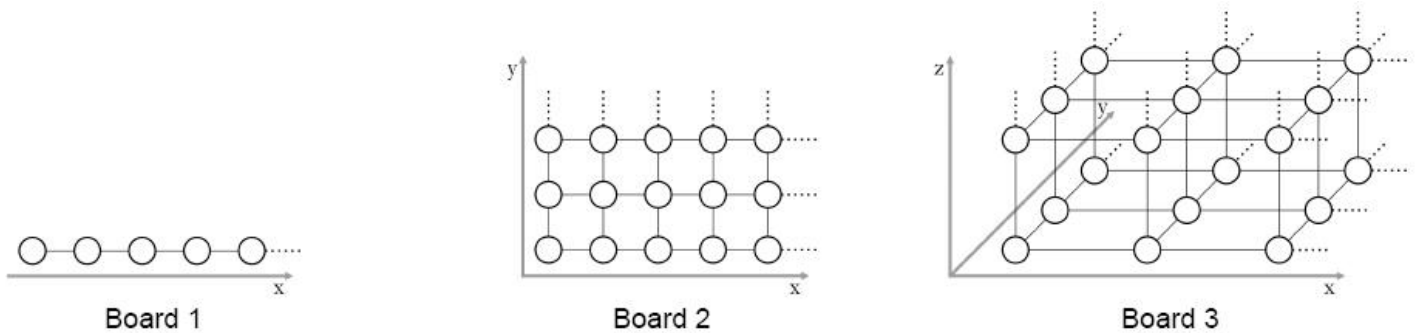# IOI07 Pairs

Mirko and Slavko are playing with toy animals. First, they choose one of three boards given in the figure below. Each board consists of cells (shown as circles in the figure) arranged into a one, two or three dimensional grid.

Mirko then places N little **toy animals** into the cells.

The **distance** between two cells is the smallest number of moves that an animal would need in order to reach one cell from the other. In one move, the animal may step into one of the adjacent cells (connected by line segments in the figure).

Two animals can hear each other if the distance between their cells is **at most D**. Slavko's task is to calculate how many **pairs of animals** there are such that one animal can hear the other.

Task

Write a program that, given the board type, the locations of all animals, and the number D, finds the desired number of pairs.

Input

The first line of input contains four integers in this order:

- The board type B $(1 \le B \le 3)$;
- The number of animals N $(1 \le N \le 100\ 000)$;
- The largest distance D at which two animals can hear each other $(1 \le D \le 100\ 000\ 000)$;
- The size of the board M (the largest coordinate allowed to appear in the input):
    - When B=1, M will be at most 75 000 000.
    - When B=2, M will be at most 75 000.
    - When B=3, M will be at most 75.

Each of the following N lines contains B integers separated by single spaces, the coordinates of one toy animal. Each coordinate will be between 1 and M (inclusive).

More than one animal may occupy the same cell.

Output

Output should consist of a single integer, the number of pairs of animals that can hear each other.

Note: use a 64-bit integer type to calculate and output the result (long long in C/C++, int64 in Pascal).

Grading

In test cases worth a total of 30 points, the number of animals N will be at most 1000.

Furthermore, for each of the three board types, a solution that correctly solves all test cases of that type will be awarded at least 30 points.

Example

Input	Input	Input
1 6 5 100	2 5 4 10	3 8 10 20
25	5 2	10 10 10
50	7 2	10 10 20

50
10
20
23

Output
4

8 4
6 5
4 4

Output
8

10 20 10
10 20 20
20 10 10
20 10 20
20 20 10
20 20 20

Output
12