Problem A. Simple Array Sum

OS Linux

Given an array of integers, find the sum of its elements.

For example, if the array $ar = [1, 2, 3]$, $1 + 2 + 3 = 6$, so return 6.

Function Description

Complete the $simpleArraySum$ function with the following parameter(s):

- $ar[n]$: an array of integers

Returns

- int: the sum of the array elements

Input Format

The first line contains an integer, n, denoting the size of the array.
The second line contains n space-separated integers representing the array's elements.

Constraints

$0 < n, ar[i] \leq 1000$

Input	Output
STDIN ----- 6 1 2 3 4 10 11 Function -------- ar[] size n = 6 ar = [1, 2, 3, 4, 10, 11]	31

Explanation

Print the sum of the array's elements: $1 + 2 + 3 + 4 + 10 + 11 = 31$.

Problem B. Breaking the Records

OS Linux

Maria plays college basketball and wants to go pro. Each season she maintains a record of her play. She tabulates the number of times she breaks her season record for *most points* and *least points* in a game. Points scored in the first game establish her record for the season, and she begins counting from there.

Example
$scores = [12, 24, 10, 24]$

Scores are in the same order as the games played. She tabulates her results as follows:

				Count	
Game	Score	Minimum	Maximum	Min	Max
0	12	12	12	0	0
1	24	12	24	0	1
2	10	10	24	1	1
3	24	10	24	1	1

Given the scores for a season, determine the number of times Maria breaks her records for *most* and *least* points scored during the season.

Function Description

Complete the *breakingRecords* function in the editor below.

breakingRecords has the following parameter(s):

- *int scores[n]:* points scored per game

Returns

- *int[2]:* An array with the numbers of times she broke her records. Index 0 is for breaking *most points* records, and index 1 is for breaking *least points* records.

Input Format

The first line contains an integer n, the number of games.
The second line contains n space-separated integers describing the respective values of $score_0, score_1, \ldots, score_{n-1}$.

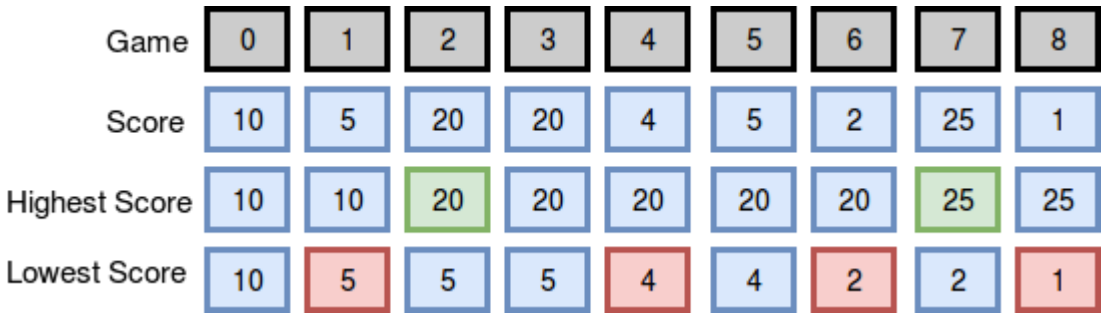
Constraints

- $1 \leq n \leq 1000$
- $0 \leq scores[i] \leq 10^8$

Input	Output
9 10 5 20 20 4 5 2 25 1	2 4

Explanation 0

The diagram below depicts the number of times Maria broke her best and worst records throughout the season:

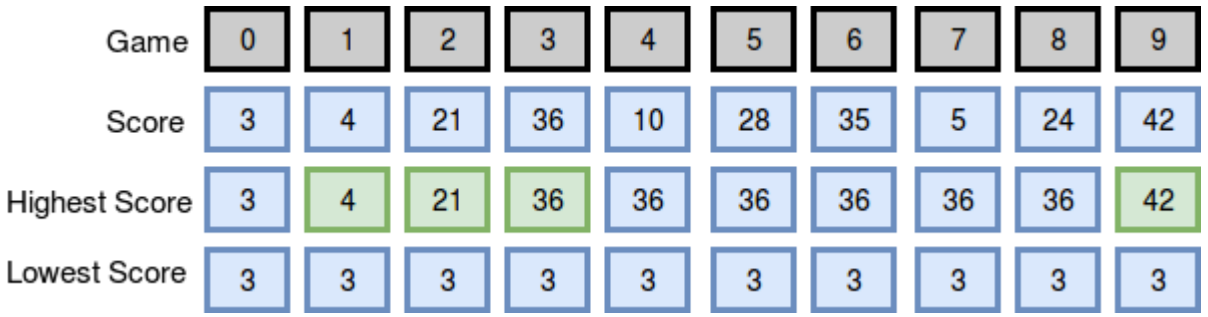


She broke her best record twice (after games 2 and 7) and her worst record four times (after games 1, 4, 6, and 8), so we print `2 4` as our answer. Note that she *did not* break her record for best score during game 3, as her score during that game was *not* strictly greater than her best record at the time.

Input	Output
10 3 4 21 36 10 28 35 5 24 42	4 0

Explanation 1

The diagram below depicts the number of times Maria broke her best and worst records throughout the season:



She broke her best record four times (after games 1, 2, 3, and 9) and her worst record zero times (no score during the season was lower than the one she earned during her first game), so we print `4 0` as our answer.

Problem C. Left Rotation

OS Linux

A $left\ rotation$ operation on a circular array shifts each of the array's elements 1 unit to the left. The elements that fall off the left end reappear at the right end. Given an integer d, rotate the array that many steps to the left and return the result.

Example
$d = 2$
$arr = [1, 2, 3, 4, 5]$

After 2 rotations, $arr' = [3, 4, 5, 1, 2]$.

Function Description

Complete the $rotateLeft$ function with the following parameters:

- $int\ d$: the amount to rotate by
- $int\ arr[n]$: the array to rotate

Returns

- $int[n]$: the rotated array

Input Format

The first line contains two space-separated integers that denote n, the number of integers, and d, the number of left rotations to perform.
The second line contains n space-separated integers that describe $arr[]$.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq d \leq n$
- $1 \leq a[i] \leq 10^6$

Input	Output
STDIN Function ----- -------- 5 4 n = 5 d = 4 1 2 3 4 5 arr = [1, 2, 3, 4, 5]	5 1 2 3 4

Explanation

To perform $d = 4$ left rotations, the array undergoes the following sequence of changes:

$$[1, 2, 3, 4, 5] \rightarrow [2, 3, 4, 5, 1] \rightarrow [3, 4, 5, 1, 2] \rightarrow [4, 5, 1, 2, 3] \rightarrow [5, 1, 2, 3, 4]$$

Problem D. The Hurdle Race

OS Linux

A video player plays a game in which the character competes in a hurdle race. Hurdles are of varying heights, and the characters have a maximum height they can jump. There is a magic potion they can take that will increase their maximum jump height by 1 unit for each dose. How many doses of the potion must the character take to be able to jump all of the hurdles. If the character can already clear all of the hurdles, return 0.

Example
$height = [1, 2, 3, 3, 2]$
$k = 1$

The character can jump 1 unit high initially and must take $3 - 1 = 2$ doses of potion to be able to jump all of the hurdles.

Function Description

Complete the *hurdleRace* function in the editor below.

hurdleRace has the following parameter(s):

- *int k*: the height the character can jump naturally
- *int height[n]*: the heights of each hurdle

Returns

- *int:* the minimum number of doses required, always 0 or more

Input Format

The first line contains two space-separated integers n and k, the number of hurdles and the maximum height the character can jump naturally.
The second line contains n space-separated integers $height[i]$ where $0 \leq i < n$.

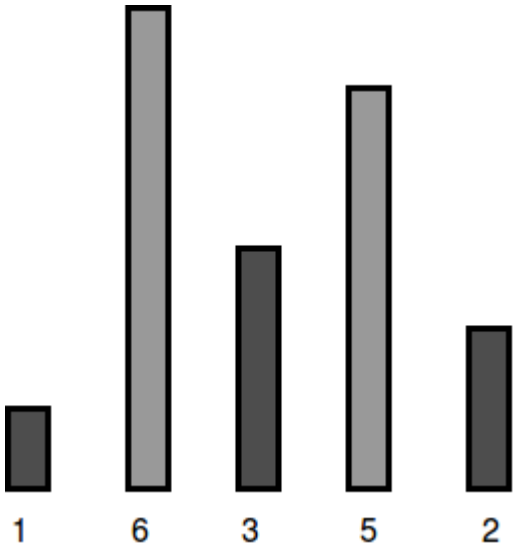
Constraints

- $1 \leq n, k \leq 100$
- $1 \leq height[i] \leq 100$

Input	Output
5 4 1 6 3 5 2	2

Explanation 0

Dan's character can jump a maximum of $k = 4$ units, but the tallest hurdle has a height of $h_1 = 6$:

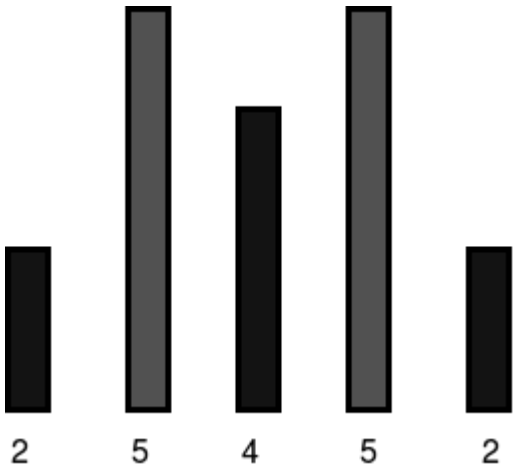


To be able to jump all the hurdles, Dan must drink $6 - 4 = 2$ doses.

Input	Output
5 7 2 5 4 5 2	0

Explanation 1

Dan's character can jump a maximum of $k = 7$ units, which is enough to cross all the hurdles:



Because he can already jump all the hurdles, Dan needs to drink 0 doses.

Problem E. Compare the Triplets

OS Linux

Alice and Bob each created one problem for HackerRank. A reviewer rates the two challenges, awarding points on a scale from *1* to *100* for three categories: *problem clarity*, *originality*, and *difficulty*.

The rating for Alice's challenge is the triplet *a = (a[0], a[1], a[2])*, and the rating for Bob's challenge is the triplet *b = (b[0], b[1], b[2])*.

The task is to calculate their comparison points by comparing each category:

- If *a[i] > b[i]*, then Alice is awarded *1* point.
- If *a[i] < b[i]*, then Bob is awarded *1* point.
- If *a[i] = b[i]*, then neither person receives a point.

Example

a = [1, 2, 3]
b = [3, 2, 1]

- For elements *0*, Bob is awarded a point because *a[0] < b[0]*.
- For the equal elements *a[1]* and *b[1]*, no points are earned.
- Finally, for elements *2*, *a[2] > b[2]* so Alice receives a point.

The return array is *[1, 1]* with Alice's score first and Bob's second.

Function Description

Complete the function *compareTriplets* with the following parameter(s):

- *int a[3]*: Alice's challenge rating
- *int b[3]*: Bob's challenge rating

Returns

- *int[2]*: the first element is Alice's score and the second is Bob's score

Input Format

The first line contains *3* space-separated integers, *a[0]*, *a[1]*, and *a[2]*, the respective values in triplet *a*.
The second line contains *3* space-separated integers, *b[0]*, *b[1]*, and *b[2]*, the respective values in triplet *b*.

Constraints

- $1 \le a[i] \le 100$
- $1 \le b[i] \le 100$

Input	Output
5 6 7 3 6 10	1 1

Explanation 0

In this example:

- $a = (a[0], a[1], a[2]) = (5, 6, 7)$
- $b = (b[0], b[1], b[2]) = (3, 6, 10)$

Now, let's compare each individual score:

- $a[0] > b[0]$, so Alice receives 1 point.
- $a[1] = b[1]$, so nobody receives a point.
- $a[2] < b[2]$, so Bob receives 1 point.

Alice's comparison score is 1, and Bob's comparison score is 1. Thus, we return the array $[1, 1]$.

Input	Output
17 28 30 99 16 8	2 1

Explanation 1

Comparing the 0^{th} elements, $17 < 99$ so Bob receives a point.
Comparing the 1^{st} and 2^{nd} elements, $28 > 16$ and $30 > 8$ so Alice receives two points.
The return array is $[2, 1]$.

Problem F. Plus Minus

OS Linux

Given an array of integers, calculate the ratios of its elements that are $positive$, $negative$, and $zero$. Print the decimal value of each fraction on a new line with 6 places after the decimal.

Note: This challenge introduces precision problems. The test cases are scaled to six decimal places, though answers with absolute error of up to 10^{-4} are acceptable.

Example
$arr = [1, 1, 0, -1, -1]$

There are $n = 5$ elements: two positive, two negative and one zero. Their ratios are $\frac{2}{5} = 0.400000$, $\frac{2}{5} = 0.400000$ and $\frac{1}{5} = 0.200000$. Results are printed as:

```
0.400000
0.400000
0.200000
```

Function Description

Complete the $plusMinus$ function with the following parameter(s):

- $int\ arr[n]$: an array of integers

Print
Print the ratios of positive, negative and zero values in the array. Each value should be printed on a separate line with 6 digits after the decimal. The function should not return a value.

Input Format

The first line contains an integer, n, the size of the array.
The second line contains n space-separated integers that describe $arr[n]$.

Constraints

$0 < n \leq 100$
$-100 \leq arr[i] \leq 100$

Input	Output
STDIN ----- 6 -4 3 -9 0 4 1 Function -------- arr[] size n = 6 arr = [-4, 3, -9, 0, 4, 1]	0.500000 0.333333 0.166667

Explanation

There are 3 positive numbers, 2 negative numbers, and 1 zero in the array.
The proportions of occurrence are positive: $\frac{3}{6} = 0.500000$, negative: $\frac{2}{6} = 0.333333$ and zeros: $\frac{1}{6} = 0.166667$.

Problem G. A Very Big Sum

OS Linux

In this challenge, you need to calculate and print the sum of elements in an array, considering that some integers may be very large.

Function Description

Complete the $aVeryBigSum$ function with the following parameter(s):

- $int\ ar[n]$: an array of integers

Return

- $long$: the sum of the array elements

Input Format

The first line of the input consists of an integer n.
The next line contains n space-separated integers contained in the array.

Output Format

Return the integer sum of the elements in the array.

Constraints
$1 \leq n \leq 10$
$0 \leq ar[i] \leq 10^{10}$

Sample Input

```
STDIN                                                   Function
-----                                                   --------
5                                                       arr[] size n
1000000001 1000000002 1000000003 1000000004 1000000005  arr[...]
```

Output

```
5000000015
```

Note:

The range of the 32-bit integer is $(-2^{31})\ to\ (2^{31}-1)\ or\ [-2147483648, 2147483647]$.

When we add several integer values, the resulting sum might exceed the above range. You might need to use long int C/C++/Java to store such sums.

Problem H. Mini-Max Sum

OS Linux

Given five positive integers, find the minimum and maximum values that can be calculated by summing exactly four of the five integers. Then print the respective minimum and maximum values as a single line of two space-separated long integers.

Example
$arr = [1, 3, 5, 7, 9]$

The minimum sum is $1 + 3 + 5 + 7 = 16$ and the maximum sum is $3 + 5 + 7 + 9 = 24$. The function prints

```
16 24
```

Function Description

Complete the $miniMaxSum$ function with the following parameter(s):

- $arr[5]$: an array of 5 integers

Print

Print two space-separated integers on one line: the minimum sum and the maximum sum of 4 of 5 elements.No value should be returned.

Note For some languages, like C, C++, and Java, the sums may require that you use a long integer due to their size.

Input Format

A single line of five space-separated integers.

Constraints

$1 \leq arr[i] \leq 10^9$

Input	Output
1 2 3 4 5	10 14

Explanation

The numbers are 1, 2, 3, 4, and 5. Calculate the following sums using four of the five integers:

1. Sum everything except 1, the sum is $2 + 3 + 4 + 5 = 14$.
2. Sum everything except 2, the sum is $1 + 3 + 4 + 5 = 13$.
3. Sum everything except 3, the sum is $1 + 2 + 4 + 5 = 12$.
4. Sum everything except 4, the sum is $1 + 2 + 3 + 5 = 11$.
5. Sum everything except 5, the sum is $1 + 2 + 3 + 4 = 10$.

Hints: Beware of integer overflow! Use a 64-bit integer to store the sums.

Need help to get started? Try the Solve Me First problem.

Problem I. Birthday Cake Candles

OS Linux

You are in charge of the cake for a child's birthday. It will have one candle for each year of their total age. They will only be able to blow out the tallest of the candles. Your task is to count how many candles are the tallest.

Example

$candles = [4, 4, 1, 3]$

The tallest candles are `4` units high. There are `2` candles with this height, so the function should return `2`.

Function Description

Complete the function $birthdayCakeCandles$ with the following parameter(s):

- $int\ candles[n]$: the candle heights

Returns

- int: the number of candles that are tallest

Input Format

The first line contains a single integer, n, the size of $candles[]$.
The second line contains n space-separated integers, where each integer i describes the height of $candles[i]$.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq candles[i] \leq 10^7$

Input	Output
4 3 2 1 3	2

Explanation 0

Candle heights are $[3, 2, 1, 3]$. The tallest candles are 3 units, and there are 2 of them.