E. Cron

Sometime the classic solution are not powerful enough and we have to design our own. For the purpose of this problem you have to implement the part of the system of task scheduling.

Each task should be executed at some particular moments of time. In our system you may set the exact value for the second, minute, hour, day of the week, day and month, when the task should be executed. Moreover, one can set a special value -1 that means any value of this parameter is valid.

For example, if the parameter string is `-1 59 23 -1 -1 -1`, the problem will be executed every day at 23:59:00, 23:59:01, 23:59:02, ..., 23:59:59 (60 times in total).

Seconds, minutes and hours are numbered starting from zero, while day, months and days of the week are numbered starting from one. The first day of the week is Monday.

There is one special case that is treated separately. If both day of the week and day are given (i.e. differ from -1) to execute the task only one of these two (at least one, if both match this is fine too) parameters should match the current time (of course, all other parameters should match too). For example, the string of parameters `0 0 12 6 3 7` means that the task will be executed both on Saturday, July 2nd, 2016 and on Sunday, July 3rd, 2016 at noon.

One should not forget about the existence of the leap years. The year is leap if it's number is divisible by 400, or is not divisible by 100, but is divisible by 4. Each leap year has 366 days instead of usual 365, by extending February to 29 days rather than the common 28.

The current time is represented as the number of seconds passed after 00:00:00 January 1st, 1970 (Thursday).

You are given the string of six parameters, describing the moments of time the task should be executed. You are also given a number of moments of time. For each of them you have to find the first moment of time strictly greater than the current when the task will be executed.

Input

The first line of the input contains six integers s, m, h, day, $date$ and $month$ ($0 \le s, m \le 59$, $0 \le h \le 23$, $1 \le day \le 7$, $1 \le date \le 31$, $1 \le month \le 12$). Each of the number can also be equal to -1. It's guaranteed, that there are infinitely many moments of time when this task should be executed.

Next line contains the only integer n ($1 \le n \le 1000$) — the number of moments of time you have to solve the problem for. Each of the next n lines contains a single integer t_i ($0 \le t_i \le 10^{12}$).

Output

Print n lines, the i-th of them should contain the first moment of time strictly greater than t_i, when the task should be executed.

Examples

input

```
-1 59 23 -1 -1 -1
6
1467372658
1467417540
1467417541
1467417598
1467417599
1467417600
```

output

```
1467417540
1467417541
1467417542
1467417599
1467503940
1467503940
```

input

```
0 0 12 6 3 7
3
1467372658
1467460810
1467547200
```

output

```
1467460800
1467547200
1468065600
```

Note

The moment of time 1467372658 after the midnight of January 1st, 1970 is 11:30:58 July 1st, 2016.