Robot Swarm Planning

Flavor Text

RoboTech is a tech company that is building automated factories to build and release large numbers of autonomous robots. Being a tech startup, they don't know what they're going to do with swarms of robots, but they're optimistic that users will come to them once the robots are built.

Your tech startup shares this optimism, but doesn't know what to do with the robots either. Your startup is building a platform for assigning these large numbers of robots to customers, like a large-scale robot-share program. And you need to show in simulations just how large-scale it can get.

Taking the production forecasts for the *Robotech* factories, and various models on how quickly these mass-produced robots will wear out, your software needs to predict when the most robots will be available for potential customers' fixed duration projects.

Problem Description

Robot production numbers are provided for each month, R_n, and should be treated as the number of new robots coming online at the start of the month (compared to the start of the previous month). You can expect $1 \le R_n \le 1e8$.

Each list of production numbers will start with a year, Y_n. $1970 \le Y_n \le 1e8$. Assume 0 robots online before Y_0. At most 1000 months of robot production numbers will need to be considered at a time, submitted in 1 – 1000 separate lists.

Robot life expectancy will be defined by a sequence of numbers, E_n, meaning that after N months, E_n is the expected proportion of robots from that batch that have been deactivated. E_0 is always 0, and the final E_n value is always 1. As an example, if $R_0 = 1000$, and $E_5 = 0.5$, then at the start of the 5th month 500 of the 1000 robots that came online at the start of the 0th month should be considered offline.

The number of projects you need to schedule is denoted N. $1 \le N \le 10$.

All projects in this problem have the same duration, D measured in whole months. $1 \le D \le 10$.

Scheduled projects may not overlap. Choose project start times, P_n, such that the sum of the number of robots at the start of each month, across all projects is maximized. If there are multiple solutions with the same such sum, choose projects such that ΣP_n (defined below) is minimized.

P_n is measured in a whole number of months from the first month of the first year in the input data. For example, if the first year of the input data was 2000, and the best time for project 1 to start is January 2002, P_1 would be 25.

Technical Details

You must implement an HTTP server with three HTTP APIs:

1. POST /data
   This API should accept application/text request body content with one number per line.

The first number will be a year, Y_n.
The remaining numbers will be the number of robots produced, R_n, in order per month.
The first R_n number is for January Y_n, the next line is for February, then March, and so on. There may be more than twelve months in one POST request, which means data can span across a year in one request.

2. DELETE /data
   This API should wipe all data previously fed in via POST /data

3. POST /analyze
   This API should accept application/text request body content with one number per line.
   The first number will be N.
   The second number will be D.
   The remaining numbers will be E_n numbers, one per line.
   This API should output application/text reponse body content with one number per line.
   Each line will be a single P_n number. Numbers should be sorted from lowest to highest.

You must implement the server using node16 and typescript. You must provide a docker image serving your HTTP server on port 2468 by default, as well as your typescript source code and Dockerfile.

As this is just a test exercise, please do not spend time on production hardening such as CORS headers, CSRF protection, external databases, or other security/reliability/scalability features beyond an efficient and bug-free algorithm.

While there is no time limit; your time and space complexity will be considered along with the correctness of output.

Worked Example

In this example the factory comes online 2025, and has a production schedule as follows:

January	February	March	April	May	June
0	5	5	5	2	1

As the factory starts production in January, the first robots completed count as being ready for February. As the factory shuts down in June, there are no new robots in July or later.

In this scenario half the robots last one month and half the robots last two months. This can lead to a non-integer number of expected robots. The expected number of robots online each month are as follows:

January	February	March	April	May	June	July
0	5	7.5	7.5	4.5	2	0.5

With this data, let's try to schedule two projects that last two months each. The best time for that would be month 3, March, as then there will be 15 expected robots online for two months combined.

If we take the same data though and try to schedule two projects (each lasting two months), the best times would be February and April. That provides 12.5 expected robots for the first project, and 12 robots for the second, for a total of 24.5 expected robots during the projects.

If you had taken the first project at March still, and tried to schedule around that, you would have scheduled the second project for May and it would have 6.5 expected robots during its duration. 15 + 6.5 is 21.5, which is less than 24.5, so this is an inferior (and thus incorrect) solution.

To illustrate data interchange used for judging the submitted docker image, the following curl commands could be used to judge a submission for the above problem:

```
curl http://localhost:2468/data -H 'Content-Type: text/plain' --data-binary @-
<< EOF
2025
0
5
5
5
2
1
EOF

# 1 2 month projects
curl http://localhost:2468/analyze -s -H 'Content-Type: text/plain' -o
answer1.txt --data-binary @- << EOF
1
2
0
0.5
1
EOF

diff answer1.txt - << EOF
3
EOF

# 2 2 month projects
curl http://localhost:2468/analyze -s -H 'Content-Type: text/plain' -o
answer2.txt --data-binary @- << EOF
2
2
0
0.5
1
EOF

diff answer2.txt - << EOF
2
4
EOF

curl -X DELETE http://localhost:2468/data
```