VK Cup 2019-2020 - Elimination Round (Engine)

A. Recommendations

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

VK news recommendation system daily selects interesting publications of one of n disjoint categories for each user. Each publication belongs to exactly one category. For each category i batch algorithm selects a_i publications.

The latest A/B test suggests that users are reading recommended publications more actively if each category has a different number of publications within daily recommendations. The targeted algorithm can find a single interesting publication of i-th category within t_i seconds.

What is the minimum total time necessary to add publications to the result of batch algorithm execution, so all categories have a different number of publications? You can't remove publications recommended by the batch algorithm.

Input

The first line of input consists of single integer n — the number of news categories ($1 \le n \le 200\,000$).

The second line of input consists of n integers a_i — the number of publications of i-th category selected by the batch algorithm ($1 \le a_i \le 10^9$).

The third line of input consists of n integers t_i — time it takes for targeted algorithm to find one new publication of category i ($1 \le t_i \le 10^5$).

Output

Print one integer — the minimal required time for the targeted algorithm to get rid of categories with the same size.

Examples

input
5 3 7 9 7 8 5 2 5 7 5
output
6

input
5 1 2 3 4 5 1 1 1 1 1
output
0

Note

In the first example, it is possible to find three publications of the second type, which will take 6 seconds.

In the second example, all news categories contain a different number of publications.

B. Double Elimination

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

The biggest event of the year – Cota 2 world championship "The Innernational" is right around the corner. 2^n teams will compete in a double-elimination format (please, carefully read problem statement even if you know what is it) to identify the champion.

Teams are numbered from 1 to 2^n and will play games one-on-one. All teams start in the upper bracket.

All upper bracket matches will be held played between teams that haven't lost any games yet. Teams are split into games by team numbers. Game winner advances in the next round of upper bracket, losers drop into the lower bracket.

Lower bracket starts with 2^{n-1} teams that lost the first upper bracket game. Each lower bracket round consists of two games. In the first game of a round 2^k teams play a game with each other (teams are split into games by team numbers). 2^{k-1} loosing teams are eliminated from the championship, 2^{k-1} winning teams are playing 2^{k-1} teams that got eliminated in this round of upper bracket (again, teams are split into games by team numbers). As a result of each round both upper and lower bracket have 2^{k-1} teams remaining. See example notes for better understanding.

Single remaining team of upper bracket plays with single remaining team of lower bracket in grand-finals to identify championship winner.

You are a fan of teams with numbers $a_1, a_2, \ldots, a_k$. You want the championship to have as many games with your favourite teams as possible. Luckily, you can affect results of every championship game the way you want. What's maximal possible number of championship games that include teams you're fan of?

Input

First input line has two integers n, k — 2^n teams are competing in the championship. You are a fan of k teams ($2 \le n \le 17; 0 \le k \le 2^n$).

Second input line has k distinct integers $a_1, \ldots, a_k$ — numbers of teams you're a fan of ($1 \le a_i \le 2^n$).

Output

Output single integer — maximal possible number of championship games that include teams you're fan of.

Examples

input
3 1 6
output
6

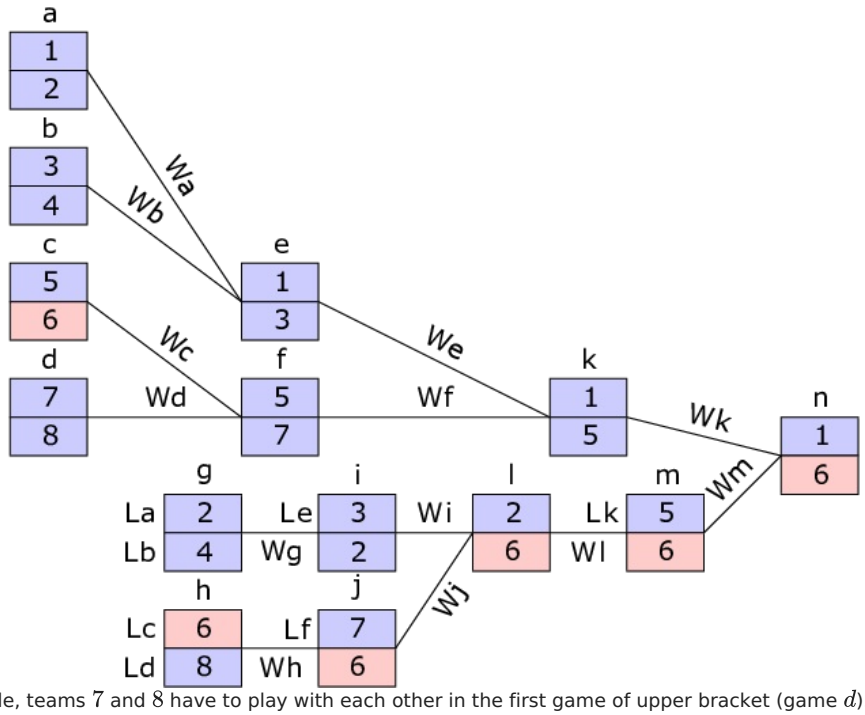
input
3 3 1 7 8
output
11

input
3 4 1 3 5 7
output
14

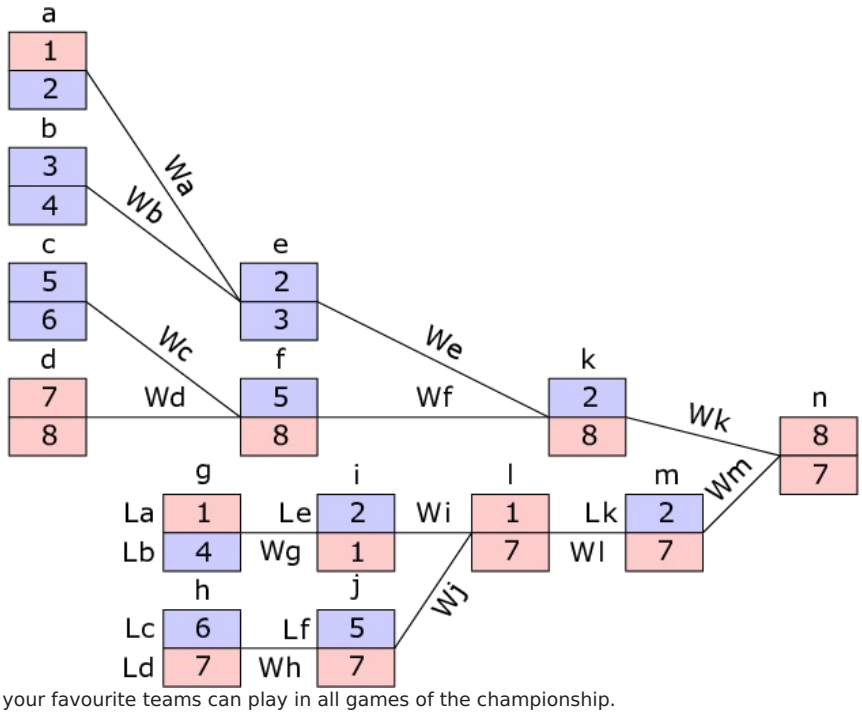
Note

On the image, each game of the championship is denoted with an English letter (a to n). Winner of game i is denoted as Wi, loser is denoted as Li. Teams you're a fan of are highlighted with red background.

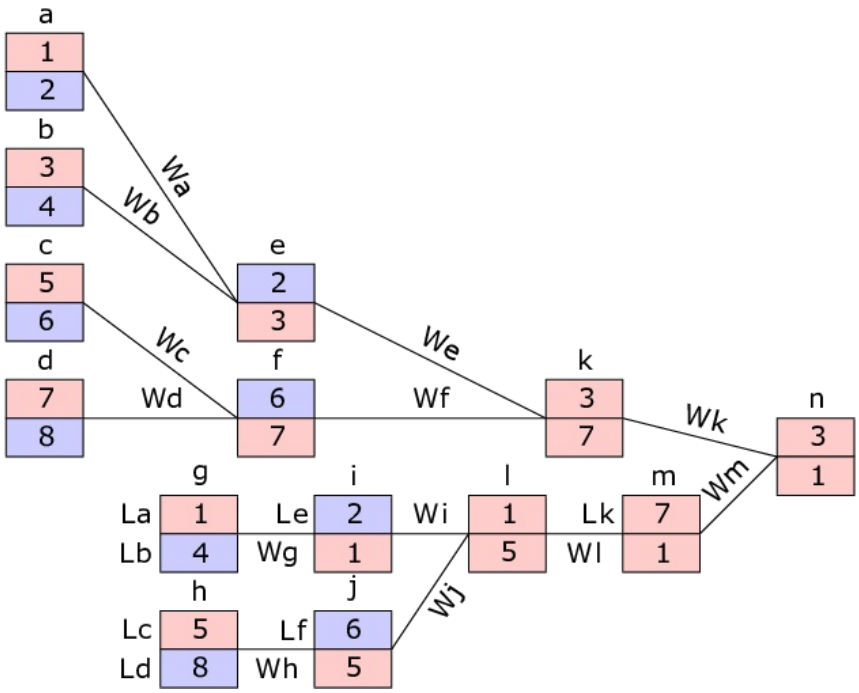
In the first example, team 6 will play in 6 games if it looses the first upper bracket game (game c) and wins all lower bracket games (games h, j, l, m).



In the second example, teams 7 and 8 have to play with each other in the first game of upper bracket (game d). Team 8 can win all remaining games in upper bracket, when teams 1 and 7 will compete in the lower bracket.



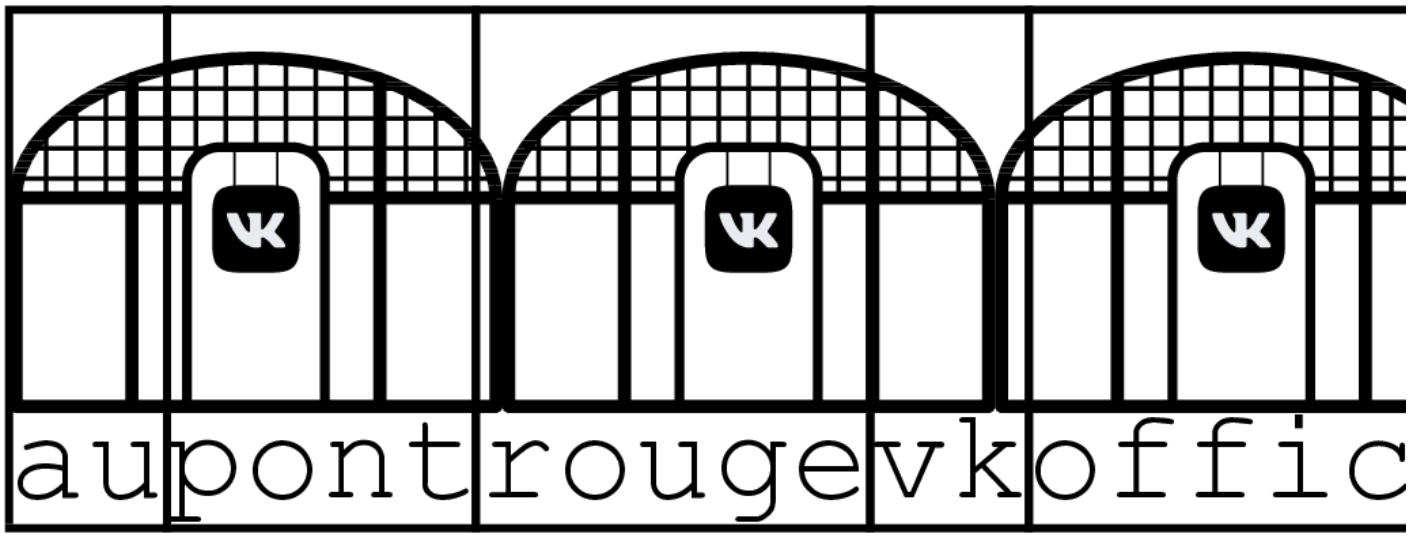
In the third example, your favourite teams can play in all games of the championship.



C. Au Pont Rouge

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

VK just opened its second HQ in St. Petersburg! Side of its office building has a huge string s written on its side. This part of the office is supposed to be split into m meeting rooms in such way that meeting room walls are strictly between letters on the building. Obviously, meeting rooms should not be of size 0, but can be as small as one letter wide. Each meeting room will be named after the substring of s written on its side.



For each possible arrangement of m meeting rooms we ordered a test meeting room label for the meeting room with lexicographically **minimal** name. When delivered, those labels got sorted **backward** lexicographically.

What is printed on kth label of the delivery?

Input
In the first line, you are given three integer numbers n, m, k — length of string s, number of planned meeting rooms to split s into and number of the interesting label ($2 \le n \le 1\,000; 1 \le m \le 1\,000; 1 \le k \le 10^{18}$).

Second input line has string s, consisting of n lowercase english letters.

For given n, m, k there are at least k ways to split s into m substrings.

Output
Output single string – name of meeting room printed on k-th label of the delivery.

Examples

input
4 2 1 abac
output
aba

input
19 5 1821 aupontrougevkoffice
output
au

Note
In the first example, delivery consists of the labels "aba", "ab", "a".

In the second example, delivery consists of 3060 labels. The first label is "`aupontrougevkof`" and the last one is "a".

D. Tourism

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Masha lives in a country with n cities numbered from 1 to n. She lives in the city number 1.

There is a direct train *route* between each pair of distinct cities i and j, where $i \neq j$. In total there are $n(n-1)$ distinct routes. Every route has a cost, cost for route from i to j may be different from the cost of route from j to i.

Masha wants to start her journey in city 1, take **exactly** k routes from one city to another and as a result return to the city 1. Masha is really careful with money, so she wants the journey to be as cheap as possible. To do so Masha doesn't mind visiting a city multiple times or even taking the same route multiple times.

Masha doesn't want her journey to have odd cycles. Formally, if you can select visited by Masha city v, take **odd** number of routes used by Masha in her journey and return to the city v, such journey is considered unsuccessful.

Help Masha to find the cheapest (with minimal total cost of all taken routes) successful journey.

Input
First line of input had two integer numbers n, k ($2 \le n \le 80; 2 \le k \le 10$): number of cities in the country and number of routes in Masha's journey. It is guaranteed that k is even.

Next n lines hold route descriptions: j-th number in i-th line represents the cost of route from i to j if $i \neq j$, and is 0 otherwise (there are no routes $i \to i$). All route costs are integers from 0 to 10^8.

Output
Output a single integer — total cost of the cheapest Masha's successful journey.

Examples

input
5 8 0 1 2 2 0 0 0 1 1 2 0 1 0 0 0 2 1 1 0 0 2 0 1 2 0
output
2

input
3 2 0 1 1 2 0 1 2 2 0
output
3

E. Strange Function

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

Let's define the function f of multiset a as the multiset of number of occurences of every number, that is present in a.

E.g., $f(\{5,5,1,2,5,2,3,3,9,5\}) = \{1,1,2,2,4\}$.

Let's define $f^k(a)$, as applying f to array a k times: $f^k(a) = f(f^{k-1}(a))$, $f^0(a) = a$.

E.g., $f^2(\{5,5,1,2,5,2,3,3,9,5\}) = \{1,2,2\}$.

You are given integers n, k and you are asked how many different values the function $f^k(a)$ can have, where a is arbitrary non-empty array with numbers of size no more than n. Print the answer modulo $998\,244\,353$.

Input

The first and only line of input consists of two integers n, k ($1 \le n, k \le 2020$).

Output

Print one number — the number of different values of function $f^k(a)$ on all possible non-empty arrays with no more than n elements modulo $998\,244\,353$.

Examples

input
3 1
output
6

input
5 6
output
1

input
10 1
output
138

input
10 2
output
33

F. Bad Cryptography

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

In modern cryptography much is tied to the algorithmic complexity of solving several problems. One of such problems is a discrete logarithm problem. It is formulated as follows:

Let's fix a finite field and two it's elements a and b. One need to fun such x that $a^x = b$ or detect there is no such x.
It is most likely that modern mankind cannot solve the problem of discrete logarithm for a sufficiently large field size. For example, for a field of residues modulo prime number, primes of 1024 or 2048 bits are considered to be safe. However, calculations with such

large numbers can place a significant load on servers that perform cryptographic operations. For this reason, instead of a simple module residue field, more complex fields are often used. For such field no fast algorithms that use a field structure are known, smaller fields can be used and operations can be properly optimized.

Developer Nikolai does not trust the generally accepted methods, so he wants to invent his own. Recently, he read about a very strange field — nimbers, and thinks it's a great fit for the purpose.

The field of nimbers is defined on a set of integers from 0 to $2^{2^k} - 1$ for some positive integer k . Bitwise exclusive or ($\oplus$) operation is used as addition. One of ways to define multiplication operation ($\odot$) is following properties:

- $0 \odot a = a \odot 0 = 0$
- $1 \odot a = a \odot 1 = a$
- $a \odot b = b \odot a$
- $a \odot (b \odot c) = (a \odot b) \odot c$
- $a \odot (b \oplus c) = (a \odot b) \oplus (a \odot c)$
- If $a = 2^{2^n}$ for some integer $n > 0$, and $b < a$, then $a \odot b = a \cdot b$.
- If $a = 2^{2^n}$ for some integer $n > 0$, then $a \odot a = \frac{3}{2} \cdot a$.

For example:

- $4 \odot 4 = 6$
- $8 \odot 8 = 4 \odot 2 \odot 4 \odot 2 = 4 \odot 4 \odot 2 \odot 2 = 6 \odot 3 = (4 \oplus 2) \odot 3 = (4 \odot 3) \oplus (2 \odot (2 \oplus 1)) = (4 \odot 3) \oplus (2 \odot 2) \oplus (2 \odot 1) = 12 \oplus 3 \oplus 2 =$
- $32 \odot 64 = (16 \odot 2) \odot (16 \odot 4) = (16 \odot 16) \odot (2 \odot 4) = 24 \odot 8 = (16 \oplus 8) \odot 8 = (16 \odot 8) \oplus (8 \odot 8) = 128 \oplus 13 = 141$
- $5 \odot 6 = (4 \oplus 1) \odot (4 \oplus 2) = (4 \odot 4) \oplus (4 \odot 2) \oplus (4 \odot 1) \oplus (1 \odot 2) = 6 \oplus 8 \oplus 4 \oplus 2 = 8$

Formally, this algorithm can be described by following pseudo-code.

```
multiply(a, b) {
  ans = 0
  for p1 in bits(a)       // numbers of bits of a equal to one
    for p2 in bits(b)   // numbers of bits of b equal to one
      ans = ans xor multiply_powers_of_2(1 << p1, 1 << p2)
  return ans;
}
multiply_powers_of_2(a, b) {
  if (a == 1 or b == 1) return a * b
  n = maximal value, such 2^{2^{n}} <= max(a, b)
  power = 2^{2^{n}};
  if (a >= power and b >= power) {
    return multiply(power * 3 / 2, multiply_powers_of_2(a / power, b / power))
  } else if (a >= power) {
    return multiply_powers_of_2(a / power, b) * power
  } else {
    return multiply_powers_of_2(a, b / power) * power
  }
}
```

It can be shown, that this operations really forms a field. Moreover, than can make sense as game theory operations, but that's not related to problem much. With the help of appropriate caching and grouping of operations, it is possible to calculate the product quickly enough, which is important to improve speed of the cryptoalgorithm. More formal definitions as well as additional properties can be clarified in the wikipedia article at link. The authors of the task hope that the properties listed in the statement should be enough for the solution.

Powering for such muliplication is defined in same way, formally $a^{\odot k} = \underbrace{a \odot a \odot \cdots \odot a}_{k \text{ times}}$.

You need to analyze the proposed scheme strength. For pairs of numbers a and b you need to find such x, that $a^{\odot x} = b$, or determine that it doesn't exist.

Input

In the first line of input there is single integer t ($1 \le t \le 100$) — number of pairs, for which you need to find the discrete logarithm.

In each of next t line there is a pair of integers a b ($1 \le a, b < 2^{64}$).

Output

For each pair you should print one integer x ($0 \le x < 2^{64}$), such that $a^{\odot x} = b$, or -1 if no such x exists. It can be shown, that if any such x exists, there is one inside given bounds. If there are several good values, you can output any of them.

Example

input
7 2 2 1 1 2 3 8 10 8 2 321321321321 2 123214213213 4356903202345442785
output
1 1 2 4 -1 6148914691236517205

```
68943624821423112
```