Persistent Stack

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	128 megabytes

Given a set of operations and a stack S, you should process operations in order. Each operation is one of the following types:

- `push x v` : add the element x to the top of S_v.
- `pop v` : remove and print the top element in S_v. If S_v is empty, print `null`.
- `min v` : print the minimum element in S_v. If S_v is empty, print `null`.

S_v means the stack structure at version v. The initial version of the stack is S_0 which is an empty stack. The `pop` and `push` operations create new versions of S enumerated sequentially starting from 1.

Input

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

Then q lines follow, each describing an operation as mentioned above. The pushed elements will be 32-bit signed integers.

It is guaranteed that any version v that appears in the operations will be a previously created version.

Output

For each query of type `min v`, print the minimum value of the S_v and for each query of type `pop v`, print the popped element. Print a single line per query.

Example

standard input	standard output
12 push 5 0 push 3 1 push 6 2 push 4 3 min 4 pop 4 min 5 push 1 2 min 6 pop 0 min 2 min 0	3 4 3 1 null 3 null