Educational Codeforces Round 3

A. USB Flash Drives

2 seconds, 256 megabytes

Sean is trying to save a large file to a USB flash drive. He has n USB flash drives with capacities equal to $a_1, a_2, ..., a_n$ megabytes. The file size is equal to m megabytes.

Find the minimum number of USB flash drives needed to write Sean's file, if he can split the file between drives.

Input

The first line contains positive integer n $(1 \le n \le 100)$ — the number of USB flash drives.

The second line contains positive integer m $(1 \le m \le 10^5)$ — the size of Sean's file.

Each of the next n lines contains positive integer a_i $(1 \le a_i \le 1000)$ — the sizes of USB flash drives in megabytes.

It is guaranteed that the answer exists, i. e. the sum of all a_i is not less than m.

Output

Print the minimum number of USB flash drives to write Sean's file, if he can split the file between drives.

input
```
3
5
2
1
3
```
output
```
2
```

input
```
3
6
2
3
2
```
output
```
3
```

input
```
2
5
5
10
```
output
```
1
```

In the first example Sean needs only two USB flash drives — the first and the third.

In the second example Sean needs all three USB flash drives.

In the third example Sean needs only one USB flash drive and he can use any available USB flash drive — the first or the second.

B. The Best Gift

2 seconds, 256 megabytes

Emily's birthday is next week and Jack has decided to buy a present for her. He knows she loves books so he goes to the local bookshop, where there are n books on sale from one of m genres.

In the bookshop, Jack decides to buy *two books of different genres*.

Based on the genre of books on sale in the shop, find the number of options available to Jack for choosing two books of different genres for Emily. Options are considered different if they differ in at least one book.

The books are given by indices of their genres. The genres are numbered from 1 to m.

Input

The first line contains two positive integers n and m $(2 \le n \le 2 \cdot 10^5, 2 \le m \le 10)$ — the number of books in the bookstore and the number of genres.

The second line contains a sequence $a_1, a_2, ..., a_n$, where a_i $(1 \le a_i \le m)$ equals the genre of the i-th book.

It is guaranteed that for each genre there is at least one book of that genre.

Output

Print the only integer — the number of ways in which Jack can choose books.

It is guaranteed that the answer doesn't exceed the value $2 \cdot 10^9$.

input
```
4 3
2 1 3 1
```
output
```
5
```

input
```
7 4
4 2 3 1 2 4 3
```
output
```
18
```

The answer to the first test sample equals 5 as Sasha can choose:

1. the first and second books,
2. the first and third books,
3. the first and fourth books,
4. the second and third books,
5. the third and fourth books.

C. Load Balancing

2 seconds, 256 megabytes

In the school computer room there are n servers which are responsible for processing several computing tasks. You know the number of scheduled tasks for each server: there are m_i tasks assigned to the i-th server.

In order to balance the load for each server, you want to reassign some tasks to make the difference between the most loaded server and the least loaded server as small as possible. In other words you want to minimize expression m_a - m_b, where a is the most loaded server and b is the least loaded one.

In one second you can reassign a single task. Thus in one second you can choose any pair of servers and move a single task from one server to another.

Write a program to find the minimum number of seconds needed to balance the load of servers.

Input

The first line contains positive number n $(1 \le n \le 10^5)$ — the number of the servers.

The second line contains the sequence of non-negative integers $m_1, m_2, ..., m_n$ $(0 \le m_i \le 2 \cdot 10^4)$, where m_i is the number of tasks assigned to the i-th server.

Output

Print the minimum number of seconds required to balance the load.

input
```
2
1 6
```
output
```
2
```

input
```
7
10 11 10 11 10 11 11
```
output
```
0
```

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

output

```
3
```

In the first example two seconds are needed. In each second, a single task from server #2 should be moved to server #1. After two seconds there should be 3 tasks on server #1 and 4 tasks on server #2.

In the second example the load is already balanced.

A possible sequence of task movements for the third example is:

1. move a task from server #4 to server #1 (the sequence m becomes: 2 2 3 3 5);
2. then move task from server #5 to server #1 (the sequence m becomes: 3 2 3 3 4);
3. then move task from server #5 to server #2 (the sequence m becomes: 3 3 3 3 3).

The above sequence is one of several possible ways to balance the load of servers in three seconds.

D. Gadgets for dollars and pounds

2 seconds, 256 megabytes

Nura wants to buy k gadgets. She has only s burles for that. She can buy each gadget for dollars or for pounds. So each gadget is selling only for some type of currency. The type of currency and the cost in that currency are not changing.

Nura can buy gadgets for n days. For each day you know the exchange rates of dollar and pound, so you know the cost of conversion burles to dollars or to pounds.

Each day (from 1 to n) Nura can buy some gadgets by current exchange rate. Each day she can buy any gadgets she wants, but each gadget can be bought no more than once during n days.

Help Nura to find the minimum day index when she will have k gadgets. Nura always pays with burles, which are converted according to the exchange rate of the purchase day. Nura can't buy dollars or pounds, she always stores only burles. Gadgets are numbered with integers from 1 to m in order of their appearing in input.

Input

First line contains four integers n, m, k, s $(1 \le n \le 2 \cdot 10^5, 1 \le k \le m \le 2 \cdot 10^5, 1 \le s \le 10^9)$ — number of days, total number and required number of gadgets, number of burles Nura has.

Second line contains n integers a_i $(1 \le a_i \le 10^6)$ — the cost of one dollar in burles on i-th day.

Third line contains n integers b_i $(1 \le b_i \le 10^6)$ — the cost of one pound in burles on i-th day.

Each of the next m lines contains two integers t_i, c_i $(1 \le t_i \le 2, 1 \le c_i \le 10^6)$ — type of the gadget and it's cost. For the gadgets of the first type cost is specified in dollars. For the gadgets of the second type cost is specified in pounds.

Output

If Nura can't buy k gadgets print the only line with the number `-1`.

Otherwise the first line should contain integer d — the minimum day index, when Nura will have k gadgets. On each of the next k lines print two integers q_i, d_i — the number of gadget and the day gadget should be bought. All values q_i should be different, but the values d_i can coincide (so Nura can buy several gadgets at one day). The days are numbered from 1 to n.

In case there are multiple possible solutions, print any of them.

input

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

output

```
3
1 1
2 3
```

input

```
4 3 2 200
69 70 71 72
104 105 106 107
1 1
2 2
1 2
```

output

```
-1
```

input

```
4 3 1 1000000000
900000 910000 940000 990000
990000 999000 999900 999990
1 87654
2 76543
1 65432
```

output

```
-1
```

E. Minimum spanning tree for each edge

2 seconds, 256 megabytes

Connected undirected weighted graph without self-loops and multiple edges is given. Graph contains n vertices and m edges.

For each edge (u, v) find the minimal possible weight of the spanning tree that contains the edge (u, v).

The weight of the spanning tree is the sum of weights of all edges included in spanning tree.

Input

First line contains two integers n and m $(1 \le n \le 2 \cdot 10^5, n - 1 \le m \le 2 \cdot 10^5)$ — the number of vertices and edges in graph.

Each of the next m lines contains three integers u_i, v_i, w_i $(1 \le u_i, v_i \le n, u_i \ne v_i, 1 \le w_i \le 10^9)$ — the endpoints of the i-th edge and its weight.

Output

Print m lines. i-th line should contain the minimal possible weight of the spanning tree that contains i-th edge.

The edges are numbered from 1 to m in order of their appearing in input.

input

```
5 7
1 2 3
1 3 1
1 4 5
2 3 2
2 5 3
3 4 2
4 5 4
```

output

```
9
8
11
8
8
8
9
```

F. Frogs and mosquitoes

2 seconds, 512 megabytes

There are n frogs sitting on the coordinate axis Ox. For each frog two values x_i, t_i are known — the position and the initial length of the tongue of the i-th frog (it is guaranteed that all positions x_i are different). m mosquitoes one by one are landing to the coordinate axis. For each mosquito two values are known p_j — the coordinate of the position where the j-th mosquito lands and b_j — the size of the j-th mosquito. Frogs and mosquitoes are represented as points on the coordinate axis.

The frog can eat mosquito if mosquito is in the same position with the frog or to the right, and the distance between them is not greater than the length of the tongue of the frog.

If at some moment several frogs can eat a mosquito the leftmost frog will eat it (with minimal x_i). After eating a mosquito the length of the tongue of a frog increases with the value of the size of eaten mosquito. It's possible that after it the frog will be able to eat some other mosquitoes (the frog should eat them in this case).

For each frog print two values — the number of eaten mosquitoes and the length of the tongue after landing all mosquitoes and after eating all possible mosquitoes by frogs.

Each mosquito is landing to the coordinate axis only after frogs eat all possible mosquitoes landed before. Mosquitoes are given in order of their landing to the coordinate axis.

Input

First line contains two integers n, m $(1 \le n, m \le 2 \cdot 10^5)$ — the number of frogs and mosquitoes.

Each of the next n lines contains two integers x_i, t_i $(0 \le x_i, t_i \le 10^9)$ — the position and the initial length of the tongue of the i-th frog. It is guaranteed that all x_i are different.

Next m lines contain two integers each p_j, b_j $(0 \le p_j, b_j \le 10^9)$ — the position and the size of the j-th mosquito.

Output

Print n lines. The i-th line should contain two integer values c_i, l_i — the number of mosquitoes eaten by the i-th frog and the length of the tongue of the i-th frog.

input

```
4 6
10 2
15 0
6 1
0 1
110 10
1 1
6 0
15 10
14 100
12 2
```

output

```
3 114
1 10
1 1
1 2
```

input

```
1 2
10 2
20 2
12 1
```

output

```
1 3
```