1. Suppose you are given an unsorted array \( A[1..n] \), which contains all but one of the \( n + 1 \) integers in the range \( 0, \ldots, n \) (so exactly one of these elements is missing from \( A \)). To simplify the problem somewhat, we will assume that \( n = 2^k - 1 \) for some integer \( k \). Hence each array element has a binary representation using \( k \) bits.

You want to determine the missing integer. You are not allowed to access an entire integer in \( A \) with a single operation. The only way to access the elements of \( A \) is by calling the function \( \text{bitvalue}(i, j) \), which returns the value of the \( j \)th bit of \( A[i] \).

Give a divide-and-conquer algorithm that finds the missing integer and makes only \( O(n) \) calls to the function \( \text{bitvalue}() \).

**Note:** There are \((n - 1) \log n \) bits, so you cannot afford to look at every bit.

2. Give asymptotic solutions (using \( \Theta() \) notation) for each of the following recurrence equations.

   (a) \( T(n) = 6T(n/4) + n \log n \)
   (b) \( T(n) = 2T(n/4) + \sqrt{n} \)
   (c) \( T(n) = 6T(n/3) + n^2 \)