Midterm

Note

This is a closed-book exam. Each problem accounts for 10 points, unless otherwise marked.

Problems

1. Prove by induction that a ring of even size can be colored with two colors and a ring of odd size with three colors such that no two adjacent nodes have the same color.

2. Construct a gray code (which is open) of length $\lceil \log_2 13 \rceil$ $(= 4)$ for 13 objects. Show how the gray code is constructed *systematically* from gray codes of smaller lengths.

3. Solve the following recurrence relation with full history:

$$\begin{cases} T(1) = 0 \\ T(n) = (n-1) + \sum_{i=1}^{n-1} T(i), \quad n \geq 2 \end{cases}$$

   You need to provide an exact, not just asymptotic, solution.

4. Find the asymptotic behavior of the function $T(n)$ defined as follows:

$$\begin{cases} T(1) = 1 \\ T(n) = 2\sqrt{n} + T(n/2), \quad n = 2^i \ (i \geq 1) \end{cases}$$

   You should try to solve this problem without resorting to the general theorem for divide-and-conquer relations discussed in class. The asymptotic bound should be as tight as possible. (Hint: guess and prove by induction.)

5. Consider binary trees where each node stores a non-negative integer. Design an algorithm that, given such a tree T and a non-negative integer k as input, determines whether T contains a branch (from the root to a leaf) such that the sum of all numbers stored on the nodes of the branch equals k. The more efficient your algorithm is, the more points you will be credited for this problem. Is there a possibility that your code may overflow? Have you avoided the problem?

6. Show all intermediate and the final AVL trees formed by inserting the numbers 4, 6, 5, 8, 9, 7, 1, 3, and 2 (in this order) into an empty tree. Please use the following ordering convention: the key of an internal node is larger than that of its left child and smaller than that of its right child. If a rotation is performed during an insertion, please also show the tree before the rotation. (15 points)

7. Rearrange the following array into a (max) heap using the bottom-up approach.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
3	5	2	8	9	14	7	6	4	1	13	10	15	11	12

Show the result after each element is added to the part of array that already satisfies the heap property. (15 points)

8. Suppose that you are given an algorithm as a *black box* (you cannot see how it is designed) that has the following properties: If you input any sequence of real numbers and an integer k, the algorithm will answer "yes" or "no," indicating whether there is a subset of the numbers whose sum is exactly k. Show how to use this black box to find the subset whose sum is k, if it exists. You should use the black box $O(n)$ times (where n is the size of the sequence).

9. A string $A = a_0a_1 \cdots a_{n-1}$ is a cyclic shift of another string $B = b_0b_1 \cdots b_{n-1}$ if there exists an index k, $0 \leq k \leq n-1$, such that $a_i = b_{(k+i) \bmod n}$ for all i, $0 \leq i \leq n-1$. For example, "`defgabc`" is a cyclic shift of "`abcdefg`". Suppose that you already have an algorithm called **Substring** that can determine whether a string is a substring of another. Design an algorithm using **Substring** that, given two strings A and B, determines whether A is a cyclic shift of B. Please present your algorithm in an adequate pseudo code and make assumptions wherever you feel necessary. Explain why your algorithm is correct and analyze its time complexity. The more efficient your algorithm is, the more points you will be credited for this problem.