D. Jamie and To-do List

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Why I have to finish so many assignments???

Jamie is getting very busy with his school life. He starts to forget the assignments that he has to do. He decided to write the things down on a to-do list. He assigns a value `priority` for each of his assignment **(lower value means more important)** so he can decide which he needs to spend more time on.

After a few days, Jamie finds out the list is too large that he can't even manage the list by himself! As you are a good friend of Jamie, help him write a program to support the following operations on the to-do list:

- $set\ a_i\ x_i$ — Add assignment a_i to the to-do list if it is not present, and set its `priority` to x_i. If assignment a_i is already in the to-do list, its `priority` **is changed** to x_i.
- $remove\ a_i$ — Remove assignment a_i from the to-do list if it is present in it.
- $query\ a_i$ — Output the number of assignments that are more important (have a **smaller** `priority` value) than assignment a_i, so Jamie can decide a better schedule. Output -1 if a_i is not in the to-do list.
- $undo\ d_i$ — Undo all changes that have been made in the previous d_i days (not including the day of this operation)

At day 0, the to-do list is empty. In each of the following q days, Jamie will do **exactly one** out of the four operations. If the operation is a $query$, you should **output the result of the query before proceeding to the next day**, or poor Jamie cannot make appropriate decisions.

Input

The first line consists of a single integer q $(1 \le q \le 10^5)$ — the number of operations.

The following q lines consists of the description of the operations. The i-th line consists of the operation that Jamie has done in the i-th day. The query has the following format:

The first word in the line indicates the type of operation. It must be one of the following four: `set`, `remove`, `query`, `undo`.

- If it is a `set` operation, a string a_i and an integer x_i follows $(1 \le x_i \le 10^9)$. a_i is the assignment that need to be set to `priority` x_i.
- If it is a `remove` operation, a string a_i follows. a_i is the assignment that need to be removed.
- If it is a `query` operation, a string a_i follows. a_i is the assignment that needs to be queried.
- If it is a `undo` operation, an integer d_i follows $(0 \le d_i < i)$. d_i is the number of days that changes needed to be undone.

All assignment names a_i only consists of lowercase English letters and have a length $1 \le |a_i| \le 15$.

It is guaranteed that the last operation is a `query` operation.

Output

For each `query` operation, output a single integer — the number of assignments that have a priority lower than assignment a_i, or -1 if a_i is not in the to-do list.

Interaction

If the operation is a $query$, you **should** output the result of the query and flush the output stream before proceeding to the next operation. Otherwise, you may get the verdict `Idleness Limit Exceed`.

For flushing the output stream, please refer to the documentation of your chosen programming language. The flush functions of some common programming languages are listed below:

- C: `fflush(stdout);`
- C++: `cout « flush;`
- Java: `System.out.flush();`

Examples

input

```
8
set chemlabreport 1
set physicsexercise 2
set chinesemockexam 3
query physicsexercise
query chinesemockexam
remove physicsexercise
query physicsexercise
query chinesemockexam
```

output

```
1
2
-1
1
```

input

```
8
set physicsexercise 2
set chinesemockexam 3
set physicsexercise 1
query physicsexercise
query chinesemockexam
undo 4
query physicsexercise
query chinesemockexam
```

output

```
0
1
0
-1
```

input

```
5
query economicsessay
remove economicsessay
query economicsessay
undo 2
query economicsessay
```

output

```
-1
-1
-1
```

input

```
5
set economicsessay 1
remove economicsessay
undo 1
```

```
undo 1
query economicsessay
```

output

```
-1
```