- This exam has 5 questions for a total of 125 marks, of which you can score at most 100 marks.
- The deadline for **receiving** your emailed answers is 14:30 hours on December 19, 2020. Email your answers to: algo2020cmi@gmail.com. **Do not** use this email address for any other communication.
- You may answer any subset of questions or parts of questions. All answers will be evaluated.
- You are free to refer to the lectures/your notes/any other material that you wish, but you are **not** permitted to confer with one another in any manner. Note that "referring" to material does *not* cover **copying** material from elsewhere, except for statements of theorems, lemmas and such. In particular, do not copy arguments wholesale from external sources.
- 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. That is, "As we saw in class ... " will not get you any credit, but "As we saw in Exercise x of Lecture y" will. You may assume that comparing any two integers takes constant time. Please feel free to ask us via the designated channels if you have questions about the questions.

Whenever you are asked to design an algorithm, you must make it as efficient as possible. The credit will depend on the running time of your algorithm as well, apart from its correctness. Also, show the correctness and analyze the running time.

1. You know a collection of times A that trains will arrive at a station. When a train arrives, there must be someone manning the station. Each employee can work at most two hours at the station. The problem is to find an assignment of slots to employees that uses the fewest number of employees. Design an algorithm for this problem, prove its correctness and analyze the running time. [25]

2. A *walk* in a graph G is a sequence of the form $\langle v_1, e_1, v_2, e_2, \ldots, e_{\ell-1} v_\ell \rangle$ where each v_i is a vertex in G, and each e_j is an edge in G of the form $e_j = (v_j, v_{(j+1)})$. Note that by this definition every walk starts and ends at a vertex. We further define the empty sequence, and a vertex by itself, to be walks. A *path* in G is a walk in which no vertex appears more than once. [25]

   The k-PATH problem is defined as follows:

- Input: A simple undirected graph G on n vertices and a positive integer k.
- Question: Does G contain a path with k vertices in it?

Using Dynamic Programming (or otherwise ...) design an algorithm which solves k-PATH in time $\mathcal{O}(2^n \cdot n^c)$ for some constant c which is independent of n and k. Prove that your algorithm is correct, and analyze its running time.

Hint: Why is it OK to assume that you know the vertex where such a path (if it exists) starts? So suppose that you are only interested in looking for paths that start at a specified vertex s of G. Construct a DP table T whose rows are indexed by the vertices of G, and whose columns are indexed by the numbers $1, 2, \ldots, k$. Each cell $T[v, i]$ contains a *collection of subsets* of vertices of G, where each set in this collection has size i. The contents of $T[v, i]$ are the set of all i-sized subsets of $V(G)$ that satisfy some property. What should this property be, so that (i) you can build up the DP table, and (ii) read out the solution once you have filled up the table?

3. We call a subset S of vertices in an undirected graph G to be 1-*sparse* if each vertex in S is adjacent to at most one other vertex in S. By a reduction from the independent set problem, prove that it is NP-complete to find whether, given a graph G and k, G has a 1-sparse subset of size at least k. [25]

4. **k-splittability:** We call a graph to be k-*splittable* if the vertices of G can be partitioned into k non-empty partitions so that there is no edge between any two vertices in the same partition. It is known that the problem of determining if a graph is 5-splittable is NP-complete. Give a reduction from this problem to show that 12-splittability i.e. determining whether a graph is 12-splittable is NP-complete. [25]

5. There are n cities $a_1, \ldots, a_n$ in a state. There is a train network that connects all the cities with each other, not necessarily by a non-stop connection. The police department gets a hint that a huge amount of stolen gold is to be transferred by some smugglers from city a_1 to city a_n by train. But a_1 and a_n are remote cities, so it is not possible to send the police force there at a short notice. Also, there is no direct train from a_1 to a_n, so the smugglers have to change train at one or more intermediate stations. The police department wants to deploy police force at some intermediate stations such that at least one station on every route from a_1 to a_n is watched, and the number of stations watched is minimized. Design a polynomial-time algorithm for this problem. Prove its correctness and analyze the time complexity. [25]