Sourav Mangla.

Problem 1 (Greedy algorithm):-

Given a list $x_1, x_2, \ldots, x_n$ of distinct real numbers, & another List $y_1, y_2, \ldots, y_n$ of distinct real numbers, reorder the x_i into a new list x_i', and reorder the y_i into a new list y_i', such that

$$\text{minimize} \max_{1 \le i \le n} |x_i' - y_i'|.$$

Algorithm:- (1) Initially we select kth smallest element from the x & y list where k is representing the number of iteration

(2) ~~We also have a variable max-difference which has~~ ~~Δk~~

(2) We will push corresponding selected element from step 1 to x' & y' at index K respectively. Where x' & y' are new reordered list.

(3) Now we calculate difference $|x_k' - y_k'|$ & store that difference into a variable, say maxd

(a) Initially maxd is zero.

(b) if incoming difference is greater than current maxd value we update maxd with current $|x_k' - y_k'|$ value.

(4) We run above steps till no element is present in x & y list.

Time analysis :-

Step 1 :- finding kth smallest elements from ~~$O(n)$~~ ~~x~~ x & y list takes $O(n)$ each, so overall step one take $O(n)$ for one iteration.

Step 2 :- Pushing or inserting element at the end of array/list takes constant time $O(1)$ (we know the size of list)

Step 3 :- updating maxd variable takes constant time.

Step 4 :- steps we are iterating this algorithm/above n times ~~&~~ step one is take $O(n)$ time so overall time complexity

$n O(n) = O(n^2)$

Time complexity $= O(n^2)$

Definition :- We define our structure such as $\{(x_1, y_1), (x_2, y_2) \ldots (x_n, y_n)\}$

Where x_i & y_i chosen from list x & y respectively & x_i & y_i ~~is~~ greater than x_{i-1} & y_{i-1} respectively

A structure $\{(x_1, y_1); (x_2, y_2) --- (x_k, y_k)\}$ is a prefix subsequence of the structure $\{(x_1, y_1), (x_2, y_2) --- (x_n, y_n)\}$ where $k \leq n$.

Lemma :-

Suppose S is a prefix subsequence which is contained in the optimal solution. Then S' be the solution which is produced by performing one step of greedy procedure which means augmenting the optimal solution S. means Appending kth smallest value from x & y to x' & y' respectively. So, S' produced ~~augmentation~~ must be contained in optimal solution.

Proof :- Let S^* be a optimal solution that contain S as a prefix subsequence. Let x_i, y_i be the elements we appended to x' & y' such that x_i & y_i are the kth smallest element for that iteration k.

(1) If S' is contained in S^* then, the lemma holds means S^* & our greedy solution has the same element x_i & y_i.

(2) If S' is not contained in S^* means S^* has x_{i+1} & y_k but S' has x_{i+1} & y_{i+1} at position i+1.

S^*	$x_1 \; - - - - \; x_i$	$x_{i+1} \; - - - \; x_K \; - - - \; x_n$
	$y_1 \; - - - - \; y_i$	$y_K \; - - - \; y_{i+1} \; - - - \; y_n$
S'	$x_1 \; - - - - \; x_i$	$x_{i+1} \; - - - - - - \; x_n$
	$y_1 \; - - - - \; y_i$	$y_{i+1} \; - - - - - - \; y_n$
	S^* & S' agrees till here	Disagree on y. values

Now, we need to create a new solution $\tilde{S}$ by swapping the position of y_K & y_{i+1}

	Position $i+1$	K	
S^*	$- - - \; x_{i+1}$	$- - - \; x_K$	$- - - \; x_n$
	$- - - \; y_K$	y_{i+1}	$- - - \; y_n$
$\tilde{S}$	$- - - \; x_{i+1}$	x_K	$- - - \; x_n$
	$- - - \; y_{i+1}$	y_K	$- - - \; y_n$

Now, Here we are able to conclude that our new solution $\tilde{S}$ has terms that are differing from the Optimal solution S^* is y_K & y_{i+1} at Position $i+1$ & K.

We know element at index K is greater than element at index $i+1$ because we are selecting K^{th} smallest element. So, $i+1$ is smaller than K (Here K is referring to index which is coming after $i+1$).

So, element $x_K > x_{i+1}$ & $y_K > y_{i+1}$

Now, if we get the difference of the terms here & we know that the difference of the solution $\hat{S}$ & S^* is not worse because at both instances the difference will always be ≥ 0. This means that our new solution structure $\hat{S}$ is not worse than that of S^*.

Theorem :-

The greedy algorithm finds a optimal reordering X & y to x' & y' respectively.

(1) Initially, an empty structure will always be a prefix subsequence of optimal solution.

(2) Then By lemma & solution induction on the number of iteration S. is still a prefix subsequence of optimal solution S^*.

(3) Since there are no more elements left when the algorithm terminates, the solution S contains all of n elements in both x' & y'. So there can be no other solution that contains S.

(4) Thus S is the optimal solution.