Fundamentals of Algorithms
CS502-Spring 2011
ASSIGNMENT #3

Deadline

Your assignment must be uploaded/submitted at or before **15th June 2011**

Uploading instructions

Please view the **assignment submission process** document provided to you by the Virtual University to upload the assignment.

Rules for Marking

It should be clear that your assignment will not get any credit if:

- The assignment is submitted after due date.
- The submitted assignment does not compile or run.
- **The assignment is copied.**

Objectives

This assignment will help you to understand the concepts of Knapsack Problem and Chain matrix Multiplication which results in efficient calculation time wise.

Guidelines

1. In order to attempt this assignment you should have full command on Lecture # 19 to Lecture # 26
2. In order to solve this assignment you have strong concepts about following topics
   - ✓ Chain Matrix Multiplication
   - ✓ Knapsack Problem

Recommended book for solving assignment

Cormen, Leiserson, Rivest, and Stein (CLRS) 2001, **Introduction to Algorithms**, (2nd ed.) McGraw Hill.

Estimated Time 4 hours

To understand the theme of both questions 90 minutes.Question1 solution implementation maximum time is 90 minutes and for Question2 solution implementation maximum time is one hour. It all depends upon your sheer concentration and devotion towards your lecture listening.

Question# 1 **(10)**

Consider the chain matrix multiplication for 4 matrices:

$A1 \;.\; A2 \;.\; A3 \;.\; A4$
$(5\times6) \quad (6\times3) \quad (3\times7) \quad (7\times10)$

Compute the cost table *m* in the dynamic programming algorithm for the chain matrix multiplication

Question# 2 **(10)**

Recall that a dynamic programming solution to the 0-1 knapsack problem can be derived from the following recurrence formula for $c[i,w]$, the value of the solution for items 1, . . . , i and maximum weight w.

$$c[i,w] = \begin{cases} 0 & \text{if } i = 0 \text{ or } w = 0, \\ c[i-1,w] & \text{if } w_i > w, \\ \max(v_i + c[i-1, w-w_i], c[i-1,w]) & \text{if } i > 0 \text{ and } w \geq w_i. \end{cases}$$

In the following example the inputs are $n = 9, W = 15$, with values v_i and weights w_i:

i	1	2	3	4	5	6	7	8	9
$\mathbf{v_i}$	15	13	12	18	20	8	13	19	22
$\mathbf{w_i}$	2	3	2	3	3	1	5	2	5

Run knapsack algorithm on this table to determine the maximum value that thief may take, also mention which items the thief should take to achieve the maximum value