Problem A. Tetrahedral Number

Time limit 2000 ms
Mem limit 1048576 kB

Problem Statement

You are given an integer N.

Print all triples of non-negative integers (x, y, z) such that $x + y + z \leq N$ in ascending lexicographical order.

▶ What is lexicographical order for non-negative integer triples?

Constraints

- $0 \leq N \leq 21$
- N is an integer.

Input

The input is given from Standard Input in the following format:

```
N
```

Output

Print all triples of non-negative integers (x, y, z) such that $x + y + z \leq N$ in ascending lexicographical order, with x, y, z separated by spaces, one triple per line.

Sample 1

Input	Output
3	0 0 0 0 0 1 0 0 2 0 0 3 0 1 0 0 1 1 0 1 2 0 2 0 0 2 1 0 3 0 1 0 0 1 0 1 1 0 2 1 1 0 1 1 1 1 2 0 2 0 0 2 0 1 2 1 0 3 0 0

Sample 2

Input	Output
4	0 0 0 0 0 1 0 0 2 0 0 3 0 0 4 0 1 0 0 1 1 0 1 2 0 1 3 0 2 0 0 2 1 0 2 2 0 3 0 0 3 1 0 4 0 1 0 0 1 0 1 1 0 2 1 0 3 1 1 0 1 1 1 1 1 2 1 2 0 1 2 1 1 3 0 2 0 0 2 0 1 2 0 2 2 1 0 2 1 1 2 2 0 3 0 0 3 0 1 3 1 0 4 0 0

Problem B. Array Reversal

OS Linux

Given an array, of size n, reverse it.

Example: If array, $arr = [1, 2, 3, 4, 5]$, after reversing it, the array should be,
$arr = [5, 4, 3, 2, 1]$.

Input Format

The first line contains an integer, n, denoting the size of the array. The next line contains n space-separated integers denoting the elements of the array.

Constraints

$1 \leq n \leq 1000$
$1 \leq arr_i \leq 1000$, where arr_i is the i^{th} element of the array.

Output Format

The output is handled by the code given in the editor, which would print the array.

Input	Output
6 16 13 7 2 1 12	12 1 2 7 13 16

Explanation 0

Given array, arr = $[16, 13, 7, 2, 1, 12]$. After reversing the array, arr = $[12, 1, 2, 7, 13, 16]$

Input	Output
7 1 13 15 20 12 13 2	2 13 12 20 15 13 1
Input	**Output**
8 15 5 16 15 17 11 5 11	11 5 11 17 15 16 5 15

Problem C. 1D Arrays in C

OS Linux

An array is a container object that holds a fixed number of values of a single type. To create an array in C, we can do `int arr[n];`. Here, arr, is a variable array which holds up to 10 integers. The above array is a static array that has memory allocated at compile time. A dynamic array can be created in C, using the malloc function and the memory is allocated on the heap at runtime. To create an integer array, arr of size n, `int *arr = (int*)malloc(n * sizeof(int))`, where arr points to the base address of the array. When you have finished with the array, use `free(arr)` to deallocate the memory.

In this challenge, create an array of size n dynamically, and read the values from stdin. Iterate the array calculating the sum of all elements. Print the sum and free the memory where the array is stored.

While it is true that you can sum the elements as they are read, without first storing them to an array, but you will not get the experience working with an array. Efficiency will be required later.

Input Format

The first line contains an integer, n.
The next line contains n space-separated integers.

Constraints

$1 \leq n \leq 1000$
$1 \leq a[i] \leq 1000$

Output Format

Print the sum of the integers in the array.

Input	Output
6 16 13 7 2 1 12	51
Input	**Output**
7 1 13 15 20 12 13 2	76

Problem D. Vector-Sort

OS Linux

You are given N integers.Sort the N integers and print the sorted order.
Store the N integers in a vector.Vectors are sequence containers representing arrays that can change in size.

- *Declaration:*

```
vector<int>v; (creates an empty vector of integers)
```

- *Size:*

`int size=v.size();`

- *Pushing an integer into a vector:*

```
v.push_back(x);(where x is an integer.The size increases by 1 aft
```

- *Popping the last element from the vector:*

```
v.pop_back(); (After this the size decreases by 1)
```

- *Sorting a vector:*

```
sort(v.begin(),v.end()); (Will sort all the elements in the vecto
```

To know more about vectors, Click Here

Input Format

The first line of the input contains N where N is the number of integers. The next line contains N integers.
Constraints
$1 <= N <= 10^5$
$1 <= V_i <= 10^9$, where V_i is the i^{th} integer in the vector.

Output Format

Print the integers in the sorted order one by one in a single line followed by a space.

Input	Output
5 1 6 10 8 4	1 4 6 8 10

Problem E. Next Round

Time limit 3000 ms
Mem limit 262144 kB
Input file stdin
Output file stdout

"Contestant who earns a score equal to or greater than the k-th place finisher's score will advance to the next round, as long as the contestant earns a positive score..." — an excerpt from contest rules.

A total of n participants took part in the contest ($n \geq k$), and you already know their scores. Calculate how many participants will advance to the next round.

Input

The first line of the input contains two integers n and k ($1 \leq k \leq n \leq 50$) separated by a single space.

The second line contains n space-separated integers $a_1, a_2, ..., a_n$ ($0 \leq a_i \leq 100$), where a_i is the score earned by the participant who got the i-th place. The given sequence is non-increasing (that is, for all i from 1 to n - 1 the following condition is fulfilled: $a_i \geq a_{i+1}$).

Output

Output the number of participants who advance to the next round.

Examples

Input	Output
8 5 10 9 8 7 7 7 5 5	6

Input	Output
4 2 0 0 0 0	0

Note

In the first example the participant on the 5th place earned 7 points. As the participant on the 6th place also earned 7 points, there are 6 advancers.

In the second example nobody got a positive score.

Problem F. Vector II

Time limit	2000 ms
Mem limit	262144 kB
OS	Linux

For n dynamic arrays A_i ($i = 0, 1, ..., n - 1$), perform a sequence of the following operations:

- pushBack(t, x): Add element x at the end of A_t.
- dump(t): Print all elements in A_t.
- clear(t): Clear A_t. If A_t is empty, do nothing.

A_i is a 0-origin array and it is empty in the initial state.

Input

The input is given in the following format.

n q
$query_1$
$query_2$
:
$query_q$

Each query $query_i$ is given by

`0` t x

or

`1` t

or

`2` t

where the first digits 0, 1 and 2 represent pushBack, dump and clear operations respectively.

Output

For each dump operation, print elements of A_t a line. Separete adjacency elements by a space character (do not print the space after the last element). Note that, if the array is empty, an empty line should be printed.

Constraints

- $1 \leq n \leq 1,000$
- $1 \leq q \leq 500,000$
- $-1,000,000,000 \leq x \leq 1,000,000,000$
- The total number of elements printed by dump operations do not exceed 500,000

Input	Output
3 13 0 0 1 0 0 2 0 0 3 0 1 -1 0 2 4 0 2 5 1 0 1 1 1 2 2 1 1 0 1 1 1 2	1 2 3 -1 4 5 1 2 3 4 5

Problem G. Vector

Time limit	1000 ms
Mem limit	262144 kB
OS	Linux

For a dynamic array $A = \{a_0, a_1, ...\}$ of integers, perform a sequence of the following operations:

- pushBack(x): add element x at the end of A
- randomAccess(p):print element a_p
- popBack(): delete the last element of A

A is a 0-origin array and it is empty in the initial state.

Input

The input is given in the following format.

q
$query_1$
$query_2$
:
$query_q$

Each query $query_i$ is given by

`0` x

or

`1` p

or

`2`

where the first digits 0, 1 and 2 represent pushBack, randomAccess and popBack operations respectively.

randomAccess and popBack operations will not be given for an empty array.

Output

For each randomAccess, print a_p in a line.

Constraints

- $1 \leq q \leq 200,000$
- $0 \leq p <$ the size of A
- $-1,000,000,000 \leq x \leq 1,000,000,000$

Input	Output
8 0 1 0 2 0 3 2 0 4 1 0 1 1 1 2	1 2 4

Problem H. Easy Fibonacci

Time limit	1000 ms
Mem limit	262144 kB
OS	Windows

Given a number N. Print first N numbers of the **Fibonacci** sequence.

Note: In order to create the **Fibonacci** sequence use the following function:

- fib(1) = 0.
- fib(2) = 1.
- fib(n) = fib(n - 1) + fib(n - 2).

Input

Only one line containing a number $N\,(1 \leq N \leq 45)$.

Output

Print the first N numbers from the **Fibonacci** Sequence .

Examples

Input	Output
7	0 1 1 2 3 5 8

Note

For more information visit Fibonacci: https://www.mathsisfun.com/numbers/fibonacci-sequence.html.

Problem I. Vector-Erase

OS Linux

You are provided with a vector of N integers. Then, you are given 2 queries. For the first query, you are provided with 1 integer, which denotes a position in the vector. The value at this position in the vector needs to be erased. The next query consists of 2 integers denoting a range of the positions in the vector. The elements which fall under that range should be removed. The second query is performed on the updated vector which we get after performing the first query.
The following are some useful vector functions:

- *erase(int position):*

```
Removes the element present at position.
Ex: v.erase(v.begin()+4); (erases the fifth element of the vector
```

- *erase(int start,int end):*

```
Removes the elements in the range from start to end inclusive of
Ex:v.erase(v.begin()+2,v.begin()+5);(erases all the elements from
```

Input Format

The first line of the input contains an integer N.The next line contains N space separated integers(1-based index).The third line contains a single integer x,denoting the position of an element that should be removed from the vector.The fourth line contains two integers a and b denoting the range that should be erased from the vector inclusive of a and exclusive of b.

Constraints

$1 \leq N \leq 10^5$
$1 \leq x \leq N$
$1 \leq a < b \leq N$

Output Format

Print the size of the vector in the first line and the elements of the vector after the two erase operations in the second line separated by space.

Input	Output
6 1 4 6 2 8 9 2 2 4	3 1 8 9

Explanation

The first query is to erase the 2nd element in the vector, which is 4. Then, modifed vector is {1 6 2 8 9}, we want to remove the range of 2~4, which means the 2nd and 3rd elements should be removed. Then 6 and 2 in the modified vector are removed and we finally get {1 8 9}

Problem J. Digit Frequency

OS Linux

Given a string, s, consisting of alphabets and digits, find the frequency of each digit in the given string.

Input Format

The first line contains a string, num which is the given number.

Constraints

$1 \leq len(num) \leq 1000$
All the elements of num are made of english alphabets and digits.

Output Format

Print ten space-separated integers in a single line denoting the frequency of each digit from 0 to 9.

Input	Output
a11472o5t6	0 2 1 0 1 1 1 1 0 0

Explanation 0

In the given string:

- 1 occurs two times.
- $2, 4, 5, 6$ and 7 occur one time each.
- The remaining digits $0, 3, 8$ and 9 don't occur at all.

Input	Output
lw4n88j12n1	0 2 1 0 1 0 0 0 2 0
Input	**Output**
1v88886l256338ar0ekk	1 1 1 2 0 1 2 0 5 0

Problem K. Triple

Time limit 1000 ms
Mem limit 262144 kB

Given an array a of n elements, print any value that appears at least three times or print `-1` if there is no such value.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 2 \cdot 10^5$) — the length of the array.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — the elements of the array.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print any value that appears at least three times or print `-1` if there is no such value.

Examples

Input	Output
7 1 1 3 2 2 2 7 2 2 3 3 4 2 2 8 1 4 3 4 3 2 4 1 9 1 1 1 2 2 2 3 3 3 5 1 5 2 4 3 4 4 4 4 4	-1 2 2 4 3 -1 4

Note

In the first test case there is just a single element, so it can't occur at least three times and the answer is `-1`.

In the second test case, all three elements of the array are equal to 2, so 2 occurs three times, and so the answer is 2.

For the third test case, 2 occurs four times, so the answer is 2.

For the fourth test case, 4 occurs three times, so the answer is 4.

For the fifth test case, 1, 2 and 3 all occur at least three times, so they are all valid outputs.

For the sixth test case, all elements are distinct, so none of them occurs at least three times and the answer is `-1`.

Problem L. Good Sequence

Time limit 2000 ms
Mem limit 262144 kB

Problem Statement

You are given a sequence of positive integers of length N, $a = (a_1, a_2, ..., a_N)$. Your objective is to remove some of the elements in a so that a will be a **good sequence**.

Here, an sequence b is a **good sequence** when the following condition holds true:

- For each element x in b, the value x occurs exactly x times in b.

For example, $(3, 3, 3)$, $(4, 2, 4, 1, 4, 2, 4)$ and $()$ (an empty sequence) are good sequences, while $(3, 3, 3, 3)$ and $(2, 4, 1, 4, 2)$ are not.

Find the minimum number of elements that needs to be removed so that a will be a good sequence.

Constraints

- $1 \leq N \leq 10^5$
- a_i is an integer.
- $1 \leq a_i \leq 10^9$

Input

Input is given from Standard Input in the following format:

```
N
a_1 a_2 ... a_N
```

Output

Print the minimum number of elements that needs to be removed so that a will be a good sequence.

Sample 1

Input	Output
4 3 3 3 3	1

We can, for example, remove one occurrence of 3. Then, $(3, 3, 3)$ is a good sequence.

Sample 2

Input	Output
5 2 4 1 4 2	2

We can, for example, remove two occurrences of 4. Then, $(2, 1, 2)$ is a good sequence.

Sample 3

Input	Output
6 1 2 2 3 3 3	0

Sample 4

Input	Output
1 1000000000	1

Remove one occurrence of 10^9. Then, $()$ is a good sequence.

Sample 5

Input	Output
8 2 7 1 8 2 8 1 8	5

Problem M. Largest and Second Largest

Time limit	1000 ms
Code length Limit	50000 B
OS	Linux

You are given an array A of N integers.
Find the **maximum** sum of **two distinct** integers in the array.

Note: It is guaranteed that there exist at least two distinct integers in the array.

Input Format

- The first line of input will contain a single integer T, denoting the number of test cases.
- Each test case consists of multiple lines of input.
  - The first line of each test case contains single integer N — the size of the array.
  - The next line contains N space-separated integers, denoting the array A.

Output Format

For each test case, output on a new line, the maximum sum of two distinct integers in the array.

Constraints

- $1 \leq T \leq 1000$
- $2 \leq N \leq 10^5$
- $1 \leq A_i \leq 1000$
- The sum of N over all test cases does not exceed $2 \cdot 10^5$.

Sample 1

Input	Output
4 3 4 1 6 7 3 7 2 1 1 5 3 5 8 2 9 4 9 2 1 2	10 12 17 3

Test case 1: The maximum sum of two distinct elements is $4 + 6 = 10$.

Test case 2: The maximum sum of two distinct elements is $7 + 5 = 12$.

Test case 3: The maximum sum of two distinct elements is $8 + 9 = 17$.

Test case 4: The maximum sum of two distinct elements is $1 + 2 = 3$.

Problem N. Juicer

Time limit 1000 ms
Mem limit 262144 kB

Kolya is going to make fresh orange juice. He has n oranges of sizes $a_1, a_2, ..., a_n$. Kolya will put them in the juicer in the fixed order, starting with orange of size a_1, then orange of size a_2 and so on. To be put in the juicer the orange must have size not exceeding b, so if Kolya sees an orange that is strictly greater he throws it away and continues with the next one.

The juicer has a special section to collect waste. It overflows if Kolya squeezes oranges of the total size strictly greater than d. When it happens Kolya empties the waste section (even if there are no more oranges) and continues to squeeze the juice. How many times will he have to empty the waste section?

Input

The first line of the input contains three integers n, b and d ($1 \le n \le 100\,000$, $1 \le b \le d \le 1\,000\,000$) — the number of oranges, the maximum size of the orange that fits in the juicer and the value d, which determines the condition when the waste section should be emptied.

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 1\,000\,000$) — sizes of the oranges listed in the order Kolya is going to try to put them in the juicer.

Output

Print one integer — the number of times Kolya will have to empty the waste section.

Examples

Input	Output
2 7 10 5 6	1

Input	Output
1 5 10 7	0

Input	Output
3 10 10 5 7 7	1

Input	Output
1 1 1 1	0

Note

In the first sample, Kolya will squeeze the juice from two oranges and empty the waste section afterwards.

In the second sample, the orange won't fit in the juicer so Kolya will have no juice at all.

Problem O. Advantage

Time limit 2000 ms
Mem limit 262144 kB

There are n participants in a competition, participant i having a strength of s_i.

Every participant wonders how much of an advantage they have over the other best participant. In other words, each participant i wants to know the difference between s_i and s_j, where j is the strongest participant in the competition, not counting i (a difference can be negative).

So, they ask you for your help! For each i ($1 \leq i \leq n$) output the difference between s_i and the maximum strength of any participant other than participant i.

Input

The input consists of multiple test cases. The first line contains an integer t ($1 \leq t \leq 1000$) — the number of test cases. The descriptions of the test cases follow.

The first line of each test case contains an integer n ($2 \leq n \leq 2 \cdot 10^5$) — the length of the array.

The following line contains n space-separated positive integers s_1, s_2, ..., s_n ($1 \leq s_i \leq 10^9$) — the strengths of the participants.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output n space-separated integers. For each i ($1 \leq i \leq n$) output the difference between s_i and the maximum strength of any other participant.

Examples

Input	Output
5 4 4 7 3 5 2 1 2 5 1 2 3 4 5 3 4 9 4 4 4 4 4 4	-3 2 -4 -2 -1 1 -4 -3 -2 -1 1 -5 5 -5 0 0 0 0

Note

For the first test case:

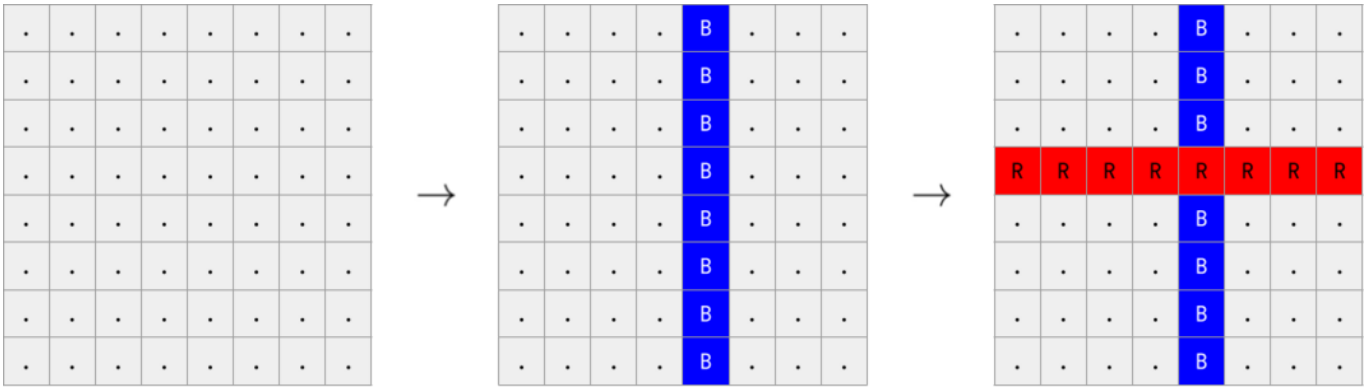
- The first participant has a strength of 4 and the largest strength of a participant different from the first one is 7, so the answer for the first participant is $4 - 7 = -3$.
- The second participant has a strength of 7 and the largest strength of a participant different from the second one is 5, so the answer for the second participant is $7 - 5 = 2$.
- The third participant has a strength of 3 and the largest strength of a participant different from the third one is 7, so the answer for the third participant is $3 - 7 = -4$.
- The fourth participant has a strength of 5 and the largest strength of a participant different from the fourth one is 7, so the answer for the fourth participant is $5 - 7 = -2$.

Problem P. Stripes

Time limit 1000 ms
Mem limit 262144 kB

On an 8×8 grid, some horizontal rows have been painted red, and some vertical columns have been painted blue, in some order. The stripes are drawn sequentially, one after the other. When the stripe is drawn, it repaints all the cells through which it passes.

Determine which color was used last.



The red stripe was painted after the blue one, so the answer is `R`.

Input

The first line of the input contains a single integer t ($1 \le t \le 4000$) — the number of test cases. The description of test cases follows. There is an empty line before each test case.

Each test case consists of 8 lines, each containing 8 characters. Each of these characters is either '`R`', '`B`', or '`.`', denoting a red square, a blue square, and an unpainted square, respectively.

It is guaranteed that the given field is obtained from a colorless one by drawing horizontal red rows and vertical blue columns.

At least one stripe is painted.

Output

For each test case, output '`R`' if a red stripe was painted last, and '`B`' if a blue stripe was painted last (without quotes).

Examples

Input	Output
4 ....B... ....B... ....B... RRRRRRRR ....B... ....B... ....B... ....B... RRRRRRRB B......B B......B B......B B......B B......B B......B RRRRRRRB RRRRRRBB .B.B..BB RRRRRRBB .B.B..BB .B.B..BB RRRRRRBB .B.B..BB .B.B..BB .... RRRRRRRR	R B B R

Note

The first test case is pictured in the statement.

In the second test case, the first blue column is painted first, then the first and last red rows, and finally the last blue column. Since a blue stripe is painted last, the answer is `B`.

Problem Q. Kefa and First Steps

Time limit 2000 ms
Mem limit 262144 kB

Kefa decided to make some money doing business on the Internet for exactly n days. He knows that on the i-th day ($1 \le i \le n$) he makes a_i money. Kefa loves progress, that's why he wants to know the length of the maximum non-decreasing subsegment in sequence a_i. Let us remind you that the subsegment of the sequence is its continuous fragment. A subsegment of numbers is called non-decreasing if all numbers in it follow in the non-decreasing order.

Help Kefa cope with this task!

Input

The first line contains integer n ($1 \le n \le 10^5$).

The second line contains n integers $a_1,\ a_2,\ \ldots,\ a_n$ ($1 \le a_i \le 10^9$).

Output

Print a single integer — the length of the maximum non-decreasing subsegment of sequence a.

Examples

Input	Output
6 2 2 1 3 4 1	3

Input	Output
3 2 2 9	3

Note

In the first test the maximum non-decreasing subsegment is the numbers from the third to the fifth one.

In the second test the maximum non-decreasing subsegment is the numbers from the first to the third one.

Problem R. 2D Array - DS

OS Linux

Given a 6×6 *2D Array*, arr:

```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```

An hourglass in A is a subset of values with indices falling in this pattern in arr's graphical representation:

```
a b c
  d
e f g
```

There are 16 hourglasses in arr. An *hourglass sum* is the sum of an hourglass' values. Calculate the hourglass sum for every hourglass in arr, then print the *maximum* hourglass sum. The array will always be 6×6.

Example

$arr =$

```
-9 -9 -9  1 1 1
 0 -9  0  4 3 2
-9 -9 -9  1 2 3
 0  0  8  6 6 0
 0  0  0 -2 0 0
 0  0  1  2 4 0
```

The 16 hourglass sums are:

```
-63, -34, -9, 12,
-10,   0, 28, 23,
-27, -11, -2, 10,

```

```
        9,  17, 25, 18
```

The highest hourglass sum is 28 from the hourglass beginning at row 1, column 2:

```
0 4 3
  1
8 6 6
```

Note: If you have already solved the Java domain's *Java 2D Array* challenge, you may wish to skip this challenge.

Function Description

Complete the function *hourglassSum* in the editor below.

hourglassSum has the following parameter(s):

- *int arr[6][6]*: an array of integers

Returns

- *int:* the maximum hourglass sum

Input Format

Each of the 6 lines of inputs $arr[i]$ contains 6 space-separated integers $arr[i][j]$.

Constraints

- $-9 \leq arr[i][j] \leq 9$
- $0 \leq i, j \leq 5$

Output Format

Print the largest (maximum) hourglass sum found in arr.

Input	Output
1 1 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 0 0 0 2 4 4 0 0 0 0 2 0 0 0 0 1 2 4 0	19

Explanation

arr contains the following hourglasses:

```
1 1 1   1 1 0   1 0 0   0 0 0
  1       0       0       0
1 1 1   1 1 0   1 0 0   0 0 0

0 1 0   1 0 0   0 0 0   0 0 0
  1       1       0       0
0 0 2   0 2 4   2 4 4   4 4 0

1 1 1   1 1 0   1 0 0   0 0 0
  0       2       4       4
0 0 0   0 0 2   0 2 0   2 0 0

0 0 2   0 2 4   2 4 4   4 4 0
  0       0       2       0
0 0 1   0 1 2   1 2 4   2 4 0
```

The hourglass with the maximum sum (**19**) is:

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

Problem S. Sereja and Dima

Time limit	1000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

Sereja and Dima play a game. The rules of the game are very simple. The players have n cards in a row. Each card contains a number, all numbers on the cards are distinct. The players take turns, Sereja moves first. During his turn a player can take one card: either the leftmost card in a row, or the rightmost one. The game ends when there is no more cards. The player who has the maximum sum of numbers on his cards by the end of the game, wins.

Sereja and Dima are being greedy. Each of them chooses the card with the larger number during his move.

Inna is a friend of Sereja and Dima. She knows which strategy the guys are using, so she wants to determine the final score, given the initial state of the game. Help her.

Input

The first line contains integer n $(1 \le n \le 1000)$ — the number of cards on the table. The second line contains space-separated numbers on the cards from left to right. The numbers on the cards are distinct integers from 1 to 1000.

Output

On a single line, print two integers. The first number is the number of Sereja's points at the end of the game, the second number is the number of Dima's points at the end of the game.

Examples

Input	Output
4 4 1 2 10	12 5

Input	Output
7 1 2 3 4 5 6 7	16 12

Note

In the first sample Sereja will take cards with numbers 10 and 2, so Sereja's sum is 12. Dima will take cards with numbers 4 and 1, so Dima's sum is 5.

Problem T. Puzzles

Time limit	1000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

The end of the school year is near and Ms. Manana, the teacher, will soon have to say goodbye to a yet another class. She decided to prepare a goodbye present for her n students and give each of them a jigsaw puzzle (which, as wikipedia states, is a tiling puzzle that requires the assembly of numerous small, often oddly shaped, interlocking and tessellating pieces).

The shop assistant told the teacher that there are m puzzles in the shop, but they might differ in difficulty and size. Specifically, the first jigsaw puzzle consists of f_1 pieces, the second one consists of f_2 pieces and so on.

Ms. Manana doesn't want to upset the children, so she decided that the difference between the numbers of pieces in her presents must be as small as possible. Let A be the number of pieces in the largest puzzle that the teacher buys and B be the number of pieces in the smallest such puzzle. She wants to choose such n puzzles that A - B is minimum possible. Help the teacher and find the least possible value of A - B.

Input

The first line contains space-separated integers n and m ($2 \le n \le m \le 50$). The second line contains m space-separated integers $f_1, f_2, \ldots, f_m$ ($4 \le f_i \le 1000$) — the quantities of pieces in the puzzles sold in the shop.

Output

Print a single integer — the least possible difference the teacher can obtain.

Examples

Input	Output
4 6 10 12 10 7 5 22	5

Note

Sample 1. The class has 4 students. The shop sells 6 puzzles. If Ms. Manana buys the first four puzzles consisting of 10, 12, 10 and 7 pieces correspondingly, then the difference between the sizes of the largest and the smallest puzzle will be equal to 5. It is impossible to obtain a smaller difference. Note that the teacher can also buy puzzles 1, 3, 4 and 5 to obtain the difference 5.

Problem U. Polycarp Training

Time limit 2000 ms
Mem limit 262144 kB

Polycarp wants to train before another programming competition. During the first day of his training he should solve exactly 1 problem, during the second day — exactly 2 problems, during the third day — exactly 3 problems, and so on. During the k-th day he should solve k problems.

Polycarp has a list of n contests, the i-th contest consists of a_i problems. During each day Polycarp has to choose **exactly one** of the contests he didn't solve yet and solve it. He solves **exactly k problems from this contest**. Other problems are discarded from it. If there are no contests consisting of at least k problems that Polycarp didn't solve yet during the k-th day, then Polycarp stops his training.

How many days Polycarp can train if he chooses the contests optimally?

Input

The first line of the input contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of contests.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$) — the number of problems in the i-th contest.

Output

Print one integer — the maximum number of days Polycarp can train if he chooses the contests optimally.

Examples

Input	Output
4 3 1 4 1	3

Input	Output
3 1 1 1	1

Input	Output
5 1 1 1 2 2	2

Problem V. Qualifying Contest

Time limit 1000 ms
Mem limit 262144 kB

Very soon Berland will hold a School Team Programming Olympiad. From each of the m Berland regions a team of two people is invited to participate in the olympiad. The qualifying contest to form teams was held and it was attended by n Berland students. There were at least two schoolboys participating from each of the m regions of Berland. The result of each of the participants of the qualifying competition is an integer score from 0 to 800 inclusive.

The team of each region is formed from two such members of the qualifying competition of the region, that none of them can be replaced by a schoolboy of the same region, not included in the team and who received a **greater** number of points. There may be a situation where a team of some region can not be formed uniquely, that is, there is more than one school team that meets the properties described above. In this case, the region needs to undertake an additional contest. The two teams in the region are considered to be different if there is at least one schoolboy who is included in one team and is not included in the other team. It is guaranteed that for each region at least two its representatives participated in the qualifying contest.

Your task is, given the results of the qualifying competition, to identify the team from each region, or to announce that in this region its formation requires additional contests.

Input

The first line of the input contains two integers n and m ($2 \le n \le 100\,000$, $1 \le m \le 10\,000$, $n \ge 2m$) — the number of participants of the qualifying contest and the number of regions in Berland.

Next n lines contain the description of the participants of the qualifying contest in the following format: Surname (a string of length from 1 to 10 characters and consisting of large and small English letters), region number (integer from 1 to m) and the number of points scored by the participant (integer from 0 to 800, inclusive).

It is guaranteed that all surnames of all the participants are distinct and at least two people participated from each of the *m* regions. The surnames that only differ in letter cases, should be considered distinct.

Output

Print *m* lines. On the *i*-th line print the team of the *i*-th region — the surnames of the two team members in an arbitrary order, or a single character "?" (without the quotes) if you need to spend further qualifying contests in the region.

Examples

Input	Output
5 2 Ivanov 1 763 Andreev 2 800 Petrov 1 595 Sidorov 1 790 Semenov 2 503	Sidorov Ivanov Andreev Semenov

Input	Output
5 2 Ivanov 1 800 Andreev 2 763 Petrov 1 800 Sidorov 1 800 Semenov 2 503	? Andreev Semenov

Note

In the first sample region teams are uniquely determined.

In the second sample the team from region 2 is uniquely determined and the team from region 1 can have three teams: "Petrov"-"Sidorov", "Ivanov"-"Sidorov", "Ivanov" -"Petrov", so it is impossible to determine a team uniquely.