

Home Work 3

CSE 101, Spring 202

Issued Thursday, May, 15. Due in class Thursday, May, 22.

State your answers legibly and concisely. Your solutions will be graded on correctness, elegance, **clarity** and originality. Your proofs should **avoid getting bogged down in too much detail**. Please note that the work handed in must be your own. Please use *staples* to fasten the pages (any loss of unstapled sheets is at your own risk). **Your handwriting must be legible** and answers must be *in proper order* for full credit to be awarded. Every problem is 20 points.

Problem 1. Read the section 15.2 from CLRS. Find an optimal parenthesization of a matrix-chain product whose sequence of dimensions is $(5, 10, 3, 12, 5, 2i)$ where i is the last digit of your social security number.

Problem 2. Let L be an array of n distinct integers. Give an algorithm to compute the length of a longest decreasing subsequence of entries in L . For example, if $L = \{3, 12, 7, 4, 6, 8, 5, 17, 11\}$, a longest decreasing subsequence is 12, 7, 6, 5. How much time and memory does your algorithm take? Can you do better?

Problem 3. Suppose you have n dollars to invest in any of m enterprises. Assume that n is an integer and that all investments must be in integer amounts. The table *return* describes the expected returns for individual investments. Specifically, $return(d, j)$ is the expected return for an investment of d dollars in enterprise j . Assume that the columns of *return* are non-decreasing, i.e., investing more money will not decrease your return.

Write a fast algorithm to determine the maximum possible expected return for investing n dollars. What is the running time of your algorithm? Is it a polynomial algorithm? Expand your algorithm to determine the optimal investment plan (i.e. how much to invest in each enterprise).

Problem 4. Describe how you can find your way out of a maze if you are given a large supply of pennies. A maze may have a large number of identical rooms and you may want to use your pennies to mark whether you have visited a particular room already.

Problem 5. A *bipartite* graph is a graph whose vertices can be partitioned into two subsets such that there is no edge that between any two vertices in the same subset. Write a fast algorithm to determine if a graph is bipartite. What is the complexity of your algorithm?