Voortgezet Programmeren (FEB23007-13)

5. Exercise

Deadline for submission: 2014-02-16 23:59 CET

Instructions

Include in each source file (in class documentation, @author) your names and student numbers. Submit the exercise as a zip file containing _only_ the source files in root of the zip. Submit via blackboard. Note that incorrectly submitted or non-compiling exercises are automatically awarded 0 points. Remember to document your code with Javadoc-annotations.

Exercise

In this exercise we will perform discrete event simulation of museum visiting behaviour of 500 000 holders of the Dutch museum pass. The simulation period is one year, and for each day of the year, there are 8 000–12 000 (uniformly distributed) visits from unique museum card holders. Note that the same person never visits two different museums in one day (these pass holders seem to enjoy passing a whole day in a single museum). We assume that these visits are ordered within a day and that the museums have relative attractiveness to the visitors given in the table below.

Museum	Attractiveness factor
Boijmans	0.08
Kunsthal	0.12
Rijksmuseum	0.2
van Gogh	0.25
Stedelijk A'dam	0.25
Groninger	0.06
Oudhuidkundige Leiden	0.04

For each visit, you should do weighted sampling to decide, according to the attractiveness factors, which museum the visitor will choose. Then, if in the current day there are already more than attractiveness factor * 8 500 visits to the chosen museum, there might be a queue to get in and the visitor might choose another museum. In this case you should re-sample the museum (s)he will visit (note that the visitor might decide to stay in the queue and will therefore go to the museum if the same museum is sampled again).

Each visitor will be uniquely identified with a museum pass sequence number. However, in order to make the simulation easily extensible, you should make a class `MuseumPass` that holds this information. For each visit, you should store the date of the visit and the museum visited (make class `Visit`). Implement the actual simulation in a separate class `Simulation`, which has a constructor taking as input the number of pass holders, an interval of the number of visits/day, the queue factor (8500 in our case), and a `List` of `Museums`.

Make a main runner class that runs initializes the list of museums and runs the simulation with the given parameters. After the simulation has finished, you should print in the main the following statistics (include accessor methods for these in `Simulation`):

- The number of unique visitors through the year
- The number of unique visitors per month
- The number of total visitors for each museum
- The minimum and the maximum museum card indices performing visits this year

Hints

- Be sure that you sample the visitors so that that are no duplicate visits from the same pass holder within a single day.
- Make extensive use of private methods in `Simulation` to maintain clarity of your code.