Problem 1
Consider a set S of m line segments (intervals) and a set P of n points on the real line. Design an $O((n+m)\log(n+m))$ time algorithm that reports every point in P that lies on a segment of S.

Problem 2
Let R be a set of n red segments in the plane and let B be a set of m blue segments in the plane. Design an algorithm that counts the number of intersections between the segments in R and the segments in B. Prove the running time, space requirement and correctness of your algorithm.

Problem 3
Given a set R of n pairwise disjoint rectilinear squares (sides are vertical or horizontal) and a set P of m points. Design an $O(n \log n)$ time algorithm that reports all points in S that lie inside a square in R. What if we consider rectangles instead? What if we allow the rectangles to intersect? Does the problem become much harder?

Problem 4
Let S be a set of m disjoint line segments and let P be a set of n points in the plane (no point lie on a segment). Given any query point q in the plane determine all points in P that p can see, that is, every point p in P such that the open segment pq does not intersect any line segment of S. Give an $O((m+n)\log(m+n))$ time algorithm.

Problem 5
Consider the following algorithm to compute the convex hull of a set S of n points in the plane.

Step 1: Sort the points in S by increasing x-coordinate.

Step 2: Recursively compute the convex hull of the left half of the point set. The resulting convex hull is denoted H_1.

Step 3: Recursively compute the convex hull of the right half of the point set. The resulting convex hull is denoted H_2.

Step 4: From H_1 and H_2 compute the convex hull H of the entire point set.

1. Assume that step 4 can be implemented in time $O(n)$. What is the running time of the algorithm? Prove your time bound.
2. Consider the edge e connecting the highest point in H_1 with the highest point in H_2. Will the edge e be an edge in H? Prove your answer.
3. Consider the points clockwise along H_1 between the highest point of H_1 to the lowest point of H_1. Can any of these points be in the convex hull, H, of the entire set? Prove your answer.
4. Give a correct implementation of step 4, that runs in $O(n)$ time. Prove the correctness and the running time of your algorithm.

Problem 6
Consider the following algorithm for the MST problem:

Algorithm 1 IMPROVING-MST

```
function IMPROVING-MST(G, w)
    T ← some spanning tree of G
    for e ∈ E [in any order] do
        T ← T + e
        C ← unique cycle in T
        f ← heaviest edge in C
        T ← T − f
    end for
    return T
end function
```

Prove its correctness and analyze its time complexity. To simplify things, you can assume the weights are different.

Problem 7

Consider the following algorithm for the MST problem:

Algorithm 2 REVERSE-MST

```
function REVERSE-MST(G, w)
    sort edges in decreasing weight w
    T ← E
    for e ∈ E [in this order] do
        if T − e is connected then
            T ← T − e
        end if
    end for
    return T
end function
```

Prove its correctness and analyze its time complexity. To simplify things, you can assume the weights are different.