2016 網際網路程式設計全國大賽
高中組決賽

- 本次比賽共 8 題，含本封面共 20 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，以題目敘述為主。
- 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。
- 所有題目的時間限制請參考 Kattis 網頁上各題之標示。
- 比賽中上傳之程式碼，使用 C 語言請用 `.c` 為副檔名；使用 C++ 語言則用 `.cpp` 為副檔名。
- 使用 `cin` 輸入速度遠慢於 `scanf` 輸入，若使用需自行承擔 Time Limit Exceeded 的風險。
- 部分題目有浮點數輸出，會採容許部分誤差的方式進行評測。一般來說「相對或絕對誤差小於 ϵ 皆視為正確」，ϵ 值以題目敘述為主。

  舉例來說，假設 $\epsilon = 10^{-6}$ 且 a 是正確答案，b 是你的答案，如果符合 $\frac{|a-b|}{\max(|a|,|b|,1)} \leq 10^{-6}$，就會被評測程式視為正確。

	題目名稱
題目 A	普遜的飛行計劃
題目 B	廢文大資料 mining
題目 C	是誰的餅乾藏在餅乾盒裡
題目 D	番茄大戰爭
題目 E	艾迪摺紙趣
題目 F	無限兔子問題
題目 G	密碼鎖
題目 H	吃吃為吃吃，是吃也

2016 網際網路程式設計全國大賽
輸入輸出範例

C 程式範例：

```c
#include <stdio.h>
int main()
{
    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. 普遜的飛行計劃

Problem ID: flight

普遜是一隻喜歡吃飯和睡覺的貓貓，不過牠有時會用剩餘的時間開著直升機到處逛逛。

有天，普遜注意到了蓬萊東路上有不少高樓大廈，這些高樓大廈排成一直線，相鄰的兩棟大樓的距離皆為一公尺。為了消磨時間，普遜決定去那兒開直升機。因為起飛跟降落是非常高難度的動作，因此普遜只打算從其中一棟高樓的頂樓出發，以直線飛行到另一棟樓的屋頂，並停在該棟高樓的頂樓，並且把直升機留在那裡，自己搭電梯下樓後回家睡覺。

然而，由於普遜的飛行技術不是很好，因此牠必須慎選飛行的路線。普遜思考了自己的飛行技術之後決定，自己只能接受飛行過程中，斜率的絕對值不到 U 的路線，但是如果飛行過程中太平坦好像也太沒有挑戰性了，因此牠希望飛行路線的斜率的絕對值至少要有 L 。換句話說，如果兩棟大樓底部的距離是 x，兩棟大樓的高度差是 y，那麼普遜只能接受 $L \leq |y/x| < U$ 的路線。符合普遜要求的路線顯然有非常多種，因此普遜開始好奇：在蓬萊東路上，究竟有多少條這樣的路線呢？附註：在普遜的心目中，僅僅是把起點與終點互換不算是兩條不一樣的路線。

Input

測試資料共包含兩行。

第一行有兩個整數 N, L, U，分別表示蓬萊東路的高樓數以及普遜能夠接受的斜率。第二行有 N 個整數，第 i 個數字 y_i 表示距離蓬萊東路路口 i 公尺處有一棟高度為 y_i 公尺的大樓。

- $2 \leq N \leq 10^5$
- $0 \leq y_i \leq 10^9$
- $0 \leq L < U \leq 10^9$

Output

請輸出一行包含一個整數，表示普遜可以接受的飛行路線的數量。

Sample Input 1	Sample Output 1
6 1 3 8 7 1 2 2 8	7

B. 廢文大資料 mining

Problem ID: fib

現在還是雲端大數據物聯網的時代，社群網站上無時無刻仍然有著海量的廢文被發表。

這次，你手上的資料換成了全球網路廢文資料集 (data set)，試圖從中得到珍貴的資訊。

你在茫茫廢文海中發現了一個使用者 — Eddy — 所發的廢文特別有深意，在數位的時代，他所發的所有廢文被編碼成了一串整數 $a_1, a_2, \ldots, a_n$。第 i 個整數代表 Eddy 在第 i 天所發的廢文。數字的大小有它的意義，當我們看了一篇被編碼成數字 x 的廢文之後，我們心靈的**正面能量值**會加上 x。當正面能量值掉到小於 0 時，我們會變得內心扭曲，言行不得體，毀家廢婚，面容猥瑣，這是很不好的事情。

為了大家好，你想要檢查 Eddy 所發的廢文會不會造成社會道德危機。

我們在看一個人發的廢文時，常常都是從某一則廢文開始，照著發表時間一篇一篇看，看到不想看為止。我們可以定義「看法」為一組 (l, r)，代表依序把 $a_l, a_{l+1}, \ldots, a_r$ 這幾篇廢文看過。而「不道德」的看法則是在觀看廢文的過程中，會有某些時刻使得起始正面能量值為 0 的人內心扭曲。例如 Eddy 發了 $100, -300, 200, 400$ 這幾篇廢文，那 $(3, 4)$ 是一組道德的看法 (觀看 $200, 400$ 的過程中正面能量值總是不小於 0)；$(1, 4)$ 則是不道德的看法，因為看完第 2 篇時正面能量值為 -200，這是內心扭曲的證據。

你想要知道，Eddy 所發的廢文，有幾種不道德的看法。

Input

測試資料共包含兩行。

第一行有一個正整數 n，表示 Eddy 所發的廢文數量。第二行有 n 個以空格隔開的整數，第 i 個數字 a_i 表示 Eddy 於第 i 天所發的廢文。

- 所有 a_i 皆滿足 $-100,000 \leq a_i \leq 100,000$
- $1 \leq n \leq 10^6$

Output

輸出一行包含一個非負整數，代表 Eddy 廢文裡不道德看法的數量。

Sample Input 1	Sample Output 1
3 -1 1 1	3

Sample Input 2	Sample Output 2
4 1 -3 2 4	6

C. 是誰的餅乾藏在餅乾盒裡

Problem ID: box

喜歡做點心餵食小蘿莉的蘿莉農最近又做了一批好吃的餅乾，他打算親手製作一個餅乾盒來裝這些餅乾，以完美呈現他對蘿莉的愛。

這個餅乾盒的體積必須要恰好為 V 立方單位才能完美的裝進所有餅乾，而不顯得過於擁擠或過於空曠。而為了方便製作餅乾盒，其形狀必須是一個長方體，且每邊長都是正整數單位長。基於節能減碳救地球的理念，蘿莉農希望能夠盡量節省材料，也就是說餅乾盒的表面積越小越好。但蘿莉農正忙著偷吃餅乾無暇思考這個問題，請你幫幫他。

Input

測試資料恰有一行，包含一個正整數 V，代表餅乾盒的體積。

- $1 \leq V \leq 10^9$

Output

請輸出四個整數於一行，分別代表最小可能的表面積以及其長寬高。如果有多組可能的解請輸出字典序最小的，也就是說先讓表面積盡量小，表面積一樣再讓長盡量小，依此類推。

Sample Input 1	Sample Output 1
24	52 2 3 4

Sample Input 2	Sample Output 2
217	510 1 7 31

Sample Input 3	Sample Output 3
514	1546 1 2 257

This page is intentionally left blank.

D. 番茄大戰爭

Problem ID: tomato

「上下因心」是現今世上最古老的世家之一，素有東方神秘力量的美名。每代會有三位家主共治著整個世家，而成為家主的人將捨棄過去的名字，從上一代家主手中接下傳承至今的家主名諱：上恩、下恩、以及卡恩。

每年五月十四日的時候，當代家主們會齊聚一堂，舉行名為「番茄大戰爭」的儀式。但數百年前一場離奇的木能寺大火，使所有典籍付之一炬。儀式究竟從何而來，其原理及意義又是什麼如今已無人理解。人們只知道，在儀式中上恩會和下恩決鬥 T 回合的猜猜拳。每回合勝利的一方將吃下一顆番茄，但如果兩者平手的話，將由卡恩吃下一顆番茄。

猜猜拳的決鬥方式為兩人感受東方神秘力量的指示，之後同時出示「剪刀、石頭、布」其中一種手勢，如果手勢相同的話視為平手，不同的話則依石頭砸爛剪刀、剪刀剪破布、布包住石頭的相生相剋規則決定勝利的一方。

在無數次的儀式中人們也漸漸從看似雜亂無章的出拳中歸納出了一些規律。人們發現，上恩跟下恩的出拳其實會不斷循環一個特定的模式。假設上恩的出拳模式為「石頭、剪刀、布」的話，那他在第一回合會出石頭，第二回合出剪刀，第三回合出布，第四回合又開始出石頭，依此類推。

由於要吃下番茄的量可能非常巨大，家主們需要事先開闢好足夠大的異次元胃袋。他們特地聘請你來根據上恩與下恩的出拳模式，預測三位家主分別需要吃下幾顆番茄。

Input

測試資料共包含三行，依序為 T, A, B。

其中正整數 T 代表決鬥的總回合數，而字串 A, B 則分別代表上恩及下恩的出拳模式，模式字串中的字元 `S`,`R`,`P` 分別代表剪刀、石頭、布。

- $1 \leq T \leq 10^9$
- $1 \leq |A|, |B| \leq 10^6$
- A, B 僅包含 `S`,`R`,`P` 三種字元

Output

請輸出三個整數於一行，依序為上恩、下恩及卡恩將在番茄大戰爭中吃下的番茄個數。

Sample Input 1	Sample Output 1
9 S PRPR	5 4 0

Sample Input 2	Sample Output 2
100 PPSP RSP	35 33 32

E. 艾迪摺紙趣

Problem ID: square

艾迪是天龍國小中一個冰雪聰明的孩子。今天學校美術課老師發給同學們一張長方形色紙，讓大家自由發揮創意。接過色紙後，艾迪感到無比興奮，因為這是他第一次看到長方形的色紙。於是他把色紙高高的舉在頭上，搖頭晃腦的細細端詳著它的色澤、形狀……。看著看著，好奇的艾迪越發覺得奇怪，臉上露出了十足困惑的神情。

這時，坐在艾迪旁邊的你，看著艾迪詭異的舉動加上疑惑的表情，你決定要一探究竟，看看艾迪腦裡到底在想些什麼。一問之下才發現原來艾迪覺得他手裡的這張長方形色紙太不對稱了，不是他所喜歡的正方形，所以他決定要把這張長方形色紙撕成一些正方形的小色紙。

但是艾迪手邊並沒有剪刀、美工刀之類的工具，他只能用以下的方式來切割色紙：先將手上的色紙沿著和其中一邊平行的方向在任意位置對折，接著沿著折線將整張色紙撕開、一分為二，然後再對這兩張色紙進行上述的操作，一直到剩下的色紙都是正方形的。但是艾迪不希望他的色紙變得太破碎，所以他希望你告訴他以這個方法切割，最少可以把原來的長方形切割成幾個正方形。（注意：艾迪原先拿到的長方形的邊長皆為整數，切出來的正方形邊長也必須為整數）

舉例來說，如果艾迪拿到一張 16×22 的長方形色紙，那一種可行的切割方法如下圖所示，最後會得到 6 個正方形，而這個切割方式所切出的正方形數量也是最少的。

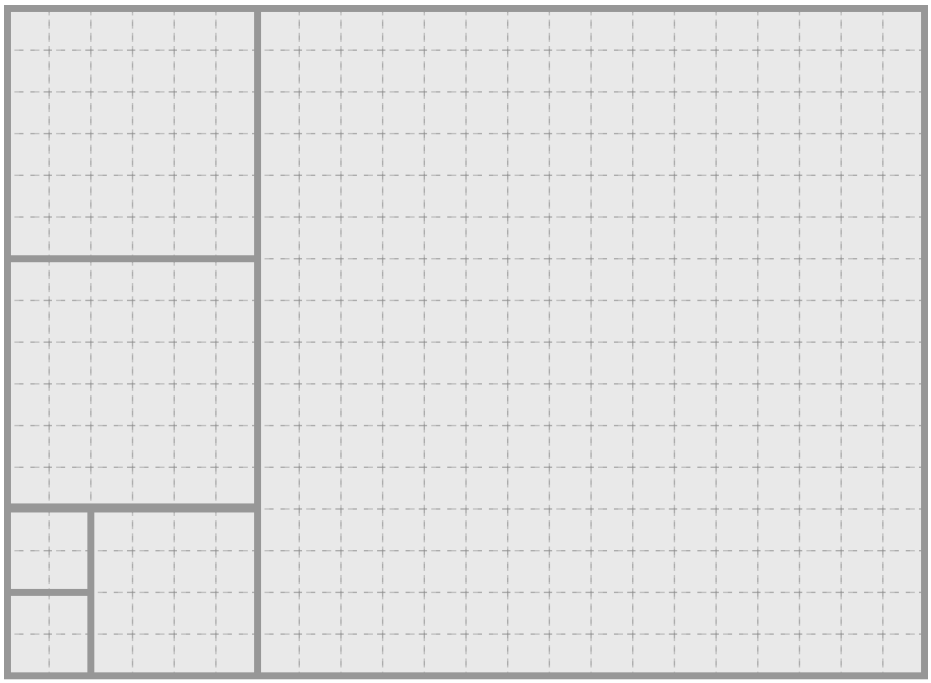
Input

測試資料只有一行，包含兩個正整數 N, M，代表艾迪的色紙大小為 $N \times M$。

- $1 \leq N, M \leq 100$
- $N \neq M$

Output

請輸出一行包含一個正整數表示這張色紙最少能切割成幾個正方形。



Sample Input 1	Sample Output 1
22 16	6

Sample Input 2	Sample Output 2
3 2	3

Sample Input 3	Sample Output 3
5 35	7

F. 無限兔子問題

Problem ID: rabbit

在一百年後的未來，你培育出了一種帶有無限繁殖基因的兔子。你決定帶著一隻能夠自體繁殖的新生兔子到月球上，成為世界最大的兔子農！

你培養出的這種兔子十分特別，每一隻兔子在新生後的兩個月會完全的發育，接著從第三個月開始每個月都會固定繁殖出一隻新生的兔子。於是你抵達月球的第一個月有一隻新生的兔子 A ，第二個月還是只有兔子 A ，在第三個月兔子 A 會繁殖出一隻新生的兔子 B ，所以總共會有兩隻兔子。第四個月兔子 A 繼續繁殖出兔子 C ，總共會有三隻兔子。第五個月兔子 A, B 各會繁殖出一隻兔子，總共有五隻兔子。以此類推，你的兔子農場很快的就充滿著兔子，不過因為你在月球上有用不完的空地，所以不必擔心空間的問題。

而為了成為偉大的兔子農，你會在每個月紀錄每隻兔子間的互動，以供未來的你進一步的研究。正式來說，如果你在第 k 個月有 M_k 隻兔子，那這個月你就會記錄下 $\frac{M_k \times (M_k - 1)}{2}$ （也就是 $\binom{M_k}{2}$ ，或是 $C_2^{M_k}$ ）筆資料。由於一百年後的未來一樣是雲端大數據時代，你把所有的資料都存放在地球上的資料庫中。而你知道地球上的資源有限，也許哪天就沒有地方存放你珍貴的資料了。所以你想要預先計算出你在某個時段內總共會紀錄的資料筆數。由於數字可能很大，請輸出資料筆數除以 $10^9 + 7$ 後的餘數。

Input

測試資料第一行有一個整數 N ，代表你關注的時段數量。

接下來有 N 行，其中第 i 行包含兩個以空白隔開的整數 s_i, t_i ，表示第 i 個時段是從第 s_i 個月到第 t_i 個月。

- $1 \leq N \leq 10^4$
- $1 \leq s_i \leq t_i \leq 10^{18}$

Output

請輸出 N 行，其中第 i 行包含一個整數表示第 i 個時段內資料筆數除以 $10^9 + 7$ 的餘數。

Sample Input 1	Sample Output 1
5 1 1 2 2 3 3 4 4 1 5	0 0 1 3 14

G. 密碼鎖

Problem ID: lock

頗旺是臺灣大學的學生，每天都開心的騎着他心愛的腳踏車上學。

然而，在臺灣大學中，有許多關於腳踏車的不可思議現象。像是，「腳踏車的稠密性」：你永遠能在兩輛腳踏車之間停入你的腳踏車；「車籃聚寶盆」：當你腳踏車停了一段時間之後，腳踏車車籃上便會自動聚集若干寶特瓶；「薛丁格的腳踏車」：在你回去牽你的腳踏車之前，你永遠不會知道你的腳踏車依然停在腳踏車位上，還是被拖吊走了。當然，這些都比不上，最可怕的「消失的腳踏車」，沒錯，就是你的腳踏車就憑空消失了。

儘管頗旺對於這些鄉野謠傳的不可思議現象不感興趣，但倘若哪天真的遇上的「消失的腳踏車」，就算一把鼻涕、一把眼淚也是找不回來的。因此，頗旺決定爲他心愛的腳踏車配上一個密碼鎖，如此一來，腳踏車就不會輕易的消失了！

但是，記憶密碼對於頗旺來說實在是太困難了，因此，他決定買一個特別製作的密碼鎖，讓他就算忘記密碼，也有機會把密碼找回來！

準確來說，頗旺所買的密碼鎖，總共有 N 位，每位分別包含數字 $1 \sim R$。而要打開這個密碼鎖，必須將每個位子都轉到對應的數字，例如，第一位是 5、第二位是 1、第三位是 4、$\cdots$。當然每個位子的數字都是環狀的，也就是說，當我把數字 1 往前轉一格時會轉到數字 2、數字 2 往後轉一格會到數字 1、數字 R 往前轉一格會轉到數字 1，數字 1 往後轉一格會轉到數字 R，以此類推。

而這個密碼鎖的特別之處在於，當你轉到不是正確的密碼時，他不僅僅只是鎖解不開來，他還會告訴你最少需要轉動多少格才能轉到正確密碼。舉例來說，這個密碼鎖有 3 位，每位分別包含數字 $1 \sim 5$。而對應的密碼是第一位爲 5、第二位爲 1、第三位爲 4，而若當你轉到第一位爲 1、第二位爲 3、第三位爲 3 時，密碼鎖會告訴你最少需要轉動 4 格才能轉到正確密碼（第一位往後轉一格、第二位往後轉兩格、第三位往前轉一格）。

頗旺拿到這個密碼鎖之後非常開心，認爲有這麼方便的功能就不用辛苦的去記憶密碼了。然而，事情總不是都那麼順利，頗旺第一天就忘記密碼了！看著心愛的腳踏車被密碼鎖鎖著無法動彈，頗旺心急如焚的想要把他解開。這時，頗旺才發現，這功能一點都不實用呀！

頗旺試了半天都毫無斬獲。剛好，身爲頗旺的好友，你看見頗旺正看著他心愛的腳踏車焦慮著，你決定來幫助他解開這個密碼鎖，看著頗旺心急的樣子，你答應他會在 $\lfloor N \lg(\lg(\lg(R))) \rfloor$ 次內把密碼解開。事不宜遲，這就開始解密碼鎖吧！($\lg(x) = \log_2(x)$，表示 2 的多少次方爲 x，e.g. $\log_2(1) = 0, \log_2(4) = 2, \log_2(5) = 2.3219\cdots, \log_2(5.14) = 2.3617\cdots$) ($\lfloor x \rfloor$ 代表小於等於 x 最大的整數)

互動說明

一開始，你的程式會從標準輸入 (standard input) 讀入兩個正整數 N、R 代表頗旺的密碼鎖有 N 位，每位分別有數字 $1 \sim R$。

而你便可以開始猜測頗旺的密碼：

當你的程式決定要猜密碼時，輸出一行包含 N 個整數，每個整數必須介於 1 到 R 之間。第一個數字代表第一位密碼、第二個數字代表第二位密碼、$\cdots$。當你輸出完成後，記得要清空 (flush) 標準輸出 (standard output)。

當我們收到你的猜測後，會把密碼鎖的結果回覆到你的標準輸入 (standard input)。回覆會包含一個非負整數，代表你最少需要轉幾格能轉到正確的密碼。

特別的是，當你猜到正確的密碼時，密碼鎖會回覆 0，你的程式必須立刻結束 (exit)。如果你在 $\lfloor N \lg(\lg(\lg(R))) \rfloor$ 次內無法找到正確的密碼，你的程式將會被強制終止。

- $10 \leq N \leq 514$
- $10^6 \leq R \leq 10^9$

以下是 C 程式 flush 的範例：

```c
#include <stdio.h>
int main(){
  int n, r, i;
  scanf( "%d%d" , &n , &r );
  for( i = 1 ; i < n ; i ++ )
    printf( "1 " );
  printf( "1\n" );
  fflush( stdout );
  long long ret;
  scanf( "%lld" , &ret );
}
```

以下是 C++ 程式 flush 的範例：

```cpp
#include <iostream>
int main(){
  int n, r;
  std::cin >> n >> r;
  for( int i = 1 ; i < n ; i ++ )
    std::cout << "1 ";
  std::cout << "1\n";
  std::cout << std::flush;
  long long ret;
  std::cin >> ret;
}
```

This page is intentionally left blank.

H. 吃吃為吃吃，是吃也

Problem ID: cook

喜歡做點心餵食小蘿莉的蘿莉農為了精進他的廚藝，特地從世界各地收集了 n 種特級食材。他打算從中挑選一些食材製作成特級餅乾，之後裝進他親手製作的精美餅乾盒中。但要挑選哪些食材好呢？蘿莉農覺得他必須親自嚐過所有可能的非空食材組合，才有辦法從這 $2^n - 1$ 種組合中挑出最適合蘿莉食用的餅乾。

我們知道，有些食材是天生絕配，搭在一起會有額外的好吃度加成；而有些食材如果混在一起的話反而會有反效果，甚至是食物中毒。更精確地來說，一個食材搭配是一個集合 s 以及一個好吃度影響值 v，如果挑選的食材組合包含 s 的話，其好吃度就會加上 v。也就是說一個食材組合的好吃度，就是其所有包含的食材搭配的好吃度總和。舉例來說，如果食材搭配 $\{1, 2\}$ 的好吃度影響值是 3，而食材搭配 $\{2, 3\}$ 的好吃度影響值是 -1，那麼食材組合 $\{1, 2, 3\}$ 的好吃度就是 $3 + (-1) = 2$。

蘿莉農嘗試各種組合的同時，他的廚藝熟練度也會逐漸上升。因此若第 i 次的食材組合好吃度是 d，則製作出來的餅乾好吃度會是 $i \times d$。為了讓這個試嚐餅乾的過程盡量愉悅而不要從此對餅乾有心理陰影，他希望找一個最好的組合嘗試順序，讓做出來的餅乾總好吃度最大。以上面的例子來說，一種可能的最好嘗試順序為 $\{2, 3\}, \{1\}, \{2\}, \{3\}, \{1, 3\}, \{1, 2, 3\}, \{1, 2\}$，其總好吃度為 $1 \times -1 + 2 \times 0 + 3 \times 0 + 4 \times 0 + 5 \times 0 + 6 \times 2 + 7 \times 3 = 32$。

Input

測試資料第一行有兩個正整數 n, m，分別代表食材數跟食材搭配數。接下來 m 行，每行會有一個字串 s_i 跟一個整數 v_i。如果 s_i 的第 j 個字元是 1 的話代表此搭配中包含第 j 種食材；反之若為 0 的話則代表不包含。而 v_i 則為此搭配的好吃度影響值。

- $1 \leq n \leq 22$
- $1 \leq m \leq 10^5$
- $|s_i| = n$
- s_i 中至少有一個字元為 1
- 所有的 s_i 皆相異
- $-100 \leq v_i \leq +100$

Output

請輸出一個整數於一行，代表最好嘗試順序的總好吃度。

Sample Input 1	Sample Output 1
2 2 01 1 10 -1	2

Sample Input 2	Sample Output 2
3 2 110 3 011 -1	32