1. (45 pts total) Professor Snape has n computer chips that are supposedly both identical and capable of testing each other's correctness. Snape's test apparatus can hold two chips at a time. When it is loaded, each chip tests the other and reports whether it is good or bad. A good chip always reports accurately whether the other chip is good or bad, but the answer of a bad chip cannot be trusted. Thus, the four possible outcomes of a test are as follows:

Chip A says	Chip B says	Conclusion
B is good	A is good	both are good, or both are bad
B is good	A is bad	at least one is bad
B is bad	A is good	at least one is bad
B is bad	A is bad	at least one is bad

   (a) (15 pts) Prove that if $n/2$ or more chips are bad, Snape cannot necessarily determine which chips are good using any strategy based on this kind of pairwise test. Assume that the bad chips can conspire to fool Snape.

   Hint: Let $\mathcal{B}$ denote the set of all bad chips and $\mathcal{G}$ the set of all good chips. Observe that there are only three types of comparisons: within $\mathcal{B}$, within $\mathcal{G}$, and between $\mathcal{B}$ and $\mathcal{G}$. What kind of results will each of these types of comparisons yield? How big are the sets?

   (b) (15 pts) Consider the problem of finding a single good chip from among the n chips, assuming that more than $n/2$ of the chips are good. Prove that $\lfloor n/2 \rfloor$ pairwise tests are sufficient to reduce the problem to one of nearly half the size.

   Hint: To reduce the problem's size, you will need to discard some chips. Only discard a pair that definitely includes a bad chip.

   (c) (15 pts) Prove that the good chips can be identified with $\Theta(n)$ pairwise tests, assuming that more than $n/2$ of the chips are good. Give and solve the recurrence that describes the number of tests.

2. (45 pts total) Consider the following basic problem. Professor Dumbledore gives you an array A consisting of n integers $A[1], A[2], \ldots, A[n]$ and asks you to output a two-dimensional $n \times n$ array B in which $B[i, j]$ (for $i < j$) contains the sum of array elements $A[i]$ through $A[j]$, i.e., the sum $A[i] + A[i+1] + \cdots + A[j]$. (The value of array element $B[i, j]$ is left unspecified whenever $i \geq j$, so it does not matter what is output for these values.)

   Dumbledore also gives you the following simple algorithm to solve this problem.

```
for i=1 to n
   for j = i+1 to n
      s = sum of array elements A[i] through A[j]
      B[i,j] = s
   end
end
```

(a) (10 pts) For some function f that you should choose, give a bound of the form $O(f(n))$ on the running time of this algorithm on an input of size n (i.e., a bound on the number of operations performed by the algorithm).

(b) (10 pts) For this same function f, show that the running time of the algorithm on an input of size n is also $\Omega(f(n))$. (This shows an asymptotically tight bound of $\Theta(f(n))$ on the running time.)

(c) (25 pts) Although Dumbledore's algorithm is the most natural way to solve the problem—after all, it just iterates through the relevant elements of B, filling in a value for each—it contains some highly unnecessary sources of inefficiency. Give a different algorithm to solve this problem, with an asymptotically better running time and prove its correctness.

3. (10 pts) Why do we analyze the average-case performance of a randomized algorithm and not its worst-case performance? Succinctly explain.

4. (10 pts extra credit each) Solve the following recurrence relations using the recurrence tree method (Chapter 4.4 in CLRS); include a diagram of your recurrence tree. If the recurrence relation describes the behavior of an algorithm you know, state its name.

(a) $T(n) = T(n-2) + n$

(b) $T(n) = T(\lceil n/2 \rceil) + 1$

(c) $T(n) = 2T(n/2) + n$

(d) $T(n) = 2T(n/2) + 1$