Bit-wise OR

Given 3 non-negative integers N, L, R. Find the **minimum** integer x such that

1. L <= x <= R
2. x **OR** N is maximized (**OR** stands for bit-wise OR operation here), i.e. for all integers y in the close interval [L,R], we have: (y **OR** N) <= (x **OR** N)

Input

There will be 3 integers N, L, R on a single line, where L <= R. You can assume that 0 <= N,L,R <= 4294967295.

Output

Print a line containing the value of x such that x **OR** N is maximized.

Example 1

```
Input:
10 10 10

Output:
10
```

Example 2

```
Input:
100 50 60

Output:
59
```

Desolate Number

Given two integers A and B, a desolate number N is defined as follows:

- N is an (A+B)-bit integer;
- the binary representation of N has exactly A 1's and B 0's (leading zeroes are ok);
- N has the maximum number of 1's adjacent to at least one 0 in its binary representation.

Your task is to find the smallest desolate number N given the values of A and B.

Input

The input consists of a single line, containing two non-negative integers A and B ($1 <= A + B <= 50$).

Output

Print one line, containing the smallest desolate number.

Example 1

```
Input:
1 1

Output:
1
```

Since we have two bits and one of them has to be zero and the other one has to be one, then the two options are `01` and `10`. They both have the same number of 1-bits adjacent to the 0-bits. But the first one is smaller, so the answer is 1.

Example 2

```
Input:
4 3

Output:
45
```

45 is binary using 7 bits is `0101101`. In this sequence every 1-bit is adjacent to at least one 0 bit sot this constraint is maximized. The only way to make this value smaller is to shift some of the 1-bits to the right, but this would decrease the number of 1-bits adjacent to the 0-bits.

Example 3

```
Input:
4 1

Output:
23
```

23 in binary using 5 bits is `10111`. With a single zero available at most two 1-bits can be adjacent to a 0-bit. If we move the 0-bit any further to the right, the value would increase.

HINT: You do not need to try every possible combination of the bits to solve this problem.

Finding Routes

Ethan Hunt wants your help to get through a high security building for a top secret mission. The building has some cameras installed at several places which are constantly monitored by the security.

Ethan has a map of the building. Each corner is identified by a positive integer less than 21. Initially Ethan is at position 1.

Write a program to find the possible routes that Ethan can take to reach the nearest corner to the target location of the building.

Input

First line consists of an integer n which is the number of corners closest to the target. n<21

Next several lines consists of pairs of positive integers separated by a space which are the adjacent corners of the building which does not have any cameras in the way (for instance, if pair 4 7 is on the line then the path between corner 4 and 7 has no cameras on the path. Also there are no other corners on that path).

The last line consists of a pair of 0's

Output

The output must list each possible route on a separate line with each corner written in the order in which they appear on the path. Also, it should include a line stating the number of paths possible for reaching the desired location in the form "There are n routes from the initial position to corner x.", where n is the number of routes and x is the desired target corner. Only include the routes which do not pass through any corner more than once (this is to avoid moving unnecessarily around in circles as time is crucial).

Example 1

```
Input:
6
1 2
1 3
3 4
3 5
4 6
5 6
2 3
2 4
0 0

Output:
1 2 3 4 6
1 2 3 5 6
1 2 4 3 5 6
1 2 4 6
1 3 2 4 6
1 3 4 6
1 3 5 6
There are 7 routes from the initial position to corner 6.
```

Example 2

```
Input:
4
2 3
3 4
5 1
1 6
7 8
8 9
```

```
2 5
5 7
3 1
1 8
4 6
6 9
0 0

Output:
1 3 2 5 7 8 9 6 4
1 3 4
1 5 2 3 4
1 5 7 8 9 6 4
1 6 4
1 6 9 8 7 5 2 3 4
1 8 7 5 2 3 4
1 8 9 6 4
There are 8 routes from the initial position to corner 4.
```

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 = L0 < L1 < ... < LM-1 < LM = N where the i-th node processes tasks Li-1+1, Li-1+2, ..., Li.

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 Ti (1 <= Ti < 10,000,000), representing the time needed to complete each task.

Output

Print one line containing the input T1, T2, ..., TN 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
```

Find Sums

You are given a multiset of N integers (multiset means that the repeated values are allowed) and a target value S. Your task is to find all distinct subsets of the given multiset for which the sum of all the elements in the subset is equal to S.

Input

The first line of the input contains two integers S (1 <= S <= 1,000) and N (1 <= N <= 12), indicating the target sum and the number of values in the multiset, respectively.

The second line contains N integers, all of which are between 1 and 100 - these are the elements of the multiset.

Output

First, print a line `Sums of S:` where S is the value given in the input. Then print one line for every subset satisfying the condition or a line containing `NONE` if there is no such subset.

For every subset, numbers are printed in decreasing order and separated by a plus sign (`+`). The subsets themselves are sorted lexicographically in decreasing order, i.e. they are sorted by their first integer, then the second integer in case of tie, and so on. Additionally, the subsets you print should not contain repetitions (i.e., you should never print two lines that are identical).

Example 1

```
Input:
4 6
4 3 2 2 1 1

Output:
Sums of 4:
4
3+1
2+2
2+1+1
```

Example 2

```
Input:
5 3
2 1 1

Output:
Sums of 5:
NONE
```

Example 3

```
Input:
400 10
150 100 60 60 50 50 50 30 20 20

Output:
Sums of 400:
150+100+60+60+30
150+100+60+50+20+20
150+100+50+50+50
150+100+50+50+30+20
150+60+60+50+50+30
150+60+50+50+50+20+20
100+60+60+50+50+50+30
```