

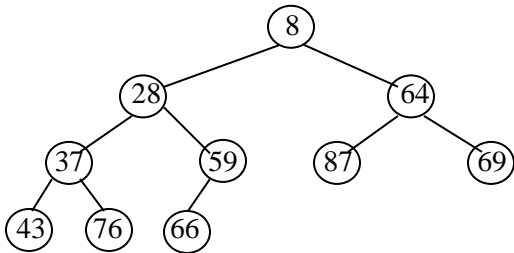
Topics: ADTs, heaps, and graphs.

1. (30 points) Programs Problem. This question will attempt to determine whether you wrote the code for programming assignment #3 and #4. There is no need for you to see sample questions if you actually participated in writing the code.
2. (30 points) Timetest3 Problem. This question will address what you learned writing your paper for timetest3.cpp. Questions could be of the form where you are given a description of a new File5.dat, and you are asked for the expected result for given ADTs compared to one of the original data files. Questions could also involve determining whether you clearly understand the source(s) of the anomalous behavior of the different ADTs.
3. (40 points) ADT Problem. Web search engines permit keyword searches. Assume that there are 100,000 keywords, 10 million document addresses, and that each document address is stored in exactly one place on a disk. There may be many pointers to that location on the disk, but there are no pointers from it. A document may have more than one keyword associated with it. The search engine companies must keep their information appearing current by ensuring that newly inserted documents are the first reported from a search.

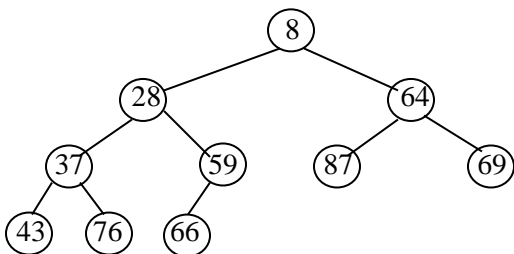
Describe the data structure(s) that you would use. Explain how you would insert and delete a document, including how many disk accesses would be involved. We do not expect you to come up with a perfect solution to this problem in 15 minutes.

4. (15 points) Heap Problem. Heap Problem. There will be one problem dealing insert, deleteMin, or BuildHeap operations with a priority queue. For each of the following binary heaps, show the state of heap after operation specified.

Insert(20)



DeleteMin()



5. Graph Problems There will be two graph problems using the following formats.

- a) Minimum Spanning Tree. Fill in the following tables using Kruskal's algorithm to find the minimum spanning tree of the graph. Use union-by-size, but not path compression. For a union of sets of the same size, the new root should be the old root with the lower array index. Stop the process when you have a minimum spanning tree. Weights are to the left of internal edges.

Edge	Weight	Action
(A, B)	37	
(A, E)	10	
(B, C)	41	
(B, E)	22	
(B, F)	3	
(C, D)	34	
(C, F)	45	
(C, G)	7	
(D, G)	25	
(D, H)	12	
(E, F)	17	
(F, G)	44	
(G, H)	29	

Find-Union Array							
A	B	C	D	E	F	G	H
0	1	2	3	4	5	6	7

Based on your tables, draw the minimum spanning tree of the graph. (Don't bother with weights.)

(A)

(B)

(C)

(D)

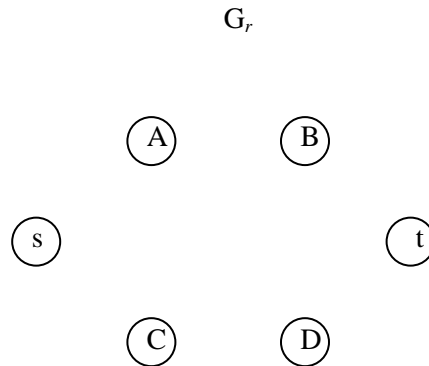
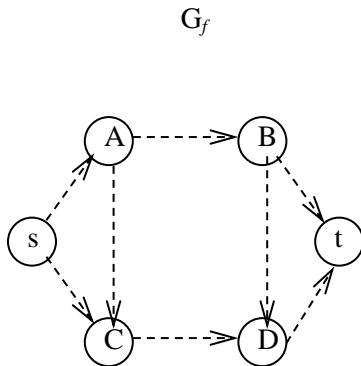
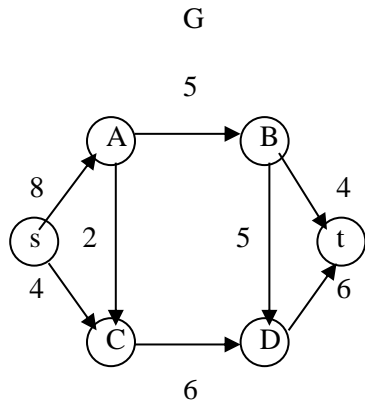
(E)

(F)

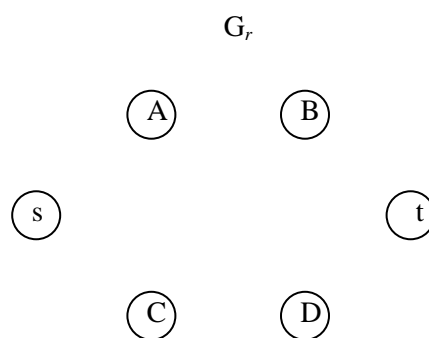
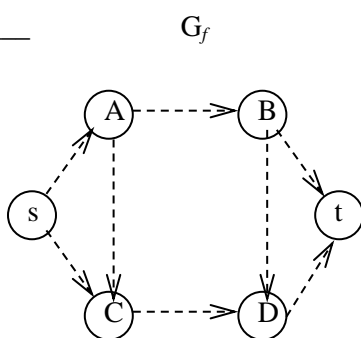
(G)

(H)

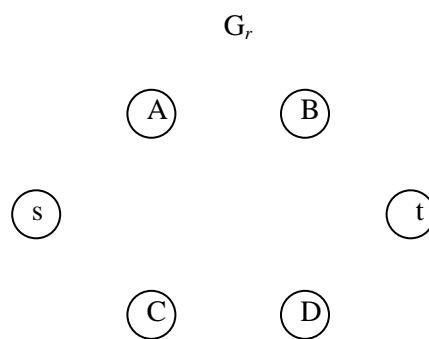
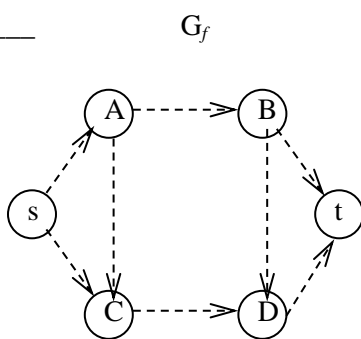
- b) Network Flow. Choosing the augmenting path that allows the largest increase in flow, find the maximum network flow for the following graph, G . Internal flows are to the left of their respective edges. Your score will be based on your completion of the augmented path line, G_f and G_r , including their initial states and back-flow edges where appropriate.



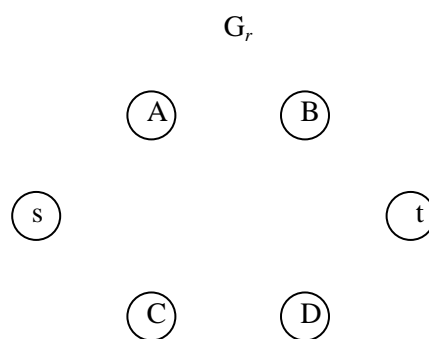
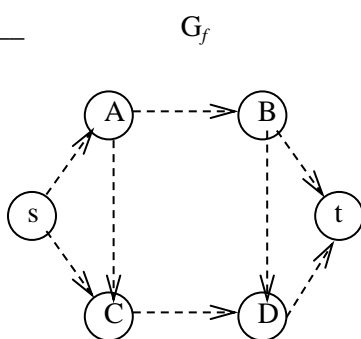
1st augmented path: _____



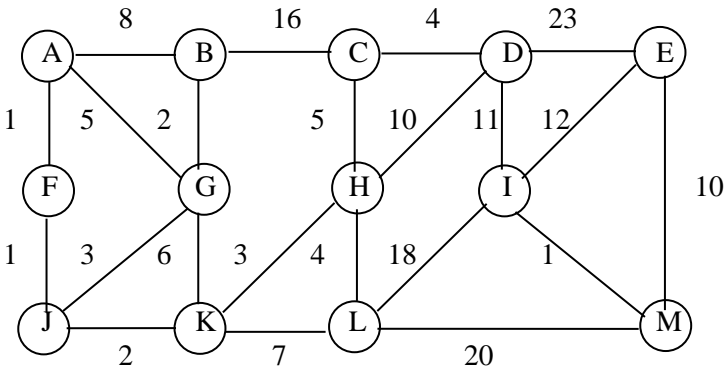
2nd augmented path: _____



3rd augmented path: _____

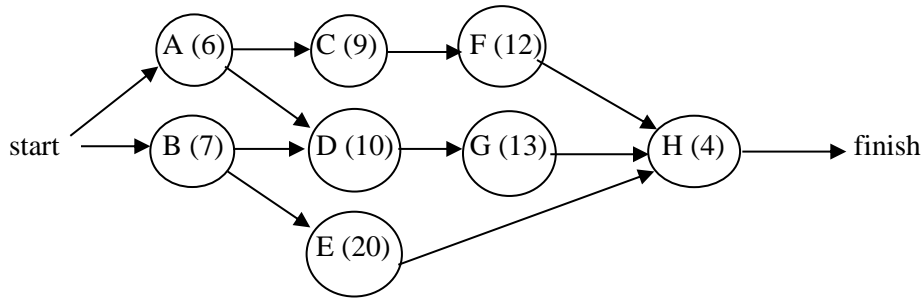


c) Shortest Path. Fill in the table using Dijkstra's algorithm to determine the shortest paths from vertex B. The weight of each interior edge is to the left of the edge.



Vertex	known	d _v	p _v
A			
B		0	---
C			
D			
E			
F			
G			
H			
I			
J			
K			
L			
M			

d) Event Node Graph. For the following Activity-node graph:

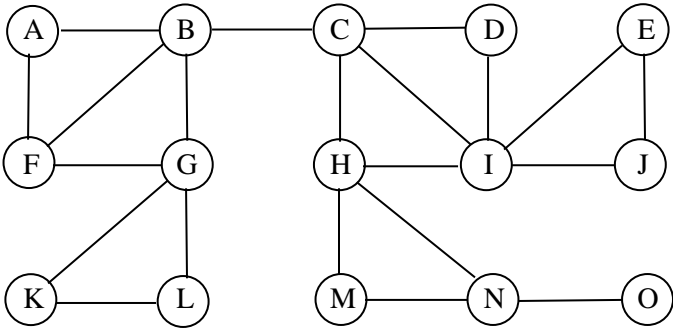


Draw the corresponding Event-node graph using numbered nodes (start numbering at left please).

Based on your graph fill in the values for the following table. There may be more columns than needed.

[illegible]

e) **Articulation Points.** Using DFS, starting at A, searching in alphabetical order whenever multiple vertices may be processed, give each vertex a number followed by its low. Then list the articulation points



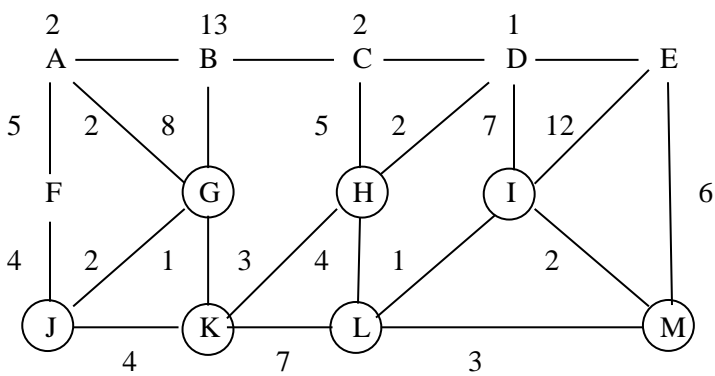
(5 points) Show DFS tree(s) below. You need not show the back edges, but may if you wish.

A 3x5 grid of circles, each containing a letter. The letters are arranged as follows:

A	B	C	D	E
F	G	H	I	J
K	L	M	N	O

[illegible]

f) Fill in the table using Prim's algorithm to determine the Minimum Spanning Tree from vertex E. The weight of each interior edge is to the left of the edge.



Vertex	known	d _v	p _v
A			
B			
C			
D			
E			
F			
G			
H			
I			
J			
K			
L			
M			

Based on your tables, draw the minimum spanning tree of the graph. (Don't bother with weights.)

