2016 網際網路程式設計全國大賽
國中組決賽

- 本次比賽共 7 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	大富翁
題目 B	兔子
題目 C	PPAP
題目 D	過馬路
題目 E	愛玩的普遜
題目 F	三角桌會議
題目 G	Special Judge

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. 大富翁

Problem ID: dice

大富翁是一款家喻戶曉的遊戲，而遊戲中許多行動皆需要透過擲骰子來完成。凡事都認真對待的你，在研究遊戲策略時發現徹底理解骰子的各種可能性是勝利的關鍵之一。因此你想寫個程式算算看當擲出 n 顆骰子時，有幾種可能的點數組合會使得點數總和為 m。在本題中我們使用的骰子都是最常見的六面骰，上面分別有一到六點。

舉例來說，3 顆骰子擲出點數總和為 5 的可能組合共有六種：$113, 122, 131, 212, 221, 311$。

Input

測試資料恰有一行，包含兩個整數 n, m。

- $1 \leq n \leq 5$
- $n \leq m \leq 6n$

Output

請輸出一個整數於一行，代表當擲出 n 顆骰子時，有幾種可能的點數組合會使得點數的總和為 m。

Sample Input 1	Sample Output 1
2 5	4

Sample Input 2	Sample Output 2
3 5	6

This page is intentionally left blank.

B. 兔子

Problem ID: rabbit

你有聽過兔子叫嗎？你知道兔子的叫聲是什麼樣子嗎？

說到兔子的叫聲，前陣子有個由安哥拉兔組成的國家叫做提比國 (Tippy)，提比國內的兔子遇到危險時會用大叫來通知同伴。安哥拉兔 (Angora rabbit) 是一種長毛兔，一般毛經過整理後很容易弄成球狀的很可愛。

提比國是由一個由獸人族組成的，叫做東部聯合 (Eastern Federation) 的國家獨立而出，最近常常邊境仍常有衝突。於是提比國在邊境佈下了大量的兔子屋 (rabbit house) 當作哨站，而每一間兔子屋會安排一隻兔子看守，一旦有獸人族的部隊靠近，就會大叫來通知同伴。通知的方式是這樣的，一般來說會從一間兔子屋開始大叫，假設那隻兔子用 R 分貝大叫，則方圓 R 公尺的兔子都會聽到，聽到的兔子會接著用 $R-1$ 分貝大叫，然後方圓 $R-1$ 公尺的兔子會聽到，以此類推。

你可以發現，只要一開始用足夠大的聲音大叫，就可以讓所有兔子都收到警報。為了保證邊境的安全，提比國國王希望不管從哪一間兔子屋大叫，都要有能力將警報散播給所有兔子，也就是說，除了一開始那隻以外，剩餘的兔子都要能至少聽到一次叫聲警報。

但大叫是很累的，每一間兔子屋的兔子想要盡量節省力氣。他們想知道從每一間兔子屋分別發起警報時，所需的最小音量是多少？

為了簡化問題，我們把所有兔子屋擺在一條直線上，每間的座標為 $x_1, x_2, \ldots, x_n$。而一間兔子屋 x_i 的兔子如果用 k 分貝大叫，則位於 $x_i - k$ 到 $x_i + k$ 的兔子屋都能聽到。

Input

測試資料包含兩行。

第一行會有一個整數 N，表示有 N 間兔子屋。

第二行會有 N 個整數 $x_1, x_2, \ldots, x_n$，分別表示 N 間兔子屋的位置。

- $1 \leq N \leq 50$
- $0 \leq x_i \leq 100$
- 保證 x_i 非遞減排序，也就是對於所有 $i < j$ 會有 $x_i \leq x_j$

Output

輸出 N 行，每行包含一個整數 r_i，分別表示從第 i 間兔子屋要發起警報，並散播給所有的兔子，所需要的最小音量分貝數。

Sample Input 1	Sample Output 1
3 1 3 5	3 2 3

C. PPAP

Problem ID: ppap

不知道為什麼，你覺得 PPAP 這四個字特別帶感，因此嘗試在生活中發掘更多的 PPAP。

受到 PPAP 這四個字獨特美感的啟發，你決定將所有擁有類似美感的字串稱為 PPAP 型字串。具體而言，如果一個非空字串能夠被分為四段等長的字串，且第一、二、四段字串完全一樣，我們就稱它為 PPAP 型字串。舉例來說，PPAP、QQBQ、TTTT、ABABXYAB 都是 PPAP 型字串；而 PENPINEAPPLEAPPLEPEN、TTTTT、PAPP 則都不是。

為了能盡情地感受 PPAP 之美，你決定找出字串 s 中最長的 PPAP 型子字串。

一個字串 a 如果能藉由刪除另一個字串 b 的頭尾若干個字元得到，我們就稱 a 為 b 的子字串。如 A、PAP、PPAP 都是 PPAP 的子字串。非空字串則代表該字串的長度必須至少為 1。

Input

測試資料恰有一行，包含一個由大寫英文字母組成的字串 s。

- $1 \leq |s| \leq 300$

Output

請輸出 s 中最長的 PPAP 型子字串。如果有不只一個的話，請輸出字典順序最小的。如果不存在任何 PPAP 型子字串，請輸出「===>{*}(*)<===」。

Sample Input 1	Sample Output 1
XQQBQQYPPAPPZ	PPAP

Sample Input 2	Sample Output 2
PENPINEAPPLEAPPLEPEN	===>{*}(*)<===

Sample Input 3	Sample Output 3
ABABYXABABXYAB	ABABXYAB

D. 過馬路

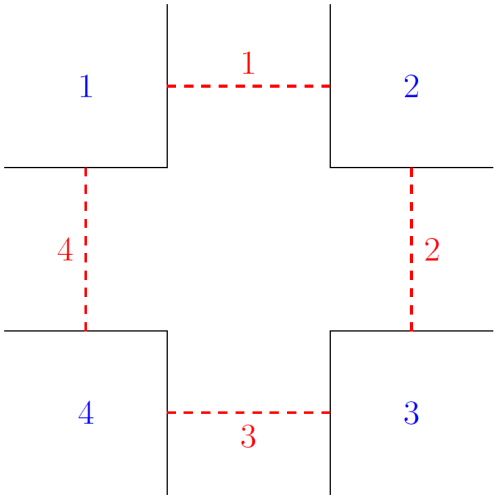
Problem ID: street

愛吃的普遜要去天龍國參加全國餡餅搶購大會 (National Pie Shopping Conference，簡稱 NPSC)。

來到會場對面時，普遜發現眼前阻礙自己與美味餡餅的關卡竟然是一個 N 叉路口！

N 叉路口周圍會有 N 個街區，我們將他們依順時針方向依序編號為 1 到 N；每個相鄰的街區之間會有一條行人穿越道，將連接街區 1 與街區 2 的行人穿越道編號為 1、將連接街區 2 與街區 3 的行人穿越道編號為 2，依此類推，特別地，街區 N 與街區 1 之間的行人穿越道編號為 N。

下圖便是一個 $N = 4$ 的 N 叉路口，其中藍色數字為街區的編號、紅色虛線為行人穿越道、紅色虛線旁的紅色數字為該行人穿越道的編號。



天龍國的行人穿越道機制很特別，在每一單位時間內只會有一條行人穿越道可以通行，這段時間結束後這條行人穿越道就不能通行，換另一條行人穿越道可以通行並持續一單位時間，重複直到每條行人穿越道恰輪過一次後，便會回到最初狀態循環。

輸入當中會告訴你這些行人穿越道可通行的順序，如果這些順序為 $4, 2, 3, 1$（即第一筆範例輸入），表示時間 1 時可通行的行人穿越道為 4、表示時間 2 時可通行的行人穿越道為 2、表示時間 3 時可通行的行人穿越道為 3、表示時間 4 時可通行的行人穿越道為 1，時間 5 又回到行人穿越道 4 且從這時開始會一直循環。

現在普遜在街區 A、NPSC 會場在街區 B，請問若普遜用最佳策略行走，他最快要在什麼

時間才能到達會場以滿足普遜的口腹之慾呢？假設普遜走路速度很快，所以只需要考慮在街區等待行人穿越道的時間，走路的時間可以忽略。

以範例測試資料 2 為例，普遜在街區 3、要前往街區 1，若他選擇在時間 3 經由行人穿越道 3 從街區 3 到街區 4、在時間 6 經由行人穿越道 4 從街區 4 到街區 1，那他在時間 6 才能到達會場；若他選擇在時間 4 經由行人穿越道 2 從街區 3 到街區 2、在時間 5 經由行人穿越道 1 從街區 2 到街區 1，那他在時間 5 就能到達會場。

Input

測試資料共包含 3 行。

第一行包含一個正整數 N，表示叉路、街區、行人穿越道的數量。

第二行包含 N 個正整數，依序表示時間 1 到時間 N（後續將會循環，如題敘所述）可通行的行人穿越道，這 N 個數字必定會是 1 到 N 的一組排列，亦即 1 到 N 都會恰出現一次。

第三行包含兩個正整數 A, B，分別表示普遜所在街區的編號與 NPSC 會場所在街區的編號。

- $4 \leq N \leq 10^6$
- 第二行的 N 個數字為 1 到 N 的一組排列
- $1 \leq A, B \leq N$
- $A \neq B$

Output

輸出一行，包含一個整數，表示普遜用最佳策略能到達 NPSC 會場的時間點。輸出該數字後請記得換行。

Sample Input 1	Sample Output 1
4 4 2 3 1 4 1	1

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

This page is intentionally left blank.

E. 愛玩的普遜

Problem ID: numbers

普遜最喜歡數字了，一堆數字照一定順序接起來的「數列」當然更讓他喜歡了。但是，傳說中最美麗的數列：「對稱數列」更是普遜追求的極致！

我們稱一個非空數列（項數大於零）$< a_i >_{i=1}^{n}$ 為「對稱數列」，若是對於所有的 k 從 1 到 n 都滿足 $a_k = a_{n+1-k}$ 。

現在普遜的好友，史冬咪，送給普遜一個「數列」來做為生日禮物。普遜為了表達對史冬咪的感謝，決定把這個「數列」經過數次操作變成一個「對稱數列」！所有操作都必須是「加入」操作，也就是在數列的某個位置加入一個**正整數**，加入後，其後方的所有數字都會往後面順移。舉例來說，$< 1, 2, 3, 4 >$ 可以經由一次「加入」操作變成 $< 1, 2, 5, 3, 4 >$、$< 1, 2, 3, 4, 514 >$ 或 $< 1126, 1, 2, 3, 4 >$，但是沒辦法變成 $< 1, 2, 4, 3 >$、$< 1, 2, 5, 4 >$ 或 $< 1, 2, 4 >$。

不難發現，能夠經由數次「加入」操作變成的「對稱數列」其實不只一個，於是普遜打算讓它變成最「好」的一個「對稱數列」。

我們比較兩個數列好壞的方法如下：

- 若兩個數列的項數不一樣多，那麼項數比較小的數列比較好。
- 若前項比較無法完成比較，則看兩個數列的第一項（即 a_1），第一項比較小的數列比較好。
- 若前項比較無法完成比較，則看兩個數列的第二項（如果存在的話），第二項比較小的數列比較好。
- 若前項比較無法完成比較，則看兩個數列的第三項（如果存在的話），第三項比較小的數列比較好。
- 以此類推比較下去，直到完成比較為止。（可以發現如果到結束都無法比較出好壞關係，表示兩個數列確實相等。）

現在給你史冬咪送給普遜的數列，請你求出普遜能藉由數次「加入」操作可以得到最「好」的「對稱數列」。

為了避免輸出過大，令計算出所求的數列為 $< b_i >_{i=1}^{m}$，請輸出 $(b_1+1) \bigoplus (b_2+2) \bigoplus \cdots (b_i+i) \bigoplus \cdots (b_m+m)$，其中 $\bigoplus$ 為位元互斥或（Bitwise XOR），即為 C/C++ 中的運算子 “^”。

例如，範例測試資料第二筆，所求的數列為 $< 1, 2, 3, 2, 1 >$，上述需輸出的值即為 $(1 +$

$1) \bigoplus (2+2) \bigoplus (3+3) \bigoplus (2+4) \bigoplus (1+5) = 0$；另外，範例測試資料第一筆中，所求的數列為 $< 4, 1, 5, 1, 4 >$，可以用同樣方法計算出輸出應為 2。

Input

測試資料共包含 2 行。

第一行包含一個正整數 n，表示史冬咪送給普遜的數列的項數。

第二行包含 n 個正整數，依序史冬咪送給普遜的數列的第一項到第 n 項。

- $1 \leq n \leq 2000$
- $1 \leq$ 史冬咪送給普遜的數列的每一項 ≤ 2000

Output

輸出一行，包含一個整數，即為題目中所要求的值。

Sample Input 1	Sample Output 1
3 5 1 4	2

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

F. 三角桌會議

Problem ID: triangle

在西元 3333 年，一個強盛的帝國從三角洲之中崛起———三角國。而統領三角國的國王就是威震三方的三角王 (King Triangle)，三角王，人如其名，是一個熱愛並且精通各種與三角相關事物的翹楚。

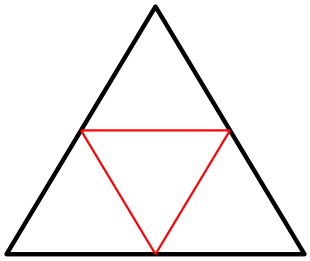
而三角王能穩定統領強盛且壯大的三角國，原因在於：建國之初，由於三角國領土十分龐大，三角王並不能有效的掌握整個國土，因此，他找來三個跟隨他已久且協助他建國的武士，並把三角國的國土一分爲三，讓每個武士各自統領一塊領地，而由三角王在中央協調與統合。

爲了隨時瞭解各領地內部及彼此之間的狀況，三角王每個月都會召開三角桌會議，將三位武士召來，彼此溝通交換訊息。而在三角桌會議中，三位武士會坐在一個三角桌邊，詳細來說，這個三角桌由上而下看是一個三角形，而三位武士分別會坐在三角形其中一個邊的中點上。此處，我們可以把每位武士視爲一個點。

而三角王認爲三角桌會議中武士 A 與武士 B 溝通的困難度會與彼此之間的距離相關。也就是說如果武士 A 與武士 B 距離 X 則武士 A 與武士 B 溝通的困難度則爲 $2X$ 。而整個三角桌會議溝通的困難度即爲兩兩溝通困難度的和，也就是第一位武士與第二位武士溝通的困難度加上第一位武士與第三位武士溝通的困難度再加上第二位武士與第三位武士溝通的困難度。

若是三角桌會議溝通的困難度太高，會使會議冗長且沒有效率，因此，三角王想知道給定一張三角桌下，三角桌會議溝通的困難度是多少。

以下爲一個範例（同時爲第一筆範例測試資料），三角形的三邊長皆爲 1 ，而三位武士分別坐在三個邊的中點上，紅色線條爲武士兩兩之間的距離：



Input

測試資料只有一行，包含三個整數 A, B, C，代表三角桌的三個邊長。

- $1 \leq A, B, C \leq 1000$
- 保證給定的三個邊長能構成合法三角形（面積大於 0）

Output

輸出一行，包含一個整數，代表三角桌會議溝通的困難程度。

Sample Input 1	Sample Output 1
1 1 1	3

G. Special Judge

Problem ID: judge

你有比過網際網路程式設計大賽，俗稱 NPSC 的比賽嗎？

啊不對，你已經在比了。

有時候 NPSC 會有一些題目的答案不只一組，所以常常會需要寫個程式來幫忙判斷。例如，讓浮點數的評分可以容許小的計算誤差，而不是純粹的字串比對，這樣我們可以讓 0.9999999999999 當作 1.0。或是在有些比賽中，對於一些分隔字元的使用採取較於不嚴格的標準，例如說題目規定要用空白字元隔開，但參賽者寫成用換行字元隔開，那也算對。可惜，NPSC 採嚴格標準。為簡化題目，我們定義分隔字元只有換行跟空白兩種，詳細字元請參考 Input 的說明。

為了方便我們明年有這種評分系統，現在要請你寫一個。

一般來說，我們會先用分隔字元把所有參賽者的輸出切開，這樣會變成一些文字片段。舉例來說 “a bb c” 會被切成 “a”, “bb”, “c”，三組字串。接著把裁判的答案也用一樣的方式切開。如此一來，我們會得到兩個字串陣列，再把這兩個字串陣列一一對應的字串拿出來比對即可。

接下來，我們要來實作比對的方式，如果發現浮點數要特別處理比對。首先我們先定義一些名詞：

- 數字：字元 ‘0’ 到 ‘9’ 之一。
- 數字字串：只有數字的字串，不能是空字串。
- 浮點數：開頭可以帶有一個正負號，再由數字字串組成。並可以帶有一個小數點 ‘.‘，小數點可以位於這串數字的任何位置（包含最前面或最後面），並把這串數字切成兩段（第一段或第二段可能爲空字串）。
  緊接著也可以接上科學記號的部分。科學記號的部分會是由一個字元 ‘e’ 或 ‘E’ 開始，接著可以接上一個正負號，再接上一個數字字串。
  舉例來說 “.2”, “1.”, “+10.e-10” 都是浮點數。
- 整數：開頭可以帶有一個正負號，再接上數字字串組成。

為簡化題目，我們先不考慮 C99/C11 有定義的 16 進位整數跟 16 進位浮點數，以及 INF, NaN 等特殊數值。

當比對兩個字串時，如果裁判方的字串是浮點數但非整數，且參賽者方的答案是整數或浮點數時，則採用浮點數判斷。在其他情況都採純粹的字串比對。至於實際浮點數誤差的部分，由於有太多邊界情況，而且你應該會抱怨你已經寫太多程式碼了，所以我們就簡化掉吧。

你的程式在解析後，需要對每一對字串依序輸出你得到的是哪兩個字串，以及他們適用哪一種比對法（字串比對或是浮點數比對）。如果切出來的字串數量不相同，你則需要在輸出到該對的時候，在缺少的那一邊做標示。詳細的輸出請參考 Output 的說明。

Input

測試資料的第一行會有兩個整數 N, M，分別表示參賽者的輸出有幾行，以及裁判的輸出有幾行。

接下來 N 行是參賽者的輸出，緊接著 M 行是裁判的輸出。

- 保證參賽者跟裁判的輸出**分別**不超過 2000 個字元 (包含換行字元)
- 保證所有字元只會有以下幾種：換行字元 '\n' (ASCII 值 10)、空白字元 ' ' (ASCII 值 32)、正號 '+' (ASCII 值 43)、負號 '-' (ASCII 值 45)、小數點 '.' (ASCII 值 46)、數字 '0' 到 '9' 以及英文大小寫字母
- 為簡化題目，保證參賽者跟裁判的輸出最後一個字元都會是換行字元

Output

假設參賽者的輸出切出 n 個字串 $a_1, a_2, \ldots, a_n$，而裁判的輸出切出 m 個字串 $b_1, b_2, \ldots, b_m$，則總共輸出 $\max(n, m)$ 行。

第 i 行輸出 a_i 跟 b_i 以及比對的方式，之間都用一個空白隔開。如果是用字串比對則輸出 "`str`" (不含引號)，如果是用浮點數比對則輸出 "`float`" (不含引號)。

如果 a_i 或 b_i 有缺少，則缺少的那方輸出 "`<missing>`" (不含引號)。另外，如果有任何一方缺少，則一定使用字串比對法。

Sample Input 1

```
3 15
IamStr
1 2  3  4 5 6   7  8  9  10 11
12           13          14 15
0.0
1.1
+2.2
-3.
+.4
+5.e+5
-.6E-6
StringBegin
8
+
.
-.
+.e+12
.e-13
+14.14e14.14
```

Sample Output 1

```
IamStr 0.0 str
1 1.1 float
2 +2.2 float
3 -3. float
4 +.4 float
5 +5.e+5 float
6 -.6E-6 float
7 StringBegin str
8 8 str
9 + str
10 . str
11 -. str
12 +.e+12 str
13 .e-13 str
14 +14.14e14.14 str
15 <missing> str
```

This page is intentionally left blank.