

Homework Assignment 1

CS 535 Design and Analysis of Algorithms
Fall Semester, 2008

Rules for Homework

On this and all future homework assignments, observe these guidelines *or credit will be deducted*:

- Write your name *clearly* at the top of *each* page. If I can't read your name *easily*, you will not get credit for the assignment.
- Begin each problem on a new page.
- Staple your finished assignment in the upper-left corner. Do not use a paper clip or fold the corner—it is your job to insure that pieces of your assignment don't get lost.
- All work must be your own with no collaboration with anyone else. The only resources that you can use in solving the homework problems are the text and class notes; nothing else is permitted. You must sign and attach the *Honesty Pledge* with this first assignment.
- Explicitly cite any references you use, including the textbook; except for the text, always give full citations—authors, titles, dates of publication, page numbers.
- Algorithms must be carefully presented, proved correct, and analyzed in terms of their time and space requirements. Follow the style of the textbook in these matters—your analyses and proofs must be mathematically rigorous.

Due: Thursday, August 28, 2008

1. IIT has a collection of time machines. These machines, once programmed, take a person back to a specific point of time. There are m such machines, where the i th machine, M_i , is pre-programmed to move p_i years back. IIT wants to give each of n grad students the opportunity to go back in time to a point in time of his or her choice. The choice of the j th student is year y_j . Ideally, each student would be assigned to the time-machine that goes as close as possible to the year of his or her own choice. Design an efficient algorithm to assign time machines to students, so that the sum of absolute differences of the pre-programmed year and year of choice is minimized.
2. You are given the coordinates of the n vertices of a convex n -sided polygon and you are to triangulate the polygon by drawing $n - 3$ non-intersecting diagonals. The cost of a triangulation is the sum of the lengths of the diagonals drawn.
 - (a) How many triangulations are possible?
 - (b) Use dynamic programming to obtain the least-cost triangulation in $O(n^3)$ time.