Assignment 8

csci2200, Algorithms

Instructions:

- Honor code: Work on this assignment with at most one partner. Between different teams, Collaboration is at level 1 [verbal collaboration only]
- Write each problem on a separate page; If a problem has multiple parts, you can write all parts on the same page, as long as you leave space in between them.

---

1. **Taking a quiz:** Consider a quiz with n questions. For each $i = 1, 2, ..., n$, question i has integral point value $v_i > 0$ and requires $m_i > 0$ minutes to solve. Suppose further that no partial credit is awarded (unlike this exam).

   Your goal is to come up with an algorithm which, given $v_1, v_2, ..., v_n, m_1, m_2, ..., m_n$ and V, computes the minimum number of minutes required to earn at least V points on the quiz. For example, you might use this algorithm to determine how quickly you can get an A on the quiz. To get you started, we'll give the choice of subproblem that you'll compute.

   The choice of subproblem: Let $MinMin(i, v)$ denote the minimum number of minutes needed to earn v points when you are restricted to selecting questions from 1 through i. We will compute $MinMin(i, v)$ for arbitrary i, v. To solve the problem we'll call $MinMin(n, V)$.

   (a) Argue briefly that the subproblem above has optimal substructure.

   (b) Give a recursive algorithm for $MinMin(i, v)$ (without dynamic programming). To get you started, here is the base case:
   - $MinMin(i, v) = 0$ for all i, and $v \leq 0$.
   - $MinMin(0, v) = \infty$ for $v > 0$

   (c) If implemented without dynamic programming, how long would it take to compute $MinMin(n, V)$? (an $\Omega()$ expression for the worst case running time is sufficient).
   We expect: A recurrence for the running time without dynamic programming, and the justification on why the solution to this recurrence has the lower bound that you claim it has.

(d) Describe (give pseudocode for) a dynamic programming algorithm to solve the problem (either top-down or bottom-up).

We expect: the pseudocode

(e) Analyze the running time of computing $MinMin(n, V)$ with dynamic programming.

2. **Load balancing:**[1] You have been hired to design algorithms for optimizing system performance. Your input is an array $J[1..n]$ where $J[i]$ or J_i represents the running time of job i; jobs do not have specific start and end times, but they can be started at any time (this is a different scenario than in the interval scheduling /activity selection problem). The running times are integers. Generally speaking, your task is to find an optimal load balance of these tasks over two processors.

   Design an algorithm for determining whether there is a subset S in J such that the running time of the elements in S sum up precisely to the same amount as the sum of the elements not in S; more formally, $\sum_{J_i \in S} J_i = \sum_{J_i \in J-S} J_i$. The algorithm should run in time $O(n \cdot N)$, where N is the sum of the running times of the n jobs.

   We expect: (1) Explanation of your choice of subproblem and its parameters (i.e. describe what function you will compute, what it returns and what are its parameters); how do you call it to solve the original problem; (2) why it has optimal substructure; (3) pseudocode and an English description of your algorithm for computing the subproblem with dynamic programming; (3) analysis of its running time.

[1] Leetcode #416