[课后作业 1](#_Toc94544747)

[**1.** ABC子串查询 1](#_Toc94544748)

[**2.** 最近字符查询 1](#_Toc94544749)

[**3.** (USACO 2021 January – Bronze 2) Even More Odd Photos 2](#_Toc94544750)

[**4.** (USACO 2020 December – Bronze 3) Stuck in a Rut 3](#_Toc94544751)

[**5.** (CSP J2 2021 – T2) 直播获奖 5](#_Toc94544752)

[**6.** (NOIP 2002 Junior – T4) 过河卒 7](#_Toc94544753)

课后作业

1. ABC子串查询

给定一个由字母'A'、'B'、'C'组成的字符串（）。给定（）个询问，每个询问包含两个整数（）。求从第个字符到第个字符所构成的子串中包含的子串"ABC"的数量。

输入样例：

|  |
| --- |
| ABCBABCA  4  1 2  1 3  2 5  1 8 |

输出样例：

|  |
| --- |
| 0  1  0  2 |

1. 最近字符查询

给定一个由大写字母组成的字符串（）。给定（）个询问，每个询问包含一个整数和一个字符（为大写字母）。求距离最近的字母与的距离，若不存在则输出。

输入样例：

|  |
| --- |
| ABACDACB  5  1 D  2 A  4 E  5 C  6 B |

输出样例：

|  |
| --- |
| 4  1  -1  1  2 |

1. (USACO 2021 January – Bronze 2) Even More Odd Photos

Farmer John is yet again trying to take a photograph of his cows ().

Each cow has an integer "breed ID" number in the range . Farmer John has a very peculiar idea in mind for his photo: he wants to partition all the cows into disjoint groups (in other words, place each cow in exactly one group) and then line up the groups so the sum of the breed IDs of the cows in the first group is even, the sum of the IDs in the second group is odd, and so on, alternating between even and odd.

What is the maximum possible number of groups Farmer John can form?

**INPUT FORMAT (input arrives from the terminal / stdin):**

The first line of input contains . The next line contains space-separated integers giving the breed IDs of the cows.

**OUTPUT FORMAT (print output to the terminal / stdout):**

The maximum possible number of groups in Farmer John's photo. It can be shown that at least one feasible grouping exists.

**SAMPLE INPUT:**

|  |
| --- |
| 7  1 3 5 7 9 11 13 |

**SAMPLE OUTPUT:**

|  |
| --- |
| 3 |

In this example, one way to form the maximum number of three groups is as follows. Place 1 and 3 in the first group, 5, 7, and 9 in the second group, and 11 and 13 in the third group.

**SAMPLE INPUT:**

|  |
| --- |
| 7  11 2 17 13 1 15 3 |

**SAMPLE OUTPUT:**

|  |
| --- |
| 5 |

In this example, one way to form the maximum number of five groups is as follows. Place 2 in the first group, 11 in the second group, 13 and 1 in the third group, 15 in the fourth group, and 17 and 3 in the fifth group.

Problem credits: Nick Wu

1. (USACO 2020 December – Bronze 3) Stuck in a Rut

Farmer John has recently expanded the size of his farm, so from the perspective of his cows it is effectively now infinite in size! The cows think of the grazing area of the farm as an infinite 2D grid of square "cells", each filled with delicious grass (think of each cell as a square in an infinite chessboard). Each of Farmer John's cows () starts out in a different cell; some start facing north, and some start facing east.

Every hour, every cow either

* Stops if the grass in her current cell was already eaten by another cow.
* Eats all the grass in her current cell and moves one cell forward according to the direction she faces.

Over time, each cow therefore leaves a barren "rut" of empty cells behind her.

If two cows move onto the same grassy cell in the same move, they share the cell and continue moving in their respective directions in the next hour.

Please determine the amount of grass eaten by each cow. Some cows never stop, and therefore eat an infinite amount of grass.

**INPUT FORMAT (input arrives from the terminal / stdin):**

The first line of input contains . Each of the next lines describes the starting location of a cow, in terms of a character that is either N (for north-facing) or E (for east-facing) and two nonnegative integers and (, ) giving the coordinates of a cell. All -coordinates are distinct from each-other, and similarly for the -coordinates.

To be as clear as possible regarding directions and coordinates, if a cow is in cell and moves north, she ends up in cell . If she instead had moved east, she would end up in cell .

**OUTPUT FORMAT (print output to the terminal / stdout):**

Print lines of output. Line in the output should describe the number of cells worth of grass that the th cow in the input eats. If a cow eats an infinite amount of grass, output "Infinity" for that cow.

**SAMPLE INPUT:**

|  |
| --- |
| 6  E 3 5  N 5 3  E 4 6  E 10 4  N 11 2  N 8 1 |

**SAMPLE OUTPUT:**

|  |
| --- |
| 5  3  Infinity  Infinity  2  5 |

**SCORING:**

* In test cases 2-5, all coordinates are at most .
* In test cases 6-10, there are no additional constraints.

Problem credits: Brian Dean

1. (CSP J2 2021 – T2) 直播获奖

【问题描述】

NOI2130即将举行。为了增加观赏性，CCF决定逐一评出每个选手的成绩，并直播即时的获奖分数线。本次竞赛的获奖率为，即当前排名前的选手的最低成绩就是即时的分数线。

更具体地，若当前已评出了个选手的成绩，则当前计划获奖人数为，其中是获奖百分比，表示对向下取整，表示和中较大的数。如有选手成绩相同，则所有成绩并列的选手都能获奖，因此实际获奖人数可能比计划中多。

作为评测组的技术人员，请你帮CCF写一个直播程序。

【输入】

输入文件名为live.in。

第一行有两个正整数。分别代表选手总数与获奖率。

第二行有个非负整数，依次代表逐一评出的选手成绩。

【输出】

只有一行，包含个非负整数，依次代表选手成绩逐一评出后，即时的获奖分数线。相邻两个整数间用一个空格分隔。

【输入输出样例1】

|  |  |
| --- | --- |
| input | output |
| 10 60  200 300 400 500 600 600 0 300 200 100 | 200 300 400 400 400 500 400 400 300 300 |

【提示与说明】

在计算计划获奖人数时，如用浮点类型的变量（如C/C++中的float、double，Pascal中的real、double、extended等）存储获奖比例，则计算时的结果可能为，也可能为，向下取整后的结果不确定。因此，建议仅使用整型变量，以计算出准确值。

【输入输出样例1说明】

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 已评测选手人数 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 计划获奖人数 | 1 | 1 | 1 | 2 | 3 | 3 | 4 | 4 | 5 | 6 |
| 已评测选手的分数从高到低排列（其中，分数线用**粗体**标出） | **200** | **300**  200 | **400**  300  200 | 500  **400**  300  200 | 600  500  **400**  300  200 | 600  600  **500**  400  300  200 | 600  600  500  **400**  300  200  0 | 600  600  500  **400**  300  300  200  0 | 600  600  500  400  **300**  300  200  200  0 | 600  600  500  400  300  **300**  200  200  100  0 |

【输入输出样例2】

|  |  |
| --- | --- |
| input | output |
| 10 30  100 100 600 100 100 100 100 100 100 100 | 100 100 600 600 600 600 100 100 100 100 |

【限制】

|  |  |
| --- | --- |
| 测试点编号 |  |
| 1~3 |  |
| 4~6 |  |
| 7~10 |  |
| 11~17 |  |
| 18~20 |  |

对于所有测试点，每个选手的成绩均为不超过的非负整数，获奖百分比是一个正整数且。

1. (NOIP 2002 Junior – T4) 过河卒

棋盘上 A 点有一个过河卒，需要走到目标 B 点。卒行走的规则：可以向下、或者向右。同时在棋盘上 C 点有一个对方的马，该马所在的点和所有跳跃一步可达的点称为对方马的控制点。因此称之为“马拦过河卒”。

棋盘用坐标表示，A 点 (0, 0)、B 点 (n, m)，以及马的位置；现在要求你计算出卒从 A 点能够到达 B 点的路径的条数，假设马的位置是固定不动的，并不是卒走一步马走一步。

【输入格式】

一行四个正整数，分别表示 BBB 点坐标和马的坐标。

【输出格式】

一个整数，表示所有的路径条数。

【输入输出样例1】

|  |  |
| --- | --- |
| input | output |
| 6 6 3 3 | 6 |

【说明/提示】

对于100% 的数据，1≤n, m≤20，0≤ 马的坐标 ≤20。

\*考虑 n, m 大小达到 2000 时的情况。