(Adv.) Competitive Programming

Submit until 19.07.2019 13:30, via the judge interface

HPI Hasso Plattner Institut
Digital Engineering · Universität Potsdam

Problem: paths (1 second timelimit)

Your friend Paul recently wrote his algorithms exam, which included the following question:

> You are given an infinite directed multigraph G, parameterized by $a, b_1, \ldots, b_{10} \in \mathbb{N}^{\geq 0}$. Let $a \rightsquigarrow_n b$ denote that there are n edges from a to b. G consists of an infinite number of nodes $[s_i]_{i \in \mathbb{N}^{\geq 0}}$, as well as the edges
>
> - $s_0 \rightsquigarrow_{(a \cdot i)} s_i$ for $i \in \mathbb{N}^{>0}$, and
> - $s_i \rightsquigarrow_{b_j} s_{(i+j)}$ for $i \in \mathbb{N}^{>0}$ and $1 \leq j \leq 10$.
>
> Describe and analyze an efficient algorithm that finds the number of distinct[1] paths from from s_0 to s_q for each q in a given finite set $Q \subset \mathbb{N}^{>0}$.

After some consideration, Paul noticed that G is always a DAG that is already topologically sorted. Remembering that paths can be counted in a topologically sorted DAG in linear time, he wrote down the algorithm and proudly concluded that it runs in $\mathcal{O}(\max Q)$ time.

Much to his dismay, his answer was deemed "too inefficient" and only received half points. Out of ideas, Paul turns to you for help. Can you find a more efficient way to solve the problem?

Input The first line of the input contains the parameter a ($0 \leq a \leq 10^9$) and the second the parameters b_1 to b_{10} ($0 \leq b_i \leq 10^9$). The third line contains $|Q|$ ($1 \leq |Q| \leq 10$), followed by a line containing the values in Q ($1 \leq q \leq 10^{18}$ for each $q \in Q$).

Output For each $q \in Q$, output a line containing the number of distinct paths from s_0 to s_q. The answers should be in the same order as the values in the input. Since the number of paths can be quite large, you should output them modulo $10^9 + 7$.

[1] Two paths are considered equal iff. they contain the same edges

Sample input

```
3
0 0 0 0 0 0 0 0 0 0
3
1 5 3
```

Sample output

```
3
15
9
```

```
1
1 2 3 0 0 0 0 0 0 0
4
1 3 6 1000
```

```
1
8
123
176838664
```