# CS345 : Algorithms II
# Semester I, 2021-22, CSE, IIT Kanpur

Assignment I

Deadline : 11:55 PM, 22 August 2020.

## Most Important guidelines

- It is only through the assignments that one learns the most about the algorithms and data structures. You are advised to refrain from searching for a solution on the net or from a notebook or from other fellow students. Remember - **Before cheating the instructor, you are cheating yourself**. The onus of learning from a course lies first on you. So act wisely while working on this assignment.
- Refrain from collaborating with the students of other groups. If any evidence is found that confirms copying, the penalty will be very harsh. Refer to the website at the link: https://cse.iitk.ac.in/pages/AntiCheatingPolicy.html regarding the departmental policy on cheating.

## General guidelines

1. There are four problems in this assignment: 2 Difficult problems, 1 Moderate, and 1 Easy. Each difficult problem carries 100 marks, the moderate one carries 75 marks, and the easy one carries 50 marks. Attempt **only** one of the 4 problems.

2. You are strongly discouraged to submit the scanned copy of a handwritten solution. Instead, you should prepare your answer using any text processing software (LaTex, Microsoft word, ...). The final submission should be a single pdf file.

3. You need to justify any claim that you make during the analysis of the algorithm. But you must be formal, concise, and precise.

4. If you are asked to design an algorithm, you may state the algorithm either in plain English or a pseudocode. But it must be formal, complete, unambiguous, and easy to read. You must not submit any code (in C++ or C, python, ...).

5. **Naming the file**:
   The submission file has to be given a name that reflects the information about the assignment number, version attempted (difficult/moderate/esay), and the roll numbers of the 2 students of the group. For example, you should name the file as **D_1_Rollnumber1_Rollnumber2.pdf** if you are submitting the solution for the difficult version of the 1st assignment. In a similar manner, the name should be **M_1_Rollnumber1_Rollnumber2.pdf** and **E_1_Rollnumber1_Rollnumber2.pdf** if you are submitting the solution for the moderate problem and the easy problem respectively of the 1st assignment.

6. Both students of a group have to upload the identical copy of their final submission. Be careful during the submission of an assignment. Once submitted, it can not be re-submitted.

7. Deadline is strict. Make sure you upload the assignment well in time to avoid last minute rush.

8. Contact TA at the email address: devansh@cse.iitk.ac.in for all queries related to the submission of the assignment. Avoid sending any such queries to the instructor.

# Difficult

## Faster algorithm for Non-dominated points in a plane

Recall the problem of non-dominated problem discussed in the second lecture of this course. We discussed two algorithms for this problem. The first algorithm takes $O(nh)$ time, where $h$ is the number of non-dominated points in the given set $P$. The second algorithm, which was based on the divide and conquer paradigm, takes $O(n \log n)$ time. As a part of this assignment, you have to design an $O(n \log h)$ time algorithm for non-dominated points. Interestingly, you have to use the insight from the first algorithm to just *slightly* modify the second algorithm so that its running time is improved to $O(n \log h)$. You must describe the algorithm and also provide the complete details of the analysis of its running time.
**Remark:** Note that $O(n \log h)$ is superior to $O(n \log n)$ in those cases where the number of non-dominated points are very few. In fact, it can be shown that if $n$ points are selected randomly uniformly from a unit square, then the expected(average) number of non-dominated points is just $O(\log n)$.

# OR

## Non-dominated points in 3 dimensions

You can extend the notion of non-dominated points to 3 dimensions as well. Spend some time to realize that it is not so simple to apply divide and conquer strategy to compute the non-dominated points in 3 dimensions. Some of you may be tempted to solve this problem by reducing an instance of this problem to three instances of 2-dimensional problem (by projecting the points on x-y, y-z, x-z plane). But that will be wrong (think over it to convince yourself). Interestingly, there is a very simple elegant algorithm using a simple data structure that computes non-dominated points in 3 dimensions. The purpose of this exercise is to make you realize this fact.

1. Design an algorithm that receives $n$ points in x-y plane one by one and maintains the non-dominated points in an online fashion. Upon insertion of $i$th point, the algorithm should take $O(\log i)$ time to update the set of non-dominated points.
   *Note:* It is perfectly fine if your algorithm only guarantees a bound of $O(i \log i)$ on the total time for insertion of $i$ points. It is not necessary that your algorithm achieves $O(\log i)$ bound on the processing carried out during insertion of $i$th point.

2. Design an $O(n \log n)$ time algorithm to compute non-dominated points of a set of $n$ points in 3 dimensions. You must use part $(a)$ above carefully.

# Moderate

## Convex Hull

In Lecture 2, we discussed an $O(n \log^2 n)$ time algorithm to compute the convex hull of a given set of $n$ points in a plane. If we can improve the time complexity of the Conquer step of this algorithm to linear, this will result in an $O(n \log n)$ time algorithm for convex hull. You have to modify the current Conquer step so that it takes at most linear time. You must provide a complete analysis of the modified Conquer step as well.

# Easy

## Counting Double-Inversions

You are given an array $A$ storing $n$ numbers. A pair $(i, j)$ with $0 \leq i < j \leq n-1$ is said to be a double-inversion if $A[i] > 2A[j]$. Design and analyze an $O(n \log n)$ time algorithm based on divide and conquer paradigm to compute the total number of double-inversions in $A$.