Problem A. Oh these volunteers

Input file: `volunteers.in`
Output file: `volunteers.out`

Holding olympiad is always a hard business and the main workers are always the volunteers. Each volunteer is subordinate of exactly one other volunteer. The only exception is Aman, the head volunteer whose number is 1. This system was brought up to split up requests of contestants between the volunteers in a most efficient way.
More formally: say volunteer numbered V gets W requests from contestants (bring some water, print solutions, etc).

- If the volunteer has no subordinates he performs all the request himself.
- If the volunteer has K ($K > 0$) subordinates and if K is a divisor of W, each of his subordinates gets W/K redirected requests and follows the same algorithm.
- If the volunteer has K ($K > 0$) subordinates and if K is not a divisor of W, all W requests are ignored.

Aman wants to minimize the number of ignored requests. He would like to apologize to all of the contestants, but he needs to know the number of ignored requests at first.
He asked a friend of his NurlashKO to help him. NurlashKO decided to redirect this problem to the contestants.

Input

First line of the input file contains the number of volunteers — N.
Next $N - 1$ lines contains 2 numbers v_i, u_i . It means that volunteer u_i is subordinate of volunteer v_i ($1 \le v_i, u_i \le N$).
Line number $N + 1$ contains M - number of queries.
Next M lines contains 2 numbers V_i, W_i - they define a query ($1 \le V_i \le N, 1 \le W \le 10^6$). W_i — total number of requests, V_i — number of volunteer.

Output

Print M lines. i-th line should contain number of ignored requests in i-th query.

Examples

volunteers.in	volunteers.out
5 1 2 1 3 2 4 2 5 2 1 10 1 20	5 0

Scoring

This problem contains 3 subtasks:

1. $1 \le N, M \le 1\,000$. This subtask worths 17 points.
2. $1 \le N, M \le 100\,000$. All queries are given to volunteer number 1. This subtask worths 24 points.
3. $1 \le N, M \le 100\,000$. This subtask worths 59 points.

Each of the next subtasks will be scored in case of all the previous subtasks are succesfully passed.

Problem B. Liar

Sergazy and Ulugbek love to play games. This task is about their favorite game.

Let $N = \{0, 1, \ldots, n-1\}$. In the other words, N is the set of all integers between 0 and $n-1$ inclusive. At the beginning, Sergazy chooses a single number x from N, but his choice is not revealed to Ulugbek. Ulugbek's goal is to find a subset of N that contains p distinct elements and **does not** contain x. We will use P to denote this subset. In order to find P, Ulugbek may ask Sergazy questions. In each question, Ulugbek chooses a set S, which is a nonempty subset of N, and asks "Does x belong to S?". Sergazy has to answer each question by saying either "yes"or "no".

Sergazy likes making life harder for Ulugbek, so when answering a question he can either tell the truth or lie. For example, assume that $x = 1$. If Ulugbek asks a query $S = \{0, 2\}$, Sergazy may lie by answering "yes" or he may tell the truth by answering "no". Luckily, Sergazy is a good friend and he promised Ulugbek that he will never lie k times in a row. That is, among any k **consecutive** questions there will be at least one that was answered truthfully.

Moreover, Sergazy is not only a liar, but he is also a tricky liar: he may change the chosen number during the game, but in such a way that does not contradict previous answers.

In addition, Sergazy will only answer to at most q questions in total. Ulugbek does not want to lose, so he asks for your help!

Example

Suppose that $n = 10$ and Sergazy chooses $x = 3$. Additionally, suppose that $k = 2$ and $p = 1$. That is, Sergazy will never lie twice in a row, and Ulugbek wants to find any 1-element subset of N that does not contain Sergazy's x.

Ulugbek: $S = \{9, 8, 7\}$

Sergazy: Yes (lie)

Ulugbek: $S = \{3, 9\}$

Sergazy: Yes (truth)

Ulugbek: $P = \{4\}$

Note that after his two questions Ulugbek can be certain that Sergazy's x is not 4.

Interaction protocol

Given n, k, p, q and a function to ask questions, find P. The parameters n, p, k and q are chosen in such a way that it is possible to find a set P that satisfies the required conditions. You need to implement one function in your solution.

$player\_answer(n, k, p, P)$, this function will be called by the grader exactly once

n: the size of the set N.

k: the number of consecutive questions among which at least one has to be answered truthfully.

p: the required size of the set P.

P: array of integers of length p; once you find the set P, you should write its elements into $P_0, ..., P_{p-1}$. Each element of P should be between 0 and $n-1$ inclusive.

This function has no return value.

Note that the value of q is not a parameter of $player\_answer$. The value of q is computed using a formula, which is different in each subtask. For details, see the Subtasks section.

Your program can ask Sergazy questions using the following function:

$get\_answer(S, sz)$ S: a non-empty array of distinct integers describing the set S you want to ask about.

Each element of S should be between 0 and $n-1$ inclusive. Note that the call to *get_answer* may modify the contents of array S. sz: the size of the set S. The elements of the set you want to ask about should be written in $S_0, ..., S_{sz-1}$. The function returns either 1, which means "yes" , or 0, which means "no" , depending on which answer jury's program decided to return. It is guaranteed that among any k consecutive queries, at least one truthful answer will be given.

Scoring

The task contains 5 subtasks

1. $n = 4, k = 2, p = 1, q = 2$. This subtask worths 9 points.
2. $4 \leq n \leq 10^4, k = 2, 1 \leq p \leq n-3, q = 2 \cdot p$. This subtask worths 10 points.
3. $2^k \leq n \leq 10^6, 2 \leq k \leq 18, p = 1, q = k$. This subtask worths 17 points.
4. $2^k \leq n \leq 10^6, 2 \leq k \leq 18, 1 \leq p \leq n - 2^k + 1, q = k \cdot p$. This subtask worths 18 points.
5. $3 * 2^k \leq n \leq 10^6, 2 \leq k \leq 18, 1 \leq p \leq n - 3 \cdot 2^k + 1, q = \max(\lfloor \frac{k \cdot p}{2} \rfloor, k)$. This subtask worths 46 points.

Note that $\lfloor x \rfloor$ denotes the greatest integer which is not greater than x. There is also an additional constraint for all subtasks: $n \cdot p \leq 4 \cdot 10^6$.

If your program tries to ask more queries than allowed, it will be terminated immediately, and it would be assumed that it produced an incorrect answer.

Sample grader

The sample grader reads the input in the following format:

line 1: n, k, p, q

line 2: x

The sample grader prints your answer and the number of queries you used and checks whether your answer is valid.

Problem C. Good segments

Input file: `segments.in`
Output file: `segments.out`

You are given an array a of length N which contains positive integers between 1 and N, inclusive. You need to calculate number of **good** segments in this array (two segments are different if they differ in at least one of the endpoints). Segment $[l, r]$ is good, if for any number x appearing in it, the number of occurrences of x in this segment is exactly x. For example, segment $(2, 1, 4, 4, 2, 4, 4)$ is good, and segment $(3, 3, 2, 2, 3, 3)$ is not (3 occurs 4 times).

Input

First line of input file contains integer number $N(1 \leq N \leq 500\,000)$. Second line contains N integers $a_1, a_2, \ldots, a_N$ $(1 \leq a_i \leq N)$.

Output

In the only line of the output file print exactly one number — number of good segments.

Examples

segments.in	segments.out
8 2 2 1 4 4 2 4 4	4

Scoring

This problem contains 8 subtasks:

1. $1 \leq N \leq 200$. This subtask worths 12 points.
2. $1 \leq N \leq 500$. This subtask worths 13 points.
3. $1 \leq N \leq 2000$. This subtask worths 12 points.
4. $1 \leq N \leq 5000$. This subtask worths 13 points.
5. $1 \leq N \leq 20\,000$. This subtask worths 12 points.
6. $1 \leq N \leq 50\,000$. This subtask worths 13 points.
7. $1 \leq N \leq 200\,000$. This subtask worths 12 points.
8. $1 \leq N \leq 500\,000$. This subtask worths 13 points.

Each subtask will be scored only in case when the solution passes all of the previous subtasks.