Codeforces Round 968 (Div. 2)

A. Turtle and Good Strings

1 second, 256 megabytes

Turtle thinks a string s is a good string if there exists a sequence of strings $t_1, t_2, \ldots, t_k$ (k is an arbitrary integer) such that:

- $k \geq 2$.
- $s = t_1 + t_2 + \ldots + t_k$, where $+$ represents the concatenation operation. For example, `abc` $=$ `a` $+$ `bc`.
- For all $1 \leq i < j \leq k$, the first character of t_i **isn't equal to** the last character of t_j.

Turtle is given a string s consisting of lowercase Latin letters. Please tell him whether the string s is a good string!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 500$). The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \leq n \leq 100$) — the length of the string.

The second line of each test case contains a string s of length n, consisting of lowercase Latin letters.

Output

For each test case, output "YES" if the string s is a good string, and "NO" otherwise.

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

input
4 2 aa 3 aba 4 abcb 12 abcabcabcabc

output
No nO Yes YES

In the first test case, the sequence of strings `a`, `a` satisfies the condition $s = t_1 + t_2 + \ldots + t_k$, but the first character of t_1 is equal to the last character of t_2. It can be seen that there doesn't exist any sequence of strings which satisfies all of the conditions, so the answer is "NO".

In the third test case, the sequence of strings `ab`, `cb` satisfies all of the conditions.

In the fourth test case, the sequence of strings `abca`, `bcab`, `cabc` satisfies all of the conditions.

B. Turtle and Piggy Are Playing a Game 2

1 second, 256 megabytes

Turtle and Piggy are playing a game on a sequence. They are given a sequence $a_1, a_2, \ldots, a_n$, and Turtle goes first. Turtle and Piggy alternate in turns (so, Turtle does the first turn, Piggy does the second, Turtle does the third, etc.).

The game goes as follows:

- Let the current length of the sequence be m. If $m = 1$, the game ends.
- If the game does not end and it's Turtle's turn, then Turtle must choose an integer i such that $1 \leq i \leq m - 1$, set a_i to $\max(a_i, a_{i+1})$, and remove a_{i+1}.
- If the game does not end and it's Piggy's turn, then Piggy must choose an integer i such that $1 \leq i \leq m - 1$, set a_i to $\min(a_i, a_{i+1})$, and remove a_{i+1}.

Turtle wants to **maximize** the value of a_1 in the end, while Piggy wants to **minimize** the value of a_1 in the end. Find the value of a_1 in the end if both players play optimally.

You can refer to notes for further clarification.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \leq n \leq 10^5$) — the length of the sequence.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^5$) — the elements of the sequence a.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output a single integer — the value of a_1 in the end if both players play optimally.

input
5 2 1 2 3 1 1 2 3 1 2 3 5 3 1 2 2 3 10 10 2 5 2 7 9 2 5 10 7

output
2 1 2 2 7

In the first test case, initially $a = [1, 2]$. Turtle can only choose $i = 1$. Then he will set a_1 to $\max(a_1, a_2) = 2$ and remove a_2. The sequence a becomes $[2]$. Then the length of the sequence becomes 1, and the game will end. The value of a_1 is 2, so you should output 2.

In the second test case, one of the possible game processes is as follows:

- Initially $a = [1, 1, 2]$.
- Turtle can choose $i = 2$. Then he will set a_2 to $\max(a_2, a_3) = 2$ and remove a_3. The sequence a will become $[1, 2]$.
- Piggy can choose $i = 1$. Then he will set a_1 to $\min(a_1, a_2) = 1$ and remove a_2. The sequence a will become $[1]$.
- The length of the sequence becomes 1, so the game will end. The value of a_1 will be 1 in the end.

In the fourth test case, one of the possible game processes is as follows:

- Initially $a = [3, 1, 2, 2, 3]$.
- Turtle can choose $i = 4$. Then he will set a_4 to $\max(a_4, a_5) = 3$ and remove a_5. The sequence a will become $[3, 1, 2, 3]$.

- Piggy can choose $i = 3$. Then he will set a_3 to $\min(a_3, a_4) = 2$ and remove a_4. The sequence a will become $[3, 1, 2]$.
- Turtle can choose $i = 2$. Then he will set a_2 to $\max(a_2, a_3) = 2$ and remove a_3. The sequence a will become $[3, 2]$.
- Piggy can choose $i = 1$. Then he will set a_1 to $\min(a_1, a_2) = 2$ and remove a_2. The sequence a will become $[2]$.
- The length of the sequence becomes 1, so the game will end. The value of a_1 will be 2 in the end.

C. Turtle and Good Pairs

2 seconds, 256 megabytes

Turtle gives you a string s, consisting of lowercase Latin letters.

Turtle considers a pair of integers (i, j) $(1 \leq i < j \leq n)$ to be a *pleasant pair* if and only if there exists an integer k such that $i \leq k < j$ and **both** of the following two conditions hold:

- $s_k \neq s_{k+1}$;
- $s_k \neq s_i$ **or** $s_{k+1} \neq s_j$.

Besides, Turtle considers a pair of integers (i, j) $(1 \leq i < j \leq n)$ to be a *good pair* if and only if $s_i = s_j$ **or** (i, j) is a pleasant pair.

Turtle wants to reorder the string s so that the number of good pairs is **maximized.** Please help him!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \leq n \leq 2 \cdot 10^5$) — the length of the string.

The second line of each test case contains a string s of length n, consisting of lowercase Latin letters.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output the string s after reordering so that the number of good pairs is maximized. If there are multiple answers, print any of them.

input
5 3 abc 5 edddf 6 turtle 8 pppppppp 10 codeforces

output
acb ddedf urtlet pppppppp codeforces

In the first test case, $(1, 3)$ is a good pair in the reordered string. It can be seen that we can't reorder the string so that the number of good pairs is greater than 1. `bac` and `cab` can also be the answer.

In the second test case, $(1, 2)$, $(1, 4)$, $(1, 5)$, $(2, 4)$, $(2, 5)$, $(3, 5)$ are good pairs in the reordered string. `efddd` can also be the answer.

D1. Turtle and a MEX Problem (Easy Version)

2 seconds, 256 megabytes

The two versions are different problems. In this version of the problem, you can choose the same integer twice or more. You can make hacks only if both versions are solved.

One day, Turtle was playing with n sequences. Let the length of the i-th sequence be l_i. Then the i-th sequence was $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$.

Piggy gave Turtle a problem to solve when Turtle was playing. The statement of the problem was:

- There was a non-negative integer x at first. Turtle would perform an arbitrary number (possibly zero) of operations on the integer.
- In each operation, Turtle could choose an integer i such that $1 \leq i \leq n$, and set x to $\operatorname{mex}^{\dagger}(x, a_{i,1}, a_{i,2}, \ldots, a_{i,l_i})$.
- Turtle was asked to find the answer, which was the **maximum** value of x after performing an arbitrary number of operations.

Turtle solved the above problem without difficulty. He defined $f(k)$ as the answer to the above problem when the initial value of x was k.

Then Piggy gave Turtle a non-negative integer m and asked Turtle to find the value of $\sum_{i=0}^{m} f(i)$ (i.e., the value of $f(0) + f(1) + \ldots + f(m)$). Unfortunately, he couldn't solve this problem. Please help him!

$^{\dagger}\operatorname{mex}(c_1, c_2, \ldots, c_k)$ is defined as the smallest **non-negative** integer x which does not occur in the sequence c. For example, $\operatorname{mex}(2, 2, 0, 3)$ is 1, $\operatorname{mex}(1, 2)$ is 0.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n, m ($1 \leq n \leq 2 \cdot 10^5, 0 \leq m \leq 10^9$).

Each of the following n lines contains several integers. The first integer l_i ($1 \leq l_i \leq 2 \cdot 10^5$) represents the length of the i-th sequence, and the following l_i integers $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$ ($0 \leq a_{i,j} \leq 10^9$) represent the elements of the i-th sequence.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$, and the sum of $\sum l_i$ over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer — the value of $\sum_{i=0}^{m} f(i)$.

input
6 3 4 2 0 2 3 2 3 3 4 7 0 1 5 3 4 5 0 2 0 4 11 1 1 5 1 3 0 3 3 2 50 2 1 2 2 1 2 1 1 7 1 2 4 1 4 9 5 4 114514 2 2 2 5 7 3 6 0 3 3 0 1 1 5 0 9 2 1 5 5 1919810 1 2 2 324003 0 3 1416324 2 1460728 4 1312631 2 0 1415195 5 1223554 192248 2 1492515 725556

output

```
16
20
1281
6
6556785365
1842836177961
```

In the first test case, when x is initially 2, Turtle can choose $i = 3$ and set x to $\operatorname{mex}(x, a_{3,1}, a_{3,2}, a_{3,3}, a_{3,4}) = \operatorname{mex}(2, 7, 0, 1, 5) = 3$. It can be proved that Turtle can't make the value of x greater than 3, so $f(2) = 3$.

It can be seen that $f(0) = 3$, $f(1) = 3$, $f(2) = 3$, $f(3) = 3$, and $f(4) = 4$. So $f(0) + f(1) + f(2) + f(3) + f(4) = 3 + 3 + 3 + 3 + 4 = 16$.

In the second test case, when x is initially 1, Turtle can choose $i = 3$ and set x to $\operatorname{mex}(x, a_{3,1}, a_{3,2}, a_{3,3}, a_{3,4}, a_{3,5}) = \operatorname{mex}(1, 1, 3, 0, 3, 3) = 2$, and choose $i = 3$ and set x to $\operatorname{mex}(x, a_{3,1}, a_{3,2}, a_{3,3}, a_{3,4}, a_{3,5}) = \operatorname{mex}(2, 1, 3, 0, 3, 3) = 4$. It can be proved that Turtle can't make the value of x greater than 4, so $f(1) = 4$.

It can be seen that $f(0) = 4$, $f(1) = 4$, $f(2) = 4$, $f(3) = 4$, and $f(4) = 4$. So $f(0) + f(1) + f(2) + f(3) + f(4) = 4 + 4 + 4 + 4 + 4 = 20$.

In the fourth test case, it can be seen that $f(0) = 3$ and $f(1) = 3$. So $f(0) + f(1) = 3 + 3 = 6$.

D2. Turtle and a MEX Problem (Hard Version)

2 seconds, 256 megabytes

The two versions are different problems. In this version of the problem, you can't choose the same integer twice or more. You can make hacks only if both versions are solved.

One day, Turtle was playing with n sequences. Let the length of the i-th sequence be l_i. Then the i-th sequence was $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$.

Piggy gave Turtle a problem to solve when Turtle was playing. The statement of the problem was:

- There was a non-negative integer x at first. Turtle would perform an arbitrary number (possibly zero) of operations on the integer.
- In each operation, Turtle could choose an integer i such that $1 \le i \le n$ and i **wasn't chosen before**, and set x to $\operatorname{mex}^{\dagger}(x, a_{i,1}, a_{i,2}, \ldots, a_{i,l_i})$.
- Turtle was asked to find the answer, which was the **maximum** value of x after performing an arbitrary number of operations.

Turtle solved the above problem without difficulty. He defined $f(k)$ as the answer to the above problem when the initial value of x was k.

Then Piggy gave Turtle a non-negative integer m and asked Turtle to find the value of $\sum_{i=0}^{m} f(i)$ (i.e., the value of $f(0) + f(1) + \ldots + f(m)$). Unfortunately, he couldn't solve this problem. Please help him!

$^{\dagger}\operatorname{mex}(c_1, c_2, \ldots, c_k)$ is defined as the smallest **non-negative** integer x which does not occur in the sequence c. For example, $\operatorname{mex}(2, 2, 0, 3)$ is 1, $\operatorname{mex}(1, 2)$ is 0.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n, m ($1 \le n \le 2 \cdot 10^5, 0 \le m \le 10^9$).

Each of the following n lines contains several integers. The first integer l_i ($1 \le l_i \le 2 \cdot 10^5$) represents the length of the i-th sequence, and the following l_i integers $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$ ($0 \le a_{i,j} \le 10^9$) represent the elements of the i-th sequence.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$ and the sum of $\sum l_i$ over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer — the value of $\sum_{i=0}^{m} f(i)$.

input

```
6
3 4
2 0 2
3 2 3 3
4 7 0 1 5
3 4
5 0 2 0 4 11
1 1
5 1 3 0 3 3
2 50
2 1 2
2 1 2
1 1
7 1 2 4 1 4 9 5
4 114514
2 2 2
5 7 3 6 0 3
3 0 1 1
5 0 9 2 1 5
5 1919810
1 2
2 324003 0
3 1416324 2 1460728
4 1312631 2 0 1415195
5 1223554 192248 2 1492515 725556
```

output

```
16
18
1281
4
6556785365
1842836177961
```

In the first test case, when x is initially 2, Turtle can choose $i = 3$ and set x to $\operatorname{mex}(x, a_{3,1}, a_{3,2}, a_{3,3}, a_{3,4}) = \operatorname{mex}(2, 7, 0, 1, 5) = 3$. It can be proved that Turtle can't make the value of x greater than 3, so $f(2) = 3$.

It can be seen that $f(0) = 3$, $f(1) = 3$, $f(2) = 3$, $f(3) = 3$, and $f(4) = 4$. So $f(0) + f(1) + f(2) + f(3) + f(4) = 3 + 3 + 3 + 3 + 4 = 16$.

In the second test case, when x is initially 1, Turtle can choose $i = 1$ and set x to $\operatorname{mex}(x, a_{1,1}, a_{1,2}, a_{1,3}, a_{1,4}, a_{1,5}) = \operatorname{mex}(1, 0, 2, 0, 4, 11) = 3$. It can be proved that Turtle can't make the value of x greater than 3, so $f(1) = 3$.

It can be seen that $f(0) = 4$, $f(1) = 3$, $f(2) = 4$, $f(3) = 3$, and $f(4) = 4$. So $f(0) + f(1) + f(2) + f(3) + f(4) = 4 + 3 + 4 + 3 + 4 = 18$.

In the fourth test case, it can be seen that $f(0) = 3$ and $f(1) = 1$. So $f(0) + f(1) = 3 + 1 = 4$.

E1. Turtle and Inversions (Easy Version)

2 seconds, 512 megabytes

This is an easy version of this problem. The differences between the versions are the constraint on m and $r_i < l_{i+1}$ holds for each i from 1 to $m - 1$ in the easy version. You can make hacks only if both versions of the problem are solved.

Turtle gives you m intervals $[l_1, r_1], [l_2, r_2], \ldots, [l_m, r_m]$. He thinks that a permutation p is interesting if there exists an integer k_i for every interval ($l_i \le k_i < r_i$), and if he lets $a_i = \max_{j=l_i}^{k_i} p_j, b_i = \min_{j=k_i+1}^{r_i} p_j$ for every integer i from 1 to m, the following condition holds:

$$\max_{i=1}^{m} a_i < \min_{i=1}^{m} b_i$$

Turtle wants you to calculate the maximum number of inversions of all interesting permutations of length n, or tell him if there is no interesting permutation.

An inversion of a permutation p is a pair of integers (i, j) ($1 \le i < j \le n$) such that $p_i > p_j$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^3$). The description of the test cases follows.

The first line of each test case contains two integers n, m ($2 \le n \le 5 \cdot 10^3, 0 \le m \le \frac{n}{2}$) — the length of the permutation and the number of intervals.

The i-th of the following m lines contains two integers l_i, r_i ($1 \le l_i < r_i \le n$) — the i-th interval.

Additional constraint on the input in this version: $r_i < l_{i+1}$ holds for each i from 1 to $m-1$.

It is guaranteed that the sum of n over all test cases does not exceed $5 \cdot 10^3$.

Output

For each test case, if there is no interesting permutation, output a single integer -1.

Otherwise, output a single integer — the maximum number of inversions.

input
6 2 0 2 1 1 2 5 1 2 4 8 2 1 4 6 8 7 2 1 3 4 7 7 3 1 2 3 4 5 6

output
1 0 8 21 15 15

In the third test case, the interesting permutation with the maximum number of inversions is $[5, 2, 4, 3, 1]$.

In the fourth test case, the interesting permutation with the maximum number of inversions is $[4, 8, 7, 6, 3, 2, 1, 5]$. In this case, we can let $[k_1, k_2] = [1, 7]$.

In the fifth test case, the interesting permutation with the maximum number of inversions is $[4, 7, 6, 3, 2, 1, 5]$.

In the sixth test case, the interesting permutation with the maximum number of inversions is $[4, 7, 3, 6, 2, 5, 1]$.

E2. Turtle and Inversions (Hard Version)

2 seconds, 512 megabytes

This is a hard version of this problem. The differences between the versions are the constraint on m and $r_i < l_{i+1}$ holds for each i from 1 to $m-1$ in the easy version. You can make hacks only if both versions of the problem are solved.

Turtle gives you m intervals $[l_1, r_1], [l_2, r_2], \ldots, [l_m, r_m]$. He thinks that a permutation p is interesting if there exists an integer k_i for every interval ($l_i \le k_i < r_i$), and if he lets $a_i = \max\limits_{j=l_i}^{k_i} p_j, b_i = \min\limits_{j=k_i+1}^{r_i} p_j$ for every integer i from 1 to m, the following condition holds:

$$\max_{i=1}^{m} a_i < \min_{i=1}^{m} b_i$$

Turtle wants you to calculate the maximum number of inversions of all interesting permutations of length n, or tell him if there is no interesting permutation.

An inversion of a permutation p is a pair of integers (i, j) ($1 \le i < j \le n$) such that $p_i > p_j$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^3$). The description of the test cases follows.

The first line of each test case contains two integers n, m ($2 \le n \le 5 \cdot 10^3, 0 \le m \le 5 \cdot 10^3$) — the length of the permutation and the number of intervals.

The i-th of the following m lines contains two integers l_i, r_i ($1 \le l_i < r_i \le n$) — the i-th interval. Note that there may exist identical intervals (i.e., there may exist two different indices i, j such that $l_i = l_j$ and $r_i = r_j$).

It is guaranteed that the sum of n over all test cases does not exceed $5 \cdot 10^3$ and the sum of m over all test cases does not exceed $5 \cdot 10^3$.

Output

For each test case, if there is no interesting permutation, output a single integer -1.

Otherwise, output a single integer — the maximum number of inversions.

input
8 2 0 2 1 1 2 5 1 2 4 8 3 1 4 2 5 7 8 7 2 1 4 4 7 7 3 1 2 1 7 3 7 7 4 1 3 4 7 1 3 4 7 7 3 1 2 3 4 5 6

output
1 0 8 18 -1 -1 15 15

In the third test case, the interesting permutation with the maximum number of inversions is $[5, 2, 4, 3, 1]$.

In the fourth test case, the interesting permutation with the maximum number of inversions is $[4, 3, 8, 7, 6, 2, 1, 5]$. In this case, we can let $[k_1, k_2, k_3] = [2, 2, 7]$.

In the fifth and the sixth test case, it can be proven that there is no interesting permutation.

In the seventh test case, the interesting permutation with the maximum number of inversions is $[4, 7, 6, 3, 2, 1, 5]$. In this case, we can let $[k_1, k_2, k_3] = [1, 6, 1, 6]$.

In the eighth test case, the interesting permutation with the maximum number of inversions is $[4, 7, 3, 6, 2, 5, 1]$.

F. Turtle and Three Sequences

3 seconds, 256 megabytes

Piggy gives Turtle three sequences $a_1, a_2, \ldots, a_n$, $b_1, b_2, \ldots, b_n$, and $c_1, c_2, \ldots, c_n$.

Turtle will choose a subsequence of $1, 2, \ldots, n$ of length m, let it be $p_1, p_2, \ldots, p_m$. The subsequence should satisfy the following conditions:

- $a_{p_1} \leq a_{p_2} \leq \cdots \leq a_{p_m}$;
- All b_{p_i} for all indices i are **pairwise distinct**, i.e., there don't exist two different indices i, j such that $b_{p_i} = b_{p_j}$.

Help him find the maximum value of $\sum\limits_{i=1}^{m} c_{p_i}$, or tell him that it is impossible to choose a subsequence of length m that satisfies the conditions above.

Recall that a sequence a is a subsequence of a sequence b if a can be obtained from b by the deletion of several (possibly, zero or all) elements.

Input

The first line contains two integers n and m ($1 \leq n \leq 3000$, $1 \leq m \leq 5$) — the lengths of the three sequences and the required length of the subsequence.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq n$) — the elements of the sequence a.

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($1 \leq b_i \leq n$) — the elements of the sequence b.

The fourth line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \leq c_i \leq 10^4$) — the elements of the sequence c.

Output

Output a single integer — the maximum value of $\sum\limits_{i=1}^{m} c_{p_i}$. If it is impossible to choose a subsequence of length m that satisfies the conditions above, output -1.

input

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

output

```
5
```

input

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

output

```
13
```

input

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

output

```
-1
```

In the first example, we can choose $p = [1, 2]$, then $c_{p_1} + c_{p_2} = 1 + 4 = 5$. We can't choose $p = [2, 4]$ since $a_2 > a_4$, violating the first condition. We can't choose $p = [2, 3]$ either since $b_2 = b_3$, violating the second condition. We can choose $p = [1, 4]$, but $c_1 + c_4 = 4$, which isn't maximum.

In the second example, we can choose $p = [4, 6, 7]$.

In the third example, it is impossible to choose a subsequence of length 3 that satisfies both of the conditions.