6051 Longest Prefix Match

In the Internet, a router forwards packets based on their destination address by consulting a forwarding table which specifies for each destination which path (if any) the packet should take next. A destination address is an unsigned integer. The forwarding table contains many distinct prefixes (thousands), and each prefix is a tuple (M, L) where M is an address mask (an unsigned integer) and L is a length specifying the number of relevant bits in the mask. A destination address D matches a prefix (M, L) if the L most significant bits of D match the L most significant bits of M.

Longest Prefix Match: whenever a packet arrives, the router must find the longest prefix from its database that matches the destination address of the packet. You are asked to implement Longest Prefix Match for a router that has to forward *a few million packets* as quickly as possible.

Input

On the first line, the input gives two integers X and Y. X denotes the number of prefixes in the router's forwarding table, and Y gives the number of packets to be forwarded. The next X lines describe the prefixes, each line describing one prefix by the mask (hexadecimal) followed by the length (base10). Each prefix is identified by number the line it appears on minus one (i.e. the first prefix has id 0, the second 1, etc). The next Y lines give the destination addresses of the packets to be forwarded, with one packet on each line (the addresses are also given in hexadecimal).

Output

The output contains Y lines. Line k represents the outcome of the longest prefix match algorithm for the k'th address in the input file. The outcome is either the identifier of the longest matching prefix or '`-1`' if no matching prefix was found.

Sample Input

```
5 5
0xFFFFFF00 24
0xFF000000 8
0xAF230000 16
0x3 31
0 0
0xFFFF0000
0xF0FF0000
0xFFFFFF00
0xAF320000
0x2
```

Sample Output

```
1
4
0
4
3
```