Programming task Customs

Memory limit: 5 MB
Time limit: 2 seconds
Input file: `customs.in`
Output file: `customs.out`

Description

In recent years, the country ABC fearing the entry of unwanted persons, has changed its policy on immigration control. At the airport, when crossing the border, all arrivals are directed into two large streams - ABC citizens and non-citizens. Citizens are controlled with a simpler procedure, while non-citizens are thoroughly checked for documents, luggage and fingerprints are also taken. Let's consider all these procedures to be customs exit procedures.

Customs work is organized as follows (see picture "Customs operation scheme"). Each customs officer has his own workplace. Each workplace has one of two types: type P for citizen control and type N for non-citizen control. Workplaces of each type are numbered sequentially, starting from 1 to a number equal to the number of workplaces in that type. For each workplace, i.e. the customs officer who works there has his own speed of immigration control. Let us assume that for a given customs officer the speed of control of any entrant is constant.

Visitors enter the customs building through one door, i.e. each arrival has its own unique moment of time when it enters the customs building. We will consider this unique arrival time as the arrival's unique identifier (arrival ID).

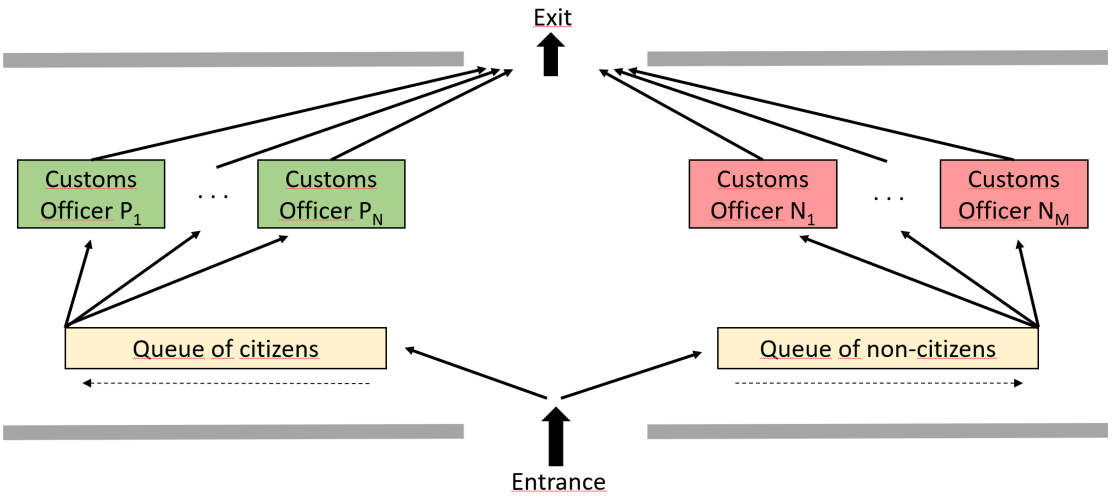
Upon entering the customs building, the person must immediately check whether a customs officer is available to serve him (according to the status of a citizen or a non-citizen). If there is a free customs officer, then person should go to the free customs officer who has the lowest workplace number. If there is no free customs officer, then person has to wait in line until a customs officer becomes free. You must not go to the wrong customs officer, i.e. there may be a situation, for example, that there is a long queue at the non-citizen customs officers, but the citizen customs officers are unemployed.

After the inspection, the person goes outside (we assume that the bad persons are taken out in a convoy). The exit time is the value obtained by adding the time spent waiting in line and the time spent at the control of the customs officer to the arrival time. The exit is big enough so that all the people who want to can go through it at the same time.

Your task is to write a program that receives data on the number of customs officers in each type, the customs officer's work speed, types of arrivals and their arrival times at the customs building, and prints the exit time of each arrival from the customs building. Results should be printed in chronological order. If there are several persons going out at the same time, then the persons with the status of a citizen should be printed first, and within the persons of the same status, the person who had a lower customs officer's workplace number should be printed.

[1..99] customs officers work for the control of each type of arrival. The duration of person service is within [1..100'000]. Arrival times or arrival IDs are in the range [1..4'000'000] and are in ascending order in the input file.

The input data is correct according to the input data format and given constraints.



Customs operation scheme

Input:

The first line of the input file contains the number of customs officers and the default time for processing one person in the form:

```
P_Officers N_Officers P_Time N_Time
```

- `P_Officers` defines the number of customs officers [1..99] who control the citizens
- `N_Officers` defines the number of customs officers [1..99] who control non-citizens
- `P_Time` defines the default control time [1..100'000] for customs officers who control citizens
- `N_Time` defines the default control time [1..100'000] for customs officers who control non-citizens

Then follows the number of lines [0.. P_Officers + N_Officers], which specify the time of arrival control for a specific customs officer. Each line is in the form

```
T Type Officer Time
```

- `Type` is a single letter that defines the type of customs officer {P, N}, i.e. whether it controls citizens (P) or non-citizens (N)
- `Officer` determines the id number [1..99] of the officer in his type (id of workplace)
- `Time` defines the control time [1..100'000] of this officer

Then follows the number of lines [0.. 4'000'000], which describes the arrivals at the customs building. Each line is in the form

```
Type ID
```

- `Type` one letter that defines the type of person {P, N}, i.e. whether the person is a citizen (P) or a non-citizen (N)
- `ID` determines the person's arrival time [1.. 4'000'000], which serves as the ID of the person

The input file is always terminated by a line containing the capital letter 'X'.

```
X
```

Output:

According to the input file, the output file contains a response for each person's arrival in the format:

```
ID Out
```

- `ID` determines the person's arrival time [1.. 4'000'000], which serves as the ID of the person
- `Out` determines the person's exit time [1.. 4'000'000'000]

If there is no record of arrivals in the input file, then the word "nothing" should be printed without quotation marks.

```
nothing
```

Example:

The content of input file `customs.in`:

```
2 3 10 50
T P 1 7
T N 2 80
P 1
N 2
N 10
N 20
N 30
N 40
P 45
P 50
P 53
N 60
X
```

The content of output file `customs.out`:

```
1 8
45 52
2 52
53 60
50 60
20 70
10 90
30 102
40 120
60 170
```