1. (10 points) True or false?

(a) If \( f(n) = O(g(n)) \), then \( f(n) \leq g(n) \)

False

(b) A and B are two sorting algorithms. If A is \( O(n^2) \) and B is \( O(n) \), then for an input of X integers, B can sort it faster than A.

False

(c) \( af(n) = bf(n) + c \) is a recurrence equation for \( f(n) \) (where a, b and c are constants).

False

(d) If \( f(n) = o(g(n)) \), then \( f(n) = O(g(n)) \).

True

(e) If \( 1 < a < b \) and \( f(n) = O(n^a) \), then \( f(n) = O(n^b) \)

True

2. (15 points). Prove that
\[
\log_a n = O(\log_b n)
\]
for any real numbers \( a > 1 \) and \( b > 1 \).

\[
\log_a n = \frac{\log_b n}{\log_b a}, \text{ let } c = \max\{1, \frac{1}{\log_b a}\}, \text{ and } n_0 = 1
\]
then for any \( n \geq n_0 \)
we have \( \log_a n \leq c \log_b n \)

3. (20 points) Given \( n \) segments of line with coordinates \([L_i, R_i]\) (i=1,2,..,n). You are asked to choose the minimal number of them to completely cover the range \([0, M]\). Describe a greedy algorithm that can be used to solve this problem. Prove the correctness of your algorithm.
Idea: We will start to cover the region \([0,M]\) from left to right. Let \(s\) be the left end of the region that we need to cover. At the beginning, \(s=0\). Find the group of segments whose left coordinate is \(\leq s\), sort them based on their coordinate \((R_i)\) and pick the segment (say \(k\)) whose right coordinate is the largest in the group. Update \(s=R_k\). Repeat the process until \(s\geq M\).

Algorithm
Input: A set \(S\) of segments with coordinates \([L_i,R_i]\), range \([0, M]\).
Output: A set \(O\) of segments that covers range \([0, M]\), such that the size of \(O\) is minimized.

Begin:
\[
\text{int } s=0; \quad \text{// the left end of the region needed to be covered} \\
\text{while } (s<M) \{ \\
\quad \text{int max}=0; \\
\quad \text{int } j=-1; \quad \text{//index of the current candidate} \\
\quad \text{for each segment } i \text{ in } S \{ \\
\quad \quad \text{if } (L_i \leq s \text{ and } R_i \geq \text{max}) \{ \\
\quad \quad \quad \text{max}= R_i; \\
\quad \quad \quad j=i \\
\quad \quad \} \\
\quad \text{Put segment } j \text{ in set } O; \\
\quad \text{remove segment } j \text{ from } S; \\
\quad s=R_j; \\
\}\}
\]

Proof:
Let \(O\) be the set returned by the greedy method.
Assume that \(O\) is not the optimal solution. Then there exists a set \((U)\) of segments that can cover the region \([0,M]\) such that \(|U|<|O|\). Let \(c=|U|\), and \(k=|O|\), then \(c<k\).

Let \(O_1, O_2, \ldots, O_k\) be the segments in \(O\), in the order that they are chosen by the greedy algorithm. Then, \(R_{O_1} < R_{O_2} < \ldots < R_{O_{k-1}} < R_{O_k}\). From the greedy algorithm we have that \(R_{O_{k-1}} < M\).

Let’s sort the segments in \(U\) based on their right coordinates. Let them be \(U_1, U_2, \ldots, U_c\). Then, \(R_{U_1} < R_{U_2} < R_{U_3} < \ldots < R_{U_c}\).

According to the greedy method we have that \(R_{U_1} \leq R_{O_1}; \) 
\(R_{U_2} \leq R_{O_2}; \ldots \) \(R_{U_c} \leq R_{O_c}; \)
Since $c < k$, then, $R_{U_c} \leq R_{O_c} \leq R_{O_k} < M$, that mean $U_1, U_2, ..., U_c$ can not cover the whole region of $[0,M]$. Thus, $U$ is not a solution. The assumption is not correct. Thus, $O$ is the optimal solution.
4. (20 points) Consider the chain product of four matrixes with the following dimensions:

\[ A_0 : 10 \times 2 \quad A_1 : 2 \times 5 \quad A_2 : 5 \times 3 \quad A_3 : 3 \times 6 \]

Finish the scoring \((N_{i,j})\) and traceback \((T_{i,j})\) tables, and write down the parenthesization that has the minimal operations.

\[
N_{ij}
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
0 & & & \\
1 & & & \\
2 & & & \\
3 & & & \\
\end{array}
\]

\[
T_{ij}
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
0 & & & \\
1 & & & \\
2 & & & \\
3 & & & \\
\end{array}
\]

\[
A_0 \times ((A_1 \times A_2) \times A_3)
\]
5. (15 points) Following is the scoring (and traceback) table for finding the longest common subsequence between sequences “BDCABA” and “ABCBDAB”. Finish the table and write down the optimal alignment that gives the longest common subsequence (if there are more than one optimal alignments, you only need to give one).

\[
\begin{array}{cccccc}
& B & D & C & A & B & A \\
A & 0 & 0 & 0 & 0 & 0 & 0 \\
B & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
C & 0 & 1 & 1 & 1 & 2 & 2 & 2 \\
D & 0 & 1 & 1 & 2 & 2 & 2 & 2 \\
A & 0 & 1 & 2 & 2 & 2 & 3 & 3 \\
B & 0 & 1 & 2 & 2 & 3 & 3 & 4 \\
\end{array}
\]

The arrows used here: 

\[
\begin{array}{c}
\text{□} = \Rightarrow \\
\text{□} = \rightarrow \\
\text{□} = \downarrow \\
\end{array}
\]

\[
\begin{align*}
M_{5,6} &= \max(M_{4,5} + 0 = 3, M_{5,5} = 3, M_{4,6} = 3) = 3 \\
M_{6,5} &= \max(M_{5,4} + 0 = 2, M_{5,5} = 3, M_{6,4} = 3) = 3 \\
M_{7,4} &= \max(M_{6,3} + 0 = 2, M_{7,3} = 2, M_{6,4} = 3) = 3 \\
M_{6,6} &= \max(M_{5,5} + 1 = 4, M_{5,6} = 3, M_{6,5} = 3) = 4 \\
M_{7,5} &= \max(M_{6,4} + 1 = 4, M_{6,5} = 3, M_{7,4} = 3) = 4 \\
M_{7,6} &= \max(M_{6,5} + 0 = 3, M_{7,5} = 4, M_{6,6} = 4) = 4
\end{align*}
\]

A B _ C _ B D A B

_ B D C A B _ A _
**Master Theorem**

The Master Theorem applies to recurrences of the following form:

\[ T(n) = aT(n/b) + f(n) \]

where \( a \geq 1 \) and \( b > 1 \) are constants and \( f(n) \) is an asymptotically positive function.

There are 3 cases:

1. If there is a constant \( \varepsilon > 0 \) such that \( f(n) = O(n^{\log_b a - \varepsilon}) \), then \( T(n) = \Theta(n^{\log_b a}) \).

2. If there is a constant \( k \geq 0 \) such that \( f(n) = \Theta(n^{\log_b a} \log^k n) \) with \( k \geq 0 \), then \( T(n) = \Theta(n^{\log_b a} \log^{k+1} n) \).

3. If there are small constants \( \varepsilon > 0 \) and \( \delta < 1 \) such that \( af(n/b) \leq \delta f(n) \), for \( n \geq d \), then \( T(n) = \Theta(f(n)) \).

6. (20 points) Use the master theorem to solve the following recurrence equations

(a) \( T(n) = 4T(n/2) + n^2 \)

\[ n^{\log_b a} = n \log \frac{1}{2} = n^2 \], \( f(n) = n^2 \). Let \( k = 0 \), then \( f(n) = \Theta(n^2 \log^k n) \). Case 2 applies. Thus, \( T(n) = \Theta(n^2 \log n) \)

(b) \( T(n) = 4T(n/2) + n \)

\[ n^{\log_b a} = n \log \frac{1}{2} = n \], \( f(n) = n \). Let \( \varepsilon = 0.5 \), is then \( f(n) = O(n^{2-\varepsilon}) \). Case 1 applies. Thus, \( T(n) = \Theta(n^2) \)

(c) \( T(n) = 4T(n/2) + n^3 \)

\[ n^{\log_b a} = n \log \frac{1}{2} = n^2 \], \( f(n) = n^3 \). Let \( \varepsilon = 0.5 \), then \( f(n) = \Omega(n^{2+\varepsilon}) \).

\[ af(n/b) = 4f(n/2) = 4((n/2)^3) = n^3 / 2 \]

Let, then \( \delta = 1/2 \) \( af(n/b) = n^3 / 2 \leq \delta f(n) \) for all \( n \geq 1 \). Case 3 applies. Thus, \( T(n) = \Theta(n^3) \)