- This exam has one question for 100 marks.
- Warning: CMI's academic policy regarding cheating applies to this exam.

Unstated assumptions and lack of clarity in solutions can and will be used against you during evaluation. You may freely refer to statements from the lectures in your arguments. You don't need to reprove these unless the question explicitly asks you to, but you must be precise. Please ask the invigilators if you have questions about the questions.

A *subsequence* of an array A is any sub-array of A, obtained by deleting zero or more elements of A *without* changing the order of the remaining elements. The input to the *Subsequence Sum* problem consists of (i) an array $A[0 \ldots (n-1)]$ of n positive integers, and (ii) a target integer T. The problem is to decide if there exists a subsequence B of A such that the sum of all the elements of B is exactly T. We define the sum of the *empty* subsequence (one with no elements) to be zero, and the "sum" of a subsequence with one element, to be that element.

1. (a) Write the *complete* pseudocode for a *non-recursive* algorithm that solves the *Subsequence Sum* problem on inputs $(A[0 \ldots (n-1)], T)$ using *dynamic programming*. The algorithm should have a worst-case running time of $\mathcal{O}(n^c T^d)$ for some small constants c, d. [40]

   You will get the credit for this part only if your psuedocode is that of a non-recursive DP algorithm which correctly solves the problem within the required time bound.

   Make sure that you correctly initialize your DP table, that you check for sentinel values wherever required, and that you always compute and store the value in any cell of the table *before* you access the value in that cell for further computation.

   (b) Explain why your algorithm of part (a) correctly solves the problem. [40]

   (c) Show that the worst-case running time of your algorithm of part (a) is $\mathcal{O}(n^c T^d)$ for some small constants c, d. What are the constants that you get? [20]