For the algorithms below, a permutation is represented as a size-\( n \) array with values and indices ranging from 1 to \( n \).

**Question 1.** Give a linear-time algorithm computing the inverse of a permutation.

**Question 2.** Give a linear-time algorithm computing an optimal sequence of swaps sorting a permutation.

**Question 3.** Give a linear-time algorithm computing the decomposition of a permutation into disjoint cycles.

**Question 4.** Let \( S \) be a set of permutations defining distance \( d_S \) over \( S_n \), such that \( S \) is stable by inversion (\( \pi \in S \Rightarrow \pi^{-1} \in S \)).

- Prove that \( d_S(\pi) = d_S(\pi^{-1}) \) for every permutation \( \pi \).
- The stability by inversion is a sufficient condition to have the above property, but is it necessary?

**Question 5.** Give sorting sequences for the following permutations, and prove they are optimal:

- \( \langle 654321 \rangle \), using block-interchanges
- \( \langle 3254761 \rangle \), using transpositions

**Question 6.** Show that \( td(\pi) \leq n - LIS(\pi) \), where \( LIS \) denotes the length of the longest increasing subsequence.

**Question 7.** Give a polynomial-time 2-approximation algorithm for the Transposition Distance problem.