Problem 1. Consider the following game for two players P1 and P2. P1 picks an integer i between 1 and an upper limit k. P2 tries to determine i by a series of guesses. A guess consists of P2 naming a number j and P1 stating whether $j < i$, $j = i$ or $j > i$. The game is over when P2 guesses $j = i$. What is an optimal strategy for P2? Assuming an optimal strategy is used, how many guesses does P2 need in order to be sure to find i in each of the following cases? Briefly explain each of your answers.

1. $k = 1023$.
2. $k = 1023$ and P1 tells P2 before the first guess that i is even.
3. $k = 1023$ and P1 tells P2 *after answering the second guess* that i is even.
4. $k = 2047$ and P1 tells P2 before the first guess that i is divisible by 8.
5. $k = 255$ and P1 tells P2 before the first guess that i is a square number, that is, $\sqrt{i} \in \mathbb{Z}^+$.
6. $k = 64$ and P1 tells P2 before the first guess that i is a power of 2, that is, $i = 2^p$ for some $p \geq 0$.

Problem 2. (CLRS, Problem 2-4) Let $A[1..n]$ be an array of n distinct numbers. Define an *inversion* as a pair of indices (i, j) such that $i < j$ and $A[i] > A[j]$. Describe how to modify merge sort so it also computes the number of inversions in the input array in $\Theta(n \log n)$.

Problem 3. Find the exact solutions to the following recurrences and prove your solutions using induction.

1. $T(1) = 5$ and $T(n) = T(n-1) + 7$ for all $n > 1$.
2. $T(1) = 3$ and $T(n) = 2T(n-1)$.

Problem 4. Solve the following recurrences using the Master Method. In both cases you can assume that n is a power of the respective b so it can be divided without remainder on every level of the recursion.

1. $T(n) = 8T(\frac{n}{2}) + n^3$.
2. $T(n) = 2T(\frac{2n}{3}) + n^2$.

Problem 5. Give a tight bound $\Theta(g(n))$ for the value returned by `Rec(1,n)` (Hint: Use the Master Method).

```
Rec(p,r)
   c = 0
   if p < r
      q = ⌊(r-p+1)/3⌋
      c = c + Rec(p,p+q-1)
      c = c + Rec(r-q+1,r)
      j = 1
      while j*j ≤ r-p+1 do
         c = c+1
         j = j+1
   return c
```

Problem 6. The Karatsuba algorithm assumes that the length n of both input integers is a power of 2. Assume an input where this is not the case. Explain what can be done to ensure that the algorithm can still recurse to single-digit numbers and compute the correct product.

Problem 7. Explain how Quicksort can be modified to sort its input into nonincreasing order.