CSE2046 Assignment 3

0-1 (Multiple) Knapsack Problem

Due date: May 29th, 2018 (for the test outputs), May 31th, 2018 (for the report and codes)

In this assignment, you are asked to design and implement algorithms for 0-1 Knapsack Problem and 0-1 Multiple Knapsack Problem.

Problem 1: 0-1 Knapsack problem is a well-known combinatorial optimization problem: Given a set of n items, $N = \{1,2,\dots,n\}$, each with a weight w_i and a value v_i, determine the subset of these items to include in a knapsack, so that weight is less than or equal to the capacity (W) of knapsack and the total value is as large as possible.

Formally:

Maximize $\sum_{i=1}^{n} v_i x_i$ // The objective is to maximize total value in the knapsack

Subject to $\sum_{i=1}^{n} w_i x_i \leq W$ // You cannot exceed capacity of the knapsack

$x_i = \{0,1\}$. // x_i is equal to 1 if item i is included, and zero otherwise.

Problem 2: 0-1 Multiple Knapsack problem is an extension of Knapsack Problem, where we have m knapsacks, $M = \{1,2,\dots,m\}$, each with capacities W_j .

Formally:

Maximize $\sum_{j=1}^{m} \sum_{i=1}^{n} v_i x_{i,j}$ // The objective is to maximize total value in all knapsacks

Subject to: $\sum_{i=1}^{n} w_i x_{i,j} \leq W_j \quad \forall j \in M$ //None of the knapsacks can exceed their capacity

$\sum_{j=1}^{m} x_{i,j} \leq 1 \quad \forall i \in N$ // Each item can be included in only one knapsack

$x_{i,j} = \{0,1\}$ // $x_{i,j}$ is equal to 1 if item i is included in knapsack j.

Project Specification:

Your team (up to 2 students) is asked to design and implement an algorithm for both knapsack problems (Problem 1 and Problem 2).

Since Knapsack problem is an NP-hard problem, it is difficult to find optimal solutions especially for large instances. Your goal is not to design an algorithm for the optimal solution, but you are requested to do your best.

You may do the following:

- Read as much as you want to learn about how to solve the (multiple) Knapsack problem. But **you have to cite** any resources you use. You may want to start with some approximation algorithms (such as local search heuristics) in chapter 12.
- You may use whichever programming language you want.

You may **not use** the following:

- Existing implementations or subroutines
- Extensive libraries (if you are not sure, check with the instructor)
- Other people's code.

Input format for Problem 1: Inputs will always be given to you as a text file. Input file format should be as following:

n W
v1 w1
v2 w2
: :
vi wi
: :
vn wn

No other input format will be accepted!

n: number of items
W: knapsack capacity
vi: value of item i
wi: weight of item i

Output format for Problem 1: Output should be a text file including following:

Total_Value
x1
x2
:
xi
:
xn

First line should include Total_Value which is the sum of the values of items included in the knapsack. Next lines include zeros and ones. **No other output format will be accepted!**

Input format for Problem 2: For multiple knapsack problem, input file format should be as following:

n m
W1 W2 .. Wm
v1 w1
v2 w2
: :
vi wi
: :
vn wn

m is the number of knapsacks and Wj is the capacity of j'th knapsack.

Output format for Problem 2:

Total_Value
x1,1 x1,2 ... x1,m
x2,1 x2,2 ... x2,m
:
xi,1 xi,2 ... xi,m
:
xn,1 xn,2 ... xn,m

Example instances: You may find example instances in the web site. There are four example inputs (sample1a.dat, sample1b.dat, sample1c.dat and sample1d.dat) for Problem 1 and one example input (sample2a.dat) for problem 2. The optimal values are given in the following table:

Input File	Optimal Solution
Sample1a.dat	35
Sample1b.dat	2,397
Sample1c.dat	54,503
Sample1d.dat	90,204
Sample2a.dat	333

Outputs for optimal solutions are also available for Sample1a.dat and Sample2a.dat.

Test instances: On May 28th, we will announce 3 test instances for Problem 1 and two test instances for Problem 2 at the web site. By May 29th, 23:59, you will be required to submit 5 seperate output text files (according to the output format) corresponding to each of these test instances. These files should be called test1a.dat, test1b.dat, test1c.dat, test2a.dat, test2b.dat and should be submitted via google classroon. The deadline is strict.

Project report: You will submit a project report by May 31th, 2019, 23:59. The project report should describe the ideas behind your algorithms as completely as possible. It should not exceed 3 pages in length in no less than 10pt.

Grading policy:

60% of your grade will be determined by your project report. Clarity and creativity of your work will significantly affect your grade.

40% of your grade will be determined by your solutions to the test instances. Studies that find solutions closer to the best possible solution will get higher grades.

Best projects will be awarded with bonus points.