Department of Computer Science and Engineering
MID EXAMINATION, FALL' 14
CSE 221: Algorithms
Total Marks: 90   Time Allowed: 2.00 Hour

[Section A]

[Questions: 3]   [Marks:30]

Question 1:   [4+3+3 = 10]

a) Show the time complexity and space complexity of the following programme? Explain your answer   **[2+2 =4]**

`for(int i = 1; i<=n;i*=2 ) { c = i+8; } for(int j=n ; j>0 ; j/=2) { a[j] = 8; }`	`for(int i = 1; i<=n;i*=2 ) { for(int j=n ; j>0 ; j/=2) { a[j] = 8; } }`
Time :	Time :
Space:	Space:

b) $T(0) = O(1)$
$T(p) = 3T(\frac{2p}{8}) + 2T(\frac{p}{8}) + O(p)$
Solve the above recurrence relation and find the $T(p)$ in terms of big O notation.   **[3]**

c) Show a worst case scenario for which quicksort comprises $O(n^2)$. Explain with pictures.**[3]**

Question 2:   [3+3+4 = 10]

a) MergeSort and QuickSort are both of the type divide and conquer. What is meant by divide and conquer? Demonstrate a simple pictorial example.   **[3]**

b) Differentiate Merge Sort and QuickSort in terms of space complexity. Analyze the time complexity of Merge Sort .   **[3]**

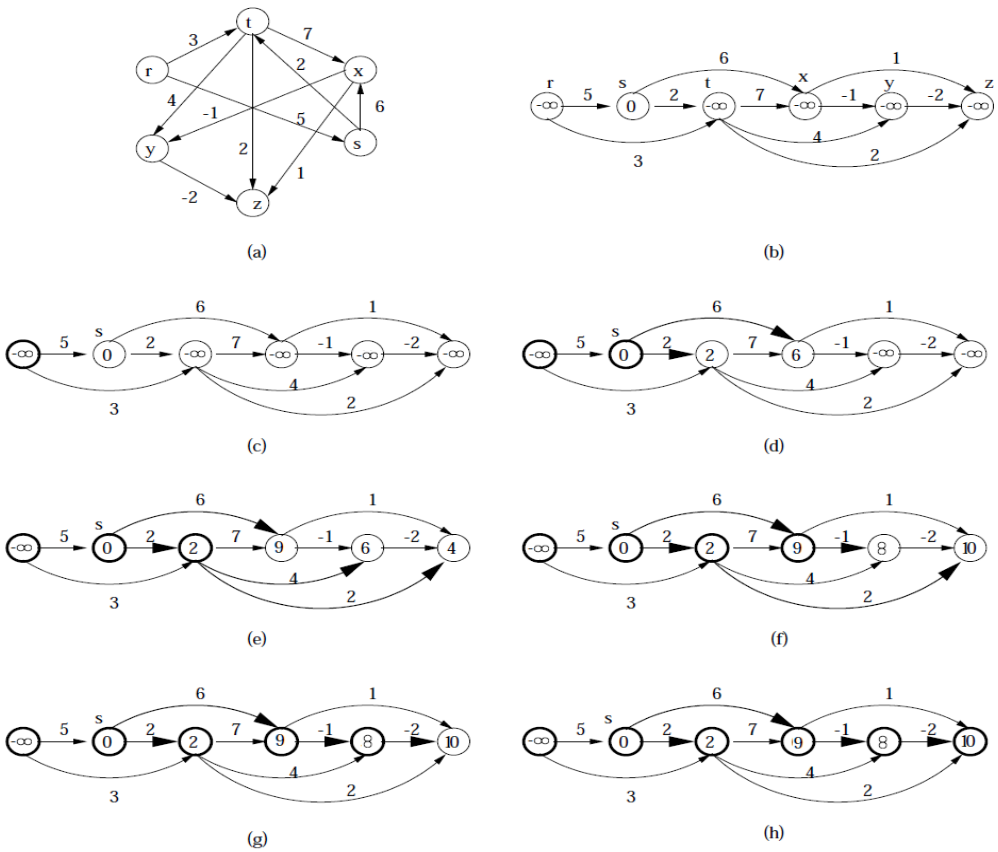
c) Draw the Max-heap of the following array? Write heapify algorithm and implement the algorithm on the following array with index 4.   **[4]**

2	43	1	3	-1	4

Question 3 : **[5+5=10]**

a) Given a Weighted Directed Acyclic Graph (DAG) and a source vertex s in it, find the longest distances from s to all other vertices in the given graph.

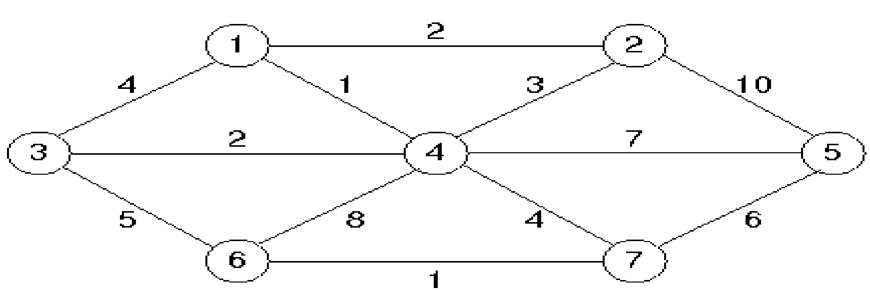
Following figure shows step by step process of finding longest paths.



Write down the algorithm for finding the longest paths. **[5]**

Hints: [In one step of your algorithm you need Create a topological order of all vertices.]

b) Find the MST of the following graph. You may use Prim / Kruskal as your preference algorithm. **[5]**



[Section B]

[Questions: 3]   [Marks:30]

Question 4 :

Compare Adjacency Matrix with Adjacency List Representation with respect to--

[5]

Cost of STORAGE:

(i) Sparse Graph ($|\mathbf{E}| \gg |\mathbf{V}|$) and Dense Graph ($|\mathbf{E}| \gg |\mathbf{V}|^2$)

(ii) Directed Graph and Undirected Graph

(iii) Weighted Graph and Un-weighted Graph

Cost of TIME:

(iv) To determine whether two vertices are connected or not

(v) To list all vertices adjacent to any particular vertex

Question 5 :   [4+6 = 10]

A project management technique called "**PERT**" involves breaking a large project into a number of tasks and determining which tasks cannot be started until other tasks have been completed. The project is then summarized in the following way:

In the Project there are 6 Tasks namely A, B, C, D, E and F.

- Task F cannot be completed until Tasks C and E are both completed.
- Similarly Task E cannot be completed until Tasks C and D are both completed.
- On the other hand Task C cannot be started until Task B is finished.
- Task D can be performed in parallel with B but Task D and B both depends on Task A.

Task-ID	Prerequisites
A	–
B	A
C	B
D	A
E	C, D
F	C, E

(a) Draw a DAG **showing the dependencies** among the tasks.

[Hint: if v depends on u then there is an edge from u to v]

(b) Solve the problem in **linear time** by **suggesting a proper order** of selecting and finishing the tasks.

Question 6 : **[3+4+(4+2)+2 = 15]**

(a) Justify that Dijkstra's algorithm will not work with graphs containing negative weighted edges.

(b) Analysis of Dijkstra Algorithm is given by the following Equation:

Total Time = $\Theta(V).T_{EXTRACTMIN} + \Theta(E).T_{DecreaseKEY}$

$T_{ExtractMin}$ = Time to Perform Extract Min Operation

$T_{DecreaseKey}$ = Time to Perform Decrease Key Operation

Then Fill up the following Table:

Queue Implementation	$T_{ExtractMin}$	$T_{DecreaseKey}$	Total Time
Linear Array			
Priority queue (Min Heap)			

(c) Write down the Bellman-Ford Algorithm. Explain why the Bellman-Ford algorithm does not need more than *N-1* iterations to find the shortest paths from vertex *s* to all other nodes, where *N* is the number of vertices. (Assume that there exists no negative cycle).

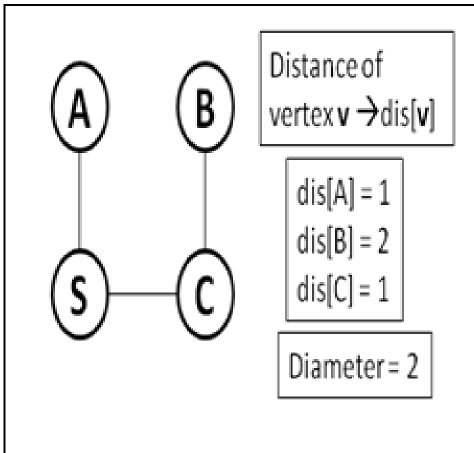
(d) Do minimum spanning tree algorithms Prim and Kruskal work for the graphs containing negative weighted edges? Why or why not?

[Section C]

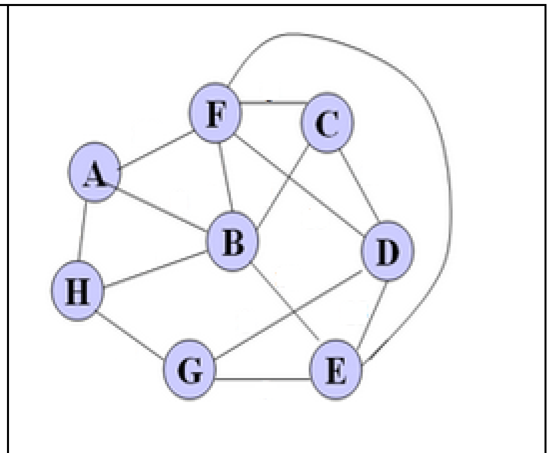
[Questions: 2] **[Marks : 30]**

Question 7:

The distance between any two vertices in an undirected unweighted graph is the length of the shortest path between them. The diameter of a graph is the maximum distance between two vertices in the graph.



(a) G_1



(b) G_2

As an example, the diameter of the graph: G_1 is max(1,2,1) = 2 where the source vertex is S. **[5+7+2=14]**

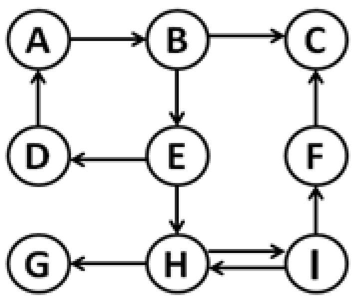
(a) Could you please find the diameter of the graph G_2?
(b) Give an algorithm to find the diameter of a graph.
(c) State the complexity of your algorithm.

Question 8: [5+(6+5) = 16]

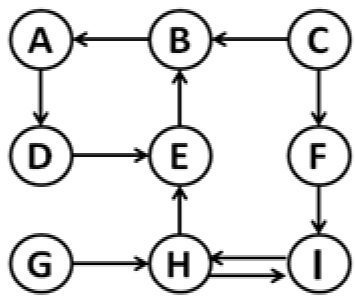
a) Classify the edges of the following digraph **G** into Back Edge, Tree Edge, Forward Edge and Cross Edge.

b) **Find out the strongly connected components** for the following graph G according to the following pseudo-code and **draw the corresponding component graph G^{scc}**.

```
SCC(G)
1. call DFS(G) to compute finishing times f[u] for all u
2. compute G^T
3. call DFS(G^T), but in the main loop, consider vertices
        inorder of decreasing f[u]
4. output the vertices in each tree of the depth-first forest
        formed in second DFS as a separate SCC
```



(a) G



(b) G^T