**Build a planner**

Offboard Infrastructure engineers at the ATC are expected to be able to reason about large systems and the complex interactions they may have. We need to be able to write software that will coordinate and manage a variety of different tasks.

Please write software that will tackle the following problem. Use whatever tools you find appropriate, whether it be your programming environment, using Google, etc. Please document you work, discuss your approach, and enumerate all the assumptions and trade-offs that you made. Please do not spend more than 8-10 hours on this, and feel free to ask questions. We are most interested in seeing how you approach a problem, how you manage the deadline, how you communicate your thoughts around the problem, and what your view of "quality" is. **We wish you to submit your software, instructions on how to run it, an example data set that we can run, and documentation on any design decisions or assumptions you made.**

Imagine you are preparing a system to be able to coordinate the execution of a large number of tasks. Many of these tasks are dependent on each other and all have different requirements. As resources, you have a heterogeneous cluster of computers. You wish to exploit parallelism and compete the entire set of tasks as quickly as possible.

We will define a list of tasks, and their dependencies, in a YAML file that looks like so:

```
task1:
  cores_required: 2
  execution_time: 100
task2:
  cores_required: 1
  execution_time: 200
  parent_tasks: task1
task3:
  cores_required: 4
  execution_time: 50
  parent_tasks: "task1, task2"
```

`cores_required` means that the all the cores must be available on the same computing resource, `execution_time` is the number of global "ticks" that pass before the task will be complete and can release all its computing resources, and the `parent_tasks` must be completely finished executing before a particular task can start.

We will then also specify the computing resources and their number of available cores as so:

```
compute1: 2
compute2: 2
compute3: 6
```

Please implement a program that reads those two files, and then emits an ordered scheduling plan that manages all the dependencies (execution time, number of cores, parent tasks). The goal is to execute the full lists of tasks as quickly as possible. The plan may look something like:

```
task1: compute1
task2: compute1
task3: compute3
```