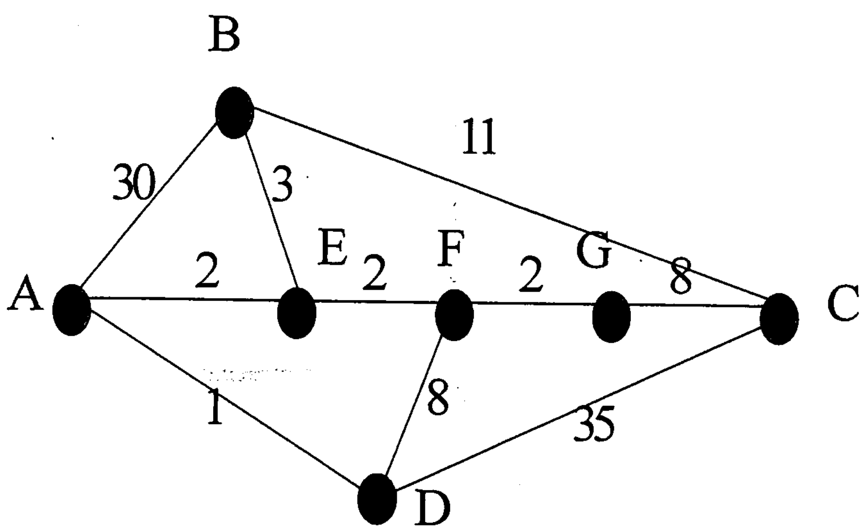
一、Data structure (50%)

1. Please find an optimal pair of shortest paths between source node A and destination node C with a pseudo code form and draw the final result. (Hint: joint computation of two link-disjoint paths) (10%)



2. Given n objects numbered from 1 to n. We wish to group them into disjoint sets. In each set we choose a canonical object (its smallest item), which will serve as a label for the set. Initially, the n objects are in n different sets each containing exactly one object, which is necessarily the label for the set. Therefore, we execute a series of n operations of two kinds:
Find(y): for a given object y, find which set contains it and return the label.
Union(A, B): given two distinct set labels, merge the two corresponding sets. Following the union, we will assume the set A and B no longer exist.
What is the total cost of a series of operations for the following four data structures: array, tree, tree with balancing, and tree with balancing and path compression and explain it shortly. (Hint: path compression means if i is a node on the path from x to its root r, then make i a child of r.) (12%)

3. Use the recursive method and the iterative method to represent the following equation: (Hint: using C code) (10%)

$$P = \frac{M \bullet (M-2) \bullet \cdots \bullet (M-2N)}{1 \bullet 2 \bullet 3 \bullet \cdots \bullet (2N+1)}$$

4. The abstract data type priority queues can be implemented by partially ordered trees. (18%)

(a) Use the linked list to implement the partially ordered tree and give an algorithm for the DELETEMAX operation (delete the maximum-priority element).(4%)

(b) What is the time complexity of your DELETEMAX algorithm? (5%)

(c) Show the partially ordered tree if the elements with priority numbers 3, 7, 5, 8, 2, 1, 6 are inserted into an empty tree. (4%)

(d) What is the result of two successive DELETEMAX operations on the tree from (c). (5%)

二、Algorithms (50%)

1. (18%) Answer each part TRUE or FALSE for the big O notation.
   a) $2n=O(n)$.
   b) $n^2 = O(n)$.
   c) $n^2 = O(n\log^2 n)$.
   d) $n\log n = O(n^2)$.
   e) $3^n = 2^{O(n)}$.
   f) $2^n = O(2^n)$.

2. (15%) A product of matrices is *fully parenthesized* if it is either a single matrix or the product of two matrix products, surrounded by parentheses. The *matrix-chain multiplication problem* can be stated as follows: given a chain $\langle A_1, A_2, \ldots, A_n \rangle$ of matrices, where for $i = 1,2,\ldots,n$, matrix A_i has dimension $p_{i-1} \times p_i$, fully parenthesize the product $A_1, A_2, \ldots, A_n$ in a way that minimizes the number of scalar multiplications. Please **solve the matrix-chain multiplication problem** and **also compute the minimum number of scalar multiplications** on a chain $\langle A_1, A_2, \ldots, A_6 \rangle$ in which the dimensions of $A_1, A_2, \ldots, A_6$ are $30\times 35, 35\times 15, 15\times 5, 5\times 10, 10\times 20, 20\times 25$, respectively.

3. (12%) Give asymptotic tight bound for $T(n) = T(\frac{n}{2}) + T(\frac{n}{4}) + T(\frac{n}{8}) + n$ (assume that $T(n)$ is constant for sufficiently small n).

4. (5%) The *0-1 knapsack problem* is posed as follows: A thief robbing a store finds n items; the ith item is worth v_i dollars and weighs w_i pounds, where v_i and w_i are integers. He wants to take as valuable a load as possible, but he can carry at most W pounds in his knapsack for some integer W. Which items should he take ? (This is called the 0-1 knapsack problem because each item must either be taken or left behind; the thief cannot take a fractional amount of an item or take an item more than once)

   In the *fractional 0-1 knapsack problem,* the setup is the same, but the thief can take fractions of items, rather than having to make a binary (0-1) choice for each item.

   Please answer which problem can be solved using the greedy strategy?