Codeforces Round #521 (Div. 3)

A. Frog Jumping

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

A frog is currently at the point 0 on a coordinate axis Ox. It jumps by the following algorithm: the first jump is a units to the right, the second jump is b units to the left, the third jump is a units to the right, the fourth jump is b units to the left, and so on.

Formally:

- if the frog has jumped an even number of times (before the current jump), it jumps from its current position x to position $x+a$;
- otherwise it jumps from its current position x to position $x-b$.

Your task is to calculate the position of the frog after k jumps.

But... One more thing. You are watching t different frogs so you have to answer t independent queries.

Input

The first line of the input contains one integer t $(1 \le t \le 1000)$ — the number of queries.

Each of the next t lines contain queries (one query per line).

The query is described as three space-separated integers a, b, k $(1 \le a, b, k \le 10^9)$ — the lengths of two types of jumps and the number of jumps, respectively.

Output

Print t integers. The i-th integer should be the answer for the i-th query.

Example

input

```
6
5 2 3
100 1 4
1 10 5
1000000000 1 6
1 1 1000000000
1 1 999999999
```

output

```
8
198
-17
2999999997
0
1
```

Note

In the first query frog jumps 5 to the right, 2 to the left and 5 to the right so the answer is $5-2+5=8$.

In the second query frog jumps 100 to the right, 1 to the left, 100 to the right and 1 to the left so the answer is $100-1+100-1=198$.

In the third query the answer is $1-10+1-10+1=-17$.

In the fourth query the answer is $10^9-1+10^9-1+10^9-1=2999999997$.

In the fifth query all frog's jumps are neutralized by each other so the answer is 0.

The sixth query is the same as the fifth but without the last jump so the answer is 1.

B. Disturbed People

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is a house with n flats situated on the main street of Berlatov. Vova is watching this house every night. The house can be represented as an array of n integer numbers $a_1, a_2, \dots, a_n$, where $a_i = 1$ if in the i-th flat the light is on and $a_i = 0$ otherwise.

Vova thinks that people in the i-th flats are disturbed and cannot sleep if and only if $1 < i < n$ and $a_{i-1} = a_{i+1} = 1$ and $a_i = 0$.

Vova is concerned by the following question: what is the minimum number k such that if people from exactly k pairwise distinct flats will turn off the lights then nobody will be disturbed? Your task is to find this number k.

Input

The first line of the input contains one integer n ($3 \le n \le 100$) — the number of flats in the house.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($a_i \in \{0, 1\}$), where a_i is the state of light in the i-th flat.

Output

Print only one integer — the minimum number k such that if people from exactly k pairwise distinct flats will turn off the light then nobody will be disturbed.

Examples

input
```
10
1 1 0 1 1 0 1 0 1 0
```
output
```
2
```

input
```
5
1 1 0 0 0
```
output
```
0
```

input
```
4
1 1 1 1
```
output
```
0
```

Note

In the first example people from flats 2 and 7 or 4 and 7 can turn off the light and nobody will be disturbed. It can be shown that there is no better answer in this example.

There are no disturbed people in second and third examples.

C. Good Array

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Let's call an array *good* if there is an element in the array that equals to the sum of all other elements. For example, the array $a = [1, 3, 3, 7]$ is good because there is the element $a_4 = 7$ which equals to the sum $1 + 3 + 3$.

You are given an array a consisting of n integers. Your task is to print all indices j of this array such that after removing the j-th element from the array it will be *good* (let's call such indices *nice*).

For example, if $a = [8, 3, 5, 2]$, the *nice* indices are 1 and 4:

- if you remove a_1, the array will look like $[3, 5, 2]$ and it is *good*;
- if you remove a_4, the array will look like $[8, 3, 5]$ and it is *good*.

You have to consider all removals **independently**, i. e. remove the element, check if the resulting array is *good*, and return the element into the array.

Input

The first line of the input contains one integer n ($2 \le n \le 2 \cdot 10^5$) — the number of elements in the array a.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^6$) — elements of the array a.

Output

In the first line print one integer k — the number of indices j of the array a such that after removing the j-th element from the array it will be *good* (i.e. print the number of the *nice* indices).

In the second line print k distinct integers $j_1, j_2, \dots, j_k$ in **any** order — *nice* indices of the array a.

If there are no such indices in the array a, just print 0 in the first line and leave the second line empty or do not print it at all.

Examples

input

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

output

```
3
4 1 5
```

input

```
4
8 3 5 2
```

output

```
2
1 4
```

input

```
5
2 1 2 4 3
```

output

```
0
```

Note

In the first example you can remove any element with the value 2 so the array will look like $[5, 1, 2, 2]$. The sum of this array is 10 and there is an element equals to the sum of remaining elements ($5 = 1 + 2 + 2$).

In the second example you can remove 8 so the array will look like $[3, 5, 2]$. The sum of this array is 10 and there is an element equals to the sum of remaining elements ($5 = 3 + 2$). You can also remove 2 so the array will look like $[8, 3, 5]$. The sum of this array is 16 and there is an element equals to the sum of remaining elements ($8 = 3 + 5$).

In the third example you cannot make the given array *good* by removing exactly one element.

D. Cutting Out

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an array s consisting of n integers.

You have to find **any** array t of length k such that you can cut out maximum number of copies of array t from array s.

Cutting out the copy of t means that for each element t_i of array t you have to find t_i in s and remove it from s. If for some t_i you cannot find such element in s, then you cannot cut out one more copy of t. The both arrays can contain duplicate elements.

For example, if $s = [1, 2, 3, 2, 4, 3, 1]$ and $k = 3$ then one of the possible answers is $t = [1, 2, 3]$. This array t can be cut out 2 times.

- To cut out the first copy of t you can use the elements $[1, \underline{\mathbf{2}}, 3, 2, 4, \underline{\mathbf{3}}, \underline{\mathbf{1}}]$ (use the highlighted elements). After cutting out the first copy of t the array s can look like $[1, 3, 2, 4]$.
- To cut out the second copy of t you can use the elements $[\underline{\mathbf{1}}, \underline{\mathbf{3}}, \underline{\mathbf{2}}, 4]$. After cutting out the second copy of t the array s will be $[4]$.

Your task is to find such array t that you can cut out the copy of t from s maximum number of times. If there are multiple answers, you may choose **any** of them.

Input

The first line of the input contains two integers n and k ($1 \le k \le n \le 2 \cdot 10^5$) — the number of elements in s and the desired number of elements in t, respectively.

The second line of the input contains exactly n integers $s_1, s_2, \dots, s_n$ ($1 \le s_i \le 2 \cdot 10^5$).

Output

Print k integers — the elements of array t such that you can cut out maximum possible number of copies of this array from s. If there are multiple answers, print **any** of them. The required array t can contain duplicate elements. All the elements of t ($t_1, t_2, \dots, t_k$) should satisfy the following condition: $1 \le t_i \le 2 \cdot 10^5$.

Examples

input

```
7 3
1 2 3 2 4 3 1
```

output

```
1 2 3
```

input

```
10 4
1 3 1 3 10 3 7 7 12 3
```

output

```
7 3 1 3
```

input

```
15 2
1 2 1 1 1 2 1 1 2 1 2 1 1 1 1
```

output

```
1 1
```

Note

The first example is described in the problem statement.

In the second example the only answer is $[7, 3, 1, 3]$ and any its permutations. It can be shown that you cannot choose any other array such that the maximum number of copies you can cut out would be equal to 2.

In the third example the array t can be cut out 5 times.

E. Thematic Contests

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp has prepared n competitive programming problems. The topic of the i-th problem is a_i, and some problems' topics may coincide.

Polycarp has to host several thematic contests. All problems in each contest should have the same topic, and **all contests should have pairwise distinct topics**. He may not use all the problems. It is possible that there are no contests for some topics.

Polycarp wants to host competitions on consecutive days, one contest per day. Polycarp wants to host a set of contests in such a way that:

- number of problems in each contest is **exactly twice** as much as in the previous contest (one day ago), the first contest can contain arbitrary number of problems;
- the total number of problems in all the contests should be maximized.

Your task is to calculate the maximum number of problems in the set of thematic contests. Note, that you should not maximize the number of contests.

Input

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

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$) where a_i is the topic of the i-th problem.

Output

Print one integer — the maximum number of problems in the set of thematic contests.

Examples

input

```
18
2 1 2 10 2 10 10 2 2 1 10 10 10 10 1 1 10 10
```

output

```
14
```

input

```
10
6 6 6 3 6 1000000000 3 3 6 6
```

output

```
9
```

input

```
3
1337 1337 1337
```

output

```
3
```

Note

In the first example the optimal sequence of contests is: 2 problems of the topic 1, 4 problems of the topic 2, 8 problems of the topic 10.

In the second example the optimal sequence of contests is: 3 problems of the topic 3, 6 problems of the topic 6.

In the third example you can take all the problems with the topic 1337 (the number of such problems is 3 so the answer is 3) and host a single contest.

F1. Pictures with Kittens (easy version)

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The only difference between easy and hard versions is the constraints.

Vova likes pictures with kittens. The news feed in the social network he uses can be represented as an array of n consecutive pictures (with kittens, of course). Vova likes all these pictures, but some are more beautiful than the others: the i-th picture has beauty a_i.

Vova wants to repost exactly x pictures in such a way that:

- each segment of the news feed of at least k consecutive pictures has at least one picture reposted by Vova;
- the sum of beauty values of reposted pictures is maximum possible.

For example, if $k = 1$ then Vova has to repost all the pictures in the news feed. If $k = 2$ then Vova can skip some pictures, but between every pair of consecutive pictures Vova has to repost at least one of them.

Your task is to calculate the maximum possible sum of values of reposted pictures if Vova follows conditions described above, or say that there is no way to satisfy all conditions.

Input

The first line of the input contains three integers n, k and x ($1 \le k, x \le n \le 200$) — the number of pictures in the news feed, the minimum length of segment with at least one repost in it and the number of pictures Vova is ready to repost.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$), where a_i is the beauty of the i-th picture.

Output

Print -1 if there is no way to repost some pictures to satisfy all the conditions in the problem statement.

Otherwise print one integer — the maximum sum of values of reposted pictures if Vova follows conditions described in the problem statement.

Examples

input

```
5 2 3
5 1 3 10 1
```

output

```
18
```

input

```
6 1 5
10 30 30 70 10 10
```

output

```
-1
```

input

```
4 3 1
1 100 1 1
```

output

```
100
```

F2. Pictures with Kittens (hard version)

time limit per test: 2.5 seconds
memory limit per test: 512 megabytes
input: standard input

output: standard output

The only difference between easy and hard versions is the constraints.

Vova likes pictures with kittens. The news feed in the social network he uses can be represented as an array of n consecutive pictures (with kittens, of course). Vova likes all these pictures, but some are more beautiful than the others: the i-th picture has beauty a_i.

Vova wants to repost exactly x pictures in such a way that:

- each segment of the news feed of at least k consecutive pictures has at least one picture reposted by Vova;
- the sum of beauty values of reposted pictures is maximum possible.

For example, if $k = 1$ then Vova has to repost all the pictures in the news feed. If $k = 2$ then Vova can skip some pictures, but between every pair of consecutive pictures Vova has to repost at least one of them.

Your task is to calculate the maximum possible sum of values of reposted pictures if Vova follows conditions described above, or say that there is no way to satisfy all conditions.

Input

The first line of the input contains three integers n, k and x ($1 \le k, x \le n \le 5000$) — the number of pictures in the news feed, the minimum length of segment with at least one repost in it and the number of pictures Vova is ready to repost.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$), where a_i is the beauty of the i-th picture.

Output

Print -1 if there is no way to repost some pictures to satisfy all the conditions in the problem statement.

Otherwise print one integer — the maximum sum of values of reposted pictures if Vova follows conditions described in the problem statement.

Examples

input

```
5 2 3
5 1 3 10 1
```

output

```
18
```

input

```
6 1 5
10 30 30 70 10 10
```

output

```
-1
```

input

```
4 3 1
1 100 1 1
```

output

```
100
```