# A - 22222

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : $100$ points

## Problem Statement

You are given a string $S$ consisting of digits.

Remove all characters from $S$ except for `2`, and then concatenate the remaining characters in their original order to form a new string.

## Constraints

- $S$ is a string consisting of digits with length between $1$ and $100$, inclusive.
- $S$ contains at least one `2`.

## Input

The input is given from Standard Input in the following format:

```
S
```

## Output

Print the answer.

## Sample Input 1

```
20250222
```

## Sample Output 1

```
22222
```

By removing `0`, `5`, and `0` from `20250222` and then concatenating the remaining characters in their original order, the string `22222` is obtained.

## Sample Input 2

```
2
```

## Sample Output 2

```
2
```

## Sample Input 3

```
22222000111222222
```

## Sample Output 3

```
22222222222
```

# B - cat

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : $200$ points

## Problem Statement

You are given $N$ strings $S_1, S_2, \ldots, S_N$, each consisting of lowercase English letters. The lengths of these strings are all distinct.

Sort these strings in ascending order of length, and then concatenate them in that order to form a single string.

## Constraints

- $2 \leq N \leq 50$
- $N$ is an integer.
- Each $S_i$ is a string consisting of lowercase English letters with length between $1$ and $50$, inclusive.
- If $i \neq j$, the length of $S_i$ is different from the length of $S_j$.

## Input

The input is given from Standard Input in the following format:

$$
\begin{array}{l}
N \\
S_1 \\
S_2 \\
\vdots \\
S_N
\end{array}
$$

## Output

Print the answer.

## Sample Input 1

```
3
tc
oder
a
```

## Sample Output 1

```
atcoder
```

When we sort (`tc`, `oder`, `a`) in ascending order of length, we get (`a`, `tc`, `oder`). Concatenating them in this order yields the string `atcoder`.

---

## Sample Input 2

```
4
cat
enate
on
c
```

## Sample Output 2

```
concatenate
```

# C - Debug

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : $300$ points

## Problem Statement

You are given a string $S$ consisting of uppercase English letters.
Apply the following procedure to $S$, and then output the resulting string:

> As long as the string contains `WA` as a (contiguous) substring, repeat the following operation:
>
> - Among all occurrences of `WA` in the string, replace the leftmost one with `AC`.

It can be proved under the constraints of this problem that this operation is repeated at most a finite number of times.

## Constraints

- $S$ is a string of uppercase English letters with length between 1 and $3 \times 10^5$, inclusive.

## Input

The input is given from Standard Input in the following format:

```
S
```

## Output

Print the resulting string after performing the procedure described in the problem statement on $S$.

## Sample Input 1

```
WACWA
```

## Sample Output 1

```
ACCAC
```

Initially, the string is $S =$ `WACWA`.

This string contains `WA` as a substring in two places: from the 1st to the 2nd character, and from the 4th to the 5th character.

In the first operation, we replace the leftmost occurrence (the substring from the 1st to the 2nd character) with `AC`, resulting in `ACCWA`.

After the first operation, the string contains `WA` as a substring in exactly one place: from the 4th to the 5th character.

In the second operation, we replace it with `AC`, resulting in `ACCAC`.

Since `ACCAC` does not contain `WA` as a substring, the procedure ends. Therefore, we output `ACCAC`.

## Sample Input 2

```
WWA
```

## Sample Output 2

```
ACC
```

Initially, the string is $S =$ `WWA`.

This string contains `WA` as a substring in exactly one place: from the 2nd to the 3rd character.

In the first operation, we replace it with `AC`, resulting in `WAC`.

Then, after the first operation, the string contains `WA` in exactly one place: from the 1st to the 2nd character.

In the second operation, we replace it with `AC`, resulting in `ACC`.

Since `ACC` does not contain `WA` as a substring, the procedure ends. Therefore, we output `ACC`.

## Sample Input 3

```
WWWWW
```

## Sample Output 3

```
WWWWW
```

Since $S$ does not contain `WA` as a substring from the start, no operations are performed and the procedure ends immediately. Therefore, we output `WWWWW`.

# D - Colorful Bracket Sequence

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : $400$ points

## Problem Statement

You are given a string $S$ consisting of six types of characters: `(`, `)`, `[`, `]`, `<`, `>`.

A string $T$ is called a colorful bracket sequence if it satisfies the following condition:

> It is possible to turn $T$ into an empty string by repeating the following operation any number of times (possibly zero):
>
> - If there exists a contiguous substring of $T$ that is one of `()`, `[]`, or `<>`, choose one such substring and delete it.
> - If the deleted substring was at the beginning or end of $T$, the remainder becomes the new $T$.
> - Otherwise, concatenate the part before the deleted substring and the part after the deleted substring, and that becomes the new $T$.

Determine whether $S$ is a colorful bracket sequence.

## Constraints

- $S$ is a string of length between $1$ and $2 \times 10^5$, inclusive.
- $S$ consists of `(`, `)`, `[`, `]`, `<`, `>`.

## Input

The input is given from Standard Input in the following format:

```
S
```

## Output

If $S$ is a colorful bracket sequence, print `Yes`; otherwise, print `No`.

## Sample Input 1

```
([])<>()
```

## Sample Output 1

```
Yes
```

For $S =$`([])<>()`, it is possible to turn it into an empty string by repeating the operation as follows:

- Delete the substring `[]` from the 2nd to the 3rd character in `([])<>()`, then concatenate the parts before and after it. The string becomes `()<>()`.
- Delete the substring `()` from the 1st to the 2nd character in `()<>()`. The string becomes `<>()`.
- Delete the substring `<>` from the 1st to the 2nd character in `<>()`. The string becomes `()`.
- Delete the substring `()` from the 1st to the 2nd character in `()`. The string becomes empty.

Thus, $S =$`([])<>()` is a colorful bracket sequence, so print `Yes`.

---

## Sample Input 2

```
([<)]>
```

## Sample Output 2

```
No
```

Since $S =$`([<)]>` does not contain `()`, `[]`, or `<>` as a contiguous substring, we cannot perform the 1st operation, and in particular $S$ is not a colorful bracket sequence. Therefore, print `No`.

---

## Sample Input 3

```
())
```

## Sample Output 3

```
No
```

It is impossible to turn $S$ into an empty string by repeating the operations.
Therefore, $S$ is not a colorful bracket sequence, so print `No`.

# E - Palindromic Shortest Path

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : $450$ points

## Problem Statement

We have a directed graph with $N$ vertices, numbered $1, 2, \ldots, N$.

Information about the edges is given by $N^2$ characters $C_{1,1}, C_{1,2}, \ldots, C_{1,N}, C_{2,1}, \ldots, C_{N,N}$. Here, each $C_{i,j}$ is either a lowercase English letter or -.

If $C_{i,j}$ is a lowercase English letter, then there is exactly one directed edge from vertex $i$ to vertex $j$ labeled $C_{i,j}$. If $C_{i,j}$ is -, there is no edge from vertex $i$ to vertex $j$.

For each integer pair $(i, j)$ with $1 \leq i, j \leq N$, answer the following question:

- Among all (not necessarily simple) paths from vertex $i$ to vertex $j$ whose concatenation of labels on the edges forms a palindrome, what is the length of the shortest such path? If there is no such path, the answer is $-1$.

## Constraints

- $1 \leq N \leq 100$
- $N$ is an integer.
- Each $C_{i,j}$ is either a lowercase English letter or -.

## Input

The input is given from Standard Input in the following format:

$$
\begin{array}{l}
N \\
C_{1,1}C_{1,2}\ldots C_{1,N} \\
C_{2,1}C_{2,2}\ldots C_{2,N} \\
\vdots \\
C_{N,1}C_{N,2}\ldots C_{N,N}
\end{array}
$$

## Output

Let $A_{i,j}$ be the answer to the question for the pair $(i, j)$. Print them in the following format:

$$\begin{array}{llll} A_{1,1} & A_{1,2} & \ldots & A_{1,N} \\ A_{2,1} & A_{2,2} & \ldots & A_{2,N} \\ \vdots & & & \\ A_{N,1} & A_{N,2} & \ldots & A_{N,N} \end{array}$$

## Sample Input 1

```
4
ab--
--b-
---a
c---
```

## Sample Output 1

```
0 1 2 4
-1 0 1 -1
3 -1 0 1
1 -1 -1 0
```

For example, consider the case $(i, j) = (1, 4)$.
By taking the path $1 \to 1 \to 2 \to 3 \to 4$, and concatenating the labels on its edges in order, we get the string `abba`, which is a palindrome.
There is no path of length at most $3$ from vertex $1$ to vertex $4$ whose concatenation of labels is a palindrome.
Thus, the answer for $(1, 4)$ is $4$.

Note that the empty string is also a palindrome.

## Sample Input 2

```
5
us---
-st--
--s--
u--s-
---ts
```

## Sample Output 2

```
0 1 3 -1 -1
-1 0 1 -1 -1
-1 -1 0 -1 -1
1 3 -1 0 -1
-1 -1 5 1 0
```

# F - Alkane

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : $500$ points

## Problem Statement

You are given an undirected tree $T$ with $N$ vertices, numbered $1, 2, \ldots, N$. The $i$-th edge is an undirected edge connecting vertices $A_i$ and $B_i$.

A graph is defined to be an **alkane** if and only if it satisfies the following conditions:

- The graph is an undirected tree.
- Every vertex has degree $1$ or $4$, and there is at least one vertex of degree $4$.

Determine whether there exists a subgraph of $T$ that is an alkane, and if so, find the maximum number of vertices in such a subgraph.

## Constraints

- $1 \leq N \leq 2 \times 10^5$
- $1 \leq A_i, B_i \leq N$
- The given graph is an undirected tree.
- All input values are integers.

## Input

The input is given from Standard Input in the following format:

$$\begin{array}{l} N \\ A_1 \quad B_1 \\ A_2 \quad B_2 \\ \vdots \\ A_{N-1} \quad B_{N-1} \end{array}$$

## Output

If there exists a subgraph of $T$ that is an alkane, print the maximum number of vertices in such a subgraph. Otherwise, print $-1$.

## Sample Input 1

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

## Sample Output 1

```
8
```

Let $(u, v)$ denote an undirected edge between vertices $u$ and $v$.

A subgraph consisting of vertices $1, 2, 3, 4, 6, 7, 8, 9$ and edges $(1, 2), (2, 3), (3, 4), (2, 6), (2, 7), (3, 8), (3, 9)$ is an alkane.

## Sample Input 2

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

## Sample Output 2

```
-1
```

## Sample Input 3

```
15
8 5
2 9
1 12
6 11
9 3
15 1
7 12
7 13
10 5
6 9
5 1
1 9
4 5
6 14
```

## Sample Output 3

```
11
```

# G - Dense Buildings

Time Limit: 5 sec / Memory Limit: 1024 MiB

Score : $600$ points

## Problem Statement

There is a city divided into $H \times W$ blocks in the north-south-east-west directions, and there is exactly one building in each block.
Specifically, in the block at the $i$-th row from the north $(1 \leq i \leq H)$ and the $j$-th column from the west $(1 \leq j \leq W)$ (hereafter referred to as block $(i, j)$), there is a building of $F_{i,j}$ floors.

Takahashi has two ways of moving. If he is on the $X$-th floor $(1 \leq X \leq F_{i,j})$ of the building in block $(i, j)$, he can:

- Move up or down one floor within the same building using **stairs**. If $X = 1$, he cannot move down; if $X = F_{i,j}$, he cannot move up.
- Choose a building with at least $X$ floors in a cardinally adjacent block, and move to the $X$-th floor of that building using a **(sky) walkway**.

Here, two blocks $(i, j)$ and $(i', j')$ are cardinally adjacent if and only if $|i - i'| + |j - j'| = 1$.

You are given $Q$ queries to be answered. The $i$-th query $(1 \leq i \leq Q)$ is the following.

> Find the minimum possible number of times that Takahashi uses **stairs** to move from the $Y_i$-th floor of the building in block $(A_i, B_i)$ to the $Z_i$-th floor of the building in block $(C_i, D_i)$.
> The count of times using stairs is incremented each time he moves up or down one floor, possibly multiple times within the same building. (For example, moving from the 1st floor to the 6th floor of a building counts as $5$ uses of stairs.)
> Note that he does not have to minimize the number of times he uses walkways.

## Constraints

- $1 \leq H \leq 500$
- $1 \leq W \leq 500$
- $1 \leq F_{i,j} \leq 10^6$
- $1 \leq Q \leq 2 \times 10^5$
- $1 \leq A_i, C_i \leq H$
- $1 \leq B_i, D_i \leq W$
- $1 \leq Y_i \leq F_{A_i,B_i}$
- $1 \leq Z_i \leq F_{C_i,D_i}$
- $(A_i, B_i, Y_i) \neq (C_i, D_i, Z_i)$
- All input values are integers.

## Input

The input is given from Standard Input in the following format:

$$
\begin{array}{l}
H \quad W \\
F_{1,1} \quad F_{1,2} \quad \ldots \quad F_{1,W} \\
F_{2,1} \quad F_{2,2} \quad \ldots \quad F_{2,W} \\
\vdots \\
F_{H,1} \quad F_{H,2} \quad \ldots \quad F_{H,W} \\
Q \\
A_1 \quad B_1 \quad Y_1 \quad C_1 \quad D_1 \quad Z_1 \\
A_2 \quad B_2 \quad Y_2 \quad C_2 \quad D_2 \quad Z_2 \\
\vdots \\
A_Q \quad B_Q \quad Y_Q \quad C_Q \quad D_Q \quad Z_Q
\end{array}
$$

## Output

Print $Q$ lines. The $i$-th line should contain the answer to the $i$-th query as an integer.

## Sample Input 1

```
3 3
12 10 6
1 1 3
8 6 7
2
1 1 10 3 1 6
1 1 6 1 2 4
```

## Sample Output 1

```
10
2
```

For the first query, for example, it is possible to move from the 10th floor of the building in block $(1, 1)$ to the 6th floor of the building in block $(3, 1)$ by using stairs a total of $10$ times, in the following manner:

- Move from the 10th floor of the building in block $(1, 1)$ to the 10th floor of the building in block $(1, 2)$ via a walkway.
- Use stairs $4$ times to go from the 10th floor down to the 6th floor of the building in block $(1, 2)$.
- Move from the 6th floor of the building in block $(1, 2)$ to the 6th floor of the building in block $(1, 3)$ via a walkway.
- Use stairs $3$ times to go from the 6th floor down to the 3rd floor of the building in block $(1, 3)$.
- Move from the 3rd floor of the building in block $(1, 3)$ to the 3rd floor of the building in block $(2, 3)$ via a walkway.
- Move from the 3rd floor of the building in block $(2, 3)$ to the 3rd floor of the building in block $(3, 3)$ via a walkway.
- Use stairs $3$ times to go from the 3rd floor up to the 6th floor of the building in block $(3, 3)$.
- Move from the 6th floor of the building in block $(3, 3)$ to the 6th floor of the building in block $(3, 2)$ via a walkway.
- Move from the 6th floor of the building in block $(3, 2)$ to the 6th floor of the building in block $(3, 1)$ via a walkway.

It is impossible to make this journey using at most $9$ uses of stairs, so we output $10$.

For the second query, if you first use a walkway to go to the building in block $(1, 2)$, and then use the stairs twice to go from the 6th floor down to the 4th floor, it is possible to move from the 6th floor of the building in block $(1, 1)$ to the 4th floor of the building in block $(1, 2)$ by using the stairs twice.