There are 125 points total. (At 5 pts. = 1%, the first exam is 20% and this is 25%.)

1. (30 pts.) The number of binary trees of size n is given by $b_1 = 1$ and by the recursion

$$b_n = b_1 b_{n-1} + b_2 b_{n-2} + b_3 b_{n-3} + \cdots + b_{n-1} b_1 \qquad \text{for } n > 1.$$

(a) In some sort of pseudocode or code, write a dynamic program to compute b_n.

Ans. This procedure uses an array `b[]` to store the values and the array is passed to the procedure. Your array might be internal and the procedure would simply return the desired `b` value.

```
void B(int n, int *b) {
    int m, j;
    b[1] = 1;
    for(m=2; m<=n; ++m) {    // loop computes b[m]
        b[m] = 0;
        for (j=1; j<m; ++j) b[m] = b[m] + b[i]*b[m-i];
    }
}
```

(b) Analyze your algorithm to determine the complexity category of its running time. (Show your work.)

Ans. The important time is the execution of the inner "`for`" loop. it is executed $\sum_{m=2}^{n}(m-1) \in \Theta(n^2)$ times. Thus the time complexity category of the algorithm is $\Theta(n^2)$.

2. (35 pts.) Indicate whether true or false. Beware of guessing:

correct answer +5pts. incorrect answer −3pts. no answer 0pts

(a) F Greedy algorithms are called "greedy" because they often take a lot of time.

(b) T Usually it is harder to prove that a greedy algorithm is correct than it is to prove that a divide and conquer algorithm is correct.

(c) F There is a good greedy algorithm for the 0-1 Knapsack Problem. [There is NO greedy algorithm.]

(d) T It is usually very difficult to determine the complexity category of the average running time of a backtracking algorithm.

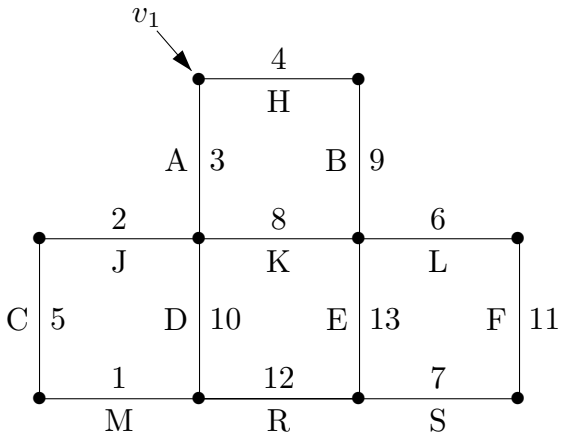
(e) T Kruskal's algorithm (choose best non-cycle edge) is better than Prim's (choose best tree edge) when the graph has relatively few edges.

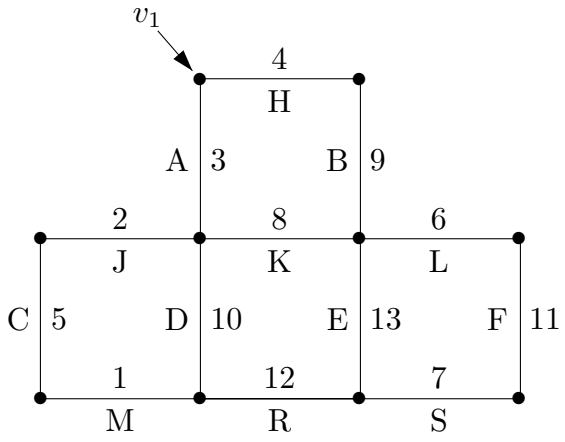
(f) F Since the recursion in Problem 1 above is linear, it can be solved by the method for linear equations in Appendix B.

(g) F Dynamic programming uses a top-down approach.

MORE

There are 2 copies of the graph so you'll have a spare.
Edges are labeled with upper case letters.





3. (20 pts.) Recall that Kruskal's algorithm greedily adds edges in a way that avoids cycles. For the graph shown above, list the edges in the order chosen by Kruskal's algorithm.

Ans. M, J, A, H, C, L, S, K, F

4. (20 pts.) Recall that Dijkstra's algorithm finds the shortest path from v_1 to all other vertices by adding edges that make shortest paths. For the graph shown above, each edge is bidirectional; that is, you can travel on either direction on it for the same cost. List the edges in the order chosen by Dijkstra's algorithm.

Ans. A, H, J, C, (K, M), L, R, F
K and M can be added in either order since there is a tie.

5. (20 pts.) An algorithm can sometimes be made faster by computing additional data; that is, data that was not asked for in the problem.

Three algorithms you have studied that do this are: (i) Prim's algorithm for minimum spanning trees, (ii) Dijstra's algorithm for single-source shortest paths, and (iii) an algorithm maximum sum of contiguous sublists.

Pick any **ONE** of these, indicate which you picked, and describe additional data that is computed. (If you do more than one, only the first will be graded.)

Ans. (i) The important additional data are, for each point v, the length of the shortest edge from v to points already in the tree.

Ans. (ii) If the source is v_1, one keeps track, for each vertex x, of the shortest path from v_1 to x using just one edge not already in the paths.

Ans. (iii) One keeps track of the largest contiguous sublist sum where the sublist must end just after the current point we're looking at in the list.

END