Prob. 1	Prob. 2	Prob. 3

Problem 1.

Problem 2.

Just like for the question 1, we will transform this problem in a 2D linear optimization problem, so that we can use the (expected) linear runing-time algorithm already studied during the course.

Let suppose that we have n red points, of coordinates $(rx_i, ry_i) \forall i \in 1,..,n$ and m green points of coordinates $(gx_i, gy_i) \forall i \in 1,..,m$. Our goal is to find a line separating the green points from the red points (we suppose that our computer is not affected by daltonism). Such a line has a Cartesian equation of the form $y = n_1 x + n_2$. We want to find n_1 and n_2 (the parameters of the line).

A point is said "under" a line of parameters (n_1, n_2) if its coordinates (x, y) verify $y \leq n_1 x + n_2$, while it is considered "above" the line if we have $y \geq n_1 x + n_2$.

This can be rewritten as $(-x)n_1 + (-1)n_2 \leq -y$ for "under", and $xn_1 + n_2 \leq y$ for "above". Those are constraints for a 2D linear optimization problem.

Thus, we will specify our constraints like $(-rx_i)n_1 + (-1)n_2 \leq -ry_i \forall i \in 1,..,n$ for red points and $gx_i n_1 + n_2 \leq gy_i \forall i \in 1,..,m$ for green points.

We can use nearly whatever we want as the function to optimize, like maximizing $f(n_1, n_2) = 1$ (really simple function to optimize) or $f(n_1, n_2) = \frac{1}{1 - n_1^2 + n_2^2}$ (so that we get normalized parameters).

Thanks to this reformulation of the problem we can easily claim that we have a $O(n + m)$ expected running time algorithm solving that problem: we use the algorithm as seen during the class where we incrementally find a solution satisfying more and more constraints.

Problem 3.