NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

SEMESTER 2 (2009/2010)

EXAMINATION FOR

CS3230: Design and Analysis of Algorithms

May 2010 **Time Allowed: 2 Hours**

INSTRUCTIONS TO CANDIDATES

1. This examination paper contains **NINE (9)** questions and comprises **EIGHT (8)** printed pages, including this page.
2. Answer **ALL** questions within the space in this booklet.
3. This is an **Open Book** examination.
4. Please write your Matriculation Number below.

MATRICULATION NO: _____

This portion is for examiner's use only

Question	Marks	Score
Problem 1	5	
Problem 2	10	
Problem 3	4	
Problem 4	4	
Problem 5	4	
Problem 6	5	
Problem 7	5	
Problem 8	5	
Problem 9	8	
Total	50	

Problem 2. (10 marks)
Solve the following recurrence relations and write the asymptotic tight bounds for T(n). **No need to write out the derivation process or prove your result.** In all cases below, T(x) denotes T(floor of x) if x is not an integer. In all cases below, the recurrence relation is for $n > 10$. For $n \leq 10$: $T(0) = T(1) = \ldots = T(10) = 1$. In the last recurrence relation, sqrt(n) stands for the square root of n.

$T(n) = T(n-4) + 1$: $T(n) =$ _____

$T(n) = T(n/10) + n^{10}$: $T(n) =$ _____

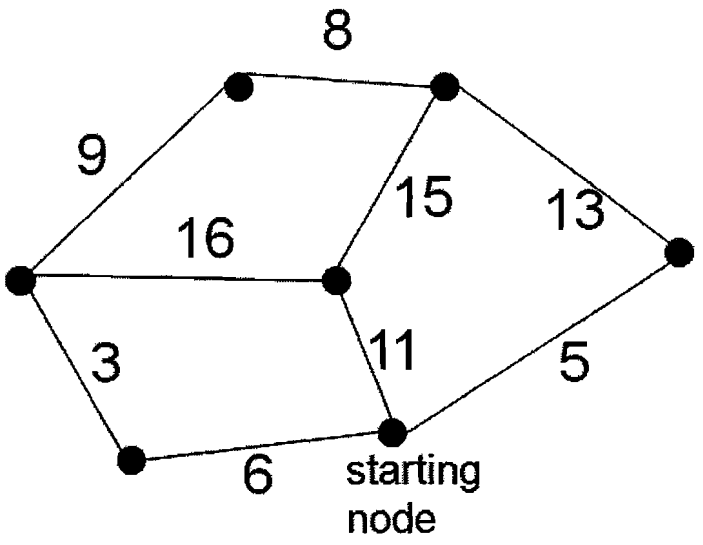
$T(n) = T(n/10) + n$: $T(n) =$ _____

$T(n) = 4T(n/2) + n^2$: $T(n) =$ _____

$T(n) = T(\text{sqrt}(n)) + 1$: $T(n) =$ _____

Problem 3. (4 marks)
Determine the optimal Huffman encoding tree for the following frequency counts: 6, 9, 10, 27, 28, 35, 50. Draw your optimal Huffman encoding tree below:

Problem 4. (4 marks)



Imagine that you apply Prim's algorithm to determine the minimum spanning tree in the above graph, **with the given starting node in the graph**. The number beside each edge indicates the weight of the edge. Write out the edges output by Prim's algorithm one by one:

The weight of 1st edge output the algorithm: _____

The weight of 2nd edge output the algorithm: _____

The weight of 3rd edge output the algorithm: _____

The weight of 4th edge output the algorithm: _____

The weight of 5th edge output the algorithm: _____

The weight of 6th edge output the algorithm: _____

Problem 5 (4 marks)
Consider coins with denominations of 1, 4, 12, and 19. How will you make change for a total amount of 30 while minimizing the total number of coins needed? Write your answer below. (As an example, if your answer is to use seven 4-cent coins, and two 1-cent coins, then you should write 30 = 4+4+4+4+4+4+4+1+1 as your answer. No need to explain why your answer minimizes the total number of coins used.)

Your answer: _____

Problem 6. (5 marks)
You are given an array A with n integers. For each integer in array A, define its occurrence count as the total number of elements in A that equal the given integer. For example, the occurrence count for 3 in the array [5, 3, 5, 171, 3, 3, 89, 1, 3] is 4 (i.e., it appears 4 times). Design an algorithm to find the element with the largest occurrence count in array A. (If there are multiple such elements, your algorithm may output any one of them.) Your algorithm should have worst-case time complexity of O(nlog(n)) and worst-case space complexity of O(n). In your answer, you should first give a high-level description of the ideas behind your algorithm, then provide concise pseudo-code, and finally succinctly explain its time/space complexity.

Problem 7. (5 points)
You are given a string S[1..n] of n characters, and you need to determine whether this string corresponds to the concatenation of some valid English words. For example, the string "helikesalgorithms" corresponds to the concatenation of three English words "he", "likes", "algorithms". The function boolean LookUpDictionary(int i, int j) is available for you to determine whether the substring S[i..j] is a **single** valid English word. Assume that this function takes $\Theta(1)$ time for each invocation in the worst-case (regardless of the value of i and j).

Design an algorithm to solve this problem (i.e., determine whether S[1..n] corresponds to the concatenation of some valid English words) using dynamic programming. Your algorithm's worst-case time complexity must be $O(n^2)$. In your answer, you should first derive the recurrence relation used for dynamic programming, then provide concise pseudo-code, and finally succinctly explain its time complexity.

Problem 8. (5 points)
In the VertexCover decision problem, we are given an undirected graph G and a positive integer k. The problem needs to determine whether G contains a set W ($|W| \leq k$) of vertices such that each edge in G is incident on some vertex in W (i.e., for each edge in G, at least one endpoint of that edge is in W). It is known that the VertexCover problem is NP-complete.

Now you are given a set F of subsets of a finite set S, and a positive integer b. The problem is to determine whether S has a subset H such that $|H| \leq b$ and H intersects with every element in F. For example, if $S = \{1, 2, 3, 4, 5, 6\}$, $F = \{\{1, 2, 3, 4\}, \{3, 4, 5\}, \{2, 3, 6\}, \{1, 4, 6\}, \{2\}\}$ and $b = 2$, then H does exist (e.g., H can be $\{2, 4\}$). Prove that this problem is NP-complete. (Hint: Try to reduce VertexCover to this problem.)

Problem 9. (8 points)
You are given a set F of subsets of a finite set S. The problem is to determine whether S can be partitioned into two subsets S_1 and S_2, such that none of the elements in F is entirely contained in S_1 or S_2. "Partition" here means that S_1 and S_2 do not intersect, and their union is exactly S. As an example, if $S = \{1, 2, 3, 4, 5, 6\}$ and $F = \{\{1, 2\}, \{3, 4, 5\}, \{2, 3, 6\}, \{1, 4, 6\}, \{2, 5\}\}$, then the answer is yes (e.g., with $S_1 = \{1, 3, 5\}$ and $S_2 = \{2, 4, 6\}$). Can you design an efficient algorithm to solve this problem (i.e., determine whether S can be partitioned into S_1 and S_2 with the above property)?

If you can, indicate "YES" first, and then design a deterministic algorithm. You algorithm's worst-case time complexity and space complexity must be both polynomial. You should first give a high-level description of the ideas behind your algorithm, then provide concise pseudo-code, and finally succinctly explain the time/space complexity.

If you feel that this cannot be done, indicate "NO" first, and then prove that the problem is NP-hard.

Note that just indicating "YES" or "NO" will not earn you any mark.

Write your answer here:

(This page is intentionally left blank for you to write the answer for Problem 9.)

END OF EXAM PAPER