Educational Codeforces Round 127 (Rated for Div. 2)

A. String Building

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given a string s. You have to determine whether it is possible to build the string s out of strings aa, aaa, bb and/or bbb by concatenating them. You can use the strings aa, aaa, bb and/or bbb any number of times and in any order.

For example:

- aaaabbb can be built as aa $+$ aa $+$ bbb;
- bbaaaaabbb can be built as bb $+$ aaa $+$ aa $+$ bbb;
- aaaaaa can be built as aa $+$ aa $+$ aa;
- abab cannot be built from aa, aaa, bb and/or bbb.

Input

The first line contains one integer t $(1 \le t \le 1000)$ — the number of test cases.

Each test case consists of one line containing the string s ($1 \le |s| \le 50$), consisting of characters a and/or b.

Output

For each test case, print YES if it is possible to build the string s. Otherwise, print NO.

You may print each letter in any case (for example, YES, yes, Yes will all be recognized as positive answer, NO, no and nO will all be recognized as negative answer).

Example

input

```
8
aaaabbb
bbaaaaabbb
aaaaaa
abab
a
b
aaaab
bbaaa
```

output

```
YES
YES
YES
NO
NO
NO
NO
YES
```

Note

The first four test cases of the example are described in the statement.

B. Consecutive Points Segment

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given n points with integer coordinates on a coordinate axis OX. The coordinate of the i-th point is x_i. All points' coordinates are distinct and given in strictly increasing order.

For each point i, you can do the following operation **no more than once**: take this point and move it by 1 to the left or to the right (i..e., you can change its coordinate x_i to $x_i - 1$ or to $x_i + 1$). In other words, for each point, you choose (separately) its new coordinate. For the i-th point, it can be either $x_i - 1$, x_i or $x_i + 1$.

Your task is to determine if you can move some points as described above in such a way that the new set of points forms a **consecutive segment** of integers, i. e. for some integer l the coordinates of points should be equal to $l, l + 1, \ldots, l + n - 1$.

Note that the resulting points should have **distinct** coordinates.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t ($1 \le t \le 2 \cdot 10^4$) — the number of test cases. Then t test cases follow.

The first line of the test case contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of points in the set x.

The second line of the test case contains n integers $x_1 < x_2 < \ldots < x_n$ ($1 \le x_i \le 10^6$), where x_i is the coordinate of the i-th point.

It is guaranteed that the points are given in strictly increasing order (this also means that all coordinates are distinct). It is also guaranteed that the sum of n does not exceed $2 \cdot 10^5$ ($\sum n \le 2 \cdot 10^5$).

Output

For each test case, print the answer — if the set of points from the test case can be moved to form a consecutive segment of integers, print YES, otherwise print NO.

Example

input

```
5
2
1 4
3
1 2 3
4
1 2 3 7
1
1000000
3
2 5 6
```

output

```
YES
YES
NO
YES
YES
```

C. Dolce Vita

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Turbulent times are coming, so you decided to buy sugar in advance. There are n shops around that sell sugar: the i-th shop sells one pack of sugar for a_i coins, but only **one pack to one customer** each day. So in order to buy several packs, you need to visit several shops.

Another problem is that prices are increasing each day: during the first day the cost is a_i, during the second day cost is $a_i + 1$, during the third day — $a_i + 2$ and so on for each shop i.

On the contrary, your everyday budget is only x coins. In other words, each day you go and buy as many packs as possible with total cost not exceeding x. Note that if you don't spend some amount of coins during a day, you can't use these coins during the next days.

Eventually, the cost for each pack will exceed x, and you won't be able to buy even a single pack. So, how many packs will you be able to buy till that moment in total?

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. Next t cases follow.

The first line of each test case contains two integers n and x ($1 \le n \le 2 \cdot 10^5$; $1 \le x \le 10^9$) — the number of shops and your everyday budget.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the initial cost of one pack in each shop.

It's guaranteed that the total sum of n doesn't exceed $2 \cdot 10^5$.

Output

For each test case, print one integer — the total number of packs you will be able to buy until prices exceed your everyday budget.

Example

input

```
4
3 7
2 1 2
5 9
10 20 30 40 50
```

```
1 1
1
2 1000
1 1
```

output

```
11
0
1
1500
```

Note

In the first test case,

- Day 1: prices are $[2, 1, 2]$. You can buy all 3 packs, since $2 + 1 + 2 \le 7$.
- Day 2: prices are $[3, 2, 3]$. You can't buy all 3 packs, since $3 + 2 + 3 > 7$, so you buy only 2 packs.
- Day 3: prices are $[4, 3, 4]$. You can buy 2 packs with prices 4 and 3.
- Day 4: prices are $[5, 4, 5]$. You can't buy 2 packs anymore, so you buy only 1 pack.
- Day 5: prices are $[6, 5, 6]$. You can buy 1 pack.
- Day 6: prices are $[7, 6, 7]$. You can buy 1 pack.
- Day 7: prices are $[8, 7, 8]$. You still can buy 1 pack of cost 7.
- Day 8: prices are $[9, 8, 9]$. Prices are too high, so you can't buy anything.

In total, you bought $3 + 2 + 2 + 1 + 1 + 1 + 1 = 11$ packs.
In the second test case, prices are too high even at the first day, so you can't buy anything.

In the third test case, you can buy only one pack at day one.

In the fourth test case, you can buy 2 packs first 500 days. At day 501 prices are $[501, 501]$, so you can buy only 1 pack the next 500 days. At day 1001 prices are $[1001, 1001]$ so can't buy anymore. In total, you bought $500 \cdot 2 + 500 \cdot 1 = 1500$ packs.

D. Insert a Progression

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a sequence of n integers $a_1, a_2, \dots, a_n$. You are also given x integers $1, 2, \dots, x$.

You are asked to insert each of the extra integers into the sequence a. Each integer can be inserted at the beginning of the sequence, at the end of the sequence, or between any elements of the sequence.

The score of the resulting sequence a' is the sum of absolute differences of adjacent elements in it $\left(\sum\limits_{i=1}^{n+x-1} |a'_i - a'_{i+1}|\right)$.

What is the smallest possible score of the resulting sequence a'?

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of testcases.

The first line of each testcase contains two integers n and x ($1 \le n, x \le 2 \cdot 10^5$) — the length of the sequence and the number of extra integers.

The second line of each testcase contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$).

The sum of n over all testcases doesn't exceed $2 \cdot 10^5$.

Output

For each testcase, print a single integer — the smallest sum of absolute differences of adjacent elements of the sequence after you insert the extra integers into it.

Example

input

```
4
1 5
10
3 8
7 2 10
10 2
6 1 5 7 3 3 9 10 10 1
4 10
1 3 1 2
```

output

```
9
15
31
13
```

Note

Here are the sequences with the smallest scores for the example. The underlined elements are the extra integers. Note that there exist other sequences with this smallest score.

- $\underline{1}, \underline{2}, \underline{3}, \underline{4}, \underline{5}, 10$
- $\underline{7}, 7, \underline{6}, \underline{4}, 2, \underline{2}, \underline{1}, \underline{3}, \underline{5}, \underline{8}, 10$
- $6, \underline{1}, 1, \underline{2}, 5, 7, 3, 3, 9, 10, 10, 1$
- $1, 3, \underline{1}, 1, 2, \underline{2}, \underline{3}, \underline{4}, \underline{5}, \underline{6}, \underline{7}, \underline{8}, \underline{9}, \underline{10}$

E. Preorder

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given a rooted tree of $2^n - 1$ vertices. Every vertex of this tree has either 0 children, or 2 children. All leaves of this tree have the same distance from the root, and for every non-leaf vertex, one of its children is the left one, and the other child is the right one. Formally, you are given a *perfect binary tree*.

The vertices of the tree are numbered in the following order:

- the root has index 1;
- if a vertex has index x, then its left child has index $2x$, and its right child has index $2x + 1$.

Every vertex of the tree has a letter written on it, either A or B. Let's define the character on the vertex x as s_x.

Let the *preorder string* of some vertex x be defined in the following way:

- if the vertex x is a leaf, then the *preorder string* of x be consisting of only one character s_x;
- otherwise, the *preorder string* of x is $s_x + f(l_x) + f(r_x)$, where $+$ operator defines concatenation of strings, $f(l_x)$ is the *preorder string* of the left child of x, and $f(r_x)$ is the *preorder string* of the right child of x.

The *preorder string* of the tree is the *preorder string* of its root.

Now, for the problem itself...

You have to calculate the number of different strings that can be obtained as the *preorder string* of the given tree, if you are allowed to perform the following operation any number of times before constructing the *preorder string* of the tree:

- choose any non-leaf vertex x, and swap its children (so, the left child becomes the right one, and vice versa).

Input

The first line contains one integer n ($2 \le n \le 18$).

The second line contains a sequence of $2^n - 1$ characters $s_1, s_2, \dots, s_{2^n-1}$. Each character is either A or B. The characters are **not separated** by spaces or anything else.

Output

Print one integer — the number of different strings that can be obtained as the *preorder string* of the given tree, if you can apply any number of operations described in the statement. Since it can be very large, print it modulo 998244353.

Examples

input
```
4
BAAAAAAAABBABAB
```
output
```
16
```

input
```
2
BAA
```
output
```
1
```

input
```
2
ABA
```
output
```
2
```

input

```
2
AAB
```

output

```
2
```

input

```
2
AAA
```

output

```
1
```

F. Permutation Counting

time limit per test: 4 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Calculate the number of permutations p of size n with exactly k inversions (pairs of indices (i, j) such that $i < j$ and $p_i > p_j$) and exactly x indices i such that $p_i > p_{i+1}$.

Yep, that's the whole problem. Good luck!

Input

The first line contains one integer t ($1 \le t \le 3 \cdot 10^4$) — the number of test cases.

Each test case consists of one line which contains three integers n, k and x ($1 \le n \le 998244352$; $1 \le k \le 11$; $1 \le x \le 11$).

Output

For each test case, print one integer — the answer to the problem, taken modulo 998244353.

Example

input

```
5
10 6 4
7 3 1
163316 11 7
136373 11 1
325902 11 11
```

output

```
465
12
986128624
7636394
57118194
```