Problem A. Automaton

Source file name:	automaton.c, automaton.cpp, automaton.java, automaton.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales P. & Gabriel Gutierrez T. - UTP Colombia`

Pepito Pérez found the next definition in **Wikipedia**: a **Nondeterministic Finite Automaton (NFA)** is represented formally by a 5-tuple, $M = (Q,\ \Sigma,\ q_0,\ F,\ \Delta)$, consisting of:

- a finite set of states Q,
- a finite set of input symbols (Alphabet) Σ,
- an initial (or start) state $q_0 \in Q$,
- a set of states F distinguished as accepting (or final) states, $F \subseteq Q$
- a transition function $\Delta : Q \times \Sigma \to P(Q)$

Here, $P(Q)$ denotes the power set of Q. Let $w = a_1 a_2 \ldots a_n$ be a word over the alphabet Σ. The Nondeterministic Finite Automaton M accepts the word w if a sequence of states, r_0, r_1, ..., r_n, exists in Q with the following conditions:

1. $r_0 = q_0$
2. $r_{i+1} \in \Delta(r_i, a_{i+1})$, for $i = 0, 1, \ldots, n-1$
3. $r_n \in F$

In words, the first condition says that the machine starts in the start state q_0. The second condition says that given each character of string w, the machine will transition from state to state according to the transition function Δ. The last condition says that the machine accepts w if the last input of w makes the machine to be in one of the accepting states. In order for w being accepted by M it is not required that every state sequence ends in an accepting state, it is sufficient if one does. Otherwise, i.e. if it is impossible at all to get from q_0 to a state from F by following w, it is said that the automaton rejects the string. The set of strings that M accepts is the language recognized by M and this language is denoted by $L(M)$.

Pepito implemented the nondeterministic finite automaton but the execution time is too big for word recognition. *Pepito* has heard that every word could be recognized in $O(|w|)$, but *Pepito* has no idea on how to do it!, Can you help him?

Input

Input begins with an integer t ($1 \leq t \leq 10$), the number of test cases. The first line of the test case contains four space-separated positive integers n m s k ($1 \leq n \leq 18$, $1 \leq m \leq 8424$, $1 \leq s \leq 26$, $1 \leq k \leq n$), which represent, the number of states, the number of transitions, the number of alphabet symbols and the number of states of acceptance, respectively. The second line contains exactly s space-separated different symbols σ_1, σ_2, σ_3, ..., σ_s ($\sigma_i \in \{a,\ b,\ c,\ d,\ \ldots,\ z\}$, σ_i is any lowercase letter of the English alphabet). The alphabet Σ is sorted lexicographically. The third line contains exactly k space-separated nonnegative integer numbers F_1, F_2, ..., F_k ($0 \leq F_i < n$, for $0 \leq i \leq k$), these are the acceptation states. The test case continues with m lines each of them containing a transition of the automaton x y c ($0 \leq x, y < n$, $c \in \Sigma$), where x y and c are the origin, destination, and label of the transition, respectively. Then a

line is presented that contains a positive integer q that represents the total of words that are going to be evaluated in the automata, finally, q lines are presented, each one containing a word w ($w \in \Sigma^+$, $1 \leq |w| \leq 10^5$, that is, the words w have a length between 1 and 10^5)

Output

For each test case, you should print q single lines, each line containing the word `Yes` or `No` depending if the word $w \in L(M)$.

Example

Input	Output
1 5 12 2 2 a b 0 2 0 3 a 0 1 b 1 1 a 1 1 b 1 3 b 2 0 a 2 1 a 2 4 b 3 2 a 3 4 b 4 2 a 4 4 a 10 aaaaaaaa abababab bbbbaaa a b abaaba bbaab babab bbbaaba bbaabbaa	Yes No Yes No No Yes No No Yes Yes

Use fast I/O methods

Problem B. TBoxes

Source file name:	tboxes.c, tboxes.cpp, tboxes.java, tboxes.py
Input:	Standard
Output:	Standard
Author(s):	Carlos Andres Arias Londoño - UTP Colombia

Toby got N boxes that must sort in ascending way; when a box comes, Toby sorts it. He wants to know the sum between the smallest and largest box that he has sorted so far.

Input

A number N ($1 \leq N \leq 10^6$) is given, the total number of boxes that Toby must sort, N spaced separated numbers are given as the size of the i-th box (B_i ($1 \leq B_i \leq 10^9$)).

Output

There must be N numbers, the result of add-up the smallest and the largest after the i-th box is fixed.

Example

Input	Output
5 1 2 3 4 5	2 3 4 5 6

Use fast I/O methods

Problem C. Concatenated Prime Numbers

Source file name:	concatenated.c, concatenated.cpp, concatenated.java, concatenated.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Stiven Cardona Monsalve - UTP Colombia`

Since he was a puppy, Tobby has been really interested in prime numbers, to the point that the identification number of his license plate was a sequence of prime numbers of equal amount of digits, that is, being n a number of six digits, if and only if $n = d_1d_2d_3d_4d_5d_6$ where $d_1d_2d_3$ is a prime number and $d_4d_5d_6$ is another prime number, then we can affirm that n is a number created by three-digit prime numbers concatenated. Tobby needs your help to know if a number meets these conditions for a certain number of digits.

For example, the number 4347 is the concatenation of two two-digit prime numbers, since both, 43 and 47 are prime numbers. The 4347 is not a concatenation of one-digit prime numbers since 4 is not a prime number.

Input

Input has multiple test cases (no more than 10^5). Each test case consists of an integer number k ($1 \leq k \leq 7$) indicating the amount of digits of the prime numbers that will form the number, and a number n ($10^{k-1} \leq n \leq 10^{1000}$) which will be the number that Tobby wants to know if it is a concatenated prime number. The test cases should be read until reach the End Of File (EOF).

Note: It is guaranteed that the amount of digits of the number n it is a multiple of k.

Output

For each test case, you should print a single line. If n is a concatenation of k-digit prime numbers, the output will be "`:)`" otherwise the output will be "`:(`". The quotation marks should not be printed, they are only to clarify.

Example

Input	Output
1 2	:)
1 3	:)
2 12331719232931374143475359616771737983	:(
1 4347	:(
2 4347	:)
1 245	:(
2 130711	:)

Use fast I/O methods

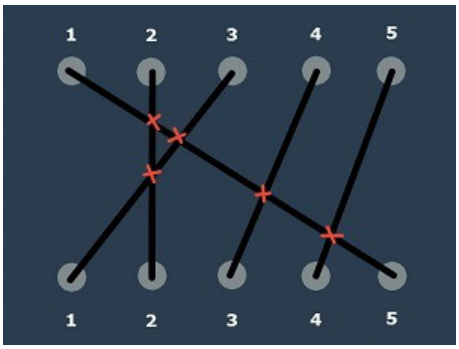
Problem D. Can not touch this

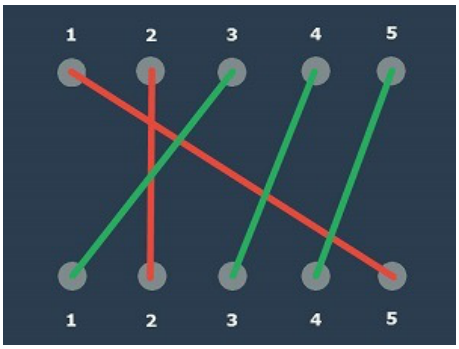
Source file name:	touchthis.c, touchthis.cpp, touchthis.java, touchthis.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Yeferson Gaitán Gómez - UTP Colombia`

Shhhhhhh! Toby is in trouble, he's deactivating a bomb and he needs a lot of concentration.

Hey! You can help him, Toby is so intelligent and he has discovered that the only way to deactivate the bomb is to cut the electricity cables in such a way that they do not touch each other.

Toby does not have too much time, he needs to cut the **minimum** number of electricity cables such that the bomb does not kill him!





In the example, if the red cables are cut, there are no crossing (touching) cables.

Input

Input begins with an integer t ($1 \leq t \leq 50$) the number of test cases, followed by t lines, each line contains an integer n ($1 \leq n \leq 10^5$), then n lines, each one with two numbers, indicating the two points that are joined by the electricity cable. Is guaranteed that in each point there will be exactly one cable that comes out of it.

Output

For each test case, you need to find the minimum number of cables that Toby has to cut so that the bomb doesn't kill him.

Example

Input	Output
1 5 1 3 2 2 3 4 4 5 5 1	2

Use fast I/O methods

Problem E. Find Phyta

Source file name:	findphyta.c, findphyta.cpp, findphyta.java, findphyta.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Yonny Mondelo - UCI Cuba`

Let's define a function **Phyta** over a sequence of integers as the largest sum that can be obtained by adding up some positive number of consecutive elements in the sequence. Then, given a sequence S with exactly N integer numbers, find the **XOR** between the **Phyta** of all non-empty segments of the sequence S. In other words, for each of the $N*(N+1)/2$ non-empty consecutive subsequences of S find its **Phyta**, and print the **XOR** of those values.

Input

The input consists of several test cases (but no more than 50), read until the end of file (EOF). The first line of each test case contains an integer N ($1 \leq N \leq 2000$) denoting the size of the sequence S. The second line contains exactly N space-separated integer numbers S_1, S_2, ..., S_N ($-10^9 \leq S_i \leq 10^9$ for $1 \leq i \leq N$) - representing the elements of S in that order.

Output

For each test case, your program must print a line with one integer number - the **XOR** between the **Phyta** of all non-empty segments of S.

Example

Input	Output
3 7 5 -13 6 11 4 4 13 11 5	-12 14

Use fast I/O methods

Problem F. Forever Alone

Source file name:	forever.c, forever.cpp, forever.java, forever.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Yeferson Gaitán Gómez - UTP Colombia`

Cupid is very bored, so he is now throwing many arrows to make the people fall in love with others, Cupid knows everyone well and has classified them into 60 different types, that is why when Cupid sees a group of people, he matches all the pairs that are of the same type and each pair will fall in love forever.

For example, if in a group of people there are $[2, 2, 1, 4, 5, 4, 2, 1]$, then Cupid will match the pairs: $(1, 2)$, $(3, 8)$ and $(4, 6)$, leaving "**Forever alone**" to 5 and 7.

Cupid wants to do his job very well, that's why he needs your help. Given n people and q queries, Cupid wants to know how many people will be "**Forever alone**" if he matches the group of people who are in the range $[l, r]$

Input

Input begins with an integer t $(1 \leq t \leq 50)$ indicating the number of test cases, followed by t cases, each case contains an integer n $(1 \leq n \leq 10^5)$, then n integers, indicating the type each person belongs to $(1 \leq a_i \leq 60)$. Then follows an integer q $(1 \leq q \leq 10^5)$, indicating the number of queries to be made by Cupid, each query will be in a new line and will have two integers, l and r $(1 \leq l \leq r \leq n)$, indicating the query range.

Output

For each query you must tell Cupid how many people will be "**Forever alone**" if he matches the group of people who are in the range $[l, r]$.

Example

Input	Output
1 6 8 7 9 7 7 8 4 1 3 4 5 2 5 1 6	3 0 2 2

Use fast I/O methods

Problem G. Table Game

Source file name:	tablegame.c, tablegame.cpp, tablegame.java, tablegame.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales P. & Gabriel Gutierrez T. - UTP Colombia`

Given the following game instructions: In a table there are N players. Initially all players start with the same amount D of money. One of the players is selected as the first player, the player next to him/her in clockwise direction is the second, and so on until the last player (Position N).Thus, the first player comes after the last one. At each turn of the game a coin, a red dice and a blue dice are thrown. The coin indicates movement direction, Heads ("cara" in Spanish) for clockwise and Tails ("sello" in Spanish) for counter-clockwise. The value of the red dice (R) indicates the total of movements and the blue dice (A) the amount of money given by the "bank" to the reached player located at R seats from the current player. Now, if the amount of money accumulated by the reached player is even, the two players located immediately at his/her right and left side are eliminated from the game and they both loose all the money they have. That money then goes to the Bank. In the other case (an odd amount of money) the player is "saved" from the game, which means that the player is eliminated, but gets to keep the money he/she accumulated. The player who gets to make the next move is given by the eliminated player's coin flip.

The game ends when only one player lefts and itâs also "saved".

You must compute the positions (in ascending order) of the "saved" players and the amount of money they had when they left the game.

Input

The input contains several test cases. Each test case begins with a line containing two values N ($2 \leq N \leq 10^4$) and D ($1 \leq D \leq 10^3$), representing the total players and initial amount of money of each player. Then $N-1$ lines follow, representing the turns that can be used at maximum. Each Line contains three values M ($M \in \{C, S\}$), R and A ($1 \leq R, A \leq 10^3$), representing respectively: The value of the coin toss (Heads ('C') o Tails ('S')), the value of the red dice and the value of the blue dice.

The input ends with EOF.

Output

For each test case you must print as many lines as "saved" players that finished the game. Each line must contain, the "saved" player position and the amount of money acquired by that player at the moment of elimination.

Example

Input	Output
5 3 C 1 2 S 2 3 C 3 2 C 2 5 10 2 S 5 3 C 3 2 C 4 2 S 5 4 S 6 5 S 5 3 C 3 2 S 6 5 C 4 2	2 5 3 5 5 6 1 2 4 13 6 5 9 7

Use fast I/O methods

Problem H. How Many Rectangles

Source file name:	rectangles.c, rectangles.cpp, rectangles.java, rectangles.py
Input:	Standard
Output:	Standard
Author(s):	Gabriel Gutierrez Tamayo - UTP Colombia

In this problem consider a matrix that will only contain the symbols '.' and '#', you must count the number of rectangles. If exists a rectangular edge (containing only symbols '#'), then this will correspond to a rectangle, regardless of the content of the inner boxes. Only rectangles with height and width greater than or equal to 3 are valid. The smallest rectangle that can appear in the matrix is:

```
###
#.#
###
```

Input

The input consists of several test cases, no more than 100, the first line of each test case contains two integers N and M ($3 \leq N, M \leq 350$), the next N lines contain M characters '.' or '#'.

Output

For each test case print the number of rectangles in the matrix.

Example

Input	Output
3 3 ### #.# ### 3 3 ### ### ### 5 5 ###.. #.### ##### ##### ####. 6 8 ######## ######## ######## ######## ######## ########	1 1 7 210

Use fast I/O methods

Problem I. Interval

Source file name:	interval.c, interval.cpp, interval.java, interval.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

Let's take a list L containing n ($1 \leq n \leq 10^6$) positive integer numbers in the interval $[1, 10^5]$, and assume you have three operations to work on it: Query, Insert and Delete.

The format for each operation is: p y, p indicates the type of the operation, 1 = Query, 2 = Insert, and 3 = Delete. y represents the value where the operation will be applied.

The **Query** operation must print a pair of numbers x z, such that $x = \max(\{i \in L \mid i < y\})$, and $z = \min(\{i \in L \mid y < i\})$

The **Insert** operation adds y to the list.

The **Delete** operation removes one occurrence of the element y in L, if the element is present.

Note 1: For the **Query** and **Delete** the element y might not exist.

Nota 2: There can be several occurrences of y inside the list.

Input

Input begins with an integer t ($1 \leq t \leq 10$), the number of test cases. The first line of the test contains a positive integer n ($1 \leq n \leq 10^6$), with the lenght on the list L. The second line contains exactly n space-separated positive integer numbers L_1, L_2, L_3, ..., L_n ($1 \leq L_i \leq 10^5$, for $1 \leq i \leq n$), that is the list L. The test case continues with a line containing a number q ($1 \leq q \leq 10^5$), the number of operations to apply over the list. In each one of the next q lines two integers p y, ($p \in \{1,\ 2,\ 3\}$, $1 \leq y \leq 10^5$) are presented.

Output

For each query of the type 1, you should print one single line containing the two integers x z. If there is not such value x then $x = -1$, if there is not such value z then $z = 100001$.

Input	Output
2 5 4 4 4 4 4 8 1 4 1 3 1 5 2 1 2 8 1 4 3 4 1 3 5 10 1 8 3 5 7 1 5 3 5 2 6 1 5 3 8 2 12 1 8	-1 100001 -1 4 4 100001 1 8 1 4 3 8 3 6 6 10

Use fast I/O methods

Problem J. Toby and stacks

Source file name:	tobystacks.c, tobystacks.cpp, tobystacks.java, tobystacks.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Manuel Felipe Pineda Loaiza - UTP Colombia`

Toby is learning about data structures, and today's data structure is the stack.

A stack is an abstract data type that serves as a collection of elements, with two main operations:

- push, which adds an element to the collection, and
- pop, which removes the most recently added element that was not removed yet.

In order to challenge his friends, Toby wants to create a lot of stacks doing the following operations:

- Take an existing stack, copy it and push one element.
- Take an existing stack, copy it and pop one element.

Note: Toby always stores the new stack.

Toby needs your help to write a program that can do all this operations efficiently.

Input

The input begins with an integer N ($1 \leq N, x \leq 10^6$) denoting the total number of operations. Each one of the next N lines describe one of the following two operations:

id 0 x, which means: "push" the element x to the copy of the stack number id

id 1, which means: "pop" the most recently added element from the copy of the stack number id

Notes:

- Suppose there is an initially empty stack with no elements and id 0.
- The id of the query is always less than the id of the new stack.
- The id for the new stack (created in each query) is a consecutive number starting with 1, in other words, the total number of copies so far.

Output

For each **push** operation, print the size of the **new stack**.

For each **pop** operation, print the element that was removed. Print -1 if the stack is empty.

Example

Input	Output
10 0 1 1 1 1 0 25 0 0 0 4 1 0 1 1 0 11 5 1 5 1 6 0 58	-1 -1 1 1 0 -1 1 -1 -1 1

Use fast I/O methods

Problem K. Universal Language II

Source file name:	universal2.c, universal2.cpp, universal2.java, universal2.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

In the holiday season at the beginning of the year Professor *Humbertov Moralov* is preparing some subjects for his course of *Automata Theory and Formal Languages*. He has given special attention to a procedure that enumerates all the words in the universal language (Σ^*) generated from an alphabet Σ:

Let $\Sigma = \{a,\ b\}$ be one sample alphabet, and let $k \in \mathbb{N}$ be the length of the words. For each k, there is a finite number of words over Σ with length k.

The words are lexicographically sorted. For convenience the empty string λ is enumerated as 0, then the words of length 1 are enumerated. In general the strings with length $k+1$ are enumerated after the ones with length k. Thus:

$$\lambda \ldots 0$$
$$a \ldots 1$$
$$b \ldots 2$$
$$aa \ldots 3$$
$$ab \ldots 4$$
$$ba \ldots 5$$
$$bb \ldots 6$$
$$aaa \ldots 7$$
$$\vdots$$

and so on.

Now Professor Humbertov Moralov needs your help to determine the word w that is located at the position p in the enumeration given by the procedure explained before over an alphabet Σ.

Input

Input begins with an integer t ($1 \leq t \leq 10^3$), the number of test cases. The first line of each test case contains an integer m ($1 \leq m \leq 26$), the size of the alphabet Σ. The second line contains exactly m space-separated different symbols σ_1, σ_2, σ_3, ..., σ_m ($\sigma_i \in \{a,\ b,\ c,\ d,\ \ldots,\ z\}$, σ_i is any lowercase letter of the English alphabet). The alphabet Σ is sorted lexicographically. The test case continues with a line containing a number n ($1 \leq n \leq 10^3$), indicating the amount of positions to be consulted. Finally each of the following n lines contain a positive integer p ($1 \leq p \leq 10^{17}$).

Output

For each test case, you should print n single lines, each one containing one word w ($1 \leq |w| \leq 56$), which is located in the position p in the enumeration procedure over the alphabet Σ.

Example

Input	Output
2 1 d 4 1 2 3 4 4 a m o r 6 112 313 104 35 281 193	d dd ddd dddd amor roma amar aro rama mora

Use fast I/O methods

Problem L. Longest Interval

Source file name:	longest.c, longest.cpp, longest.java, longest.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Manuel Felipe Pineda Loaiza - UTP Colombia`

Initially you have an empty set of intervals.

In each step you can perform one of the following operations:

- Find the lenght of the longest interval that contains x i. e. $\max(R_i - L_i + 1), \forall i \mid L_i \leq x \leq R_i$
- Add the interval $[L, R]$ to the set of intervals
- Remove some previously added interval

Input

The input begins with a number N ($1 < N < 10^5$), each one of the following N lines begins with a number $op \in \{0, 1, 2\}$ which represents one of the following types of query.

- 0 x_i, prints the lenght of the longest interval that contains x_i ($0 \leq x_i < N$).
- 1 l_i r_i, adds an interval to the set of intervals ($0 \leq l_i \leq r_i < N$).
- 2 k_i, removes the k-th interval. k is the identifier of the interval in the input starting at 0.

Note: No interval will be removed more than once.

Output

For each query of type 0, print the lenght of the longest interval that contains x.

Example

Input	Output
10 1 1 7 1 6 6 1 1 5 1 7 9 0 5 0 6 0 9 2 0 0 3 0 6	7 7 3 5 1

Use fast I/O methods

Problem M. How Many Digits does N have? II

Source file name:	howmany.c, howmany.cpp, howmany.java, howmany.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

Recently, Professor *Humbertov Moralov* was organizing his workplace, he needed space to be able to locate his christmas decorations. For this reason, he began by selecting documents to either throw in the trash or to recycle. In this process he found some freehand notes with the puzzle: How many digits does $N = 2^{2003} \cdot 5^{2007}$ have?

Using this interesting puzzle as our starting point, the problem you are asked to solve now is: Given two nonnegative integers a, b ($0 \leq a,\ b \leq 10^5$), compute the total number of occurrences of the digits 0, 1, 2, ..., 9. And finally the total number of digits of $N = 2^a \cdot 5^b$.

Note: it is guaranteed that $|a - b| \leq 5 \times 10^3$.

Input

Input begins with an integer t ($1 \leq t \leq 4 \times 10^5$), the number of test cases, followed by t lines, each one containing two nonnegative integers a, b ($0 \leq a,\ b \leq 10^5, \quad |a - b| \leq 5 \times 10^3$).

Output

For each test case, you should print out eleven single lines. The first 10 lines contain the number of occurrences of each digit. The eleventh line contains a positive integer with the total number of digits of N.

Example

Input	Output
4 10 8 8 10 2003 0 0 2007	8 0 0 0 1 0 0 0 0 0 9 8 0 1 0 0 1 0 0 0 0 10 68 71 62 70 57 58 66 58 46 47 603 145 148 131 145 137 128 124 149 140 156 1403

Use fast I/O methods