Accounting

Statement

Fluffy is doing financial planning and examining his spending. For the upcoming month, he has sketched out a list of financial events. Each event is associated to its dollar value v_i to him. A friend sending him money would increase the account balance (positive v_i), while buying the hottest video game on its release date would decrease it (negative v_i).

However, the bank absolutely refuses to let Fluffy overdraw from his account. Hence, he might have to skip some of the events (e.g. refusing his friends' money, or not purchasing the game), in order to maintain his account balance. In other words, after each event, Fluffy's bank account balance **must not be negative**.

Can Fluffy make it through the month without incurring the wrath of the bank? What's the largest number of events he can accept, without causing an overdrawn situation?

Input

The first line of input consists of a single integer N, the number of events.

The second line of input consists of N integers $v_1\ v_2\ \ldots\ v_N$. These are the values of the events, going forward in time, from beginning to the end of the month.

Constraints

- $1 \leq N \leq 2 \times 10^5$
- $|v_i| \leq 10^9$

Output

Print out the **maximum** number of events Fluffy can accept.

Examples

Sample Input	Expected Output
6 6 -5 2 -3 1 -2	5

Notes

1. A skeleton file has been given to help you. You should not create a new file or rename the file provided. You should develop your program using this skeleton file.
2. You are free to define your own helper methods and classes (or remove existing ones) if it is suitable but you must put all the new classes, if any, in the same skeleton file provided.

Skeleton File

You are given the skeleton file `Accounting.java`. You should see the following contents when you open the file:

```
/**
 * Name          :
 * Matric. No    :
*/

import java.util.*;

public class Accounting {
    private void run() {
        // implement your "main" method here
    }

    public static void main(String args[]) {
        Accounting runner = new Accounting();
        runner.run();
    }
}
```