AIPO 2024

February 17, 2024

Contributors

The AIPO organisers would like to thank the following people.

For leading the question writing:

- Dr Sabin Tabrica
- Andrew Nash
- Marcu Ştefan-Bogdan
- Zou Fangda

Time limits and input/output bounds will become available when the submission system opens.

1 Greatest Common Divisor

We have learnt that the Euclidian algorithm is the optimal method to calculate the greatest common divisor of two positive natural numbers. What if we have a sequence of numbers for which we need to calculate the greatest common divisor?

The problem considers a sequence of n positive natural numbers and requires to determine the answer for one of the following tasks:

1. The greatest common divisor of the n numbers.
2. The greatest common divisor that can be obtained is by choosing exactly $n-1$ numbers from the sequence.
3. The greatest common divisor that can be obtained is by choosing exactly $n-2$ numbers from the sequence.

Input

- On the first line, an integer c representing the task to solve
- Line 2 contains n the number of integers in the sequence
- The following n lines contain the numbers of the sequence, one number per line

Output

One integer representing the answer for the required task

Constraints

- **For task 1 and task 2:** $3 \leq n \leq 100\,000$.
- **For task 3:** $3 \leq n \leq 2\,000$.
- the numbers in the sequence $2 \leq a_i \leq 2^{63}-1$

Example

Sample Input 1

```
1
5
48
40
20
16
80
```

Sample Output 1

```
4
```

Explanation 1

The first is to be solved. Hence, we need to determine the gcd of all the numbers $48, 40, 20, 16 and 80$.

The answer is 4.

Sample Input 2

```
2
5
48
40
20
16
80
```

Sample Output 2

```
8
```

Explanation 2

The second task is to be solved. If 20 is eliminated, then we have to find the $gcd(16, 48, 40, 80) = 8$, which is the maximum possible.

The answer is 8.

Sample Input 3

```
3
5
48
40
20
16
80
```

Sample Output 3

```
20
```

Explanation 3

The third task is to be solved. If 16 and 48 are eliminated, then we find that $gcd(40, 20, 80) = 20$, which is the maximum possible.

The answer is 20.

2 Justin's Justification Journey

Justin is diligently working on a text editor. A document consists of one or more paragraphs, each ending with a new line character (Enter key press). Within a paragraph, any two consecutive words are separated by space(s) (one or more). Justin has to write his documents on very unusal page sizes. The maximum number of characters that can fit on any line on the page is uniquely determined by an integer (Max) for that page.

Justin's current task is to implement the functionality to justify each paragraph in the document both to the left and to the right.

To do this, He must split each paragraph into lines of length Max (each ending with Enter), maximizing the number of words on each line **without** splitting any words between two lines like th-is.

For left-right justification, he needs to distribute spaces evenly between words on each line so that the position of the last character on the line equals Max. The only exception is the last line of a paragraph, which should remain left-aligned (the normal way of formatting text like this document - with all words separated by a single space, even if the line is not full).

Generally, for left-right justification, it's unlikely that the alignment can be achieved by placing an exactly equal number of spaces between every two consecutive words on a line. Justin believes it's more aesthetically pleasing if, when some consecutive words require additional space compared to other pairs, these spaces should be distributed starting from the beginning of the line.

Write a program that reads the length of a line and the given text and aligns the text both to the left and to the right.

Input

- On the first line, two space separated integers N, representing the number of inputted paragraphs and Max representing the maximum length of a line
- The following N lines contain the document

Output

The justified page according to the required specifications.

Constraints

- $2 \leq \text{Max} \leq 1\,000$.
- length(word) ≤ 25
- length(word) $\leq Max$
- length(paragraph) $\leq 1\,000$ chars

Example

Sample Input 1

```
1 20
Justin has plenty good bats.
```

Sample Output 1

```
Justin   has  plenty
good bats.
```

Explanation 1

On the first line, there have been placed 3 and respectively 2 spaces between the words. No extra spaces were added on the second line as it is the end of the paragraph and only requires to be left aligned.

Sample Input 2

```
2 20
Joan is old.
Melissa has many birthdays.
```

Sample Output 2

```
Joan is old.
Melissa   has   many
birthdays.
```

Explanation 2

For the first line, there were no additional spaces required since the whole paragraph fits within Max length. The second paragraph of the text has been broken into two lines. Three spaces have been added between 'Melissa' and 'has' as well as between 'has' and 'many' so that the line was justified; the rest of the paragraph was only aligned to the left because it fits within the Max length of one line.

3 Lamentable Laptop Loan Losses

The AIPO has launched a computer rental scheme for students to borrow laptops from a library. The school has L laptops available to borrow. Each day a certain number of laptops are taken out by students, who use them for a number of days, and then return them. When a laptop is borrowed, we do not know when it is going to be returned.

One particular day, the AIPO discovered that they had no laptops to lend, since they were all being borrowed by students. So, they decided from then on to introduce a scheme to keep track of the number of laptops taken and returned every day.

To keep track of the inventory each day the net number of laptops (the number of laptops returned minus the number of laptops taken in that day) is recorded. Unfortunately, there is more bad news - the recording system is very buggy, and some records have been lost!

According to the given records, it appears that at the end of some days, there were more than L laptops in the library, and at the end of other days, there were fewer than 0 laptops - impossible situations.

Your task is, given a set of records, to determine the minimum possible number of days for which the records have been lost.

Input/Output

Input

The first line contains two space-separated integers L, T, the number of laptops in the scheme and the number of records that were successfully recorded.

T lines follow, corresponding to a record from the end of a day. You may assume that on day 0, before the records begin, that there are 0 laptops in the inventory.

Each line contains a single integer T_i. If $T_i < 0$ this means that on this particular day $|T_i|$ more laptops were borrowed on this particular day than were returned. Conversely, if $T_i > 0$ this means that on this particular day $|T_i|$ more laptops were returned on this particular day than were borrowed.

Output

A single line containing a single integer M - the minimum number of records that need to be added to make the set of inputted records valid.

Subtasks & Constraints

(See next page if you need an explanation of how Subtask scoring works)

Subtask 1: $T = 2, L \leq 100,000$, 20 points

Subtask 2: $L = 1, T \leq 100,000$ 10 points

Subtask 3: $L < 10, T < 100$, 20 points

Subtask 4: $1 \leq L, T \leq 100,000$, 50 points

Examples

Sample Input 1

```
5 2
-1
-1
```

Sample Output

```
1
```

Sample Input 2

```
4 5
2
-2
1
-2
-4
```

Sample Output 2

```
2
```

Explanation of Sample Input/Output 1 In the begining, the AIPO has 0 laptops available (with $L = 5$ out on loan), and the $T = 2$ records that exist are for two days where there was a net loss of two laptops. The minimum number of days that are missing are 1 - e.g., if there was a day before the given records where 2 more laptops were returned than were borrowed.

Explanation of Sample Input/Output 2 In this case we start with 0 laptops, with 4 out on loan. One possible minimal solution in this case would be is if there was a missing record of 1, between the records with 1 and -2, and a missing record of 4 before the -4.

Explanation of subtasks

For some tasks, we have divided the test cases into **subtasks**.

Each subtask is allocated a certian number of points - for this particular problem, the subtasks are worth 20,10,20 and 50 points repsectively.

To get the points for a sub-task, your solution must solve **all** of the test cases in that particular sub-task within 1 second. E.g., solving 2/3 cases for subtask 1 scores 0 points, where solving 3/3 cases will score 20 points.

An important detail is that you do not have to (but can if you wish) solve all of the sub-tasks in the same submission - e.g. if you make a submission that solves Subtask 1 and scores 20 points, and a seprate solution that solves only subtask 2 and scores 10 points - **your total score on that problem will be 40 points**.

4 Marisa and Magic Onions

Marisa loves eating onions, especially magic onions! She found n magic onions in the Forest of Magic. The i-th magic onion will add a_i health points to Marisa. If a_i is negative, it means that the i-th magic onion is poisonous and will decrease Marisa's health points instead. Marisa's initial health points are 0, and the health points cannot be less than 0 at any time. Since these onions are magical, Marisa can only eat an onion when she first encounters it. She cannot go back and eat a previous onion with a smaller index after eating a later one with a larger index. In other words, Marisa cannot eat the j-th magic onion for any $j < i$ after she eats the i-th magic onion.

Marisa wants to eat as many magic onions as possible, can you help her find out the maximum number of magic onions she can eat?

Input

- The first line contains a single integer n — the number of magic onions.
- The second line contains n integers $a_1, a_2, \cdots, a_n$ — the number of health points that each magic onion adds.

Output

Print a single integer — the maximum number of magic onions Marisa can eat.

Constraints

- **Subtask 1:** $1 \le n \le 10, -1 \le a_i \le 1$ (10 points)
- **Subtask 2:** $1 \le n \le 100, -100 \le a_i \le 100$ (20 points)
- **Subtask 3:** $1 \le n \le 1000, -1000 \le a_i \le 1000$ (20 points)
- **Subtask 4:** $1 \le n \le 10^5, -10^6 \le a_i \le 10^6$(50 points)

Sample Input

```
6
4 -4 1 -3 1 -3
```

Sample Output

```
5
```

Explanation

Marisa can eat all the magic onions except for the second one. Her health points after eating each magic onion will be $4, 5, 2, 3, 0$.

Sample Input 2

```
2
-1 1
```

Sample Output 2

```
1
```

Explanation

Marisa can only eat the second magic onion. She cannot go back and eat the first one after eating the second one.

Sample Input 2

```
3
-1 -2 -3
```

Sample Output 2

```
0
```

Explanation

Marisa can not eat any of the onions.

5 Brian's Bright Balancing Battle

Bulb Town has embarked on an ambitious project to revamp its street lighting system, spearheaded by the young and innovative engineer, Brian. The town boasts a network of N intersections connected by M one-way streets, each intersection featuring a pair of advanced, energy-efficient streetlights. The first of these lights casts its glow over half of each outgoing street, while the second illuminates half of each incoming street. This means, for example, that the first half of the street between intersections A and B is lit by the first light at A, and the second half by the second light at B. For safety reasons, a street is considered too bright if the entire length of the street is lit, that is, if both its ends are illuminated.

Brian's challenge is to create a lighting plan that maximizes the number of streetlights turned on to enhance urban visibility and safety without making any street overly bright by illuminating both of its ends. This careful balance aims to prevent any street from being overly bright or completely dark, ensuring pedestrian safety.

Input

- On the first line, two strictly positive integers N and M, separated by a space, representing the number of intersections and streets in Bulb Town
- On each of the next M lines, a pair of strictly positive integers A and B, separated by a space, between 1 and N, representing a street that is going from intersection A and arriving in intersection B

Output

- On the first line, one single natural number that represents the maximum number of streetlights that can be safely lit.
- On the next M lines will contain a number between 0 and 3 signaling the lit status of the streetlight respecting the following convention:
  - 0: both streetlights at the intersection are off;
  - 1: only the first streetlight at the intersection is on;
  - 2: only the second streetlight at the intersection is on;
  - 3: both streetlights at the intersection are on

Constraints

- $1 \leq N \leq 8\,191$
- $1 \leq M \leq 20\,000$
- There are no streets that unite an intersection with itself

Sample Input

```
4 4
1 2
4 1
4 2
4 3
```

Sample Output

```
6
2
3
3
2
```