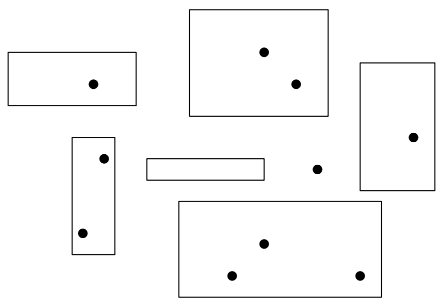
CS 473 (Fall 2026)
Homework 1 (due Sep 10 Thu 10am)

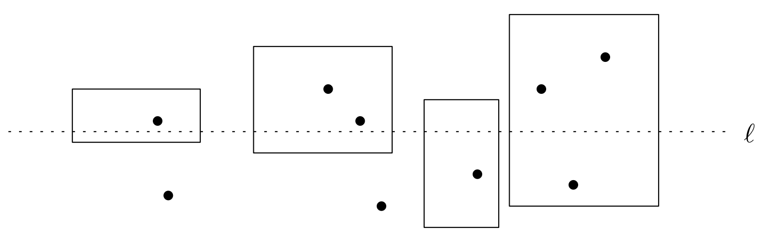
Instructions: Carefully read `http://courses.grainger.illinois.edu/cs473/fa2026/policies.html#hw` and `http://courses.grainger.illinois.edu/cs473/fa2026/integrity.html` (in particular, remember to include the 3 disclosure statements, on collaborators, outside sources, and AI use, for each problem).

Problem 1.1: We are given a set R of n_1 rectangles in 2D, where the sides of the rectangles are parallel to the x- and y-axes (i.e., the rectangles are not rotated) and the rectangles do not overlap. We are also given a set P of n_2 points in 2D. For every point $p \in P$, we want to find the rectangle $r_p \in R$ that contains p. (Note that if p is not covered by any of the rectangles in R, then r_p is undefined, but otherwise r_p is unique because of the nonoverlapping assumption.) Let $n = n_1 + n_2$.



(a) (35 pts) First give an $O(n \log n)$-time algorithm for the special case of the problem where all rectangles of R are assumed to intersect a given horizontal line ℓ.

[Hint: sort. . .]



(b) (65 pts) Now describe an algorithm to solve the general problem in $O(n \log^2 n)$ time or better.

[Hint: use divide-and-conquer and part (a) as a subroutine.][1]

[1] As usual, in a multi-part question, if you are unable to solve (a), you can still do (b) under the assumption that (a) has been solved.

Problem 1.2:

(a) (20 pts) Given two sets of numbers $A, B \subseteq \{0, \ldots, N-1\}$, define $A \oplus B = \{a + b : a \in A,\ b \in B\}$. Describe how to compute $A \oplus B$ in $O(N \log N)$ time.

[For example: $\{1, 5, 7\} \oplus \{2, 4, 8\} = \{3, 5, 7, 9, 11, 13, 15\}$. Hint: use polynomial multiplication or convolution.]

(b) (40 pts) Given two sets of numbers $A, B \subseteq \{0, \ldots, N-1\}$, define $A \boxplus B = \{a + b : a \in A,\ b \in B, a < b\}$. Describe an algorithm to compute $A \boxplus B$ in $O(N \log^2 N)$ time.

[For example: $\{1, 5, 7\} \boxplus \{2, 4, 8\} = \{3, 5, 9, 13, 15\}$.]

(c) (40 pts) Given three sets of numbers $A, B, C \subseteq \{0, \ldots, N-1\}$, define $f(A, B, C) = \{a + b + c : a \in A,\ b \in B,\ c \in C,\ a < b < c\}$. Describe an algorithm to compute $f(A, B, C)$ in $O(N \log^3 N)$ time.

Optional Problem 1.3 (not graded): We are given a text string $T = t_1 t_2 \cdots t_n \in \Sigma^*$ of length n over alphabet Σ. We are also given a sequence of pairs $P = \langle (a_1, b_1), \ldots, (a_m, b_m) \rangle$ with $a_i, b_i \in \Sigma$. We say that there is a *match* iff there exists a position i such that for every $j \in \{1, \ldots, m\}$, we have $t_{i+j} = a_j$ or $t_{i+j} = b_j$. Describe an $O(n \log n)$-time algorithm to determine whether there is a match. (Do not assume that $|\Sigma|$ is a constant.)

[For example: for $T = 01\mathbf{230}1032$ and $P = \langle (1, 2), (0, 3), (0, 1) \rangle$ with $\Sigma = \{0, 1, 2, 3\}$, there is a match (at the position shown in bold).]