1.加法

Score： 5

输入两个数a、b，输出两数之和。

Standard Input

一行两个数a，b

Standard Output

一行一个数，即a+b（小数部分保留最后一位有效数字（比如：1.20请输出为1.2））

Samples

Input	Output
`1.1 1`	`2.1`

数据范围：对于60%的数据：-1e50<a,b<1e50，且a，b皆为整数
对于100%的数据：-1e50<a,b<1e50，且a，b的小数位不超过30位

tag:大数加法（萌新可以在洛谷刷刷高精度的题）

Question2.压缩字符串

给你一个字符串s，输出压缩后的该字符串以及该字符串压缩后的长度。压缩规则如下：
从s的第一个字符开始。对于 s 中的每组连续重复字符：
如果这一组长度为 1 ，则不变。
否则，该组改为x（该重复字符）+长度的模式（比如讲“aaa”改为“a3”，将“bbbbbbbbbbb”改为“b11”）。

Standard Input

共一行，表示字符串s（s只由小写字母构成且长度位于1与1e5之间）

Standard Output

共两行：
第一行为压缩后的字符串，第二行为压缩后的字符串的长度

Samples

Input	Output
`aabbbbbbbbbbc`	`a2b10c` 6

Tag：不涉及啥算法

Question3.投硬币

瑄有一个爱好是收集硬币，他已经收集了 2020 年的硬币，谁也不知道他现在到底有多少枚硬币。他最喜欢的游戏就是「抛硬币」，每次投掷硬币都会让他心情愉快。这个游戏想必大家也都玩过，将一枚硬币抛到地面上，结果要么正面朝上要么反面朝上。

瑄长大了，不再满足于仅仅投掷一枚硬币。今天他从他的无尽的硬币库存中取出了 **N**个硬币，并在无尽的地面上依次投掷取出的每一枚硬币，投掷好的硬币瑄会把他们按照投掷顺序依次排成一排，他认为如果存在连续排列的三个硬币都是正面或者都是反面，那么这个排列是丑陋的，反之这个排列就是美观的。他想知道这**N** 个硬币来排列，有多少个丑陋的排列。虽然瑄数学很优秀，但他只想单纯的玩游戏，并不想将宝贵的时间花在思考这个问题上面，作为瑄好朋友的你，可以帮助他找出丑陋的排列的个数吗？由于答案可能很大，你只需要输出排列数对1000000007取模后的答案

Standard Input

一个正整数N(1≤*N*≤10000)，表示硬币的个数。

Standard Output

输出一个整数，即丑陋排列的个数对1000000007取模后的答案。

Samples

Input	Output
4	6

tag:动态规划（萌新们可能还得学学取模运算的一些基本性质）

Question4.好朋友

幼儿园里有n个小朋友，他们的关系很简单，只有朋友和不熟两种。而秉承着我朋友的朋友就是我的朋友的观念，当任意两个小朋友A、B结成好朋友时，A与A的好朋友们和B与B的好朋友们也会成为好朋友。最开始小朋友们都是互相不认识的，而接下来的每一天都会有一对小朋友一起玩耍，如果两个小朋友不熟他们会结成好朋友，如果已经是好朋友了则会无事发生。小朋友SYQ很关心大家的交友状况，她很好奇经过多少天幼儿园里的所有小朋友都会成为好朋友，你能告诉她吗。

若a、b是好朋友，b、c是好朋友，那么a、c也是好朋友。

Standard Input

第一行包含两个整数N，M（2<=N<=1e5, 1<=M<=2e5）分别表示小朋友的数量和有多少天。

接下来M行，每行包括两个数ai，bi（1<=a,b<=N且a不等于b）表示第i天小朋友a会和小朋友b一起玩。

Standard Output

输出一个整数t，表示在第t天小朋友们一起玩耍后全幼儿园的小朋友都会成为好朋友。若在M天后全部小朋友仍未成为好朋友，则输出-1。

Samples

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

Note

第一天1，2成为了好朋友，第二天3，4成为了好朋友，第三天1，2，3，4成为了好朋友。

tag:并查集

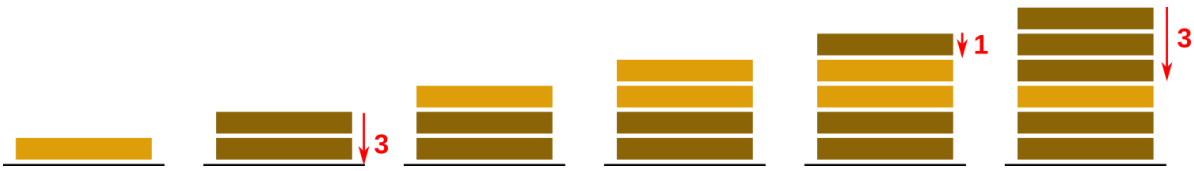
Question5. Napoleon Cake

This week Arkady wanted to cook some pancakes (to follow ancient traditions) and make a problem about that. But then he remembered that one can't make a problem about stacking pancakes without working at a specific IT company, so he decided to bake the Napoleon cake instead.

To bake a Napoleon cake, one has to bake **n** dry layers first, and then put them on each other in one stack, adding some cream. Arkady started with an empty plate, and performed the following steps **n** times:

- place a new cake layer on the top of the stack;
- after the **i**-th layer is placed, pour **ai** units of cream on top of the stack.

When xx units of cream are poured on the top of the stack, top xx layers of the cake get drenched in the cream. If there are less than **x** layers, all layers get drenched and the rest of the cream is wasted. If x=0x=0, no layer gets drenched.



The picture represents the first test case of the example.

Help Arkady determine which layers of the cake eventually get drenched when the process is over, and which don't.

Input

Each test contains multiple test cases. The first line contains the number of test cases **t**(1≤t≤20000). Description of the test cases follows.

The first line of each test case contains a single integer **n** (1≤n≤2·1e5) — the number of layers in the cake.

The second line of each test case contains **n** integers a1,a2,...,an (0≤ai≤n) — the amount of cream poured on the cake after adding each layer.

It is guaranteed that the sum of **n** over all test cases does not exceed 2·1e5.

Output

For each test case, print a single line with **n** integers. The **i**-th of the integers should be equal to **1** if the **i**-th layer from the bottom gets drenched, and **0** otherwise.

Example

input

```
3
6
0 3 0 0 1 3
10
0 0 0 1 0 5 0 0 0 2
3
0 0 0
```

output

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

tag:差分 前缀和（大家加油读题呀）