1. (5%) (a)Describe all operations associated with a single and a doubly linked lists,

   (5%) (b) Compare the differences of insertion methods for both data structures.

2. (5%) (a) Describe all operations associated with "Stack" in JAVA or C++.

   (5%) (b) Define a data structure "Stack" with a single linked list. (5%)

   (5%) (c) Describe how to implement two stacks using one array. The total number of elements in both stacks is limited by the array length; all stack operations should run in $O(1)$ time.

3. (12%) Let $G = (V, E)$ be a connected directed graph with a weight function $w: E \rightarrow \mathbf{R}$, where R is the set of real numbers, and $V = \{1, 2, 3, \ldots, n\}$ is the set of vertices. For convenience we assume that there is a weight matrix $W = (w_{ij})$ where w_{ij} is 0 if $i = j$;　w_{ij} is the weight of the directed edge (i, j) if $i \neq j$ and $(i, j) \in E$; w_{ij} is ∞ if $i \neq j$ and $(i, j) \notin E$. A vertex v in a simple path $p = \langle v_1, v_2, \ldots, v_l \rangle$ is said to be **intermediate** if $v \in \{v_2, v_3, \ldots, v_{l-1}\}$. Let $d_{ij}^{(k)}$ be the weight of a shortest path from vertex i to vertex j for which all intermediate vertices are in the set $\{1, 2, \ldots, k\}$. Then it is easy to see that $d_{ij}^{(0)} = w_{ij}$.

   (6%) (a) Give a recursive definition of $d_{ij}^{(k)}$ for $0 \leq k \leq n$.

   (6%) (b) Based on the recursive definition above, write an algorithm assuming an input W to compute the matrix $D^{(n)} = d_{ij}^{(n)}$.

4. (13%) Let $G = (V, E)$ be a connected undirected graph with a weight function $w: E \rightarrow \mathbf{R}$, where $\mathbf{R}$ is the set of real numbers and $|V| = n$. Kruskal's algorithm for finding minimum spanning tree can be summarized as follows. First, choose an edge in the graph with minimum weight. Successively add edges with minimum weight that do not form a simple cycle with those edges already chosen. Stop after $n - 1$ edges have been chosen.

   Prove correctness of Kruskal's algorithm.

5. (10%) Let x be an input of an optimization problem, Opt(x) be the optimum solution of x, A be a polynomial time algorithm for x, and ε be any positive real number. If A satisfies $|Opt(x) - A(x)| \leq \varepsilon Opt(x)$ for all x, then we say A is an ε-approximation algorithm.

   If G is an undirected graph, a **vertex cover** of G is a subset of the nodes where every edge of G touches at least one of those nodes.　We want to find the cover as small as possible. Consider the following ε-approximation algorithm for finding a vertex cover of G:

   > 1. $C := \phi$;
   > 2. **while there is an edge [u,v] in G**
   >   **add u, v to C and delete them from G;**

   Note that C forms a vertex cover of G. What is ε and the complexity of the above algorithm? Explain your answer!　Suppose there are n nodes and m edges in G.

6. (5%) Consider the following algorithm which divides the list in half each time. Once the length of the list is one, square the number and return it. Multiply the two values returned from each half of the list. Suppose the input has n numbers linked as a list, where the function Listhead returns the number stored in the first node of the list. Now you are asked to analyze how many multiplications are used in the algorithm. Let $T(n)$ be the number of multiplications used for a list of length n. Write down the recurrence and solve it.
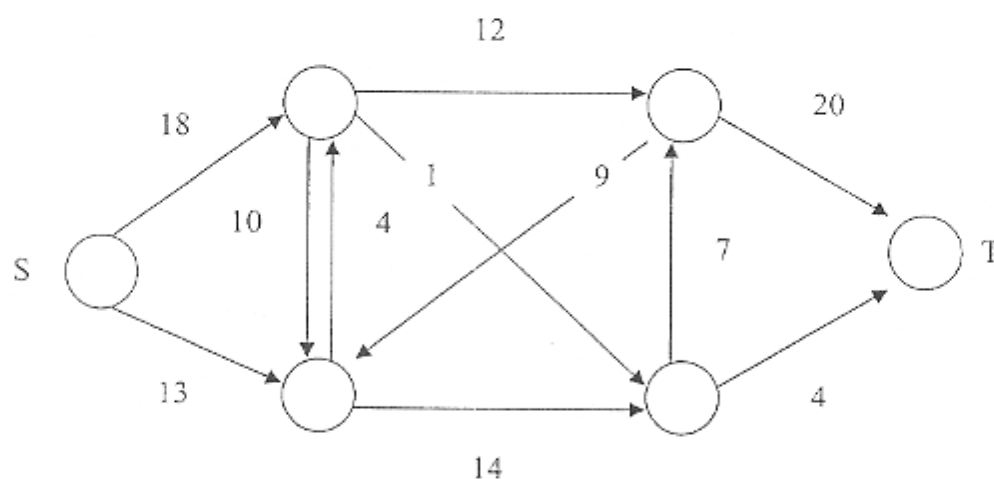
> **Procedure Count(list){**
> **If**　　**Length(list)=1 return (Listhead(list)*Listhead(list))**
> **Else**　**{c = Count(first half of list);**
> 　　　　　**d = Count(second half of list);}**
> **Return(c*d);}**

7. (10%) Consider the following flow network. Find the maximum flow and show a minimum cut.

8. (10%) Prove that a node of a binary tree has at most one parent.

9. Consider a sequence of keys: 4, 7, 12, 15, 3, 5, 14, 18. Please draw the result by inserting these keys into an empty

(5%) (a) binary search tree.

(5%) (b) red-black tree. (Please also show the color of each node)

(5%) (c) AVL tree.