Let's learn about list comprehensions! You are given three integers $X, Y$ and $Z$ representing the dimensions of a cuboid along with an integer $N$. You have to print a list of all possible coordinates given by $(i, j, k)$ on a 3D grid where the sum of $i + j + k$ is not equal to $N$. Here, $0 \leq i \leq X; 0 \leq j \leq Y; 0 \leq k \leq Z$

**Input Format**

Four integers $X, Y, Z$ and $N$ each on four separate lines, respectively.

**Constraints**

Print the list in lexicographic increasing order.

**Sample Input 0**

```
1
1
1
2
```

**Sample Output 0**

```
[[0, 0, 0], [0, 0, 1], [0, 1, 0], [1, 0, 0], [1, 1, 1]]
```

**Explanation 0**

***Concept***

You have already used lists in previous hacks. List comprehensions are an elegant way to build a list without having to use different for loops to append values one by one. This example might help.

***Example:*** You are given two integers x and y . You need to find out the ordered pairs ( i , j ) , such that ( i + j ) is not equal to n and print them in lexicographic order.( 0 <= i <= x ) and ( 0 <= j <= y) This is the code if ***we dont use list comprehensions in Python***.

```
python x = int ( raw_input()) y = int ( raw_input()) n = int ( raw_input()) ar = [] p = 0 for i in range ( x + 1 ) : for j in range( y + 1): if i+j != n: ar.append([]) ar[p] = [ i , j ] p+=1 print ar
```

Other smaller codes may also exist, but using list comprehensions is always a good option. ***Code using list comprehensions:***

```
python x = int ( raw_input()) y = int ( raw_input()) n = int ( raw_input()) print [ [ i, j] for i in range( x + 1) for j in range( y + 1) if ( ( i + j ) != n )]
```

**Sample Input 1**

```
2
2
2
2
```

**Sample Output 1**

```
[[0, 0, 0], [0, 0, 1], [0, 1, 0], [0, 1, 2], [0, 2, 1], [0, 2, 2], [1, 0, 0], [1, 0, 2], [1, 1, 1], [1, 1, 2], [1, 2, 0], [1, 2, 1], [1, 2, 2], [2, 0, 1], [2, 0, 2], [2, 1, 0], [2, 1, 1], [2, 1, 2], [2, 2
```