Problem J
Stacking Up



Jack and Jason's Pancakes, CC BY-SA 4.0, via Wikimedia Commons

Stacy has recently started work at Stacks"R"Us, a leading manufacturer of stack-related products such as children's blocks, pancake spatulas, and dining hall tray dispensers. As a brand new employee, she has been tasked with testing the latest product in Stacks"R"Us's line of automated electronic stack machines, the *Stackulator 3000*.

The Stackulator 3000 holds a single stack of positive integers in its memory, and supports three instructions, `1`, `d`, and `+`. The first two instructions are fairly standard:

- `1`: push the number 1 onto the top of the stack.
- `d`: duplicate the number on top of the stack, that is, take the number currently on top of the stack and push another copy of it on top. It is an error to execute the `d` instruction on an empty stack.

Unfortunately, due partly to miscommunication between the designer and the hardware team, and partly to confusion about the difference between a stack *pointer* and a stack *entry*, the `+` instruction ended up being a little wonky:

- `+`: pop the top two numbers from the stack, *decrement all remaining stack entries by one*, then add the two popped elements and push the result onto the stack. It is an error to execute the `+` instruction on a stack containing fewer than two integers.

For example, executing the program `1d+11+` produces the sequence of stack states illustrated in Figure J.1, ultimately resulting in a stack containing two entries, 1 on the bottom and 2 on the top.

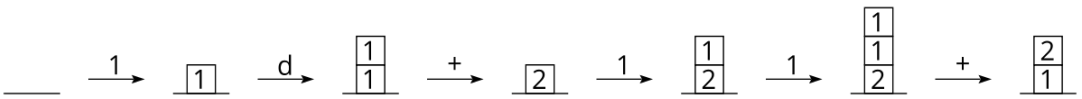


Figure J.1: Execution of `1d+11+`

If a stack entry of 1 is decremented during an addition operation, it is simply removed from the stack, since the stack can only hold positive integers.

Since she is still an apprentice stack tester, Stacy was assigned to work with Stan, a more experienced tester. Stan proposes that they split the work evenly: Stan will come up with stacks, and it will then be Stacy's job to create Stackulator 3000 programs which should generate the given stacks. They can then run Stacy's programs to ensure the Stackulator 3000 gives the expected results.

Stacy could do this, but she is now too busy looking for a different job. That's where you come in: please help Stacy by creating Stackulator 3000 programs that can generate the stacks given by Stan.

Input

The first line of input consists of an integer $1 \leq n \leq 1\,000$. The following line contains n space-separated positive integers, indicating the desired contents of the stack from bottom to top. Each integer x will be in the range $1 \leq x \leq 10^6$.

Output

Output a single line containing a valid Stackulator 3000 program, which, when executed beginning with an empty stack, results in a stack holding the desired contents. Your program must consist of only the characters `1`, `d`, and `+` and be no more than $100\,000$ characters in length. If there is more than one valid program which results in the desired stack, you may output any of them.

Sample Input 1	Sample Output 1
3 1 1 1	1dd

Sample Input 2	Sample Output 2
3 1 2 3	11+1+1+1+1+11+1+1+11+1+