1. For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t assuming that the algorithm to solve the problem takes $f(n)$ microseconds. (1 second = 1,000,000 microseconds). Recall that $\log n$ denotes the logarithm in base 2 of n. Your input sizes do not have to be very precise. Close approximations are good enough.

	1 Second	**1 Minute**	**1 Hour**
$\log n$			
sqrt(n)			
n			
$n\log n$			
n^2			
n^3			
2^n			
$n!$			

2. Arrange n^2 apples in a square. From each row find the largest one and let A be the smallest of these. From each column find the smallest one and let B be the largest of these. Which one is bigger, A or B? Give reasons.

3. Consider the following recursive algorithm.

   ALGORITHM *Min1*($A[0..n-1]$)
   // Input: An array $A[0..n-1]$ of real numbers
   if ($n = 1$) **return** A[0]
   else *temp* $\leftarrow$ *Min1*($A[0..n-2]$)
   if (*temp* $\leq A[n-1]$ **return** *temp*
   else return $A[n-1]$

   a. What does this algorithm compute?
   b. Set up a recurrence relation for the algorithm's basic operation count and solve it.

4. Show that $n^3 - n$ is always divisible by 3.

5. Is $n^5 - n$ always divisible by 5?

6. Given x, compute x^{62} in only eight multiplications (no divisions allowed!).