The Best Plagiarism Detector

Time limit: 1s

Plagiarism is an emerging issue in academia and that becomes more concerning due to the pandemic. To deal with that, MCU, an academic institution in Indonesia, conducts research to find the best plagiarism detector to use. The 'best' is mainly defined from effectiveness, which is measured via f-score, the harmonic mean between precision and recall. F-score value is in between 0 to 1 inclusive and it is calculated as follows:

$$f\text{-}score = \frac{2 \times precision \times recall}{precision + recall}$$

Precision is the proportion of copied programs suspected by the similarity detector to total suspected programs. Recall on the other hand, is the proportion of copied programs suspected by the similarity detector to total copied programs.

Given N plagiarism detectors, your task is to sort the detectors based on their effectiveness. If there are two or more plagiarism detectors with the same effectiveness (i.e., the difference is less than 0.01), kindly prioritise the one with the shortest processing time. If there are two or more plagiarism detectors with the same effectiveness and efficiency, prioritise the one entered first in the list of plagiarism detectors.

Input

The program initially accepts N as the number of plagiarism detectors. Per detector, it accepts program name (as an alphanumeric string), followed by three metrics for calculating f-score (its number of copied and suspected programs, total suspected programs, and total copied programs) and execution time (in hour minute second format). All inputted numbers are non-negative, and their values can be up to 10,000.

Output

Names of the similarity detectors sorted based on given criteria, separated by line break.

Example input:

3 # number of plagiarism detectors

stephen # first detector's name

5 10 8 # first detector's metrics: copied and suspected programs, total suspected programs, total copied programs

1 30 23 # first detector's processing time, 1 hour 30 minutes 23 seconds

vincent # second detector's name

5 10 8 # second detector's metrics: copied and suspected programs, total suspected programs, total copied programs

1 28 27 # second detector's processing time, 1 hour 28 minutes 27 seconds

strange # third detector's name

4 8 5 # third detector's metrics: copied and suspected programs, total suspected programs, total copied programs

0 18 3 # third detector's processing time, 18 minutes 3 seconds

Example output:

strange

vincent

stephen