Educational Codeforces Round 32

A. Local Extrema

1 second, 256 megabytes

You are given an array a. Some element of this array a_i is a *local minimum* iff it is strictly less than both of its neighbours (that is, $a_i < a_{i-1}$ and $a_i < a_{i+1}$). Also the element can be called *local maximum* iff it is strictly greater than its neighbours (that is, $a_i > a_{i-1}$ and $a_i > a_{i+1}$). Since a_1 and a_n have only one neighbour each, they are neither local minima nor local maxima.

An element is called a *local extremum* iff it is either local maximum or local minimum. Your task is to calculate the number of local extrema in the given array.

Input

The first line contains one integer n ($1 \le n \le 1000$) — the number of elements in array a.

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 1000$) — the elements of array a.

Output

Print the number of local extrema in the given array.

input
3 1 2 3
output
0

input
4 1 5 2 5
output
2

B. Buggy Robot

2 seconds, 256 megabytes

Ivan has a robot which is situated on an infinite grid. Initially the robot is standing in the starting cell $(0, 0)$. The robot can process commands. There are four types of commands it can perform:

- U — move from the cell (x, y) to $(x, y + 1)$;
- D — move from (x, y) to $(x, y - 1)$;
- L — move from (x, y) to $(x - 1, y)$;
- R — move from (x, y) to $(x + 1, y)$.

Ivan entered a sequence of n commands, and the robot processed it. After this sequence the robot ended up in the starting cell $(0, 0)$, but Ivan doubts that the sequence is such that after performing it correctly the robot ends up in the same cell. He thinks that some commands were ignored by robot. To acknowledge whether the robot is severely bugged, he needs to calculate the maximum possible number of commands that were performed correctly. Help Ivan to do the calculations!

Input

The first line contains one number n — the length of sequence of commands entered by Ivan ($1 \le n \le 100$).

The second line contains the sequence itself — a string consisting of n characters. Each character can be U, D, L or R.

Output

Print the maximum possible number of commands from the sequence the robot could perform to end up in the starting cell.

input
4 LDUR
output
4

input
5 RRRUU
output
0

input
6 LLRRRR
output
4

C. K-Dominant Character

2 seconds, 256 megabytes

You are given a string s consisting of lowercase Latin letters. Character c is called k-dominant iff each substring of s with length at least k contains this character c.

You have to find minimum k such that there exists at least one k-dominant character.

Input

The first line contains string s consisting of lowercase Latin letters ($1 \le |s| \le 100000$).

Output
Print one number — the minimum value of k such that there exists at least one k-dominant character.

input
abacaba
output
2

input
zzzzz
output
1

input
abcde
output
3

D. Almost Identity Permutations

2 seconds, 256 megabytes

A permutation p of size n is an array such that every integer from 1 to n occurs exactly once in this array.

Let's call a permutation an *almost identity permutation* iff there exist at least $n - k$ indices i ($1 \le i \le n$) such that $p_i = i$.

Your task is to count the number of *almost identity* permutations for given numbers n and k.

Input
The first line contains two integers n and k ($4 \le n \le 1000$, $1 \le k \le 4$).

Output
Print the number of *almost identity* permutations for given n and k.

input
4 1
output
1

input
4 2
output
7

input
5 3
output
31

input
5 4
output
76

E. Maximum Subsequence

1 second, 256 megabytes

You are given an array a consisting of n integers, and additionally an integer m. You have to choose some sequence of indices $b_1, b_2, ..., b_k$ ($1 \le b_1 < b_2 < ... < b_k \le n$) in such a way that the value of $\sum_{i=1}^{k} a_{b_i} \bmod m$ is maximized. Chosen sequence can be empty.

Print the maximum possible value of $\sum_{i=1}^{k} a_{b_i} \bmod m$.

Input
The first line contains two integers n and m ($1 \le n \le 35$, $1 \le m \le 10^9$).

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^9$).

Output
Print the maximum possible value of $\sum_{i=1}^{k} a_{b_i} \bmod m$.

input
4 4 5 2 4 1
output
3

input
3 20 199 41 299
output
19

In the first example you can choose a sequence $b = \{1, 2\}$, so the sum $\sum_{i=1}^{k} a_{b_i}$ is equal to 7 (and that's 3 after taking it modulo 4).

In the second example you can choose a sequence $b = \{3\}$.

F. Connecting Vertices

4 seconds, 256 megabytes

There are n points marked on the plane. The points are situated in such a way that they form a regular polygon (marked points are its vertices, and they are numbered in counter-clockwise order). You can draw $n - 1$ segments, each connecting any two marked points, in such a way that all points have to be connected with each other (directly or indirectly).

But there are some restrictions. Firstly, some pairs of points cannot be connected directly and have to be connected undirectly. Secondly, the segments you draw must not intersect in any point apart from the marked points (that is, if any two segments intersect and their intersection is not a marked point, then the picture you have drawn is invalid).

How many ways are there to connect all vertices with $n - 1$ segments? Two ways are considered different iff there exist some pair of points such that a segment is drawn between them in the first way of connection, but it is not drawn between these points in the second one. Since the answer might be large, output it modulo $10^9 + 7$.

Input
The first line contains one number n ($3 \le n \le 500$) — the number of marked points.

Then n lines follow, each containing n elements. $a_{i,j}$ (j-th element of line i) is equal to 1 iff you can connect points i and j directly (otherwise $a_{i,j} = 0$). It is guaranteed that for any pair of points $a_{i,j} = a_{j,i}$, and for any point $a_{i,i} = 0$.

Output
Print the number of ways to connect points modulo $10^9 + 7$.

input
```
3
0 0 1
0 0 1
1 1 0
```
output
```
1
```

input
```
4
0 1 1 1
1 0 1 1
1 1 0 1
1 1 1 0
```
output
```
12
```

input
```
3
0 0 0
0 0 1
0 1 0
```
output
```
0
```

G. Xor-MST

2 seconds, 256 megabytes

You are given a complete undirected graph with n vertices. A number a_i is assigned to each vertex, and the weight of an edge between vertices i and j is equal to $a_i \; xor \; a_j$.

Calculate the weight of the minimum spanning tree in this graph.

Input
The first line contains n ($1 \le n \le 200000$) — the number of vertices in the graph.

The second line contains n integers $a_1, a_2, ..., a_n$ ($0 \le a_i < 2^{30}$) — the numbers assigned to the vertices.

Output
Print one number — the weight of the minimum spanning tree in the graph.

input
```
5
1 2 3 4 5
```
output
```
8
```

input
```
4
1 2 3 4
```
output
```
8
```