Note: You should solve all the problems in the Mandatory Part, and one of the two problems in the Optional Part.

**Mandatory Part**

**Problem 1. External queue.** Show how to efficiently implement a queue in external memory. That is, show how to support two operations:

- **Push** \((a)\): pushes the element \(a\) at the end of the queue.
- **Pop**: removes the element from the front of the queue and returns it.

Your implementation should support any \(m\) operations on the queue using \(O(m/B + 1)\) block operations.

**Problem 2. Reference point for EMD.** Recall that, for two point-sets \(A, B \subset \mathbb{R}^2\), with \(|A| = |B|\),

\[
\text{EMD}(A, B) = \min_{\pi: A \rightarrow B} \sum_{a \in A} \|a - \pi(a)\|_2
\]

where \(\pi\) is 1-to-1.

Construct a reference point function \(r(A)\) for EMD (under translations). That is, give a function \(r(\cdot)\) such that for any two sets \(A, B\), if \(t^* = r(B) - r(A)\) is a translation that moves \(r(A)\) to \(r(B)\), then

\[
\text{EMD}(t^*(A), B) \leq c \cdot \min_{t \in T} \text{EMD}(t(A), B)
\]

for some constant \(c \geq 1\). As before, \(T\) is the set of all translations in \(\mathbb{R}^2\).

**Problem 3. Unit distances under the \(\ell_1\) norm.** Determine asymptotically the maximum number of unit distances between \(n\) points in \(\mathbb{R}^2\), under the \(\ell_1\) norm.

**Optional Part**

**Theoretical Problem. Instability of the minimum enclosing circle.** Let \(P\) be a set of \(n\) points in \(\mathbb{R}^2\), and let \(x\) be the center of the minimum enclosing circle of \(P\). Let \(P'\) be a set obtained
from \( P \) by perturbing each point by at most some distance \( \epsilon \). That is, there exists \( \epsilon > 0 \), and a bijection \( \sigma : P \to P' \) such that for each \( p \in P \),

\[
\|p - \sigma(p)\|_2 \leq \epsilon
\]

Let \( x' \) be the center of the minimum enclosing circle of \( P' \). Show that \( \|x - x'\|_2 \) cannot be upper-bounded by a function that depends only on \( \epsilon \).

**Programming Problem. Directed Hausdorff Under Translation.** Implement a Java applet that simulates the algorithm from slide 13 of lecture 21. The algorithm should take as input two sets of points \( A \) and \( B \). Then it should do the following: if there is a translation \( t \) such that \( DH(t(A), B) \leq r \), it should find a translation \( t' \) such that \( D(t'(A), B) \leq (1 + \epsilon)r \). It is OK if \( \epsilon r \) is set to be equal to the diameter of a screen pixel (i.e., the grid used in the algorithm can coincide with the grid induced by screen pixels).

To implement the above, the algorithm should construct approximations of the sets \( T(a) \) for \( a \in A \). Then, it should check if the intersection of all those sets is non-empty. If so, it should visualize the resulting translation.