Problem A. Daybreak Frontline

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Given a string s, you can form pair (i, j) if $s_i = s_j$. We call set of pairs good if every index belongs to only one pair and if there is no two pairs $(i, j), (k, l)$ such that $i \leq k \leq j \leq l$. Determine if there is a good set of pairs.

Input

First line contains string $s(1 \leq |s| \leq 10^5)$ consisting of lowercase english letters.

Output

Output 'YES'(without quotes) if there exists a good set of pairs, and output 'NO' otherwise.

Examples

standard input	standard output
abbacc	YES
abab	NO

Note

In first example we can form pairs (1, 4), (2, 3), (5, 6), it's easy to see that it satisfies all conditions. In second example there is no good set of pairs. Hint: Solution is similar to regular brackets problem.

Problem B. Прокачка линкед листа

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

На одной из предыдущих лаб вам доводилось реализовывать связный список. Сегодня вашей задачей будет расширение имеющегося функционала.

Ваша задача написать функции *int count*(*int x*), *int getNth*(*int n*) и *void insertLast*(*int x*).

- *int count*(*int x*) — возвращает количество элементов в листе равных x.
- *int getNth*(*int n*) — возвращает значение элемента являющегося n-м от начала листа. Нумерация начинается с нуля. Если данного элемента нет, функция должна вернуть -1.
- *void insertLast*(*int x*) — добавляет число x в конец листа.

Кроме этих функций, вам больше ничего менять не нужно.

Удачи ! ;)

Example

standard input	standard output
8 insertFirst 10 insertLast 20 insertFirst 10 cnt 10 getNth 0 getNth 1 getNth 2 insertLast 30	2 10 10 20 10 10 20 30

Problem C. Doki-Doki

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given q queries of three types:

1 h : Add new tree with height $1 \leq h \leq 10^9$ to the end.

2 d : Water level rises by $0 \leq d \leq 10^9$. It is guranteed that water level never will be greater than 10^9. At the beginning, water level is 0.

3 : Output the height of first tree not covered by water. Tree with height h is covered by water if $h \leq c$, where c is level of water. If all trees are covered by water output -1.

Input

First line contains integer $1 \leq q \leq 5 * 10^5$, number of queries. In next q lines there are 1 or 2 integers - the queries.

Output

Output answer to the queries.

Example

standard input	standard output
9 1 5 1 10 1 5 2 5 3 2 5 3 1 15 3	10 -1 15

Note

First example:

1. Add tree with height 5 to the end : [5].

2. Add tree with height 10 to the end : [5, 10].

3. Add tree with height 5 to the end : [5, 10, 5].

4. Water level rises by 5, current water level 5.

5. First tree larger than 5 is 10.

6. Water level rises by 5, current water level is 10.

7. There is no tree with height larger than 10, answer is -1.

8. Add tree with height 15 to the end : [5, 10, 15]

9. First tree larger than 10 is 15.

Problem D. Super Prime

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

A Prime number will be called a natural number greater than one and divisible only by one and by itself. Let's write out all Prime numbers in ascending order and denote i-th number p_i (number 2 will be 1-th). For example, $p_1 = 2, p_2 = 3, p_3 = 5, p_{52} = 239$.

Let's say that P_i is a super-simple number if $i = p_k$ for some k. in Other words, a super — simple number is a Prime number whose number in the list of Prime numbers ordered in ascending order is a Prime number.

Given a positive integer k. let's Order all super-Prime numbers in ascending order. Find the k-th super Prime number in this order.

Input

Input contains a positive integer k ($1 \leq k \leq 500$).

Output

In the output print the k-th super-prime number.

Examples

standard input	standard output
1	3
2	5
3	11
100	3911

Problem E. ОПН

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Эдсгер Дейкстра изобрёл алгоритм для преобразования выражений из инфиксной нотации в ОПЗ. Алгоритм получил название «сортировочная станция», за сходство его операций с происходящим на железнодорожных сортировочных станциях. Инфиксная нотация -- это форма математических записей, которую использует большинство людей (например, 3 + 4 или 3 + 4 *(2 - 1)). Как и алгоритм вычисления ОПЗ, алгоритм сортировочной станции основан на стеке. В преобразовании участвуют две текстовых переменных: входная и выходная строки. В процессе преобразования используется стек, хранящий ещё не добавленные к выходной строке операции. Преобразующая программа читает входную строку последовательно символ за символом (символ -- это не обязательно буква), выполняет на каждом шаге некоторые действия в зависимости от того, какой символ был прочитан.

"Wikipedia"

На одной из предыдущих практик вам уже доводилось вычислять значение выражения в ОПН. Сегодня вам нужно будет найти значение выражения в ~~человеческой~~ инфиксной нотации.

Для простоты, будут использоваться только операции сложения, вычитания, умножения, числа 0 до 9 и скобочки.

Все операции являются бинарными, то есть тестов вроде $1 + (-1)$ не будет.

Гарантируется что все строки в тестах являются корректной записью какого либо математического выражения.

Также, гарантируется что ответ и числа в ходе промежуточных вычислений не превысят 10^9.

Упрощённая версия алгоритма решающего задачу звучит следующим образом. Мы будем поддерживать 2 стека. Первый будет содержать операции(op), второй числа(num) ожидающие обработки.

Также у каждой операции будет приоритет.

- (,) - низкий
- $+, -$ - средний
- $*$ - высокий

Выражение обрабатывается с лева на право. Приобработке очередного символа c выполняются следующие действия :

- Если c является числом. Добавляем его в стек num.
- Если c является открывающейся скобкой. Добавляем её в op.
- Если c является бинарной операцией. Пока op не пуст и операция на вершине стека имеет равный или больший приоритет чем у c, применяем операцию из стека и удаляем её. После, добавляем c в op.
- Если c является закрывающей скобкой. Пока не дойдём до открывающей скобки. Выполняем операцию на вершине op и удаляем её. Когда дошли до открывающей скобки, удаляем её.

В конце выполняем все оставшиеся в стеке op.

Examples

standard input	standard output
2+2*2	6
(2+2)*2	8
(1+(2+3)*4-5+(5))	21