Final Exam Name_____ ID_____

It's a closed book and notes exam. You have 120 minutes.

Question	Points	
1	6	
2	6	
3	8	
4	10	
5	20	
6	10	
7	20	
8	20	
Total:	100	

1. (6 points)

Indicate whether each of the following is true or false.

a) Top-down dynamic programming without table look-up is faster than bottom-up dynamic programming with table look-up because top-down avoids building the table altogether.

b) Recall the divide-and-conquer algorithm for selecting the k^{th} smallest of n elements. Suppose that we use groups of size 3 instead of size 5. This also leads to a linear time algorithm.

c) Depth-first-search assigns pre and post numbers for all vertices in a directed graph. Suppose pre(x) = 2, post(x) = 7, pre(y) = 3, and post(y) = 4, then the edge from y to x is a forward edge.

2. The quicksort algorithm is one of the most frequently used sorting algorithms. It provides highly efficient sorting and simple implementation logic. The essence of quicksort is to choose a pivot value, and partition the input array into 2 sub-arrays, with one sub-array containing values greater than the pivot value, and the other sub-array containing values less than the pivot value. Then quicksort is called recursively on these 2 smaller sub-arrays, until every element is sorted.

The choice of pivot values has great impact on partition layout, thus affects the overall efficiency of the quicksort algorithm.

A few methods for choosing pivot values are given below. Please give a brief remark on the pros & cons of them. (6 points)

a) Choose the pivot value from fixed positions, such as the first element in the input array.

b) Find the first, middle, and last element in the input array, and use median value of these 3 numbers as the pivot value. (NOTE: when there are 2k elements in the input array, either the k-th or the (k+1)-th element could be used as the middle element)

c) Use the SELECT algorithm discussed in course (and also in text book "Introduction to Algorithms"), select the exact median value of the input array as the pivot value, in O(n) time.

3. Running Time (8 points)

1) Given a directed Graph *G=(V,E), |V| = n, and |E| = m*. If m = nlogn, what is the running time of Dijkstra's algorithm using binary heap implementation? How about using array implementation?

2) During the running of the procedure RANDOMIZED-QUICKSORT, how many calls are made to the random-number generator RANDOM in the worst case? How about in the best case? Give your answer in terms of θ notation.

4. (10 points)

Give a tight asymptotic upper bound for the following two recurrences below, solve (a) using Master Theorem, and solve (b) using recursion tree method.

a） T(n) = 2T(n/8) + $\sqrt[3]{n}$ (note:$\log_8^2 = 1/3$)

b） T(n) = T(2n/3) + n T(1) = 1

(note: $\sum_{k=0}^{n} x^k = \frac{1-x^{n+1}}{1-x}$)

5. (20 points)

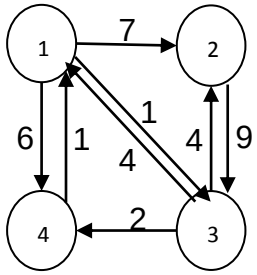
An array A[1…n] is said to have a majority element if more than half of its entries are the same. Given an array, the task is to design an efficient algorithm to tell whether the array has a majority element, and, if so, to find that element. Design a divide-and-conquer algorithm to solve this problem, your answer should include:

1) a general description of your idea;

2) pesudocode of your algorithm;

3) running time analysis of your algorithm.

6. (10 points)



Given a directed graph as above, use FLOYD-WARSHALL algorithm (see below for your reference) to computer the shortest path between any pair of vertices in the graph. Please show the values of $D^{(0)}$, $D^{(1)}$, $D^{(2)}$, $D^{(3)}$, and $D^{(4)}$.

```
FLOYD-WARSHALL(W)
1 n ←rows[W]
2 D(0) ← W
3 for k ← 1 to n
4       do for i ← 1 to n
5             do for j ← 1 to n
6                   do d_ij^k ← min(d_ij^(k−1), d_ik^(k−1) + d_kj^(k−1))
7 return D(n)
```

7. Greedy Algorithm (20 points)

A server has *n* customers waiting to be served. The service time required by each customer is known in advance: it is t_i minutes for customer *i*. So if, for example, the customers are served in order of increasing *i*, then the i^{th} customer has to wait $\sum_{j=1}^{i} t_j$ minutes.

We wish to minimize the total waiting time

$$\mathrm{T} = \sum_{i=1}^{n} (time\ spent\ waiting\ by\ customer\ i)$$

Give an efficient algorithm for computing the optimal order in which to process the customers, and justify your algorithm is correct.

8. Dynamic Programming (20 points)

.Let $S = \{a_1, a_2, \ldots, a_n\}$ be a set of n positive integers and let k be an integer. Give an *O(kn)*-time bottom-up dynamic programming algorithm to decide if there is a subset *U* of *S* that $\sum_{a_i \in U} a_i = k$. Your algorithm should return T (for 'true') if *U* exists and *F* (for 'false') otherwise.

Your answer should include:

1) the recurrence relation and a clear justification for it (merely writing down a recurrence won't do).

2) pseudocode for the algorithm (including computing information that allows retrieval of the subset, if necessary).

(blank)