Problem 1
Does $P = (p_1, \ldots, p_n)$ form a simple Polygon?

Problem 2 (Polygon Intersection)
Do two simple polygons intersect?

Problem 3 (Segment Intersection Test)
Are $n$ line segments pairwise disjoint?
Problem 4 (Segment Intersection)

Given \( n \) line segments, construct all intersections.

Problem 5 (Segment Arrangement)

Given \( n \) line segments, construct their Arrangement.

Problem 6 (Map Overlay)

Given sets \( S \) and \( T \) of pairwise disjoint line segments, construct the Arrangement of \( S \cup T \).
Segment Intersection

**Trivial Algorithm.** Test all pairs. $O(n^2)$ time and linear space.
Worst-case optimal for Problem 4...

In case of few intersections, we would like to have sub-quadratic time.

**Lower bound** $\Omega(n \log n)$ from Element Uniqueness.

**Problem 7** Given a set $I$ of $n$ intervals $[l_i, r_i] \subset \mathbb{R}$, $1 \leq i \leq n$, compute all intersecting pairs.

**Theorem 1**
Problem 7 can be solved in $O(n \log n + k)$ time and $O(n)$ space, where $k$ is the number of intersecting pairs.
Line Sweep

Idea. Move a line \( \ell \) (*sweep line*) from left to right, such that at any time all intersections to the left of \( \ell \) are known.

We do not make any general position assumption here, that is, several segments can start, end, and/or intersect at the same point. Imagine the sweep line infinitesimally twisted.
**Sweep line status (SLS).** Sequence $L$ of segments that intersect the current sweep line, sorted by $y$-coordinate.

**Event point (EP).** Point where SLS changes when moving $\ell$ (discretization).

**Event point schedule (EPS).** Sequence $E$ of event points to be processed (not all known in advance), sorted lexicographically.

With every EP $p$ we store

- a list $\text{end}(p)$ of segments ending at $p$;
- a list $\text{begin}(p)$ of segments that begin at $p$;
- a list $\text{int}(p)$ of segments that intersect a neighboring (in SLS) segment at $p$.

With every segment we store pointers to the ($\leq 2$) entries in $\text{int}()$ lists and a pointer to its appearance in $L$. 
Invariants.

i) $L$ is the sequence of segments from $S$ which intersect $\ell$, sorted by $y$-coordinate ($\leq$);

ii) $E$ contains all event points (endpoints from segments in $S$ and all points of intersection from segments adjacent in $L$) that are to the right of $\ell$;

iii) All intersections between segments from $S$ that are to the left of $\ell$ have been reported.
**Event point handling.** Consider an EP $p$.

1) If $\text{end}(p) \cup \text{int}(p) = \emptyset$, localize $p$ in $L$.

2) Report all pairs of segments from $\text{end}(p) \cup \text{begin}(p) \cup \text{int}(p)$ as intersecting.

3) Remove all segments in $\text{end}(p)$ from $L$.

4) Reverse the subsequence in $L$ that is formed by the segments from $\text{int}(p)$.

5) Insert segments from $\text{begin}(p)$ into $L$, sorted by slope.

6) Test the topmost and bottommost segment in SLS from $\text{begin}(p) \cup \text{int}(p)$ for intersection with its successor and predecessor, respectively, and update EP if necessary.
**Update of EPS.** Insert an EP $p$ for intersection of segments $s$ and $t$.

1) If $p$ does not yet appear in $E$, insert it.

2) If $s$ or $t$ are contained in some int($\cdot$) list of some other EP $q$, remove them there and possibly remove $q$ from $E$ (if end($q$) $\cup$ begin($q$) $\cup$ int($q$) = $\emptyset$).

3) Insert $s$ and $t$ into int($p$).

**Sweep.**

1) Insert all segment endpoints into begin($\cdot$) and end($\cdot$) lists of a corresponding EP in $E$.

2) As long as $E \neq \emptyset$, handle the first EP and then remove it from $E$. 
Runtime Analysis

Initialization: $O(n \log n)$.

Handling of an EP $p$:

$O(\#\text{intersecting pairs} + |\text{end}(p)| \log n + |\text{int}(p)| + |\text{begin}(p)| \log n + \log n)$.

Altogether:

$O(k + n \log n + k \log n) = O((n + k) \log n)$.

Space. Clearly $|S| \leq n$. At begin $|E| \leq 2n$ and $|S| = 0$. Never more than $2|S|$ intersection EPs, therefore linear space overall.

Theorem 2 Problem 4 and Problem 5 can be solved in $O((n + k) \log n)$ time and $O(n)$ space.

Theorem 3 Problem 1, Problem 2 and Problem 3 can be solved in $O(n \log n)$ time and $O(n)$ space.
Improvements

The presented algorithm is due to Jon Bentley and Thomas Ottmann (1979).

\[O(n \log n + k) \text{ time and } O(n + k) \text{ space [Bernard Chazelle and Herbert Edelsbrunner (1988)]}\]

expected \(O(n \log n + k)\) time using \(O(n + k)\) space [Ketan Mulmuley (1988)]

expected \(O(n \log n + k)\) time using \(O(n)\) space [Kenneth Clarkson and Peter Shor (1989)]

\(O(n \log n + k)\) time and linear space [Ivan Balaban (1995)]