Module-10.5(Week-3-PracticeDay-1)

M. Replace MinMax

Given a number N and an array A of N numbers. Print the array after doing the following operations:

- Find **minimum** number in these numbers.
- Find **maximum** number in these numbers.
- Swap **minimum** number with **maximum** number.

Input

First line contains a number N $(2 \leq N \leq 1000)$ number of elements.

Second line contains N numbers $(-10^5 \leq A_i \leq 10^5)$

It's **guaranteed** that all numbers are distinct.

Output

Print the array after the **replacement** operation.

Example

input

```
5
4 1 3 10 8
```

output

```
4 10 3 1 8
```

F. Way Too Long Words

Given a string S. Print the **origin string** if it's **not too long** otherwise, print the **special abbreviation**.

Note: The string is called **too long**, if its length is strictly more than **10** characters. If the string is **too long** then you have to print the string in the following manner:

- Print the **first** character in the string.
- Print number of characters between the first and the last characters.
- Print the **last** character in the string.

For example: "localization" will be "l10n", and "internationalization" will be "i18n".

Input

The first line contains a number T $(1 \leq T \leq 100)$ number of test cases.

Each of the T following lines contains a string S $(1 \leq |S| \leq 100)$ where |S| is the length of the string.

It's guaranteed that S contains only lowercase Latin letters.

Output

For each test case, print the result string.

Example

input

```
4
word
localization
internationalization
pneumonoultramicroscopicsilicovolcanoconiosis
```

output

```
word
l10n
i18n
p43s
```

G. Conversion

Given a string S. Print the **origin** string after replacing the following:

- Replace every **comma** character ',' with a space character.
- Replace every **capital character** in S with its respective **small character** and **Vice Versa**.

Input

Only one line contains a string S $(1 \leq |S| \leq 10^5)$ where |S| is the length of the string and it consists of **lower** and **upper** English letters and **comma** character ','.

Output

Print the string after the **conversion**.

Example

input

```
happy,NewYear,enjoy
```

output

```
HAPPY nEWyEAR ENJOY
```

I. Palindrome

Given a string S. Determine whether S is **Palindrome** or **not**

Note: A string is said to be a **palindrome** if **the reverse** of the string is **same** as the string. For example, "abba" is **palindrome**, but "abbc" is not **palindrome**.

Input

Only one line contains a string S $(1 \leq |S| \leq 1000)$ where |S| is the length of the string and it consists of **lowercase** letters only.

Output

Print "**YES**" if the string is **palindrome**, otherwise print "**NO**".

Examples

input

```
abba
```

output

```
YES
```

input

```
icpcassiut
```

output

```
NO
```

input

```
mam
```

output

```
YES
```

D. Strings

Given two strings A and B. Print three lines contain the following:

- The size of the string A and size of the string B separated by a space
- The string produced by **concatenating** A and B $(A + B)$.
- The two strings separated by a single space respectively, after **swapping** their first character.

For more clarification see the example below.

Input

The first line contains a string A $(1 \leq |A| \leq 10)$ where |A| is the length of A.

The second line contains a string B $(1 \leq |B| \leq 10)$ where |B| is the length of B.

Output

Print the answer required above.

Example

input

```
abcd
ef
```

output

```
4 2
abcdef
ebcd af
```

Note

Declaration:

```
string a = "abc";
```

Size:

```
int len = a.size();
```

Concatenate two strings:

```
string a = "abc";
string b = "def";
string c = a + b; // c = "abcdef".
```

Accessing i *element*:

```
string s = "abc";
char   c0 = s[0];   // c0 = 'a'
char   c1 = s[1];   // c1 = 'b'
char   c2 = s[2];   // c2 = 'c'
s[0] = 'z';         // s = "zbc"
```