2015 網際網路程式設計全國大賽
國中組初賽

- 本次比賽共 6 題，含本封面共 16 頁。
- 全部題目的輸入都來自**標準輸入**。
  輸入中可能包含多組輸入，依題目敘述分隔。
- 全部題目的輸出皆輸出到螢幕 (**標準輸出**)。
  輸出和裁判的答案必須完全一致，英文字母大小寫不同或有多餘字元皆視為答題錯誤。
- 比賽中上傳之程式碼，請依照以下規則命名:
  1. 若使用 C 做為比賽語言則命名為 `pa.c`, `pb.c`, 以此類推。
  2. 若使用 C++ 做為比賽語言則命名為 `pa.cpp`, `pb.cpp`, 以此類推。
- `cin` 輸入經測試發現速度遠慢於 `scanf` 輸入，
  答題者若使用需自行承擔因輸入速度過慢導致 Time Limit Exceeded 的風險。
- 使用 `scanf` 或 `printf` 處理長整數 (`long long int`) 時，請使用 `%lld`。
- 每一題的執行時間限制，請參考 Kattis 上的題目敘述。

2015 網際網路程式設計全國大賽
解題程式輸入輸出範例

C 程式範例：

```c
#include <stdio.h>
int main(void)
{
    int cases;
    scanf("%d", &cases);
    for (int i = 0; i < cases; ++i)
    {
        long long a, b;
        scanf("%lld %lld", &a, &b);
        printf("%lld\n", a + b);
    }
    return 0;
}
```

C++ 程式範例：

```cpp
#include <iostream>
int main()
{
    int cases;
    std::cin >> cases;
    for (int i = 0; i < cases; ++i)
    {
        long long a, b;
        std::cin >> a >> b;
        std::cout << a + b << std::endl;
    }
    return 0;
}
```

A. 換換換

Problem ID: guess

貓咪一族一直都很擅長一個小遊戲。

這個遊戲很簡單，遊戲主持人會先拿出三個具有蓋子功能的容器，傳統上會使用貝殼，但近年來也會使用杯子或是小蓋子等物品。接著遊戲主持人也會拿出一顆粒狀的標記物，通常會用小石頭或是豆子。接下來，主持人會先把貝殼們面朝下蓋住並排成一直線，之後翻開其中一個貝殼並把豆子再放進貝殼中，接著再蓋起來。之後主持人會開始一直交換貝殼，直到交換結束後，會重新排成一直線，玩家要負責猜出標記物到底在哪個貝殼裡。

這個小遊戲在人類圈裡一直沒有一個固定的中文名稱，有人說是猜貝殼或是猜豆子等等。至於英文名稱也是有各種說法，像是 shell game, thimblerig 或是 cups and balls trick。而非常擅長這個遊戲的貓咪一族，甚至把這個遊戲列入教材。於是身為貓咪一族的喵喵跟貓貓，也曾經在小學的時候學過了這個遊戲，以及這個遊戲的歷史。

這個遊戲可以追溯到貓咪一族統治古希臘的時候就已有畫作描繪有許多貓咪在市集玩著這個遊戲，堪稱為歷史悠久。但是這個遊戲在傳進人類世界之後，也變得不再只是單純的眼力遊戲，而貪婪的人類將之用於賭博，並會在遊戲中運用各種障眼法跟魔術手法來進行作弊。為了維護這個遊戲的神聖性，也為了避免善良的貓咪一族受到邪惡的人類的欺負，貓咪一族甚至有法律嚴禁貓咪跟人類玩這個遊戲。

最近貓咪一族的量子電腦有了突破性的進展，不僅是運算速度的突破，也變得非常容易量產。連喵喵跟貓貓家的寵物鼠都可以配一台量子電腦來玩「當個創世貓（Meowcraft）」。而喵喵跟貓貓他們也因為好玩，開發了你這個會寫上古程式語言 C 與 C++ 的貓工智能。

只是他們不知道你的能力究竟有多強，只好開始翻閱論文研究，試圖找個好方法來評估你的能力。但是在他們埋進紙堆之前，他們決定先讓你寫個程式來玩這個歷史悠久的小遊戲，來證明你有足夠的能力來接受他們更進一步的測試。

為了證明你不是充滿臭蟲（Bug）的貓工智能，你能夠快速地撰寫出一個程式來玩這個遊戲嗎？

由於貓貓跟喵喵並沒有要刁難你的意思，他們只要求你寫出一個「給你目前豆子在哪個貝殼，接著只**交換一次**後，回答豆子最後位於第幾個貝殼中」的程式。

當然，這個遊戲是非常神聖的，身為貓咪一族的貓貓跟喵喵絕對不會作弊。

Input

輸入只有兩行。

第一行是一個整數 n，表示豆子放在從左邊數來第 n 個貝殼之中。第二行有兩個整數 x, y，表示交換第 x 個與第 y 個貝殼的位置。

- $1 \leq n, x, y \leq 3$
- $x \neq y$

Output

請輸出一行恰包含一個整數 k，表示最後豆子位於從左邊數來第 k 個貝殼中。

Sample Input 1	Sample Output 1
1 1 2	2

Sample Input 2	Sample Output 2
1 2 3	1

B. 胖胖數

Problem ID: candy

哇，好多糖果唷！

胖胖魚最近從瘦瘦蚯那邊獲得了不少糖果，大方的瘦瘦蚯希望胖胖魚可以幫忙把糖果分送給大家。

假設胖胖魚手上有 n 顆糖果，他會把糖果分給盡量多人，使得每個人獲得的糖果數量都一樣，而且每個人**至少有兩顆**。畢竟如果只被分到一顆糖果的話，他們會覺得很孤單，糖果也會因此變得不好吃。

舉例而言，如果胖胖魚手上有 $n = 21$ 顆糖果的話，他會分給含自己共七個人，每個人有三顆糖果。但如果只有 $n = 1$ 顆糖果的話，胖胖魚只好自己吃掉那顆變難吃的糖果。

聰明的瘦瘦蚯跟胖胖魚馬上就發現一個大問題：有些糖果數量會逼迫胖胖魚自己吃掉全部的糖果！這真是太胖了，因此他們為這種數字取了個貼切的名字「胖數」。舉例來說，$n = 1, 2, 5, 13, 37, 43, 97$ 都是胖數，而 $n = 4, 6, 14, 42$ 等整數則都不是胖數。

此時，剛吃完大薯的胖胖天路過此處，他覺得胖數太弱了，富有惡趣味的他決定將其中某些更胖的數命名為「胖胖數」。如果在由左至右寫下一個數的過程中一直都是胖數的話，我們就稱它為胖胖數。像 $n = 5, 37, 137, 2333$ 都是胖胖數。以 2333 為例，在從左至右的寫下的過程會有 $2, 23, 233, 2333$ 這四個胖數，所以 2333 是個胖胖數。另外，$n = 43$ 是胖數，但不是胖胖數，因為 4 不是個胖數。

好奇的他們決定問你：「有多少個介於 L, R 之間的整數（包含 L 和 R 本身）是胖胖數呢？」例如，當 $L = 1, R = 11$ 的時候有 $1, 2, 3, 5, 7, 11$ 這六個胖胖數介於 L, R 之間。

為了避免你只是胡亂回答剛好猜中，你必須回答 T 個這種問題。

Input

輸入的第一行有一個整數 T，代表有幾個問題。接下來 T 行，每行有兩個整數 L, R，代表一個問題。

- $1 \leq T \leq 10000$
- $1 \leq L \leq R \leq 10^{18}$

Output

對於每個問題 L, R，請輸出一個整數於一行，代表共有多少個胖胖數介於 L, R 之間（包含 L 和 R 本身）。

Sample Input 1	Sample Output 1
3 1 11 10 100 31415926535 31415926535	6 13 0

C. 跳躍的波利

Problem ID: path

喵喵跟貓貓以前很喜歡玩一款線上遊戲。這遊戲裡有很多可愛的怪物，當年很受女生們的喜愛。例如「波利」就是非常受歡迎的可愛怪物之一。

最近喵喵和貓貓又重新回來懷舊這個童年的回憶，不過他們不是在玩原本的遊戲，而是「波利賽跑」這個新的小遊戲。

波利是一種粉紅色的果凍狀生物，平常都用「跳躍」的方式移動，只是方向是隨機的很難預測。於是把波利用障礙物圍起來之後，像賽馬一樣，來猜看看哪隻波利會最先跳到終點，是個很有趣的小遊戲。贏家會獲得一些稀有道具，而輸了也不會有什麼損失。所以就算輸了，看著波利跳跳跳也十分有趣。

喵喵跟貓貓為了獲得稀有道具，已經盯著波利們跳來跳去好一陣子了。喵喵跟貓貓都很好奇波利的行為是否有可預測的地方，畢竟他們最近上的資訊課講到了在電腦上的隨機。老師有提到在資訊科學上，要製造真正的隨機是困難的，所以實務上都以偽隨機（pseudo-random）的方式來產生隨機數。如果產生的方式太簡單，那麼結果可能會是預測的。於是喵喵跟貓貓開始認真紀錄波利們跳躍的路線，試圖從中去分析並預測波利的動向。

他們對每一隻波利會記錄他一開始的位置，以及每一次跳躍後的位置。而在這款遊戲中，座標系是二維座標系，所以波利位置可以用一組座標 (x, y) 表示。

接下來，喵喵跟貓貓想要分析波利的行為，可是光一隻波利就有很多資料了，他們想要你幫忙寫個程式來處理，你的程式一次只會收到一隻波利的移動資訊。除此之外，他們也不知道一開始要從何下手分析，所以打算先從簡單的開始。他們現在想要知道，這隻波利從頭到尾總共改變了幾次方向。

他們想請你幫忙寫個程式，給你一隻波利的移動記錄，問你這隻波利總共改變方向了幾次。這一步改變方向的定義是，這一步跳躍的方向跟上一步跳躍的方向不同。

舉例來說，如果波利依序跳過 $(0,0) \to (1,1) \to (2,2) \to (1,2) \to (0,2) \to (1,1) \to (2,0)$，可以從下圖發現，波利在第三跟第五步的時候改變了方向。

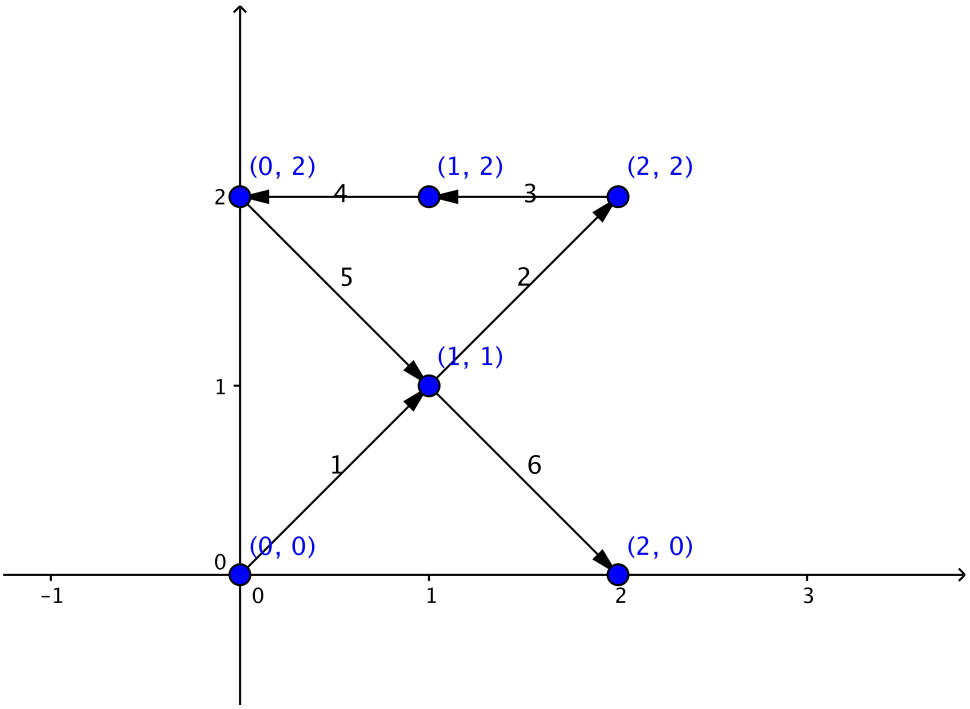


Figure C.1: 第一筆範例（邊上的編號表示是第幾步）

Input

輸入的第一行有一個整數 N，表示這隻波利有 N 筆位置紀錄。接下來有 N 行，每一行有兩個整數 x_i, y_i ，代表第 i 筆紀錄中波利的位置。

第一筆是波利一開始的位置，第二筆是波利跳第一步之後的位置，第三筆是波利跳第二步之後的位置，以此類推。

- $3 \leq N \leq 100$
- $0 \leq x, y \leq 1024$
- 任兩個**連續**位置紀錄不會相同。也就是說當 $1 \leq i < N$ 的時候保證 $(x_i, y_i) \neq (x_{i+1}, y_{i+1})$。但要注意兩筆不連續的位置紀錄可能會相同，因為波利可能會過一陣子跳回同一個座標上。

Output

輸出的第一行須包含一個整數 k，表示波利總共改變了幾次方向。接下來應有 k 行，每行有一個整數 p_i，表示第 p_i 步的時候改變了方向。請由小到大依序輸出 p_i 。

請注意，$p_i \neq 1$。因為根據定義，第一步永遠不會是改變方向的一步。

Sample Input 1

```
7
0 0
1 1
2 2
1 2
0 2
1 1
2 0
```

Sample Output 1

```
2
3
5
```

Sample Input 2

```
5
0 0
0 1
1 1
1 0
0 0
```

Sample Output 2

```
3
2
3
4
```

This page is intentionally left blank.

D. 傳紙條是很辛苦的

Problem ID: cipher

喵喵跟貓貓喜歡在上課的時候傳紙條，為了避免被傳遞紙條的同學知道紙條上寫了些什麼，喵喵跟貓貓之間約定了一個加解密的方法。他們預先約定了一組 a 到 z 的排列，稱之為「密碼表」，而加密就是根據那個排列來置換成新的字，解密就反向操作。

為了替換方便起見，他們只會使用小寫字母以及底線 '_' 用來隔開每個字。需要注意的是，替換只會替換英文字母，底線並不會被替換。

舉例來說，如果密碼表是 "rjfzowngxeqkmcihtdyvlbpasu"，則表示原文的 'a' 對應到 'r'、'b' 對應到 'j'、'c' 對應到 'f' .....以此類推。而 "this_is_a_secret_message" 加密後會變成 "vgxy_xy_r_yofdov_moyyrno"。

然而，喵喵跟貓貓覺得每張紙條都要這樣加密實在是太累了。因此他們想請你寫支程式，幫他們的訊息進行加密或解密。

Input

輸入恰有四行。

第一行是一個為 "encrypt" 或 "decrypt" 的字串，表示要加密或者是解密。第二行有一個長度為 26 的小寫字母字串，為加解密用的密碼表。第三行有一個正整數 N，代表要加密或解密的訊息長度。第四行有一個長度為 N 的字串 S，代表要加密或解密的訊息。

- $1 \leq N \leq 100$
- S 中只會有英文小寫字母或底線 '_'。

Output

請輸出一行，包含加密或解密的結果。

請注意不要輸出**任何**多餘的字元，且不要忘記最後的換行。

Sample Input 1

```
encrypt
rjfzowngxeqkmcihtdyvlbpasu
24
this_is_a_secret_message
```

Sample Output 1

```
vgxy_xy_r_yofdov_moyyrno
```

Sample Input 2

```
decrypt
rjfzowngxeqkmcihtdyvlbpasu
24
vgxy_xy_r_yofdov_moyyrno
```

Sample Output 2

```
this_is_a_secret_message
```

Sample Input 3

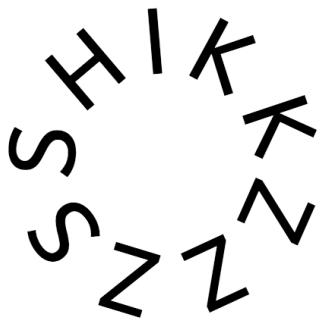
```
decrypt
psivaykmbjgtcouhwdrxzelnfq
52
iuokdpxztpxbuor_mada_br_fuzd_ytpk_ohri_br_ru_plaruca
```

Sample Output 3

```
congratulations_here_is_your_flag_npsc_is_so_awesome
```

E. 小希取名字

Problem ID: name



取名字一直都是件困擾眾人的事情，每當玩線上遊戲要決定角色名字的時候小希都很煩惱。這次，沒有取名字天份的小希決定借助喵喵的力量來完成這個困難的任務。

小希會先讓喵喵在地上順時針寫下一圈文字，再從中選出角色名字。注意喵喵因為寫下的是**一圈**文字，如果她寫下的是「真難取名字」，那麼其實跟寫下「名字真難取」或「取名字真難」是一樣的。

為了讓名字看起來有趣一些，小希會挑一段**最長且沒有重複字元的子字串**作為角色名字。以右圖為例，如果喵喵寫下 "IKKZZZSSH" 的話，小希會以 "SHIK" 作為角色名字。現在給你喵喵寫下的文字，請你回答小希所選的角色名字的長度。

Input

輸入的第一行有一個整數 T，代表有幾筆測試資料。

每筆測試資料的輸入為一個字串 s 於一行，代表喵喵在地上寫下的一圈文字。

- $1 \leq T \leq 30$
- s 的長度至少為 1，最長不超過 100000。
- 在 s 中只會有大寫英文字母。

Output

對於每筆測試資料請輸出一個整數於一行，代表小希的角色名字的長度。

Sample Input 1	Sample Output 1
2 XDDORZQQ IKKZZZSSH	5 4

F. 頗旺愛數樹

Problem ID: div

頗旺最喜歡上學了，他每天最快樂的事就是到學校學習知識。當然，今天也不例外！

頗旺到學校後，發現今天上的是最有趣的算樹課。而算樹課，顧名思義就是教大家如何算樹。在課堂上，老師說了要學會算樹就要先學會數樹，所以教大家如何數樹。

放學之後，頗旺迫不及待地唱著老師所教的口訣，開心地邊唱邊跳地回家。「一棵樹，兩棵樹，三棵樹，四棵樹，五棵樹，六棵樹，一棵樹 $\cdots\cdots$ 哎呀，又數錯了！六棵樹，七棵樹，八棵樹 $\cdots\cdots$」

一如往常地頗旺來到了樹林大道，這是頗旺回家的必經之路。樹林大道上共有 N 棵樹，而頗旺這時突然很好奇，他如果「M 棵樹一數」，能不能恰好把所有的樹都數完。例如，若他「3 棵樹一數」，則數樹的過程會包含 $3, 6, 9, 12, \ldots$ 棵樹。這種情況（$M = 3$）下，則可以「恰好數完」$24 = 3 \times 8$ 棵樹但「不能恰好數完」25 棵樹，因為會剩餘 1 棵樹。

不過，頗旺才剛學會數樹，口訣記的還不是很熟，所以他目前頂多只能一次數 11 棵樹。除此之外，他覺得 1 跟 7 長得太像了，常常讓他數錯。所以，他不喜歡一棵樹一數、七棵樹一數，他怕數一數就忘記數幾棵了。

雖然這是個很好的練習數樹的機會，但頗旺又很想趕快回家吃叔叔煮的玉蜀黍，所以他想拜託你幫他確認 M 棵樹一數能不能恰好數完 N 棵樹。

Input

第一行有一個正整數 T，代表接下來有 T 筆測試資料。

每筆測試資料只有一行，包含兩個正整數 N, M，以單一空白字元隔開，分別代表樹林大道上樹的數量，以及頗旺一次想數幾棵樹。

- $1 \leq T \leq 10$
- $1 \leq N < 10^{100000}$
- $2 \leq M \leq 11$ 且 $M \neq 7$，也就是說 M 只可能是 $2, 3, 4, 5, 6, 8, 9, 10, 11$ 其中之一。

Output

對於每筆測試資料請輸出一行。如果 M 棵一數恰好可以數完樹林大道上的樹請輸出「Yes」，否則請輸出「No」，皆不包含引號。

Sample Input 1

```
2
514 2
514 3
```

Sample Output 1

```
Yes
No
```

Sample Input 2

```
3
10077696 6
555111444 5
123456654321 11
```

Sample Output 2

```
Yes
No
Yes
```

Sample Input 3

```
1
10000000000000000000 5
```

Sample Output 3

```
Yes
```