1. Suppose you are given an unsorted array $A[1..n]$, which contains all but one of the integers in the range $0, \ldots, n$ (so exactly one of these elements is missing from $A$). The problem is to determine the missing integer in $O(n)$ time. Each element of $A$ is represented in binary, and the only operation available is the function $\text{bit}(i, j)$, which returns the value of the $j$th bit of $A[i]$ and takes constant time. Show that using only this operation, it is still possible to determine the missing integer in $O(n)$ time, by giving a divide-and-conquer algorithm. Be sure to explicitly state and then solve the recurrence equation for the running time of your algorithm. (It may simplify your answer if you assume that $n$ is of the form $2^k - 1$ for some integer $k$; if so, you are free to make this assumption.)

2. Suppose we could multiply two $3 \times 3$ matrices using 25 scalar multiplications and a constant number of scalar additions and subtractions. Set up and solve the recurrence relations to analyze the resulting divide-and-conquer algorithm for matrix multiplication.

3. Suppose we could multiply two $3 \times 3$ matrices using $r$ scalar multiplications and a constant number of scalar additions and subtractions. How small would $r$ have to be to make the resulting divide-and-conquer algorithm for matrix multiplication asymptotically faster than Strassen’s matrix multiplication algorithm? Justify your answer.