Пътищата към нулата

Един ден президентът на Галактиката Зейфод Бийблброкс попадна в целочислената решетка на едно тъпо тримерно пространство след случайно телепортиране, наложило се поради неминуемия сблъсък на кораба на Зейфод с една малка планета. И добре, че се намираше в положителния октант, защото единствения начин да се излезе оттам беше през нулата на пространството, като единственото нещо, което можеше да прави беше да отиде в целочислена точка, чиято сума от координати е с единица по-малка от сумата на координатите на точката, в която се намираше в момента. И то използвайки двата си крака и трите си ръце! Положението се усложняваше и от факта, че двете му глави не можеха да стигнат до споразумение в кой момент коя координата да бъде намалена с единица, за да бъде определена точката на скока. В резултат движенията на Зейфод Бийблброкс бяха направлявани един път от първата глава, и после от втората глава, после пак от първата и т.н., като всяка глава сменяше посоката на движението му.
(Дъглас Адамс, Пътеводител на галактическия стопаджия, Бард, София, 2002)
Напишете програма, която да определи дали Зейфод Бийблброкс ще се измъкне от тъпото тримерно пространство и ако да, колко са възможните различни пътища за достигане на нулата.

Жокер: Задачата е усложнен вариант на задачата 'Paths', която вече решихме.

Там: path_counts[i, j] = path_counts[i - 1, j] + path_counts[i, j - 1]

В тази задача можем спокойно да сметнем, че се движим наобратно от (0,0,0) към (X,Y,Z) и да преброим по-колко начина можем да стигнем до (X,Y,Z). Те ще са същия брой, колкото начините започвайки от (X,Y,Z) и завършвайки в (0,0,0). Ако всичко беше наред целевата функция на динамичното програмиране за тази задача (path_counts - тримерна DP) щеше да е:

path_counts[i, j, к] = path_counts[i - 1, j, k] + path_counts[i, j - 1, k] + path_counts[i, j, k - 1]

ПРОБЛЕМ 1: За съжаление условието, че не можем да се движим два пъти по едно измерение прави горната формула невалидна, тъй като path_counts[i - 1, j, k] включва в себе си и броя начини идвайки от path_counts[i - 2, j, k]. Настоящото условие ни забранява да направим преходите [i - 2, j, k] -> [i - 1, j, k] -> [i, j, k], тъй като правим два поредни хода по едно и също измерение (0-вото). Същите съображения важат и за path_counts[i, j - 1, k] и path_counts[i, j, k -1]. Какво да направим за да се справим с проблема?

РЕШЕНИЕ: Ще направим нашия модел по-прецизен и по-детайлен, добавяйки четвърто измерение в нашата path_counts(или DP) таблица с размер 3 елемента.

Значението на path_counts[i, j, k, d] ще е: по-колко начина можем да достигнем точката (i, j, k), идвайки в нея по измерение d (d = 0..2). Тоест, при d=0 ще броим само начините, идвайки от (i-1, j, k). При d=1 ще броим само начините идвайки от (i, j-1, k). При d=2 ще броим само начините, идвайки от (i, j, k-1). В този случай:

path_counts_3d[i, j, k] = path_counts[i, j, k, 0] + path_counts[i, j, k, 1] + path_counts[i, j, k, 2]
Тук не сме изтървали нищо, тъй като броим всички начини, идвайки от всички възможни посоки в точката (i, j, k)

В крайна сметка целевите функция ще станат 3 (за всеки елемент от 4-тото измерени):

path_counts[i, j, k, 0] = path_counts[i-1, j, k, ???] + path_counts[i-1, j, k, ???]
path_counts[i, j, k, 1] = path_counts[i, j-1, k, ???] + path_counts[i, j-1, k, ???]
path_counts[i, j, k, 2] = path_counts[i, j, k-1, ???] + path_counts[i, j, k-1, ???]

Вие решете какво ще има на местата на ???.

ПРОБЛЕМ 2: Не можем да инициализираме началните граници path_counts[0, 0, i] = path_counts[0, i, 0] = path_counts[i, 0, 0] = 1, тъй като това означава да се движим i пъти подред, по едно и също измерение, което е забранено по условие.

РЕШЕНИЕ 2: Можем да достигнем точките (i, i, 0), (i, 0, i) и (0, i, i) по два начина:
Първият начин (завършвайки по измерение 0) за (i, i, 0) е: (0,0,0) -> (0,1,0) -> (1, 1, 0) -> (1, 2, 0) -> ...-> (i-1, i, 0) -> (i, i, 0).
Вторият начин (завършвайки по измерение 1) за (i, i, 0) е: (0,0,0) -> (1,0,0) -> (1, 1, 0) -> (2, 1, 0) -> ...-> (i, i-1, 0) -> (i, i, 0)
Симетрично е за точките (i, 0, i) и (0, i, i).
Всички останали точки (i, j, 0), (i, 0, j) и (0, i, j) за i != j са недостижими, поради условието, че не можем да направим два хода в едно и също измерение.
Следователно ще имаме следната инициализация:

path_counts[i, j, 0, d] = 0 за d = 0..2 и i != j
path_counts[i, 0, j, d] = 0 за d = 0..2 и i != j

path_counts[0, i, j, d] = 0 за d = 0..2 и i != j
и
path_counts[i, i, 0, 0] = 1 path_counts[i, i, 0, 1] = 1 path_counts[i, i, 0, 2] = 0, тъй като не можем да дойдем от точка (i, i, -1) path_counts[i, 0, i, 0] = 1 path_counts[i, 0, i, 1] = 0, тъй като не можем да дойдем от точка (i, -1, i) path_counts[i, 0, i, 2] = 1
path_counts[0, i, i, 0] = 0, тъй като не можем да дойдем от точка (i, -1, i)
path_counts[0, i, i, 1] = 1 path_counts[0, i, i, 2] = 1

Всичко останало попълваме с -1, което означава, че още не сме го изчислили.

След това попълвате четиримерната динамична таблица рекурсивно, използвайки горните рекурентни формули (path_counts[i, j, k, 0] = ...; path_counts[i, j, k, 1]=...; path_counts[i, j, k, 2]=...)

Отговорът на задачата, разбира се, ще е всичките възможни начини за влизане в точка (X, Y, Z), а те са: path_counts[X, Y, Z, 0] + path_counts[X, Y, Z, 1] + path_counts[X, Y, Z, 2]

Input Format

На първия ред от стандартния вход е даден броя на примерните 3D координати T. На нов ред за всеки пример се задава точка от положителния октант на тримерното пространство с координати (X, Y, Z) - три цели положителни числа, не по-големи от 10.

Constraints

1 <= T <= 6
1 <= X, Y, Z <= 10
Няма право да се движи два скока подред в едно и също измерение.

Output Format

На стандартния изход за всеки пример се извежда по едно число - броят на възможните различни пътища до тримерната нула.

Sample Input 0

```
3
2 2 2
4 1 1
5 4 3
```

Sample Output 0

30
0
588