Competitive Programming SS24

Submit until end of contest

Problem: bus (1.0 second timelimit)

After attending *Basics of digital systems* during her first semester, Lena decided to get some more practice. She wants to build a project involving many programmable components that talk to each other. Of course she learned that, in order to work around the n^2 point to point connections, a bus for communication would be the way to go. However she thought a traditional, single parallel bus would be lame. Instead, she invented a connection architecture utilizing the same amount of wiring a parallel bus would need.

Every component has an incoming and an outgoing port of 64 bits. Ports can only have a single multicore cable attached. The components are arranged in a loop, every outgoing port is wired up to the incoming port of another component. Each component i for $1 \leq i < n$ has its outgoing connection to component $i + 1$, component n has its outgoing connection to component 1.

Components have three modes: *send*, *receive* and *passthrough*. For now, only one component can be in *send* and one component can be in *receive* mode simultaneously, while all others have to maintain *passthrough* mode.

Lena also had a few inverters lying around, which she thought would be fun to add to the build. After all, it would look cool and make the design more obscure. Behind every outgoing port directly before the attached cable she passes some bit signals through inverters, so that the outgoing bit is inverted.

Lena now wants to program some receiver routines for some point-to-point connections and needs to know which bits to invert to get the originally sent data at the receiving component.

Input The first line contains n, the number of components ($2 \leq n \leq 5 \cdot 10^5$) and q, the number of connections, for which she needs your help ($1 \leq q \leq 10^5$).

The second line contains n hexadecimal numbers[1] x_i, representing the pattern of inverters at the outgoing port of component i ($0 \leq x_i < 2^{64}$, $1 \leq i \leq n$). A 1 marks an inverter in front of the respective outgoing bit. The following q lines each contain two distinct indices $1 \leq s, r \leq n$, representing a connection from s to r Lena needs to program.

Output For each connection query print which incoming bits Lena needs to invert to get the originally sent data as a line containing the bit pattern encoded as decimal integer.

[1] You might want to use https://en.cppreference.com/w/cpp/io/manip/hex to read them.

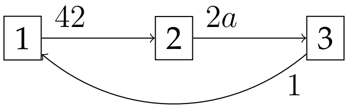

Figure 1: The sample case. $n = 3$, arrows are connections pointing from an outgoing to an incoming port, labeled with their inverter pattern at their start.

Sample input

```
3 3
42 2a 1
1 2
1 3
2 1
```

Sample output

```
66
104
43
```