DELFT UNIVERSITY OF TECHNOLOGY
Faculty of Electrical Engineering, Mathematics and Computer Science

TI2306 Algorithm Design
Resit Digital test – part 1

April 18, 2019, 90 minutes

- Usage of the book, notes or a calculator during this test is not allowed.
- This digital test contains 3 questions (worth a total of 10 points) contributing 10/20 to your grade for the digital test. The other digital test contributes the other 10/20 to your grade. Note that the final mark for this course also consists for $\frac{2}{3}$ of the unrounded score for the written exam (if both ≥ 5.0).
- The storyline throughout the questions can be ignored: each question can be answered independently of the others and in any order.
- The spec tests in WebLab **do not represent your grade**.
- There is an API specification of the Java Platform (Standard Edition 8) available at `https://weblab.tudelft.nl/java8/api`.
- When an implementation is asked, please provide the **most efficient** implementation in terms of asymptotic time complexity. Providing a suboptimal implementation can lead to a subtraction of points.
- The material for this tests consists of module 1 (Greedy Algorithms) and 2 (Divide and Conquer) of the course and the corresponding book chapters and lectures.
- Total number of pages (without this preamble): 1.
- Exam prepared by M. de Weerdt and S. Hugtenburg

Mason is providing a computing service. For each computing job j he receives its (estimated) runtimes: the initialization of t_j seconds, to be done by his central server, and the main computation of p_j seconds, which starts immediately after the initialization and is done in the cloud (which we assume can be scaled up such that all jobs could be run in parallel if needed). Mason wants to schedule the initializations of the jobs such that these do not overlap, and such that the computation of all jobs is done as quickly as possible (minimizing the latest end time of any job in the cloud).

1. (4 points) Given are the number of jobs n and two sequences of length n: one with the initialization times, and one with the main computation times, in no particular order.

   Implement a greedy algorithm to determine the start times of all initializations such that the latest end time is minimized. Return the **start time** of the initialization of the last job according to this schedule.

2. (1 point) Someone else is in charge of the producing the timetable and your answer to the previous question is ignored (i.e., not necessary for answering this question).

   In addition to the input for the first question, you are given two more sequences: a sequence S with at position i the start time of the ith job that will be run, and a sequence I with at position i the index j of this job (i.e., the ith job that will be run). Example: $I[7] = 2$ means that the seventh job to run is the job with index two.

   Implement an algorithm to efficiently compute the latest **end time** of (the computation in the cloud given) this schedule.

3. (5 points) Given is the same input as for the previous question.

   Mason is now asked whether he can run a high-priority job at time x. Before he decides, he wants to know how many jobs would be delayed if he started the initialization of this job as soon as the last job started before time x completed its initialization. Implement a divide-and-conquer algorithm that returns for a given time x, the number of jobs that are initialized at or later than x.
   (Hint: an $\Omega(n)$ solution is worth at most 2 points.)

End of exam questions