2014 網際網路程式設計全國大賽
高中組初賽

- 本次比賽共 7 題，含本封面共 24 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。
- 每一題的執行時間限制如下表所示。
  執行期間該電腦不會有別的動作，也不會使用鍵盤或滑鼠。

	題目名稱	執行時間限制
題目 **A**	禹鍺的鞋櫃	2 秒
題目 **B**	寧寧數貓咪	1 秒
題目 **C**	寧寧潤髮尾	2 秒
題目 **D**	小小郭的密碼	2 秒
題目 **E**	Dynamic Programming	3 秒
題目 **F**	校園偶像計劃	2 秒
題目 **G**	華麗內餡格狀超好吃巧克力	3 秒

2014 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("%I64d %I64d", &a, &b);
        printf("%I64d\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;
}
```

題目 A
禹錡的鞋櫃

執行時間限制: 2 秒

禹錡有句名言：「無論再怎麼難看的款式，仍是會有人喜歡，所以不怕賣不出去。」

禹錡是一個鞋店老闆，他有一個兩層的櫃子，上層放著一排高級鞋，下層放著一排山寨鞋。為了簡化問題，每一雙高級鞋都可以用一個正整數來表示它的種類，而山寨鞋是仿冒高級鞋而成，所以每一雙山寨鞋會各別與某一雙高級鞋同種類。

禹錡是一個很有愛心的人，他想要捐給慈善團體一些高級鞋與山寨鞋，雖然他也有順便清理鞋櫃的意思。但在捐鞋之餘，他有個特殊的要求，他希望在捐完鞋子後，高級鞋與山寨鞋的數量相同。而且在不改變剩餘鞋子的順序下重新排好後，第 i 雙高級鞋與第 i 雙山寨鞋是同種類。

現在禹錡遇到了一個問題，他不知道該捐哪些高級鞋，因此禹錡求助於你，請你找出最小的 K，使得禹錡選**任意**的 K 雙高級鞋捐出去都至少有一種相應的山寨鞋捐法使他達成目的。

■ 輸入說明

輸入的第一行有一個正整數 T，代表測試資料的筆數。

每一筆測試資料的第一行有兩個以隔開正整數 N, M。第二行有 N 個正整數 A_i 表示高級鞋。第三行有 M 個正整數 B_i 表示山寨鞋。

- $T \leq 70$
- $2 \leq N \leq 1000, 1 \leq M \leq 100000$
- 每雙鞋子的種類為一個不超過 100000 的正整數。

■ 輸出說明

對於每一筆測試資料若禹錡可以達到條件，請輸出一個數字表示最小的 K 值。

■ 範例輸入

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

■ 範例輸出

```
1
2
4
1
```

題目 B
寧寧數貓咪

執行時間限制: 1 秒

住在喵星上的寧寧是個十足的貓咪愛好者，為了能夠每天環繞在貓咪之中，她在家裡養了非常多的貓咪。由於貓咪的數量龐大，縱使飼料批發商已經以非常低廉的價錢賣飼料給寧寧了，貓咪飼料的開銷仍依然龐大。為了避免浪費太多錢，寧寧每隔一段時間便會統計究竟有幾隻貓咪在她家中，以控制飼料的量。

然而，隨著貓口膨脹，寧寧漸漸無法計算究竟有幾隻貓咪在她家中。為此，寧寧發明了一種新方法來計算貓咪的數量，稱之為「寧寧點貓」。寧寧點貓的方法很簡單。寧寧首先選擇 N 相異的質數 $P_1, P_2, P_3, \ldots, P_N$ 。接著寧寧會對貓咪們下 N 次指示，第 i 次時寧寧會要求貓咪們每 P_i 隻組成一團，最後落單找不到組的貓咪便來找寧寧。

如此經過 N 次的指示之後，寧寧便能知道貓咪的數量除以這些質數的餘數，並可以逆推出貓咪的數量了！當然，不論質數怎麼選，可能的貓咪數量依然會有無限種，但寧寧相信貓咪的數量一定是這所有可能中最少的那個！

發明了這個方法之後，寧寧便迫不及待的對貓咪們下了指示，但她很快的就發現了一個問題：她不知道該怎麼逆推出貓咪的數量！由於明天就要買飼料了，寧寧沒時間從頭數過，她找上傳說中數學非常好的你。究竟，你能不能幫助寧寧算出貓咪的數量呢？

■ 輸入說明

輸入的第一行有一個正整數 T 代表測試資料的筆數，接下來的每一行代表一筆測試資料。

每一筆測試資料的開頭有一個正整數 N 代表寧寧選了幾個質數。接下來跟著 $2N$ 個整數，奇數項是寧寧所選的質數 P_i，而其對應的偶數項 R_i 便是最後來找寧寧的貓咪數量。

- $T \leq 1000$
- $N \leq 1000$
- $2 \leq P_i \leq 1000$
- $0 \leq R_i < P_i$

■ 輸出說明

對於每一筆測試資料，請輸出貓咪的數量到底是多少。

如果貓咪的數量大於等於 955049953 隻，請輸出 -1 來告訴寧寧該少養些貓咪了。

■ 範例輸入

```
3
2 3 1 5 4
3 5 1 7 1 11 1
10 2 0 3 2 5 4 7 5 11 1 13 1 17 1 19 1 23 1 29 1
```

■ 範例輸出

```
4
1
-1
```

■ 範例說明

第三筆測試資料中，貓咪的數量為 955049954 ，故必須輸出 -1 。

題目 C
寧寧潤髮尾

執行時間限制: 2 秒

住在喵星上的寧寧是個十足的貓咪愛好者，為了能夠每天環繞在貓咪之中，她在家裡養了非常多的貓咪。

由於寧寧養的貓咪實在太多了，最近常常有朋友跟鄰居向寧寧建議少養點貓。這讓既愛貓咪又認為朋友有理的寧寧心情相當浮躁，髮尾也跟著分岔。

為了讓寧寧能順心一點，芷芷決定用獨門研製的潤髮乳來幫寧寧潤髮。這個潤髮乳使用的方式非常特別，如果你晚上洗澡後在某個分岔點塗上潤髮乳，那麼分岔的前 1mm 便會黏起來。而若某個分岔，岔出的長度並不到 1mm ，那麼他們就會整個黏起來。另外因為芷芷的手非常巧，所以幫寧寧使用潤髮乳的時候芷芷可以任意的選擇他想要在哪些分岔點上使用。

不幸的，獨門潤髮乳只能當天製作當天使用，且一天只能使用一次 (但可以同時用在很多分岔點上) ，否則潤髮效果就會大減 (只有 $1\mu\text{m}$ 的效果)。而獨門潤髮乳的製程又非常複雜，所以如果芷芷當天想幫寧寧潤髮，便整天都讀不到書了！

為了能估算自己之後究竟要補多少進度，芷芷想要知道，若使用最佳的策略幫助寧寧潤髮，究竟需要花幾天才能讓所有分岔黏合呢？此外，由於想要知道自己的潤髮乳效果到底如何，芷芷還想要知道在最佳策略下每天寧寧的頭髮會有多少分岔。

現在，芷芷已經將寧寧的頭髮用 1mm 的長度間隔表達成一棵二元樹。但不諳數學的芷芷顯然無法自己找出最佳策略。你能夠幫助芷芷嗎？

■ 輸入說明

輸入的第一行有一個正整數 T ，代表測試資料的筆數。

每筆測試資料的第一行包含一個正整數 N，代表那棵表示寧寧頭髮的樹有幾個節點，從 0 編號到 $N-1$ 。

下一行包含了 $N-1$ 個整數，第 i 個數字 F_i 代表編號 i 的節點接在編號 F_i 的節點下面。

- $T \leq 20$
- $3 \leq N \leq 200000$
- $0 \leq F_i < i$
- 保證每個節點至多只會有兩個子節點。

■ 輸出說明

對每筆測試資料請輸出一行。

若總共需要 A_i 天才能讓寧寧的頭髮完全沒有分岔，請輸出 $A_i + 1$ 個用空格隔開的整數，分別代表第 0 天至第 A_i 天時寧寧頭髮的分岔數量。(所以最後一個數字一定是 0) 若有多種策略的天數相同，請使用讓答案序列字典序最小的那組。(從第一個數字開始往後比較，找到第一個不同的數字，數值較小的字典序便比較小)

■ 範例輸入

```
2
6
0 0 1 2 2
3
0 1
```

■ 範例輸出

```
2 1 0
0
```

■ 範例說明

1. 在第一筆範例中，第一天會有一個分岔碰到結尾而消失，另一個則向下黏合，第二天時剩下的分岔也黏合了。以下分別是第零天、第一天、第二天時寧寧髮尾的狀況。

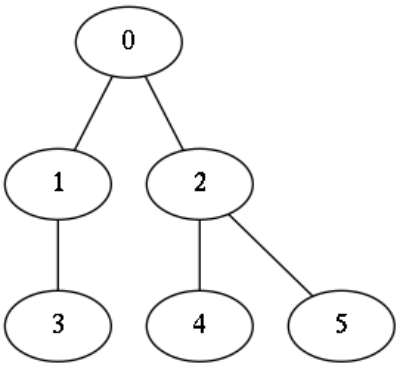

Figure 1: 第零天

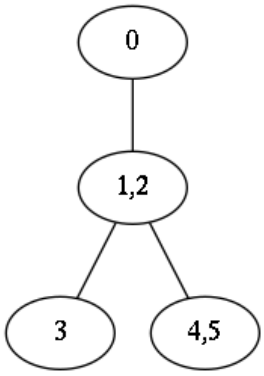

Figure 2: 第一天

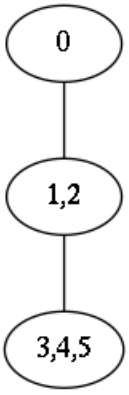

Figure 3: 第二天

2. 在第二筆範例中，寧寧的頭髮根本沒有分岔，所以第 0 天時的分岔數便是 0 。

本頁留白。

題目 D
小小郭的密碼

執行時間限制: 2 秒

註冊網站的時候，想一個好密碼總是個痛苦的過程。太簡單的密碼怕被破解，太困難的密碼怕記不住，用以前用過的密碼則怕這個網站的資料外洩，獨特的密碼又不知道該取什麼才好。

身為一隻聰明又懶惰的小小郭，他決定選個數字當成密碼就好，因為他每次使用特殊字元當密碼，都會落得記不起來的下場，令他十分難過。為了避免忘記密碼，聰明小小郭的密碼會是至少兩個連續正整數 $x, x+1, x+2, \ldots, x+n$ 的乘積，這樣他就只要記住小小的 x 跟 n 他就可以算出他大大的密碼 $x \times (x+1) \times (x+2) \times \cdots \times (x+n)$。例如 $56 = 7 \times 8$ 和 $210 = 5 \times 6 \times 7$ 都有可能是他的密碼，而 514 和 120908 因為沒有辦法寫成至少兩個連續正整數的乘積，所以不可能是他的密碼。

小小郭覺得小於 l 的密碼太簡單了，大於 r 的密碼太困難了，可以請你幫他算看看在 l 到 r 之間他有幾種可用的密碼嗎？

■ 輸入說明

輸入的第一行有一個正整數 T，代表測試資料的筆數。

每一筆測試資料為一行兩個整數 l, r，代表小小郭覺得適合的密碼範圍。

- $1 \leq T \leq 10000$
- $1 \leq l \leq r \leq 10^{18}$

■ 輸出說明

對於每一筆測試資料請輸出一行，包含一個整數，表示在 l 到 r 之間他有幾種密碼可以用。

■ 範例輸入

```
2
1 10
5 30
```

■ 範例輸出

```
2
5
```

■ 範例說明

- 第一筆測試資料中合適的密碼有 $2, 6$。
- 第二筆測試資料中合適的密碼有 $6, 12, 20, 24, 30$。

題目 E
Dynamic Programming

執行時間限制: 3 秒

古可魚語被發現後已經過了一年。可魚國的考古學家們雖然成功的將其翻譯為**現代可魚語**，但因為現代可魚語太複雜，無法直接使用電腦分析*魚法結構*。因此他們求助於你，希望你能將其翻譯成**簡化可魚語**。

以下分別定義現代可魚語、簡化可魚語的魚法。其中「$\langle name \rangle$」定義了魚法中「$name$」類型的 non-terminal（中間變數），它總是需要被其他定義換掉；「abc」這種字型的字則代表普通的字串；$[defn]^*$ 代表 $defn$ 這個 * 定義重複**零次或多次**（如 $[,\ \langle variable \rangle]^*$ 代表「」（空字串）、「, $\langle variable \rangle$」、「, $\langle variable \rangle$, $\langle variable \rangle$」、...）。

1. $\langle constant \rangle$ 是十進位整數，$\langle variable \rangle$ 是 `%`n（$n = 0, 1, 2, \ldots$）或任意 C 識別字（identifier）如 `%0`、`%514`、`npsc2014`，但**不能**是 `let`, `in`, `if`, `then`, `else`, `fn`。
2. 在現代可魚語只有 $\langle expr_0 \rangle$ 一種 non-terminal，而一個現代可魚語句子是一個符合 $\langle expr_0 \rangle$ 魚法的字串。$\langle expr_0 \rangle$ 可以被以下任何一種取代：
   - $\langle constant \rangle$、$\langle variable \rangle$、`(` $\langle expr_0 \rangle$ `)` 或 `fn (` $\langle variable \rangle$ `[,` $\langle variable \rangle$`]*` `) =>` $\langle expr_0 \rangle$
   - `let` $\langle variable \rangle$ `=` $\langle expr_0 \rangle$ `in` $\langle expr_0 \rangle$
   - `if` $\langle expr_0 \rangle$ `then` $\langle expr_0 \rangle$ `else` $\langle expr_0 \rangle$
   - $\langle expr_0 \rangle$ `(` $\langle expr_0 \rangle$ `[,` $\langle expr_0 \rangle$`]*` `)`

   以上的定義中，`fn (...) => ...` 代表「建立一個函數」，就像 javascript 的 `function(...){...}` 構造。詳細執行方法之後會以**可魚直譯器**解釋。
3. 簡化可魚語則有 $\langle expr_1 \rangle$ 和 $\langle value_1 \rangle$ 兩種 non-terminal。一個簡化可魚語句子是一個符合 $\langle expr_1 \rangle$ 魚法的字串，而 $\langle expr_1 \rangle$ 可以是
   - $\langle value_1 \rangle$
   - $\langle variable \rangle$ `(` $\langle value_1 \rangle$ `[,` $\langle value_1 \rangle$`]*` `)`
   - `let` $\langle variable \rangle$ `=` `fn (` $\langle variable \rangle$ `[,` $\langle variable \rangle$`]*` `) =>` $\langle expr_1 \rangle$ `in` $\langle expr_1 \rangle$
   - `let` $\langle variable \rangle$ `=` $\langle variable \rangle$ `(` $\langle value_1 \rangle$ `[,` $\langle value_1 \rangle$`]*` `)` `in` $\langle expr_1 \rangle$
   - `if` $\langle value_1 \rangle$ `then` $\langle expr_1 \rangle$ `else` $\langle expr_1 \rangle$

   $\langle value_1 \rangle$ 可以是 $\langle constant \rangle$、$\langle variable \rangle$ 或「`fn (` $\langle variable \rangle$ `[,` $\langle variable \rangle$`]*` `) =>` $\langle expr_1 \rangle$」

在現代可魚語中，$\langle expr_0 \rangle$ 的長度延伸到盡量遠處，所以 `fn (w)=> w(0)(1)` 等同 `(fn (w)=> (w(0)(1)))`。當然，簡化可魚語句子都是現代可魚語句子。翻譯過程中，對任意未知的集合 S^*，我們必須保證翻譯前與翻譯後的句子以**可魚直譯器**解釋的結果永遠一樣：

```
set insert(S_new, S) {
    shadow ← { (var,v) | ∀var,v,w s.t. (var,v) ∈ S ∧ (var,w) ∈ S_new }
    return (S \ shadow) ∪ S_new
}
// S, S_e, S* ⊆ variable × value
value translate(expr) { return translateHelper(expr,some fixed S*); }
value translateHelper(expr, S) {
    switch (expr) {
    case( c ): return c; // if c is a ⟨constant⟩
    case( var ): return v if (var,v) ∈ S // if var is a ⟨variable⟩
    case( (e) ): return translateHelper(e, S);
    case( fn (x1,x2,...,xk) => e ): return ⟨S,e,x1,x2,...,xk⟩;
    case( display(e) ): return printf(translateHelper(e, S));
    case( let var = e1 in e2 ):
        v ← translateHelper(e1, S);
        return translateHelper(e2, insert({(var,v)}, S));
    case( if e1 then e2 else e3 ):
        v ← translateHelper(e1, S);
        return translateHelper(v? e2 : e3, S);
    case( e0(e1,e2,...,ek) ):
        vk ← translateHelper(ek, S);
        ...
        v2 ← translateHelper(e2, S);
        v1 ← translateHelper(e1, S);
        ⟨Se,e,x1,x2,...,xk⟩ ← translateHelper(e0, S);
        return translateHelper(e, insert({(x1,v1),(x2,v2),...,(xk,vk)}, Se);
    }
}
```

為了確保翻譯的品質，考古學家們決定限制只能使用以下四種重寫規則，且規則 1、2、3 總是必須優先於規則 4 使用。

1. 一 *expr* 語句依其前後語句的條件（敘述於底下）及種類，在不影響解釋結果的前提下，可以做如下數種變換：

```
    ... // part A
    ... e0(e1,...,ei,...,ek) ... // part B
⇒ //  ( 0 ≤ i ≤ k 、 part B 不在「 fn (...) => 」中、前面沒有 let )
    ... // part A
    let %m = ei in ... e0(e1,...,%m,...,ek) ... // part B
```

```
    ... // part A
    ... (fn (x1,...,xk) => e0)(v1,...,vk) ... // part B
⇒ //  ( vi 都是 ⟨value1⟩ ，其餘條件同上)
    ... // part A
    let %m = fn (x1,...,xk) => e0 in ... %m(v1,...,vk) ... // part B
```

```
    ... // part A
    ... if e1 then e2 else e3 ... // part B
⇒ //  (其餘條件同上)
    ... // part A
    let %m = e1 in ... if %m then e2 else e3 ... // part B
```

「不在「fn (...) =>」中、前面沒有 let」意味以下都不可行，即使 S* 可能無法分辨

```
   (fn (x) => let z = h(w) in z)(0)
⇒ let %0 = let z = h(w) in z in (fn(x) => %0)(0)
```

```
   let k = fn (x) => x in f(k,g(z))
⇒ let %0 = g(z) in let k = fn(x) => x in f(k,%0)
```

2. 在不影響解釋結果的前提下，調整任意「let *var* = ⟨*expr*$_0$⟩ in」語句的**內外** 嵌套順序，如

```
    let x = let y = let z = w in z in y in x
⇒
    let x = let z = w in let y = z in y in x
```

3. 對於「let *var* = ⟨*constant*⟩ in *e*」或「let *var* = ⟨*variable*⟩ in *e*」語句，我們可以把 *e* 中**所有的** *var* 都替換成等號右邊的值，例如經過兩次變換後，

```
    let x = %1 in let w = h(x) in let z = 7 in mul(x,w,z)
⇒
    let w = h(%1) in mul(%1,w,7)
```

4. 若一「if e_1 then e_2 else e_3」語句不是在 ⟨$expr_0$⟩ 最尾端的位置，在不影響解釋結果的前提下，可以做如下的變換

```
    ... // part A
    ... if e1 then e2 else e3 ... // part B
⇒ //  (因為 if 不在最尾巴， part B 後面一定有 ... )
    ... // part A
    let %n = fn (%m) => ... %m ... // part B
                            ... in // part C
    if e1 then %n(e2) else %n(e3)
```

注意在這個變換中，「//part C」總是必須把所有外面的語句都抓進來。所以假如

```
    f( let flag = if x then y else z in g(flag) )
```

使用規則 4 重寫的話，結果是

```
    let %0 = fn (%1) => f( let flag = %1 in g(flag) ) in
    if x then %0(y) else %0(z)
```

而**不是**

```
    f( let %0 = fn (%1) => let flag = %1 in g(flag)
       in       if x then %0(y) else %0(z) )
```

■ 輸入說明

輸入的第一行有一個正整數 T，代表測試資料的筆數。

每一筆測試資料為一句現代可魚語句子，輸入皆以空白或換行隔開，且後面都有單獨一行「`-----`」。排版僅供參考。

- $T \leq 30$
- 保證輸入的句子都正確。
- 字串/符號個數不超過 21000 個。
- 變數名稱長度不超過 31 個字元，且不會重複，且一定不是 `%` 開頭。

■ 輸出說明

對於每筆測試資料請輸出翻譯後的簡化可魚語句子，並在其後輸出一行「`=====`」。翻譯過程中所新增的 $\%n$ 變數不可重複，且編號由上到下必須依序是 $0, 1, \ldots$。

遇到「`let` x `=` e_1 `in` e_2」語句時，必須在「`in`」之後換行；遇到「`fn (...) =>` e」語句時，若 e 是 `let ...` 或 `if ...`，則必須在「`=>`」之後換行，並於輸出 e 時內縮兩個空格；遇到「`if` e_1 `then` e_2 `else` e_3」時，必須在 `then` 後換行、`else` 前後換行，且輸出 e_2, e_3 時都內縮兩個空格。除此之外不可換行，且字串、符號之間都需空一個空白。

■ 範例輸入

```
2
let x =
  let z = 514 in
  let w = ( let v = 31 in add1 ( v , z , y ) ) in
  w in
( fn ( u ) => u ( u ( u ) ) ) ( fn ( k ) => k ) ( sub1 ( x ) )
-----
let x = pow ( 2 , n ) in
let v = if greater ( x , 8 ) then false else true in
if v then 7 else 29
-----
```

■ 範例輸出

```
let w = add1 ( 31 , 514 , y ) in
let %0 = sub1 ( w ) in
let %1 = fn ( u ) =>
  let %2 = u ( u ) in
  u ( %2 ) in
let %3 = %1 ( fn ( k ) => k ) in
%3 ( %0 )
=====
let x = pow ( 2 , n ) in
let %0 = greater ( x , 8 ) in
let %1 = fn ( %2 ) =>
  if %2 then
    7
  else
    29 in
if %0 then
  %1 ( false )
else
  %1 ( true )
=====
```

題目 F
校園偶像計劃

執行時間限制: 2 秒

校園偶像 (school idols) 最近在各地都蔚為風潮，在穗香學園也不例外。一群對流行偶像文化感興趣的學生們組成了「偶像研究社」，並且創立了一個校園偶像團體。就和籃球社、棒球社等等社團附屬的社隊一樣，參與偶像研究社並創立校園偶像團體的學生們，不一定就是未來想要從事相關行業的人，只是因為有充分的熱誠和十足的興趣，想要在自己喜歡的領域展現自己青春的一面 ——校園偶像也就是因此而受到人們關注。

最近，為了響應這類校園偶像的活動，一個名為「我愛偶像」的比賽將在幾個月後舉辦。「我愛偶像」是一個校園偶像團體的全國大賽，只要是高中的校園偶像團體都可以參加。比賽首先會舉辦地區預賽，各區的前兩名隊伍則可以晉級最後的全國決賽。

熱衷於流行偶像文化的穗香學園偶像研究社成員們當然也要參與這場盛會。緊張又期待的心情可比棒球社的成員參加日本高中的夏季甲子園。由於種種原因，大家決定要從社員中選出九個人，並以九人一組的方式演出 (九個人登台似乎是現在的流行趨勢)。但也因為這項決定，使得要選擇哪些社員參賽便成了一個大問題。

根據分析，目前校園偶像們的吸引力大致上是由「笑容 (smile)」、「純真 (pure)」和「酷炫 (cool)」三個面向組成的。如果越想要吸引觀眾和專業評審們的目光，就越得要加強這三大方面的表現。(當然，這些只是表現的技巧，實質的功力如歌唱舞蹈等等絕對也是必要的。) 另外，為了簡化問題，我們假設對於每一首歌曲，也都可以將它們分成「笑容」、「純真」和「酷炫」這三類。並且，我們簡單地定義一首歌的「整體表現」為「所有表演的成員對於該首歌類型的能力值之平均」。舉例而言，假設有一首歌屬於「笑容」類別，則該首歌的整體表現則以參加的九人的「笑容」能力值之平均值計算。隊伍的表演總分則是以每首歌的整體表現加總之。

在大家做完能力評估後，我們已經知道了穗香學園偶像研究社所有社員的三項能力值。表演的歌曲也已決定是以數首自行創作的歌曲參加，故所有屆時要演唱的歌曲的類型都已經知道了。於是給定大家的能力和歌曲類型，請你幫幫眾人找出最佳的九人組合。

■ 輸入檔說明

輸入的第一行有一個正整數 T，代表測試資料的筆數。

每一筆測試資料的第一行共有兩個正整數 N, M，代表穗香學園偶像社的成員人數。接下來有 N 行，每一行都有四個非負實數 u_i, s_i, p_i, c_i 分別代表其中一個社員的學號及其三項能力值：「笑容」、「純真」和「酷炫」。所有的實數都精確到小數點下第三位。每筆測試資料的最後一行共有 M 的字元，表達參賽的歌曲類別。每一個 s 都對應到一首「笑容」類的歌曲；每一個 p 都對應到一首「純真」類的歌曲；每一個 c 都對應到一首「酷炫」類的歌曲。

輸入檔所有於同一行的欄位內容都將以恰一個逗點「,」(不含引號) 分開。另外，輸入的成員資料將會按照學號從小到大的順序逐行呈現。

- $T \leq 50$
- $9 \leq N \leq 25$
- $1 \leq M \leq 10$
- $0 \leq s_i, p_i, c_i \leq 5000$
- 社員們的學號 u_i 恰為 1 到 N 之正整數

■ 輸出檔說明

對於每一筆測試資料，請輸出兩行。

第一行請輸出最佳選擇可以得到的表演總分 (即可能的最高表演總分)，請將數值四捨五入輸出至小數點下第三位。

第二行請輸出最佳選擇之九位成員的學號，並以恰一個逗點「,」(不含引號) 分開。成員的輸出順序請依照其學號從小到大輸出。

各筆測試資料所對應到的最佳解答都保證只有一組。

(注意：請不要在任何地方自行加上額外空白或其他字元。)

■ 範例輸入

```
2
10,3
1,2000.000,5000.000,3000.000
2,4000.000,2000.000,2000.000
3,1000.000,5000.000,3000.000
4,3000.000,2000.000,3000.000
5,2000.000,1000.000,5000.000
6,1000.000,3000.000,1000.000
7,1000.000,5000.000,3000.000
8,5000.000,3000.000,5000.000
9,4000.000,4000.000,4000.000
10,5000.000,1000.000,2000.000
s,p,c
12,5
1,796.501,13.025,994.582
2,676.916,409.636,561.610
3,848.811,443.469,586.526
4,221.629,653.935,608.985
5,589.992,555.418,413.737
6,539.929,334.949,46.011
7,547.726,103.542,559.104
8,934.054,673.369,373.407
9,860.740,995.345,3.964
10,553.694,620.526,438.781
11,48.880,503.052,638.534
12,213.942,398.149,963.539
s,p,s,c,c
```

■ 範例輸出

```
9444.444
1,2,3,4,5,7,8,9,10
2894.192
1,2,3,5,7,8,9,10,12
```

本頁留白。

題目 G
華麗內餡格狀超好吃巧克力

執行時間限制: 3 秒

小小郭不喜歡吃巧克力，但是因為他實在太夯了，每年情人節都會收到巨量巧克力，這讓他非常困擾。他只好想辦法把收到的巧克力分送給親朋好友們。

現在小小郭想要分送一片大小為 $n \times m$ 的華麗內餡格狀超好吃巧克力，而小小郭的好朋友胖胖天獲得先行選擇的權利。因為巧克力實在太多了，小小郭希望胖胖天可以從裡面拿走至少兩格巧克力，而且拿走的部分必需是一個長方形。每格巧克力因為內餡不同，會各自有一個好吃度。現在胖胖天想要使得他選擇的部分平均好吃度最大，請你幫幫他。

■ 輸入說明

輸入的第一行有一個正整數 T，代表測試資料的筆數。

每一筆測試資料開始有一行兩個整數 n, m，代表華麗內餡格狀超好吃巧克力的大小。接下來 n 行，每行有 m 個整數 $a_{i,j}$ 代表每格的好吃度。

- $1 \leq T \leq 1000$
- $1 \leq n, m \leq 500$
- $n \times m \geq 2$
- $1 \leq a_{i,j} \leq 10^8$
- 至少 99% 的測試資料中 $1 \leq n, m \leq 100$

■ 輸出說明

對於每一筆測試資料請輸出一行，包含一個分數 p/q 表示最大的平均好吃度，其中 p, q 皆為正整數且 $\gcd(p, q) = 1$。

■ 範例輸入

```
2
3 3
8 1 6
3 5 7
4 9 2
4 4
1 15 14 4
12 6 7 9
8 10 11 5
13 3 2 16
```

■ 範例輸出

```
7/1
29/2
```

■ 範例說明

- 第一筆測試資料中最好的是由好吃度 $5, 9$ 的格子組成的長方形
- 第二筆測試資料中最好的是由好吃度 $15, 14$ 的格子組成的長方形