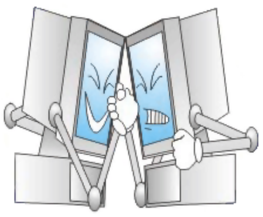


Pripreme za IOI 2008
06. avgust 2008. godine
predavač: Slobodan Mitrović
e-mail: boba5555@gmail.com

Strukture podataka

Zadatak 1. [SPOJ Increasing Subsequences] Given a sequence of $N(1 \leq N \leq 10,000)$ integers $S_1, ..., S_N(0 \leq Si < 100,000)$, compute the number of increasing subsequences of S with length $K(1 \leq K \leq 50$ and $K \leq N)$; that is, the number of K-tuples $i_1, ..., i_K$ such that $1 \leq i_1 < ... < i_K \leq N$ and $S_{i1} < ... < S_{iK}$.

Input The first line contains the two integers N and K. The following N lines contain the integers of the sequence in order.

Output Print a single integer representing the number of increasing subsequences of S of length K, modulo 5,000,000.

Input	**Output**

4 3
1
2
2
10
	2

Zadatak 2. [SPOJ Distinct Increasing Subsequences] Given a sequence of $N(1 \leq N \leq 10,000)$ integers $S_1, ..., S_N(0 \leq S_i < 1,000,000,000)$, compute the number of distinct increasing subsequences of S with length $K(1 \leq K \leq 50$ and $K \leq N)$.

Input The first line contains the two integers N and K. The following N lines contain the integers of the sequence in order.

Output Print a single integer representing the number of distinct increasing subsequences of S of length K, modulo 5,000,000.

Input	**Output**

4 3
1
2
2
10
	1

Zadatak 3. [SPOJ Intervals] You are given n closed integer intervals $[a_i, b_i]$ and n integers $c_1, ..., c_n$. Write a program that:

- reads the number of intervals, their endpoints and integers $c_1, ..., c_n$ from the standard input,

- computes the minimal size of a set Z of integers which has at least c_i common elements with interval $[a_i, b_i]$, for each $i = 1, 2, ..., n$,
- writes the answer to the standard output.

Input The input begins with the integer t, the number of test cases. Then t test cases follow.

For each test case the first line of the input contains an integer $n(1 \leq n \leq 50000)$ - the number of intervals. The following n lines describe the intervals. Line $(i + 1)$ of the input contains three integers a_i, b_i and c_i separated by single spaces and such that $0 \leq a_i \leq b_i \leq 50000$ and $1 \leq c_i \leq b_i - a_i + 1$.

Output For each test case the output contains exactly one integer equal to the minimal size of set Z sharing at least c_i elements with interval $[a_i, b_i]$, for each $i = 1, 2, ..., n$.

Input	**Output**

1
5
3 7 3
8 10 3
6 8 1
1 3 1
10 11 1
	6

Zadatak 4. [SPOJ Matrix Summation] A NxN matrix is filled with numbers. BuggyD is analyzing the matrix, and he wants the sum of certain submatrices every now and then, so he wants a system where he can get his results from a query. Also, the matrix is dynamic, and the value of any cell can be changed with a command in such a system.

Assume that initially, all the cells of the matrix are filled with 0. Design such a system for BuggyD. Read the input format for further details.

Input The first line of the input contains an integer t, the number of test cases. t test cases follow.

The first line of each test case contains a single integer $N(1 \leq N \leq 1024)$, denoting the size of the matrix.

A list of commands follows, which will be in one of the following three formats (quotes are for clarity):

- "SET x y num" - Set the value at cell (x, y) to $num(0 \leq x, y < N)$.
- "SUM x1 y1 x2 y2" - Find and print the sum of the values in the rectangle from $(x1, y1)$ to $(x2, y2)$, inclusive. You may assume that $x1 \leq x2$ and $y1 \leq y2$, and that the result will fit in a signed 32-bit integer.
- "END" - Indicates the end of the test case.

Output For each test case, output one line for the answer to each "SUM" command. Print a blank line after each test case.

```
Input                  Output
1                      1
4                      12
SET 0 0 1              12
SUM 0 0 3 3            13
SET 2 2 12
SUM 2 2 2 2
SUM 2 2 3 3
SUM 0 0 2 2
END
```

Zadatak 5. [SPOJ Counting inversions] You are given a sequence A of $N (N \leq 250000)$ integers between 1 and 50000. On this sequence you have to apply $M (M \leq 10000)$ operations of the form: modify the i-th element in the sequence and then say how many inversions are there in the sequence. The number of inversions in a sequence is given by the number of pairs (i, j) with $i < j$ and $Ai > Aj$.

Input The first line of input contains the number N and the next line contains the numbers that form the sequence. After that follows the number M and then M lines, each containig 2 integers X and Y, meaning that new value of the X-th element of the sequence is Y and that you should count the number of inversions in the modified sequence.

Output Output must contain M lines, the i-th line of output containg the number of inversions in the sequence after the first i operations.

```
Input                  Output
10                     17
2 6 6 4 7 6 3 5 9 1    18
7                      16
8 8                    13
5 1                    14
5 6                    8
10 5                   6
7 1
10 10
4 6
```

Zadatak 6. [Timus Factory] There are three machines on the new toy factory: A, B and C. The factory makes toys by processing each toy on these machines in order A, B, C. Your task is to create N toys as soon as possible. You know the time to process each toy on each machine: a_i, b_i and c_i. You can select an arbitrary order of processing toys. The second machine is so fast that at least one of the following two statements holds: $max(b_i) \leq min(a_i)$ or $max(b_i) \leq min(c_i)$.

Input The first line of the input contains the number of toys $N (1 \leq N \leq 100000)$. The next N lines contain three integers each: a_i, b_i and $c_i (1 \leq a_i, b_i, c_i \leq 1000000)$.

Output Output the minimal possible processing time on the first line. The second line must contain an example of optimal processing order a permutation of toy numbers from 1 to N.

Input	**Output**
5	33
3 1 6	2 1 3 5 4
1 1 2	
5 2 5	
7 1 4	
10 2 8	
1	16
5 4 7	1

Zadatak 7. [USACO 2008 Februar GOLD - Hotel] The cows are journeying north to Thunder Bay in Canada to gain cultural enrichment and enjoy a vacation on the sunny shores of Lake Superior. Bessie, ever the competent travel agent, has named the Bullmoose Hotel on famed Cumberland Street as their vacation residence. This immense hotel has $N (1 \leq N \leq 50,000)$ rooms all located on the same side of an extremely long hallway (all the better to see the lake, of course).

The cows and other visitors arrive in groups of size $D_i (1 \leq D_i \leq N)$ and approach the front desk to check in. Each group i requests a set of D_i contiguous rooms from Canmuu, the moose staffing the counter. He assigns them some set of consecutive room numbers $r..r + D_{i-1}$ if they are available or, if no contiguous set of rooms is available, politely suggests alternate lodging. Canmuu always chooses the value of r to be the smallest possible.

Visitors also depart the hotel from groups of contiguous rooms. Checkout i has the parameters X_i and D_i which specify the vacating of rooms $X_i..X_i + D_{i-1} (1 \leq X_i \leq N - D_{i+1})$. Some (or all) of those rooms might be empty before the checkout.

Your job is to assist Canmuu by processing $M (1 \leq M < 50,000)$ checkin/checkout requests. The hotel is initially unoccupied.

Input

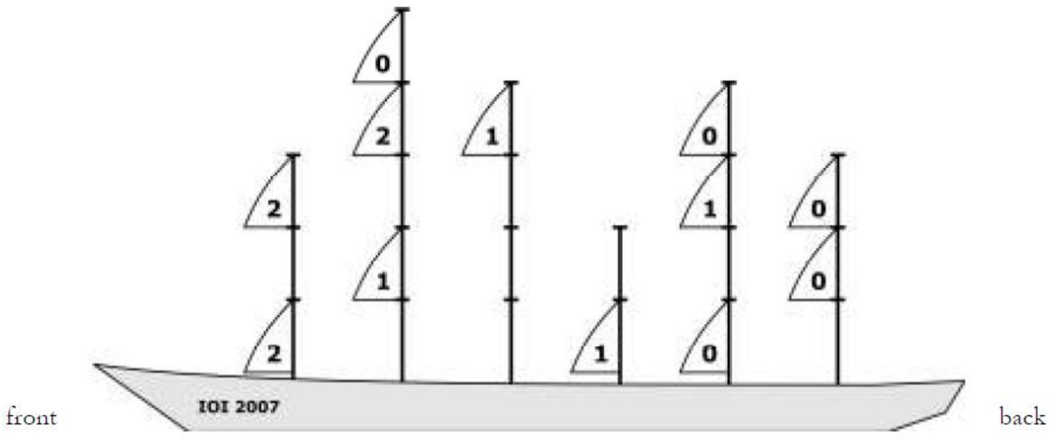
- Line 1: Two space-separated integers: N and M
- Lines 2..M+1: Line i+1 contains request expressed as one of two possible formats: (a) Two space separated integers representing a check-in request: 1 and D_i (b) Three space-separated integers representing a check-out: 2, X_i, and D_i

Output Lines 1.....: For each check-in request, output a single line with a single integer r, the first room in the contiguous sequence of rooms to be occupied. If the request cannot be satisfied, output 0.

Input	**Output**
10 6	1
1 3	4
1 3	7
1 3	0
1 3	5
2 5 5	
1 6	

Zadatak 8. [IOI 2007 Sails] A new pirate sailing ship is being built. The ship has N masts (poles) divided into unit sized segments the height of a mast is equal to the number of its segments. Each

mast is fitted with a number of sails and each sail exactly fits into one segment. Sails on one mast can be arbitrarily distributed among different segments, but each segment can be fitted with at most one sail.
Different configurations of sails generate different amounts of thrust when exposed to the wind. Sails in front of other sails at the same height get less wind and contribute less thrust. For each sail we define its inefficiency as the total number of sails that are behind this sail and at the same height. Note that "in front of" and "behind" relate to the orientation of the ship: in the figure below, "in front of" means to the left, and "behind" means to the right.
The total inefficiency of a configuration is the sum of the inefficiencies of all individual sails.



This ship has 6 masts, of heights 3, 5, 4, 2, 4 and 3 from front (left side of image) to back.
This distribution of sails gives a total inefficiency of 10. The individual inefficiency of each sail is written inside the sail.

Write a program that, given the height and the number of sails on each of the N masts, determines the smallest possible total inefficiency.

Input The first line of input contains an integer $N(2 \leq N \leq 100000)$, the number of masts on the ship. Each of the following N lines contains two integers H and $K(1 \leq H \leq 100000, 1 \leq K \leq H)$, the height and the number of sails on the corresponding mast. Masts are given in order from the front to the back of the ship.

Output Output should consist of a single integer, the smallest possible total inefficiency.

Input	**Output**

6
3 2
5 3
4 1
2 1
4 3
3 2
	10

Zadatak 9. [USACO 2006 April GOLD - The Milk Queue] Every morning, Farmer John's $N(1 \leq N \leq 25,000)$ cows all line up for milking. In an effort to streamline the milking process, FJ has designed a two-stage milking process where the line of cows progresses through two barns in sequence, with milking taking part sequentially in both barns. Farmer John milks cows one by one as they go through the first barn, and his trusty sidekick Farmer Rob milks the cows (in the same order) as they are released from the first barn and enter the second barn.

Unfortunately, Farmer John's insistence that the cows walk through both barns according to a single ordering leads to some inefficiencies. For example, if Farmer John takes too long to milk a particular cow, Farmer Rob might end up sitting idle with nothing to do for some time. On the other hand, if Farmer John works too fast then we might end up with a long queue of cows waiting to enter the second barn.

Please help Farmer John decide on the best possible ordering of cows to use for the milking, so that the last cow finishes milking as early as possible. For each cow i we know the time $A(i)$ required for milking in the first barn and the time $B(i)$ required for milking in the second barn. Both $A(i)$ and $B(i)$ are in the range 1...20,000.

Input

- Line 1: A single integer, N.
- Lines 2..1+N: Line i+1 contains two space-separated integers $A(i)$ and $B(i)$ for cow i.

Output Line 1: The minimum possible time it takes to milk all the cows, if we order them optimally.

Input	**Output**
3	16
2 2	
7 4	
3 5	

Input details: There are three cows. Cow 1 takes 2 units of time in both barns, cow 2 takes 7 units of time in the first barn and 4 in the second, and cow 3 takes 3 units of time in the first barn and 5 in the second.

Output details: Ordering the cows in the order 3, 1, 2, will allow for milking to complete within only 16 units of time.

Zadatak 10. [SPOJ Card Trick] The magician shuffles a small pack of cards, holds it face down and performs the following procedure:

1. The top card is moved to the bottom of the pack. The new top card is dealt face up onto the table. It is the Ace of Spades.
2. Two cards are moved one at a time from the top to the bottom. The next card is dealt face up onto the table. It is the Two of Spades.
3. Three cards are moved one at a time
4. This goes on until the nth and last card turns out to be the n of Spades.

This impressive trick works if the magician knows how to arrange the cards beforehand (and knows how to give a false shuffle). Your program has to determine the initial order of the cards for a given number of cards, $1 \leq n \leq 20000$.

Input On the first line of the input is a single positive integer, telling the number of test cases to follow. Each case consists of one line containing the integer n.

Output For each test case, output a line with the correct permutation of the values 1 to n, space separated. The first number showing the top card of the pack, etc...

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

Zadatak 11. [Savezno Srbija Terorista] Najveća zgrada u svetskoj prestionici Beonisu ima N spratova. Na svakom spratu se nalazi bazen. Terorista Joca želi da sruši zgradu, tako sto će porušiti prvi sprat, a samim tim i celu zgradu. Za svaki sprat su poznate sledeće informacije:

- W_i - težina vode koja se nalazi u bazenu na i-tom spratu na početku
- L_i - težina vode koju i-ti sprat može da izdrži
- C_i - cena dinamita potrebnog da se poruši i-ti sprat

Sva voda iz srušenog sprata odlazi na sledeći niži. Ako težina vode na nekom spratu postane veća od dozvoljene koju može da izdrži, pod se ruši i sva voda prelazi nadole.

Vas zadatak je da pomognete teroristi Joci da sa što manje novca sruši prvi sprat.

Ulaz U prvom redu ulaza je prirodan broj $N (1 \leq N \leq 100000)$, koji predstavlja broj spratova. U svakom od sledećih N redova nalaze se po tri cela broja $W_i, L_i, C_i (0 \leq W_i, L_i, C_i \leq 2000000000)$, veličine naznačene u tekstu zadatka za i-ti sprat. Suma svih W_i i C_i za $(1 \leq i \leq N)$ će uvek biti manja od 2000000000.

Izlaz Na standardni izlaz ispisati minimalnu količinu novca potrebnu da se banka potopi. Zatim u sledećih nekoliko redova upisati redne brojeve spratova koje treba srušiti za minimalno rešenje, u redosledu rušenja.

Ulaz	**Izlaz**
4	5
10 50 100	3
0 100 3	2
100 100 2	
10 50 6	

Objašenjenje: Terorista ruši treći, pa drugi sprat sa minimalnom cenom 5. Drugi sprat se ne ruši sam, jer je težina vode nakon eksplozije jednaka trećini izdržljivosti. Kada bi se rušio samo prvi sprat cena bi bila 100, a kada bi se rušio četvrti cena bi bila 6.

Zadatak 12. [SPOJ Brackets] We will call a **bracket word** any word constructed out of two sorts of characters: the opening bracket "(" and the closing bracket ")". Among these words we will distinguish **correct bracket expressions**. These are such bracket words in which the brackets can be matched into pairs such that

- every pair consists of an opening bracket and a closing bracket appearing further in the bracket word
- for every pair the part of the word between the brackets of this pair has equal number of opening and closing brackets

On a bracket word one can do the following operations:

- **replacement** – changes the i-th bracket into the opposite one
- **check** – if the word is a correct bracket expression

Write a program which

- reads (from standard input) the bracket word and the sequence of operations performed,
- for every check operation determines if the current bracket word is a correct bracket expression,
- writes out the outcome (to standard output).

Input Ten test cases (given one under another, you have to process all!). Each of the test cases is a series of lines. The first line of a test consists of a single number $n (1 \leq n \leq 30000)$ denoting the length of the bracket word. The second line consists of n brackets, not separated by any spaces. The third line consists of a single number m – the number of operations. Each of the following m lines carries a number k denoting the operation performed. $k = 0$ denotes the check operation, $k > 0$ denotes replacement of k-th bracket by the opposite.

Output For every test case your program should print a line:
Test i:
where i is replaced by the number of the test and in the following lines, for every check operation in the i-th test your program should print a line with the word YES, if the current bracket word is a correct bracket expression, and a line with a word NO otherwise. (There should be as many lines as check operations in the test.)

Input
```
4
()((
4
4
0
2
0
and 9 test cases more
```

Output
```
Test 1:
YES
NO
and 9 test cases more
```