GPC-UPC Group Contest I

A. Jeff and Digits

1 second, 256 megabytes

Jeff's got n cards, each card contains either digit 0, or digit 5. Jeff can choose several cards and put them in a line so that he gets some number. What is the largest possible number divisible by 90 Jeff can make from the cards he's got?

Jeff must make the number without leading zero. At that, we assume that number 0 doesn't contain any leading zeroes. Jeff doesn't have to use all the cards.

Input

The first line contains integer n $(1 \le n \le 10^3)$. The next line contains n integers $a_1, a_2, ..., a_n$ ($a_i = 0$ or $a_i = 5$). Number a_i represents the digit that is written on the i-th card.

Output

In a single line print the answer to the problem — the maximum number, divisible by 90. If you can't make any divisible by 90 number from the cards, print -1.

input
```
4
5 0 5 0
```
output
```
0
```

input
```
11
5 5 5 5 5 5 5 5 0 5 5
```
output
```
5555555550
```

In the first test you can make only one number that is a multiple of 90 — 0.

In the second test you can make number 5555555550, it is a multiple of 90.

B. The Eternal Immortality

1 second, 256 megabytes

Even if the world is full of counterfeits, I still regard it as wonderful.

Pile up herbs and incense, and arise again from the flames and ashes of its predecessor — as is known to many, the phoenix does it like this.

The phoenix has a rather long lifespan, and reincarnates itself once every $a!$ years. Here $a!$ denotes the factorial of integer a, that is, $a! = 1 \times 2 \times ... \times a$. Specifically, $0! = 1$.

Koyomi doesn't care much about this, but before he gets into another mess with oddities, he is interested in the number of times the phoenix will reincarnate in a timespan of $b!$ years, that is, $\frac{b!}{a!}$. Note that when $b \ge a$ this value is always integer.

As the answer can be quite large, it would be enough for Koyomi just to know **the last digit of the answer in decimal representation**. And you're here to provide Koyomi with this knowledge.

Input

The first and only line of input contains two space-separated integers a and b $(0 \le a \le b \le 10^{18})$.

Output

Output one line containing a single decimal digit — the last digit of the value that interests Koyomi.

input
```
2 4
```
output
```
2
```

input
```
0 10
```
output
```
0
```

input
```
107 109
```
output
```
2
```

In the first example, the last digit of $\frac{4!}{2!} = 12$ is 2;

In the second example, the last digit of $\frac{10!}{0!} = 3628800$ is 0;

In the third example, the last digit of $\frac{109!}{107!} = 11772$ is 2.

C. Hungry Sequence

1 second, 256 megabytes

Iahub and Iahubina went to a date at a luxury restaurant. Everything went fine until paying for the food. Instead of money, the waiter wants Iahub to write a Hungry sequence consisting of n integers.

A sequence $a_1, a_2, ..., a_n$, consisting of n integers, is *Hungry* if and only if:

- Its elements are in increasing order. That is an inequality $a_i < a_j$ holds for any two indices i, j $(i < j)$.
- For any two indices i and j $(i < j)$, a_j must **not** be divisible by a_i.

Iahub is in trouble, so he asks you for help. Find a Hungry sequence with n elements.

Input

The input contains a single integer: n $(1 \le n \le 10^5)$.

Output

Output a line that contains n space-separated integers a_1 a_2, ..., a_n $(1 \le a_i \le 10^7)$, representing a possible Hungry sequence. Note, that each a_i must not be greater than 10000000 (10^7) and less than 1.

If there are multiple solutions you can output any one.

input
```
3
```
output
```
2 9 15
```

input
```
5
```
output
```
11 14 20 27 31
```

D. Flowers

1.5 seconds, 256 megabytes

We saw the little game Marmot made for Mole's lunch. Now it's Marmot's dinner time and, as we all know, Marmot eats flowers. At every dinner he eats some red and white flowers. Therefore a dinner can be represented as a sequence of several flowers, some of them white and some of them red.

But, for a dinner to be tasty, there is a rule: Marmot wants to eat white flowers only in groups of size k.

Now Marmot wonders in how many ways he can eat between a and b flowers. As the number of ways could be very large, print it modulo 1000000007 $(10^9 + 7)$.

Input
Input contains several test cases.

The first line contains two integers t and k $(1 \le t, k \le 10^5)$, where t represents the number of test cases.

The next t lines contain two integers a_i and b_i $(1 \le a_i \le b_i \le 10^5)$, describing the i-th test.

Output
Print t lines to the standard output. The i-th line should contain the number of ways in which Marmot can eat between a_i and b_i flowers at dinner modulo 1000000007 $(10^9 + 7)$.

input
```
3 2
1 3
2 3
4 4
```
output
```
6
5
5
```

- For $K = 2$ and length 1 Marmot can eat (R).
- For $K = 2$ and length 2 Marmot can eat (RR) and (WW).
- For $K = 2$ and length 3 Marmot can eat (RRR), (RWW) and (WWR).
- For $K = 2$ and length 4 Marmot can eat, for example, ($WWWW$) or ($RWWR$), but for example he can't eat ($WWWR$).

E. Another Problem on Strings

2 seconds, 256 megabytes

A string is binary, if it consists only of characters "0" and "1".

String v is a *substring* of string w if it has a non-zero length and can be read starting from some position in string w. For example, string "010" has six substrings: "0", "1", "0", "01", "10", "010". Two substrings are considered different if their positions of occurrence are different. So, if some string occurs multiple times, we should consider it the number of times it occurs.

You are given a binary string s. Your task is to find the number of its substrings, containing exactly k characters "1".

Input
The first line contains the single integer k $(0 \le k \le 10^6)$. The second line contains a non-empty binary string s. The length of s does not exceed 10^6 characters.

Output
Print the single number — the number of substrings of the given string, containing exactly k characters "1".

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

input
```
1
1010
```
output
```
6
```

input
```
2
01010
```
output
```
4
```

input
```
100
01010
```
output
```
0
```

In the first sample the sought substrings are: "1", "1", "10", "01", "10", "010".

In the second sample the sought substrings are: "101", "0101", "1010", "01010".

F. Random Task

1 second, 256 megabytes

One day, after a difficult lecture a diligent student Sasha saw a graffitied desk in the classroom. She came closer and read: "Find such positive integer n, that among numbers $n + 1$, $n + 2$, ..., $2 \cdot n$ there are exactly m numbers which binary representation contains exactly k digits one".

The girl got interested in the task and she asked you to help her solve it. Sasha knows that you are afraid of large numbers, so she guaranteed that there is an answer that doesn't exceed 10^{18}.

Input
The first line contains two space-separated integers, m and k $(0 \le m \le 10^{18}; 1 \le k \le 64)$.

Output
Print the required number n $(1 \le n \le 10^{18})$. If there are multiple answers, print any of them.

input
```
1 1
```
output
```
1
```

input
```
3 2
```
output
```
5
```

G. Little Pony and Harmony Chest

4 seconds, 256 megabytes

Princess Twilight went to Celestia and Luna's old castle to research the chest from the Elements of Harmony.

A sequence of positive integers b_i is harmony if and only if for every two elements of the sequence their greatest common divisor equals 1. According to an ancient book, the key of the chest is a harmony sequence b_i which minimizes the following expression:

$$\sum_{i=1}^{n} |a_i - b_i|.$$

You are given sequence a_i, help Princess Twilight to find the key.

Input
The first line contains an integer n ($1 \le n \le 100$) — the number of elements of the sequences a and b. The next line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 30$).

Output
Output the key — sequence b_i that minimizes the sum described above. If there are multiple optimal sequences, you can output any of them.

input
```
5
1 1 1 1 1
```
output
```
1 1 1 1 1
```

input
```
5
1 6 4 2 8
```
output
```
1 5 3 1 8
```

H. Holes

1 second, 64 megabytes

Little Petya likes to play a lot. Most of all he likes to play a game «Holes». This is a game for one person with following rules:

There are N holes located in a single row and numbered from left to right with numbers from 1 to N. Each hole has it's own power (hole number i has the power a_i). If you throw a ball into hole i it will immediately jump to hole $i + a_i$, then it will jump out of it and so on. If there is no hole with such number, the ball will just jump out of the row. On each of the M moves the player can perform one of two actions:

- Set the power of the hole a to value b.
- Throw a ball into the hole a and count the number of jumps of a ball before it jump out of the row and also write down the number of the hole from which it jumped out just before leaving the row.

Petya is not good at math, so, as you have already guessed, you are to perform all computations.

Input
The first line contains two integers N and M ($1 \le N \le 10^5$, $1 \le M \le 10^5$) — the number of holes in a row and the number of moves. The second line contains N positive integers not exceeding N — initial values of holes power. The following M lines describe moves made by Petya. Each of these line can be one of the two types:

- $0\ a\ b$
- $1\ a$

Type 0 means that it is required to set the power of hole a to b, and type 1 means that it is required to throw a ball into the a-th hole. Numbers a and b are positive integers do not exceeding N.

Output
For each move of the type 1 output two space-separated numbers on a separate line — the number of the last hole the ball visited before leaving the row and the number of jumps it made.

input
```
8 5
1 1 1 1 1 2 8 2
1 1
0 1 3
1 1
0 3 4
1 2
```
output
```
8 7
8 5
7 3
```

I. Remainder Problem

4 seconds, 512 megabytes

You are given an array a consisting of 500000 integers (numbered from 1 to 500000). Initially all elements of a are zero.

You have to process two types of queries to this array:

- $1\ x\ y$ — increase a_x by y;
- $2\ x\ y$ — compute $\sum_{i \in R(x,y)} a_i$, where $R(x, y)$ is the set of all integers from 1 to 500000 which have remainder y modulo x.

Can you process all the queries?

Input
The first line contains one integer q ($1 \le q \le 500000$) — the number of queries.

Then q lines follow, each describing a query. The i-th line contains three integers t_i, x_i and y_i ($1 \le t_i \le 2$). If $t_i = 1$, then it is a query of the first type, $1 \le x_i \le 500000$, and $-1000 \le y_i \le 1000$. If $t_i = 2$, then it it a query of the second type, $1 \le x_i \le 500000$, and $0 \le y_i < x_i$.

It is guaranteed that there will be at least one query of type 2.

Output
For each query of type 2 print one integer — the answer to it.

input
```
5
1 3 4
2 3 0
2 4 3
1 4 -4
2 1 0
```
output
```
4
4
0
```

J. Ant colony

1 second, 256 megabytes

Mole is hungry again. He found one ant colony, consisting of n ants, ordered in a row. Each ant i $(1 \le i \le n)$ has a strength s_i.

In order to make his dinner more interesting, Mole organizes a version of «Hunger Games» for the ants. He chooses two numbers l and r $(1 \le l \le r \le n)$ and each pair of ants with indices between l and r (inclusively) will fight. When two ants i and j fight, ant i gets one battle point only if s_i divides s_j (also, ant j gets one battle point only if s_j divides s_i).

After all fights have been finished, Mole makes the ranking. An ant i, with v_i battle points obtained, is going to be freed only if $v_i = r - l$, or in other words only if it took a point in every fight it participated. After that, Mole eats the rest of the ants. Note that there can be many ants freed or even none.

In order to choose the best sequence, Mole gives you t segments $[l_i, r_i]$ and asks for each of them how many ants is he going to eat if those ants fight.

Input

The first line contains one integer n $(1 \le n \le 10^5)$, the size of the ant colony.

The second line contains n integers $s_1, s_2, ..., s_n$ $(1 \le s_i \le 10^9)$, the strengths of the ants.

The third line contains one integer t $(1 \le t \le 10^5)$, the number of test cases.

Each of the next t lines contains two integers l_i and r_i $(1 \le l_i \le r_i \le n)$, describing one query.

Output

Print to the standard output t lines. The i-th line contains number of ants that Mole eats from the segment $[l_i, r_i]$.

input

```
5
1 3 2 4 2
4
1 5
2 5
3 5
4 5
```

output

```
4
4
1
1
```

In the first test battle points for each ant are $v = [4, 0, 2, 0, 2]$, so ant number 1 is freed. Mole eats the ants $2, 3, 4, 5$.

In the second test case battle points are $v = [0, 2, 0, 2]$, so no ant is freed and all of them are eaten by Mole.

In the third test case battle points are $v = [2, 0, 2]$, so ants number 3 and 5 are freed. Mole eats **only the ant 4**.

In the fourth test case battle points are $v = [0, 1]$, so ant number 5 is freed. Mole eats the ant 4.