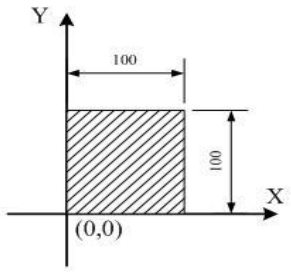
Problem 1. 判斷座標是否在正方形的範圍內

(Time Limit: 2 seconds)

問題描述 :

有一正方形，長、寬均為 100，且起始座標為(0,0)。請寫一支程式可以輸入「點」的座標，並判斷「點」是否在正方形的範圍內。如果「點」的位置剛好在邊界的話也算是在正方形範圍內(例：x=100，y=10)。



輸入說明:

第一行為一個正整數 N ($0 < N < 10$)，代表共有幾組測試資料。之後接下來有 N 組數據，每一組輸入有兩正整數字，分別代表 X 與 Y 座標。

輸出說明:

依序輸出座標在正方形範圍內或外，最後必須有換行字元。

範例:

Sample Input:	**Sample Output:**
2 10 20 150 150	inside outside

Problem 2. 地雷檢查

(Time Limit: 2 seconds)

問題描述 :

有一個8 × 8 的戰場,今天地雷超人會在這上面放*N*顆地雷,預防敵人入侵，而這些地雷是採米字攻擊，如果這些地雷在直、橫、斜上有其他的地雷，便會自動爆炸，為了防止這類情況，所以在地雷超人畫完地雷藍圖後，要幫他檢查是否可以這樣放，不行請打 X。

輸入說明:

第一列輸入一個正整數 *n* (0 < *n* <10)。其後有 *n* 個 test cases，每一個 case 輸入 0/1 的整數(空白間隔)8 × 8 矩陣。不同的 case 以一空行隔開。

輸出說明:

每個 case 輸出一行結果，判斷每個 1 的直、橫、斜是否皆為 0，是的話印出大寫 O，不是則印出大寫 X，最後必須有換行字元。

範例:

Sample Input:	Sample Output:
2 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1	O X

Problem 3. 立柱包牌法

(Time Limit: 2 seconds)

問題描述 ：

包牌一直是許多彩卷迷用來提高中獎機率的手段,而立柱包牌法則是其中一種包牌的手法。立柱包牌法的順序如下：

- 選擇可能會出現的號碼。彩迷要由 1~49 個號碼 (以大樂透為例) 中選出若干個在下期很有機會會出現的號碼。舉例而言,令這位彩迷選了 2 ， 18 ， 19 ， 22 ， 26 ， 27 ， 28 ， 31 ， 33 ， 37 ， 42 ， 48 ， 49 。
- 將號碼分成六類。接著彩迷會設立 6 個柱子，並把選出的號碼分別歸類到不同的柱子上。每個柱子致少會有一個數字。舉例而言，下面是這位使用者的歸類。

柱 1	柱 2	柱 3	柱 4	柱 5	柱 6
2	18	22	26	33	42
48	19	27	28	37	49
			31		

- 最後，使用組合的方式，把所有可能的號碼組合都列出來，而這些組合就是彩迷所要投注的號碼。挑選的方式很簡單，上述共有 6 個柱子，每個柱子上一次可以取出一個號碼。因為共有 6 個柱子，所以每一次都可以挑出 6 個號碼來。以上述的例子而言，所有的排列組合數共有 2*2*2*3*2*2=96 組號碼。

立柱包牌有一個特色,那就是屬於同一柱子的號碼一定不會出現在同一組的號碼中。這個特色可以大量地減少組合的個數。舉例而言,這位彩迷共選出 13 個號碼，在 13 個號碼中取 6 個號碼，總共的組合數為 C(13,6) = 1716 ，若要將這些號碼都買下來，金額為 1716*50=85800 ，是一個很龐大的開銷。而若使用立柱包牌，總共的組合數只有 96 組，這是因為立柱包牌法把所有“ 不會一起出現號碼 ”的組合都給刪掉了 (舉例而言，這位彩迷認為 18 號與 19 號不可能一起出現，於是將它們歸於同一柱中，所以同時出現 18 及 19 的組合就會被忽略掉) 。就算把所有的組合都買下來，也只要花 96*50=4800 元。

請寫一個程式，接受使用者的設定，並輸出所有的立柱組合。

輸入說明:

我們的輸入包含 6 行。每一行代表一個柱子的內容。一個柱子中會包含 1 個以上的數字，數字由小到大排列彼此用逗點隔開。

輸出說明:

請依照字典順序將所有的組合輸出。每一行為一個組合，同一行數字間以逗號分隔，最後必須有換行字元。

範例:

Sample Input:	Sample Output:
2,48 18,19 22,27 26,28,31 33,37 42,49	2,18,22,26,33,42 2,18,22,26,33,49 2,18,22,26,37,42 2,18,22,26,37,49 2,18,22,28,33,42 2,18,22,28,33,49 2,18,22,28,37,42 2,18,22,28,37,49 2,18,22,31,33,42 2,18,22,31,33,49 2,18,22,31,37,42 2,18,22,31,37,49 2,18,27,26,33,42 2,18,27,26,33,49 2,18,27,26,37,42 2,18,27,26,37,49 2,18,27,28,33,42 2,18,27,28,33,49 2,18,27,28,37,42 2,18,27,28,37,49 2,18,27,31,33,42 2,18,27,31,33,49

	2,18,27,31,37,42 2,18,27,31,37,49 2,19,22,26,33,42 2,19,22,26,33,49 2,19,22,26,37,42 2,19,22,26,37,49 2,19,22,28,33,42 2,19,22,28,33,49 2,19,22,28,37,42 2,19,22,28,37,49 2,19,22,31,33,42 2,19,22,31,33,49 2,19,22,31,37,42 2,19,22,31,37,49 2,19,27,26,33,42 2,19,27,26,33,49 2,19,27,26,37,42 2,19,27,26,37,49 2,19,27,28,33,42 2,19,27,28,33,49 2,19,27,28,37,42 2,19,27,28,37,49 2,19,27,31,33,42 2,19,27,31,33,49 2,19,27,31,37,42 2,19,27,31,37,49 48,18,22,26,33,42 48,18,22,26,33,49 48,18,22,26,37,42 48,18,22,26,37,49 48,18,22,28,33,42 48,18,22,28,33,49 48,18,22,28,37,42 48,18,22,28,37,49 48,18,22,31,33,42 48,18,22,31,33,49 48,18,22,31,37,42 48,18,22,31,37,49

	48,18,27,26,33,42 48,18,27,26,33,49 48,18,27,26,37,42 48,18,27,26,37,49 48,18,27,28,33,42 48,18,27,28,33,49 48,18,27,28,37,42 48,18,27,28,37,49 48,18,27,31,33,42 48,18,27,31,33,49 48,18,27,31,37,42 48,18,27,31,37,49 48,19,22,26,33,42 48,19,22,26,33,49 48,19,22,26,37,42 48,19,22,26,37,49 48,19,22,28,33,42 48,19,22,28,33,49 48,19,22,28,37,42 48,19,22,28,37,49 48,19,22,31,33,42 48,19,22,31,33,49 48,19,22,31,37,42 48,19,22,31,37,49 48,19,27,26,33,42 48,19,27,26,33,49 48,19,27,26,37,42 48,19,27,26,37,49 48,19,27,28,33,42 48,19,27,28,33,49 48,19,27,28,37,42 48,19,27,28,37,49 48,19,27,31,33,42 48,19,27,31,33,49 48,19,27,31,37,42 48,19,27,31,37,49

Problem 4. 算式中序轉後序

(Time Limit: 2 seconds)

問題描述 :

給予一個由變數與整數加上+-*/四則運算與冪次^運算的數學式， 變數用一個英文字母表示，大寫與小寫的名稱表示不同的變數。算式中可以用括號分隔開優先進行的運算。 四則運算與冪次運算的優先次序是依照冪次大於乘除大於加減。 運算符號在兩運算子中間的表示方式為中序表示，中序表示法有時需要藉助括號來表示運算的先後次序。 請將用中序表示的運算式改成後序表示方式。後序表示法一般並不需要藉由括號來區分運算的先後次序。

輸入說明:

第一列輸入一個正整數 n ($0 < n < 10$)。其後有 n 個 test cases，每一個 case 輸入為一行長度在 512 個字母內的字串，用中序表示法表達的算式，運算元與運算子之間可能有可能沒有空白。

輸出說明：

依序輸出各 test case 的後序表示的算式，每個變數或運算符號之間必須以個空白隔開，最後必須有換行字元。

範例:

Sample Input:	Sample Output:
2 a + b * (c - d)^e 234 - 123 * (A-B)^C	a b c d - e ^ * + 234 123 A B - C ^ * -

Problem 5. Maze for Robot

(Time Limit: 2 seconds)

Problem Description

The Fun-To-Play company is designing a maze to train a robot. The maze has several check points, each with 3 doors labeled as a, b, and c. The robot has to choose a door to continue his journey. Some of the check points have a hidden door which will only show up when the battery of the robot runs out. There is no hint for which door leads to which check point and some of the doors may lead back to the same check point. The robot will keep moving until the battery runs out. If the robot stops at a check point with the hidden door when the battery runs out, the robot gets a credit. Assume that the time needed for a robot to choose a door and move on to a check point is always the same, and, for simplicity sake, can be considered as one time unit. Also assume that the battery can run for *L* time units. Your mission is to find the number of different ways for the robot to get credits with the given maze and battery life.

The maze may be modeled by a finite state machine *M*. The input alphabets are a, b, and c. Each check point is a state of *M*. The entry is the starting state and the check points with hidden doors are the final states. State *i* with input a maps to state *j* if the door "a" in check point *i* leads to check point *j*. Then the solution is the same as finding the different number of strings over alphabets {a,b,c} with length *L* that are accepted by machine *M*.

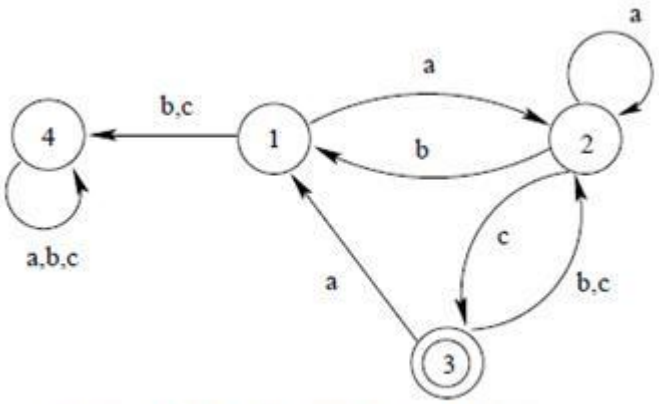


Figure 1: Finite state machine *M*

Figure 1 shows an example of a maze represented by a finite state machine *M*. State 1 is the starting state and the final states are indicated by double circles. Given *L*=4, there are only 4 possible strings aaac, abac, acbc and accc that can be accepted by *M*.

Technical Specification

1. There are N ($1 \leq N \leq 100$) states.
2. The state 1 is always the starting state.
3. Each state has exactly 3 transitions with input a, b and c respectively.
4. There is at least 1 final state.

Input Format

The first line of the input file contains an integer indicating the number of test cases to follow. The first line of each test case contains a positive integer *N* (less than 100) indicating the number of states in the test case. Each of the *N* following lines will contain the description of one state. Each state is described by a sequence of five integers, the first as the name of the state, followed by three indicating the transitions to the next state of each input alphabet a, b and c respectively. The last integer is 1 if the state is a final state and 0 otherwise. The last line of the test case is an integer (not larger then 20) denoting the length of the input strings.

Output Format

For each test case, output a line with a single integer, indicating the number of different strings that are accepted by the finite state machine (the number of different ways that the robot may take to win).

Example

Sample Input:	Sample Output:
3 4 1 2 4 4 0 2 2 1 3 0 3 1 2 2 1 4 4 4 4 0 4 8 1 2 3 4 0 2 1 5 6 0 3 1 5 7 0 4 1 6 7 0 5 2 3 8 0 6 2 4 8 0 7 3 4 8 0 8 5 6 7 1 9 5 1 2 4 5 1 2 1 3 5 0 3 4 2 5 1 4 3 1 5 0 5 5 5 5 0 19	4 4920 0