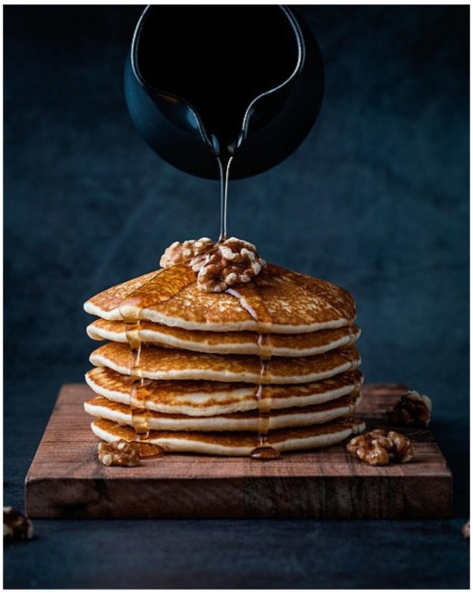
Yet Another Sorting Problem (YASP)

"It is lovely day for pancakes."

As she said that, Kou invented yet another sorting problem (yasp). In this problem, we are given a *stack* of integers and we are asked to sort that stack in ascending order from top to bottom. The only allowed operation, denoted by *flip(t)*, is to flip all elements from the *t*-th elements to the top upside down. Note that the 1st element is the bottom element while the N-th element refers to the top element if the size of the stack is N. For example, consider the following four stacks of integers.



```
    4       2       5       1       (top)
    3       3       4       2
    2       4       3       3
    5       5       2       4
    1       1       1       5       (bottom)
   (a)     (b)     (c)     (d)
```

Performing *flip(3)* on stack (a) will result in stack (b). Similarly, by applying *flip(2)*, stack (b) will become stack (c). Finally, we can apply *flip(1)* on stack (c) and get the stack sorted as shown in stack (d).

We need to figure out a sequence of *flip* operations that transforms the given stack into a sorted one.

Input

The input consists of a single line. The stack of integers will be given in this line from top to bottom. The size of the stack is between 1 and 30, and those integers are between 1 and 100.

Output

Print one line containing (K+1) flip operations:

- Among the first K integers, the i-th integer specifies the *t* value of the i-th flip operation.
- The last, i.e., the (K+1)-th integer should always be `0`.

If there are multiple solutions, output any of them.

Example 1

```
Input:
1 2 3 4 5

Output:
0
```

Example 2

```
Input:
5 4 3 2 1

Output:
1 0
```

Example 3

```
Input:
5 1 2 3 4

Output:
1 2 0
```

Toy Blocks

Ayu is playing with toy blocks. Ayu decides to build two towers with those blocks. She wants to use up all of the blocks she has and the number of blocks used in two towers should not differ by more than one. Besides, every block has a height and she wants to minimize the height difference between two towers.

Input

The first line of the input contains one integer N (1 <= N <= 100), the number of toy blocks. Each of the following N lines contains one integer indicating the height h (1 <= h <= 450) of that block.

Output

Print one line, containing two space separated integers, the heights of two towers. The smaller number goes first.

Example 1

```
Input:
3
100
90
200

Output:
190 200
```

Grievous Labyrinth

This is a survival game. The game takes place in a labyrinth, which has N rooms connected by some one-way doorways. You are transported to room 1 at the beginning of the game and you want to reach room N (the exit) through those doorways. You have 100 HPs (health points) initially and you can either earn or lose a given amount of HPs whenever you reach a room.

Given the description of the game, determine whether it is possible to reach the exit (i.e., room N) or not before the number of HPs falls down to zero (or below).

Note that you can enter the same room multiple times.

Input

The first line of the input contains one integer N (1 <= N <= 100), indicating the number of rooms. Then N line follows, describing the N rooms, each of which has the following format `V K ...` where:

- V is the number of HPs to be earned or lost by entering this room (the loss is indicated by a negative value of V); |V| <= 100 and V = 0 for room 1 and room N
- K is the number of doorways leaving this room;
- The remaining K integers represent the list of rooms reachable by the doorways leaving this room; 0 <= K <= N.

Output

Print one line containing `winnable` or `hopeless` to indicate if this game is winnable or not.

Example 1

```
Input:
5
0 1 2
-60 1 3
-60 1 4
20 1 5
0 0

Output:
hopeless
```

Example 2

```
Input:
5
0 1 2
20 1 3
-60 1 4
-60 1 5
0 0

Output:
hopeless
```

Example 3

```
Input:
5
0 1 2
21 1 3
-60 1 4
-60 1 5
```

```
0 0

Output:
winnable
```

Example 4

```
Input:
5
0 1 2
20 2 1 3
-60 1 4
-60 1 5
0 0

Output:
winnable
```

Batch Processing

Since Light Kingdom did not pay you salary for a whole year, you decided to leave and work for Conflict Empire. Now, you are again an operator of a super computing center and in control of M nodes.

One day, a research institute from Conflict Empire submitted N computational tasks numbered from 1 to N. Given the running time needed for each task, you are to distribute the tasks among the available nodes such that the node with heaviest workload completes as early as possible. Restriction: every node must process at least one task and must process a contiguous subsection of tasks. That is, you need to find a sequence $0 = L_0 < L_1 < \ldots < L_{M-1} < L_M = N$ where the i-th node processes tasks $L_{i-1}+1, L_{i-1}+2, \ldots, L_i$.

Input

The first line of the input contains two integers N and M (1 <= M <= N <= 500), representing the number of tasks and the number of nodes you control, respectively. The second line contains N integers T_i ($1 <= T_i < 10,000,000$), representing the time needed to complete each task.

Output

Print one line containing the input $T_1, T_2, \ldots, T_N$ divided into M parts such that the maximum sum of a single part is minimized. You should use character '/' to separate the parts and there must be a space character between every numbers or '/'s.

If there is more than one solution, print the one that minimizes the sum of the first part, then the second part and so on.

Example 1

```
Input:
9 3
100 200 300 400 500 600 700 800 900

Output:
100 200 300 400 500 / 600 700 / 800 900
```

Example 2

```
Input:
5 4
100 100 100 100 100

Output:
100 / 100 / 100 / 100 100
```

Shortest Path

Cirno has learned how to solve single-source shortest path problem. She decides to give you a challenge to check if you are as talented as she is.

You are working with a directed graph G with n nodes with weights assigned to edges. The node E is the destination node.

Given an integer B, find the number of nodes that have a path to the destination node and the distance is less than or equal to B.

The distance of a path is the sum of the weights of its edges.

Input

The first line contains 4 integer n, E, B, m. n is the number of nodes, E is the node number for the destination node, B is the limit, and m is the number of edges. 1 <= n <= 100, 1 <= E <= n, 1 <= B <= 10^9, 0 <= m <= n*n

In each of the following m lines, there are 3 integers u,v,w, which means there is an edge from u to v with weight w. 1<=u,v<=n, 0 < w < 10^9

There may be parallel edges or self loops in the input graph.

Output A integer, the number of nodes that have desired paths to E.

Example 1

```
Input:
100 1 123 0

Output:
1
```

There is no edge in the graph, the only node that has a path to E is E itself.

Example 2

```
Input:
5 1 1 4
2 1 1
3 1 1
4 5 1
5 3 1

Output:
3
```