Reverse Polish notation

Reverse Polish notation (RPN), also known as Polish postfix notation or simply postfix notation, is a mathematical notation in which operators follow their operands, in contrast to Polish notation (PN), in which operators precede their operands. It does not need any parentheses as long as each operator has a fixed number of operands.

You need to write a program that transforms an infix expression to a equivalent RPN according to the following specifications.

1. The infix expression is in the input file in the format of one character per line, with a maximum of 50 lines. For example, `(1+1)*(4*5+1)-4` would be in the form:

```
(
1
+
1
)
*
(
4
*
5
+
1
)
-
4
```

Jan Łukasiewicz was a Polish logician and philosopher best known for Polish notation.

2. There will be only one infix expression in the input file, and it will be an expression with a valid syntax.
3. All operators are binary operators `+, -, *, /` .
4. The operands will be one digit numerals: 0, 1, 2, … , 9.
5. The operators `*` and `/` have the highest precedence. The operators `+` and `-` have the lowest precedence. Operators at the same precedence level associate from left to right. Parentheses act as grouping symbols that override the operator precedence.

Input

There will be multiple lines in the input file as specified above.

Output

The output file will have a postfix expression all on one **line** with no whitespace between symbols and a single newline character at the end.

Example

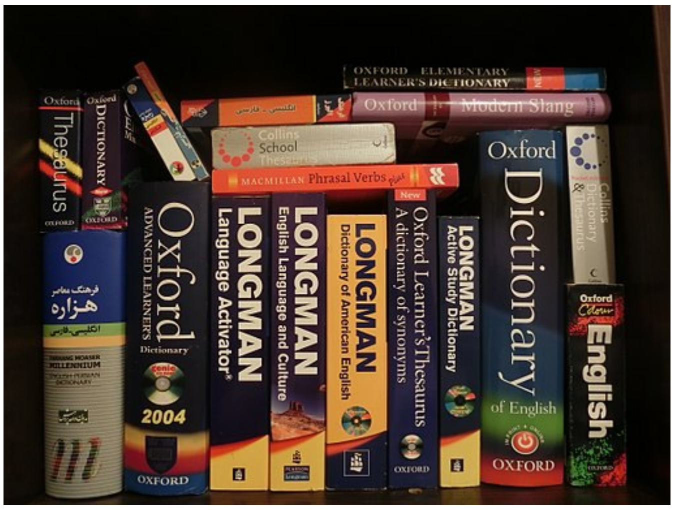
```
Input:
(
1
+
1
)
*
```

```
(
4
*
5
+
1
)
-
4

Output:
11+45*1+*4-
```

Albert’s Dream

Albert has a dream: he wants to create his own dictionary (Merriam Webster is his idol and he hopes to have his own name on a thick book one day). Albert realizes that he does not know enough words to create a complete dictionary. But he has many many books and if he could only get all of the unique words out of those, he would be happy.



You are Albert’s best friend and majoring in computer science, so you offer to help with processing all the texts and selecting only the unique words. You are going to write a program that processes text and produces a list of unique words in alphabetical order. Before you do that, Albert gives you his definition of a word. A *word* is a sequence of one or more consecutive alphabetic characters in upper or lower case. Albert also wants your program to be case insensitive, that is words like "APPLE", "apple", "appLE" should be treated as the same.

Input

The input is a text with up to 5,000 lines. Each line has at most 200 characters. The input if terminated by EOF.

Output

A list of different/unique words that appear in the input text, one per line. You are guaranteed that the number of unique words in the text is no more than 5,000.

Example 1

```
Input:
Albert has a dream: he wants to create his own dictionary (Merriam Webster is
his idol and he hopes to have his own name on a thick book one day).

Output:
a
albert
and
book
create
day
dictionary
dream
has
have
he
his
hopes
idol
is
merriam
name
on
one
own
thick
to
wants
webster
```

Example 2

```
Input:
121()* &* abc *awre@#

Output:
abc
awre
```

Ferry

Ferries used to carry cars across the river. In your village, there is still a ferry that can take up to `N` cars and needs `T` minutes to cross the river. A car may arrive at either river bank and wait to be carried to the opposite bank. The ferry operates continuously between the banks as long it is carrying at least one car or there is at least one car waiting on either side. Whenever the ferry arrives at one bank, it unloads cars carried and loads up to `N` cars waiting at that bank. When there are more than `N` cars waiting, they are loaded on the first-come-first-serve basis. If there is no car waiting on either bank, the ferry stops and waits until one car arrives. The ferry is initially on the left bank. You are asked to determine at what time each car arrives at the other bank.

Input The first line of input contains three integers `N`, `T` and `M` (`1 <= N, T, M <= 10,000`). Each of the following `M` lines gives the arrival time of a car and the bank at which the car arrives (`left` or `right`). The cars are ordered by their arrival times (so the arrival times are non-decreasing) and the time spent on loading and unloading can be ignored.

Output For each car, you should print one line containing one number, the time at which the car is unloaded at the opposite bank.

Example 1

```
Input:
2 10 10
0 left
10 left
20 left
30 left
40 left
50 left
60 left
70 left
80 left
90 left

Output:
10
30
30
50
50
70
70
90
90
110
```

Example 2

```
Input:
2 10 3
10 right
25 left
40 left

Output:
30
40
60
```

Bit-wise OR

Given 3 non-negative integers N, L, R. Find the **minimum** integer x such that

1. L <= x <= R
2. x **OR** N is maximized (**OR** stands for bit-wise OR operation here), i.e. for all integers y in the close interval [L,R], we have: (y **OR** N) <= (x **OR** N)

Input

There will be 3 integers N, L, R on a single line, where L <= R. You can assume that 0 <= N,L,R <= 4294967295.

Output

Print a line containing the value of x such that x **OR** N is maximized.

Example 1

```
Input:
10 10 10

Output:
10
```

Example 2

```
Input:
100 50 60

Output:
59
```

Prime Gap

A **prime gap** is the difference between two successive prime numbers. The *n*-th prime gap, denoted g(n) is the difference between the (n+1)-th and the n-th prime numbers, i.e.

g(n) = p(n+1) - p(n)

The first 7 prime numbers are 2, 3, 5, 7, 11, 13, 17, and the first 6 prime gaps are 1, 2, 2, 4, 2, 4.

Shinya Yukimura is interested in prime gaps and he need some experimental data to verify his hypothesis. More specifically, given a closed interval [a,b], Shinya wants to find the two adjacent primes p1 and p2 (a <= p1 <= p2 <= b) such that the prime gap between p1 and p2 is minimized (i.e. p2-p1 is the minimum). If there are multiple prime pairs that have the same prime gap, report the first pair. Shinya also wants to find the two adjacent primes p3 and p4 (a <= p3 < p4 <= b) that maximize the gap between p3 and p4 (choose the first pair if there are mote than one such pairs).

Please write a program to help Shinya.

Input

Two integer values a,b, with a < b. The difference between a and b will not exceed 1,000,000. 1 <= a <= b <= 2,147,483,647.

Output

If there are no adjacent primes in the interval [a,b], output -1 followed by a newline.

Otherwise the output should be 4 integers: `p1 p2 p3 p4` as mentioned above separated by a space and followed by a newline (no space after p4).

Example 1

```
Input:
1 20

Output:
2 3 7 11
```

Example 2

```
Input:
13 16

Output:
-1
```

In the first example test case, the prime gap between 13 and 17 also has the largest value 4, but the pair (7,11) appears before (13,17), so we output 7 11 instead of 13 17.