装 订 线

任课教师：

装 订 线

学号：

装 订 线

姓名：

装 订 线

班级：

西 安 电 子 科 技 大 学

考试时间 120 分钟

试　　题

题号	一	二	三	四	五	六	七	总分
分数	16	20	12	14	16	14	8	
得分								

1. 考试形式：闭卷☑ 开卷☐

2. 考试日期：　　年　　月　　日（答案直接答在试卷上，不要超出装订线）

1. Answer T/F for the following: (2 * 8 points)

(1) $5n^2 + 3n\lg n + 1024 \in O(n^3)$

(2) $5n^2 + 3n\lg n + 1024 \in \Omega(n^3)$

(3) $5n^2 + 3n\lg n + 1024 \in \Theta(n^2)$

(4) Insertion-Sort will run in worst-case if the input array is in reverse sorted order-that is, in decreasing order.

(5) Quick-Sort always runs faster than Insertion-Sort for any input of size n.

(6) When solving Matrix-Chain-Multiplication problem by Dynamic Programming strategy, the recursive algorithm without memoization will run in $O(n^3)$ time, although overlapping-subproblems exist.

(7) Dijkstra's algorithm can be used to calculate the single-source shortest paths if negative weight edges exist.

(8) TSP(Traveling Salesman Problem) is a NP-Complete problem.

2．Single Choice (2*10 points)

(1) The worst-case running time of Quick Sort is (　　)

A. $\Theta(n^2)$　　B. $\Theta(n\lg n)$　　C. $\Theta(n)$　　D. $\Theta(n^3)$

(2) Which of following array forms a min-heap (　　)

A. 3　5　9　4　8　10　12　　B. 3　5　9　7　8　10　12

C. 3　7　9　6　8　5　12　　D. 3　5　10　8　7　9　12

(3) When using max-heap to implement a Max-Priority queue, all operations on an *n*-element heap, including INSERT(S, x)、MAXIMUM(S)、EXTRACT-MAX(S) and INCREASE-KEY(S, x, k), take time of (　　)

A. $\Theta(n)$　　B. $\Theta(n\lg n)$　　C. $O(\lg n)$　　D. $O(1)$

(4) In the 3 main steps of Divide and Conquer strategy, which step usually results in

recursive fashion ( )

A. Divide B. Conquer C. Combine D. None of the above

(5) Which of the following sorting algorithm is NOT stable ( )

A. Insertion Sort B. Merge Sort

C. Heap Sort D. Counting Sort

(6) If the input is generated by a random process that distributes elements uniformly, which algorithm can finish sorting in Θ(n) time ( )

A. Quick Sort B. Counting Sort

C. Radix Sort D. Bucket Sort

(7) Which design strategy is used in Assembly-Line Scheduling problem efficiently( )

A. Divide and Conquer B. Dynamic Programming

C. Greedy D. Brute Force

(8) Which design strategy is used in Huffman codes efficiently ( )

A. Divide and Conquer B. Dynamic Programming

C. Greedy D. Brute Force

(9) In the DP recursive equation used for 0/1 knapsack problem, c*[i,j]* represents the maximum value of the objects that fit in the knapsack, selecting objects from 1 through *i* with the sack's weight capacity equal to *j*. If the weight of the 4^{th} object is 6, and its value is 20, you are given c[3, 4] = 7, c[3, 10] = 25, then c[4, 10] = ( )

A. 7 B. 20 C. 25 D. 27

(10) In the Back-Tracking paradigm, we generate problem states in ( ) way.

A. Depth First Search B. Breadth First Search

C. D-Search D. None of the above

3. Evaluate the following recursions using the Master Method (4 * 3 points)

(1) $T(n) = 4T(n/2) + 1$

(2) $T(n) = 2T(n/2) + n\lg n$

(3) T(n) = 2T(n/4) + n

4. Divide and Conquer Strategy (14 points)

(1) (6 points) In the general method of Divide-and-Conquer strategy, assume we have the following procedures:

--- Divide(…) : Divide an original problem into smaller subproblems;

--- Combine(…) : combine the solutions to subproblems into the solution for the original problem.

Fill in the procedure names for the blanks (a)~(c) in the pseudo code skeleton DC(…) for a general Divide-and-Conquer algorithm:

```
DC (…){
    if (problem size is small enough)
        solve the problem directly and return;

    (a) _____(…)
    for each sub_problem
        (b) _____(…)
    (c) _____(…)
}
```

(2) (8 points) Using Divide-and-Conquer, we can obtain a merge-sort algorithm, assume we have procedure MERGE (A,p,q,r) which merges sorted A[p..q] with sorted A[q+1..r], write down the pseudo code for MergeSort(A,p,r) to sort A[p..r].

5. Dynamic Programming Strategy(16 points)

(1) (6 points) Describe the mainly steps of Dynamic Programming strategy for solving optimal problems.

(2) (4 points) Describe the difference of 'the value of optimal solution' and 'optimal solution' in Dynamic Programming algorithms, give their exact meanings in the Floyd-Warshall algorithm for all-pairs shortest paths.

(3) (6 points) Given the optimal substructure used in Matrix Chain Multiplication problem's DP algorithm (Write down the recursive equation and explain it). What should we do to construct the optimal solution when calculating the minimum number of scalar multiplications?

6. Calculations (14 points)

(1) (5 points) Given the following integers in array A, find the maximum subarray (write down the calculating steps, and the max sum and subarray respectively).

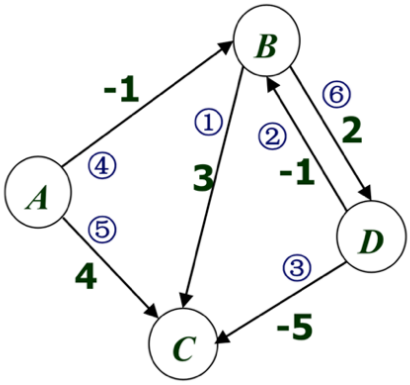
A = (13, -18, 20, -5, 8, 12, -15, -22, 25, -4)

(2) (4 points) Given the following set A of activities with start and finish times (s_i, f_i), $1 \le i \le n$, select a maximal set S of "non-overlapping" activities (the activities were already sorted initially by their finish times) .

Index	1	2	3	4	5	6	7	8	9	10
s_i	1	4	2	5	3	5	7	9	2	12
f_i	4	5	6	7	8	9	10	11	12	15

(3) (5 points) Given the following directed graph G, fill in the blanks (a)~(e) in calculating the lengths of single-source shortest paths using Bellman-Ford algorithm. Source vertex is A and the order of edge relaxation is marked by ①~⑥, initial value d[A] is set to 0.

Edge(u,v)	①	②	③	④	⑤	⑥
Target v	C	B	C	B	C	D
Initial(d[v])	∞	∞	∞	∞	∞	∞
Pass 1(d[v])	∞	∞	∞	-1	*(a)*	*(b)*
Pass 2(d[v])	*(c)*	-1	*(d)*	-1	-4	1
Pass 3(d[v])	-4	*(e)*	-4	-1	-4	1



7. Algorithm Design(8 points)

There is a row of *n* houses, where each house can be painted one of three colors: red, blue, or green. The cost of painting each house with a certain color is different. You have to paint all the houses such that no two adjacent houses have the same color.

The cost of painting each house with a certain color is represented by an n*3 cost matrix. For example, costs[1][1] is the cost of painting house 1 with the color red; costs[2][3] is the cost of painting house 2 with color green, and so on...

Design an algorithm to find the minimum cost to paint all houses.

For Example, n=3:

Input: costs = [[17,2,17],
[16,16,5],
[14,3,19]]

Minimum cost should be : 10

By painting house 1 into blue, paint house 2 into green, paint house 3 into blue, we can get minimum cost: 2 + 5 + 3 = 10.