6, 3, 11

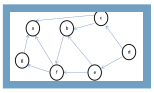
B	4, 6, 11, 2, 3, 1
C	11, 6, 4, 6, 11, 3, 2, 3, 1
D	11, 6, 4, 3, 2, 1
E	11, 6, 3, 4, 2, 1

Explanation:

visit the root, then DFS the left tree, then DFS the right tree

#3

EDIT

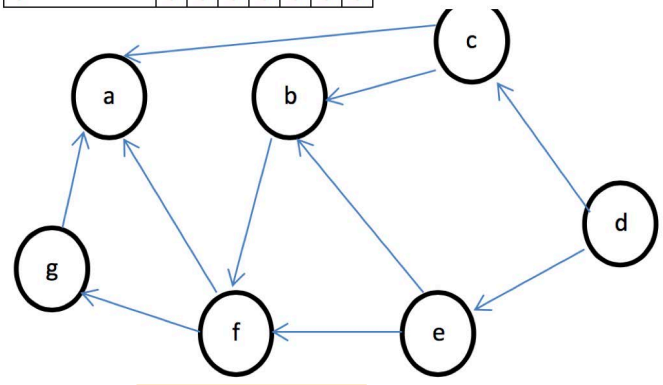


Perform a pre-order DFS of this graph, starting at node d.
- do not write down the same node twice
- when picking which neighbor to visit first, go in alphabetical order

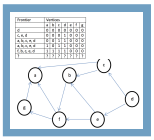
ANSWER CHOICE

A	d, c, a, b, e, f, g
B	d, c, a, b, f, g, e
C	d, c, e, a, b, f, g
D	d, c, a

Frontier	Vertices						
	a	b	c	d	e	f	g
d	0	0	0	0	0	0	0
c, e, d	0	0	0	1	0	0	0
a, b, c, e, d	0	0	1	1	0	0	0
a, b, c, e, d	1	0	1	1	0	0	0
f, b, c, e, d	1	1	1	1	0	0	0
?	?	?	?	?	?	?	?



#4



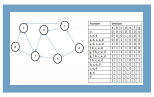
The pre-order DFS from node d is d, c, a, b, f, g, e.
Write down the frontier at each step, and mark a node visited (with a 1) when you push its children onto the frontier. Do not add visited nodes to the frontier.

What is the next step?

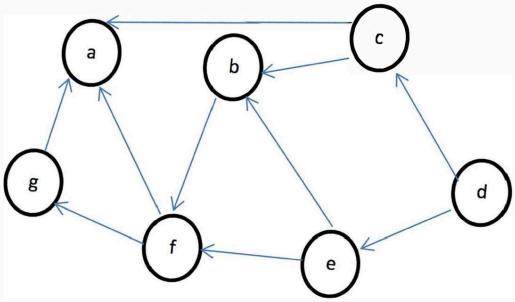
ANSWER CHOICE

A	f, b, c, e, d || a1 b1 c1 d1 e0 f1 g1
B	g, f, b, c, e, d || a1 b1 c1 d1 e0 f1 g0
C	f, b, c, e, d || a1 b1 c1 d1 e0 f1 g1
D	g, f, b, c, e, d || a1 b1 c1 d1 e0 f1 g1
E	g, f, b, c, e, d || a1 b1 c1 d1 e0 f0 g1

#5



The pre-order DFS
Write down the fro
(with a 1) when you
add visited nodes t



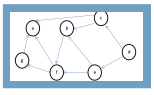
Frontier	Vertices						
	a	b	c	d	e	f	g
d	0	0	0	0	0	0	0
c, e, d	0	0	0	1	0	0	0
a, b, c, e, d	0	0	1	1	0	0	0
a, b, c, e, d	1	0	1	1	0	0	0
f, b, c, e, d	1	1	1	1	0	0	0
g, f, b, c, e, d	1	1	1	1	0	1	0
g, f, b, c, e, d	1	1	1	1	0	1	1
f, b, c, e, d	1	1	1	1	0	1	1
b, c, e, d	1	1	1	1	0	1	1
c, e, d	1	1	1	1	0	1	1
e, d	1	1	1	1	1	1	1
d	1	1	1	1	1	1	1
	1	1	1	1	1	1	1

Here is the rest of the traversal.

Does your group have any questions about DFS or this example?

#6

EDIT



Now, suppose you do DFS from node f. Which nodes will *not* be discovered?

ANSWER CHOICE

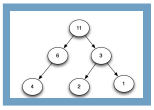
A	c, d
B	c, d, b
C	c, d, b, e
D	c, d, b, e, g
E	none

Explanation:

DFS from f will visit f, then visit a, then visit g (assuming we pick the neighbor we visit first in alphabetical order).

#7

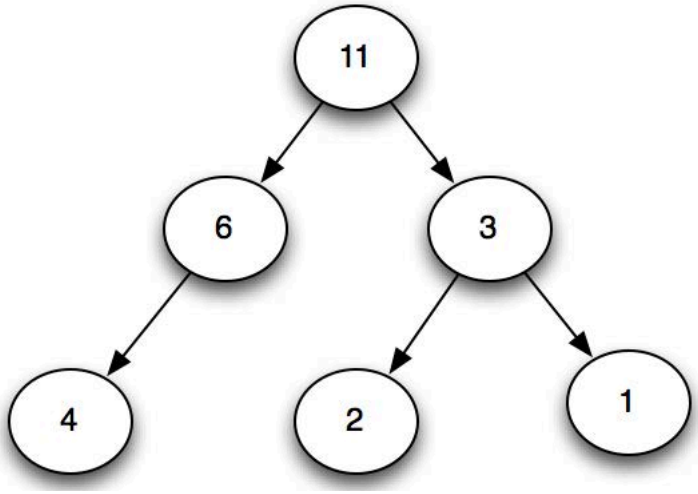
EDIT



Remember breadth-first search of trees? What is the BFS traversal from the root of this tree? Look at left before right.

ANSWER CHOICE

A	4, 2, 1, 6, 3, 11
B	4, 6, 11, 2, 3, 1
C	11, 6, 4, 6, 11, 3, 2, 3, 1
D	11, 6, 4, 3, 2, 1
E	11, 6, 3, 4, 2, 1

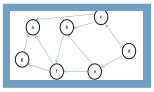


Explanation:

visit the root at depth 1, then left to right visit everything at depth 2, then left to right visit everything at depth 3

#8

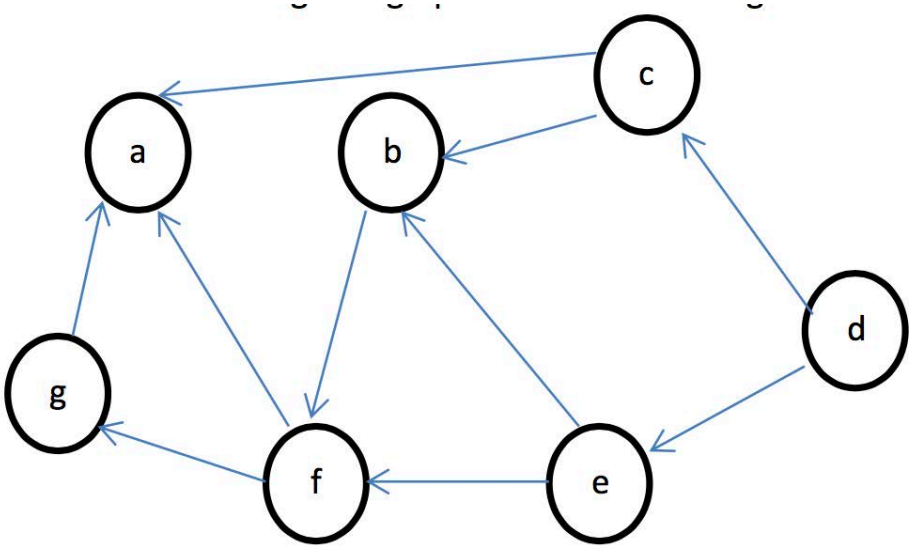
EDIT



Perform a BFS traversal of this graph, starting at node d.
- do not write down the same node twice
- when picking which neighbor to visit first, go in alphabetical order

ANSWER CHOICE

A	d, c, a, b, e, f, g
B	d, c, a, b, f, g, e
C	d, c, e, a, b, f, g
D	d, c, e, b, f, a, g

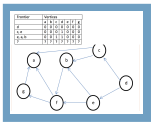


Explanation:

visit d, then visit everything new that is 1 hop from d (c,e), then visit everything new that is 2 hops from d (a,b,f), then visit everything new that is 3 hops from d (g).

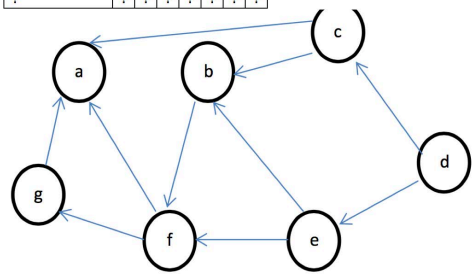
#9

EDIT



The table shows the frontier and which vertices have been visited in a BFS traversal from node d.
What is the next step?

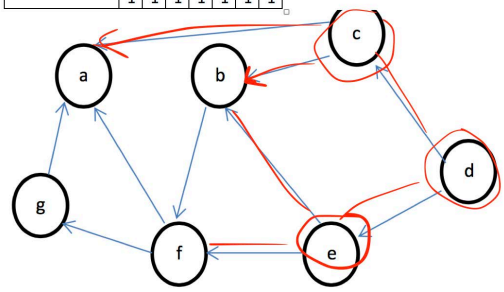
Frontier	Vertices						
	a	b	c	d	e	f	g
d	0	0	0	0	0	0	0
c, e	0	0	0	1	0	0	0
e, a, b	0	0	1	1	0	0	0
?	?	?	?	?	?	?	?



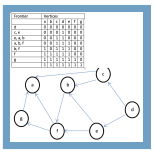
ANSWER CHOICE

A	a, b, b, f || a0 b0 c1 d1 e1 f0 g0
B	a, b || a0 b0 c1 d1 e1 f0 g0
C	a, b, f || a0 b0 c1 d1 e1 f0 g0
D	f, a, b || a0 b0 c1 d1 e1 f0 g0
E	f, e, a || a0 b1 c1 d1 e1 f0 g0

Frontier	Vertices						
	a	b	c	d	e	f	g
d	0	0	0	0	0	0	0
c, e	0	0	0	1	0	0	0
e, a, b	0	0	1	1	0	0	0
a, b, f	0	0	1	1	1	0	0
b, f	1	0	1	1	1	0	0
f	1	1	1	1	1	0	0
g	1	1	1	1	1	1	0
	1	1	1	1	1	1	1



#10



The table shows the frontier and which vertices visited in a BFS traversal from node d.

Here are all the steps in the BFS.

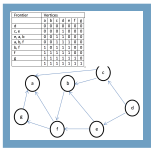
How could you use the BFS to find the smallest number of hops from d to any node?

Explanation:

Let h(d, X) be the hops from d to node X. When you visit a node X by dequeueing it from the frontier, for every *unvisited* neighbor Y, h(d,Y) = 1+h(d, X).

#11

EDIT



Does your group have any questions about BFS or this example?

#12

EDIT

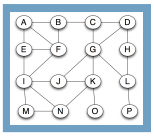
Have you ever heard of "6 degrees of separation"? It is the observation that any two people tend to be very few hops away (often <= 6) in the social network (e.g., a friend of a friend is 2 hops away). e.g., https://oracleofbacon.org/

Devise an experiment you could actually run to determine the

number of hops between you and Donald Trump (where an edge (X,Y) means X and Y are acquaintances).

#13

EDIT

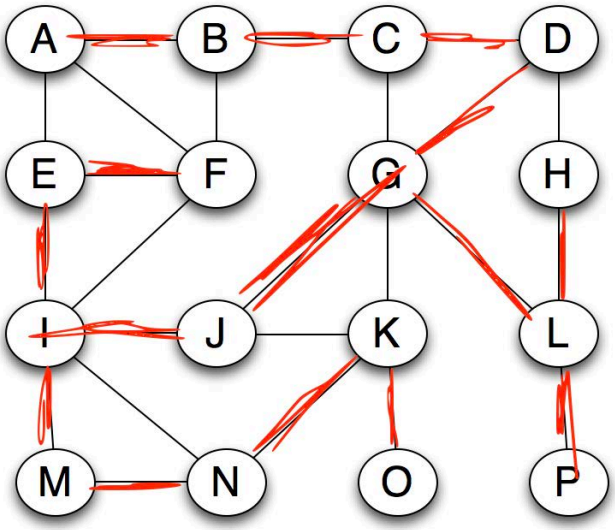


Now for a bigger example. Consider the undirected graph here. Perform a DFS traversal:
- starting at node A
- when there are two or more out-edges consider them in alphabetical order

When we tried the DFS traversal, the edges we followed traced out the following tree (given by its edges).

A->B, B->C, C->D, D->G, G->J, J->I, J->K, I->E, I->M, M->N, K->O, G->L, L->H, L->P

Did we make a mistake?



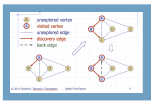
ANSWER CHOICE

A	J->K is not in the tree and N->K is in the tree
B	No mistakes
C	A->E and A->F are in the tree
D	F->B, F->A, F->I are in the tree
E	L->H is not in the tree

Forgot E->F, too

#14

EDIT



We can think of one product of a DFS traversal to be a "DFS tree". The nodes in this tree are the nodes discovered by the

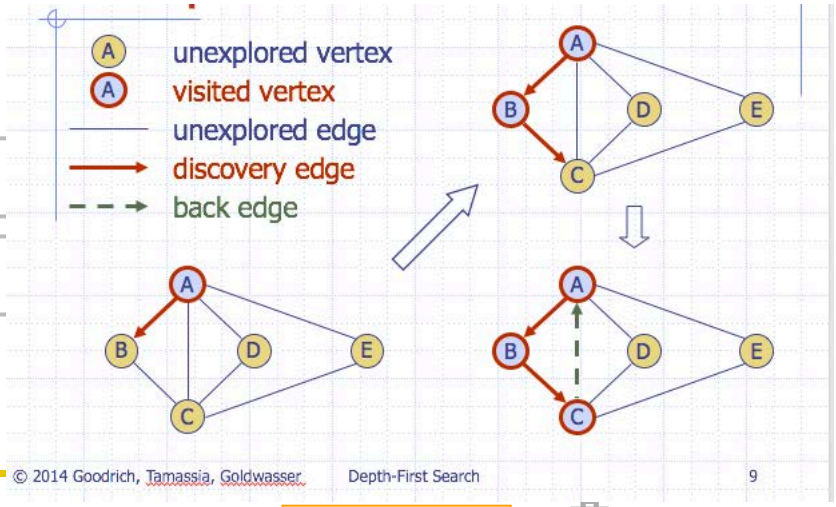
traversal, and the edges in the tree are "discovery edges", or edges where the traversal found a new node.

In the image is an example of part of a DFS traversal starting at node A. Traversing edge A->B discovers the unvisited B, so A->B is a "discovery edge".

When we visit C, we draw a "back edge" to A. This "back edge" leads to a node we've already visited. Only discovery edges are in the DFS tree.

DFS Trees will be useful for many algorithms; first we'll do some exercises with them.

Do you understand the concepts of DFS trees, discovery edges, back edges?



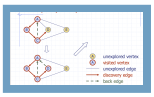
ANSWER CHOICE

A yes

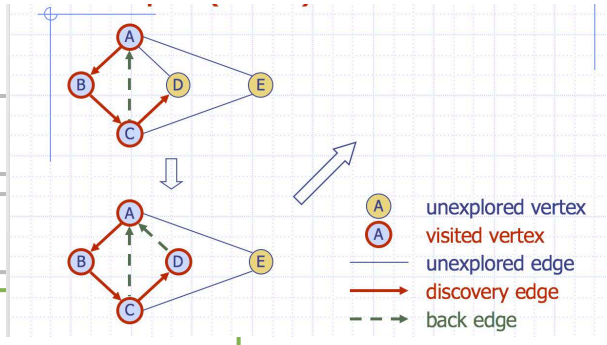
B no, I don't understand

#15

EDIT



Let's continue the traversal. What does the next step in the DFS look like?



ANSWER CHOICE

A visit A, mark discovery edge A->E

B visit A, mark discovery edge A->E

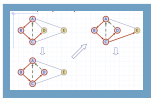
C mark discovery edge C->E

D mark back edge C->E

E mark back edge E->C

#16

EDIT



Let's continue the traversal. What does the next step in the DFS look like?

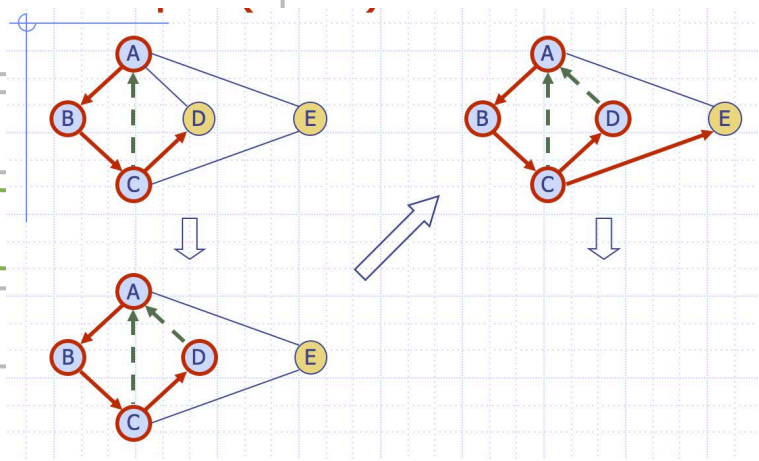
ANSWER CHOICE

A visit E, mark discovery edge A->E

B visit E, mark back edge A->E

C visit E, mark back edge E->A

D visit E, mark discovery edge E->A



#17

EDIT

DFS can be used for lots of algorithms, such as computing the path between two vertices, testing if G is connected, computing a "spanning tree" (next time) of G, computing "connected components", computing whether G has a cycle.

Describe how you would use DFS to compute whether G has a cycle. You must refer to the DFS tree in your answer.

If we find any back edges then G has a cycle.

#18

EDIT

Do you have any questions about graph traversals? (yes / no, followed by question)