Put your *name* and *section number* on your solution and (if more than one sheet) staple.
Turn in Section 1 of the NP-completeness homework separately.

1. Assume that we have a special computer that can find the maximum of several values at a time. On the tth comparison step it can find the maximum of $f(t)$ numbers in an array. The format of the operation (executed on the tth comparison step) is

   k $\leftarrow$ max_index(A,i,j)

   where $j-i+1 \leq f(t)$. It assigns to k the index of a maximum value from $A[i], A[i+1], \ldots, A[j]$.

   (a) Give an efficient algorithm to find the maximum of n numbers where $f(t) = t + 1$. Write the pseudo-code. Try to minimize the number of comparison steps. Exactly how many comparison steps does your algorithm use?

   (b) Give an efficient algorithm to find the maximum of n numbers where $f(t) = 2^{t-1} + 1$. Write the pseudo-code. Try to minimize the number of comparison steps. Exactly how many comparison steps does your algorithm use?

   (c) CHALLENGE PROBLEM (not part of your grade). Give an efficient algorithm to find the maximum of n numbers where $f(t) = 2^t$. Write the pseudo-code. Try to minimize the number of comparison steps. Exactly how many comparison steps does your algorithm use?

2. Assume that your are using Bubble Sort to sort a list of n elements. Each permutation is equally likely except that the smallest and largest values start in the first and last positions in either order.

   (a) What is the average number of exchanges for $n = 2$. Justify.

   (b) What is the average number of exchanges for $n = 3$. Justify.

   (c) What is the average number of exchanges for $n = 4$. Justify.