2014 網際網路程式設計全國大賽
國中組, 高中組模擬測試

- 本次比賽共 6 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，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。
- 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。

表 1: 題目資訊

	題目名稱	執行時間限制
題目 **A**	北極熊大遷徙	1 秒
題目 **B**	南極企鵝大遷徙	1 秒
題目 **C**	新 · 烤餅乾	2 秒
題目 **D**	棒球練習場	1 秒
題目 **E**	捷運路線	2 秒
題目 **F**	古可魚語	30 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
北極熊大遷徙

執行時間限制: 1 秒

因為全球暖化的關係，北極各處的浮冰正在慢慢融化之中。部份北極熊所在的浮冰已經融化到不堪居住的程度，於是這些北極熊興起遷徙的念頭。

已經融化到不堪居住的浮冰 A 上有 a 隻北極熊，牠們現在打算遷徙到有 b 隻北極熊居住的浮冰 B。你要回答的是：經過北極熊大遷徙以後，浮冰 B 上總共會有多少隻北極熊。

■ 輸入說明

輸入檔第一行有一個數字 $T(T \leq 10)$ 代表總共有幾筆測試資料。

每一筆測試資料有兩個整數 a 和 b $(1 \leq a, b < 2^{31})$，代表有 a 隻北極熊即將從浮冰 A 遷徙到原本有 b 隻北極熊的浮冰 B。

■ 輸出說明

針對每一筆測試資料，輸出浮冰 B 上最後會有多少隻北極熊。

■ 範例輸入

```
3
24 47
33 20
11 30
```

■ 範例輸出

```
71
53
41
```

題目 B
南極企鵝大遷徙

執行時間限制: 1 秒

因為全球暖化的關係，南極各處的浮冰正在慢慢融化之中。部份企鵝所在的浮冰已經融化到不堪居住的程度，於是這些企鵝興起遷徙的念頭。

已經融化到不堪居住的浮冰 A 上有 a 公斤的企鵝，牠們現在打算遷徙到有 b 公斤的企鵝居住的浮冰 B。你要回答的是：經過企鵝大遷徙以後，浮冰 B 上總共會有多少公斤的企鵝。

■ 輸入說明

輸入檔第一行有一個數字 $T(T \leq 10)$ 代表總共有幾筆測試資料。

每一筆測試資料有兩個浮點數 a 和 b $(1 \leq a, b \leq 50)$，代表有 a 公斤的企鵝即將從浮冰 A 遷徙到原本有 b 公斤重的企鵝的浮冰 B。

■ 輸出說明

針對每一筆測試資料，輸出浮冰 B 上最後會有多少公斤的企鵝，四捨五入至小數第二位。

■ 範例輸入

```
3
24.23 47.33
33.43 20.92
11.38 30.74
```

■ 範例輸出

```
71.56
54.35
42.12
```

題目 C
新 · 烤餅乾

執行時間限制: 2 秒

不久前舉辦的一年一度烤餅乾世界大賽，很不幸地因為大會出的題目有點複雜，導致選手都在計算自己有多少模具可以使用，而沒有足夠時間將選手們各自的烤餅乾技巧完全發揮出來，以致於烤出來的餅乾都很普通，沒有到驚為天人的美味。大會對此感到非常可惜，認為這樣就喪失了原本舉辦烤餅乾大賽的真諦，因此大會決定今年要破例，在今天舉辦今年第二場的烤餅乾世界大賽，讓各位選手能夠做出最得意的作品！

小櫻雖然在前一陣子的比賽中，因為沒有完全發揮所長而沒有得名，但是這次不一樣，對面即將到來的烤餅乾世界大賽，小櫻非常地有把握，相信自己一定可以在自己的堅強實力以及你的幫助之下，重新奪回冠軍！

這次的大會決定要做一些大更動，首先更換的是模具，這次的模具五花八門，各種形狀都有；再來賽制的部分也改為回合制，一個回合是兩個人，比賽題目當場公布，公布題目之後再挑選模具進行烤餅乾。然而由於每種模具都只有一個，因此先挑的人會有優勢，為了避免不公平以及增加比賽的刺激感，大會制訂了一個規則：

1. 兩個人先抽一個公正的六面骰，六面骰的每一面都有一個數字，兩人的六面骰的數字都不會重複 (總共會有 12 個不同的數字)
2. 擲骰子，朝上的那一面數字較大者可以獲得優先選擇模具的權利

在擲骰子前就可以看到自己以及對手的所有數字為多少，小櫻想知道她總共可以有多少種情況可以獲得優先權呢？

■ 輸入說明

輸入的第一行有一個正整數 $T(T \leq 100)$，代表測試資料的組數。

每一組測試資料有兩行，每行有六個正整數 $N_i(0 < N_i \leq 10^6)$，各以一個空白隔開，第一行代表小櫻獲得的六面骰上的六個數字；第二行代表對手獲得的六面骰上的六個數字。每一組測試資料中的數字皆不重複。

■ 輸出說明

對於每一筆測試資料請輸出一行，包含一個整數表示小櫻有幾種情況會贏。

■ 範例輸入

```
2
5 10 15 20 25 30
2 4 8 16 32 64
4 2 1 6 5 3
7 11 9 10 8 12
```

■ 範例輸出

```
20
0
```

■ 範例說明

- 第一筆測試資料，小櫻有以下幾種情況會贏：
  - 擲出 5，可贏過對手 $2, 4 \Rightarrow 2$ 種情況會贏
  - 擲出 10，可贏過對手 $2, 4, 8 \Rightarrow 3$ 種情況會贏
  - 擲出 15，可贏過對手 $2, 4, 8 \Rightarrow 3$ 種情況會贏
  - 擲出 20，可贏過對手 $2, 4, 8, 16 \Rightarrow 4$ 種情況會贏
  - 擲出 25，可贏過對手 $2, 4, 8, 16 \Rightarrow 4$ 種情況會贏
  - 擲出 30，可贏過對手 $2, 4, 8, 16 \Rightarrow 4$ 種情況會贏

  因此共 20 種情況會贏。
- 第二筆測試資料，由於小櫻所有數字皆小於對方的數字，因此 0 種情況會贏。

題目 D
棒球練習場

執行時間限制: 1 秒

公元四八八八年，由於空間傳送裝置的普及，人類對於能夠將物品傳送至不同空間，早就習以為常，並且作了許多應用。

有一天，皮皮和球球決定到「眼明手快」棒球場上進行終極棒球的決鬥。這場決鬥的規則很簡單，打到最多球的人就贏了。不過有趣的事情是，當棒球出現在某個位置 (x, y) 之後，下一秒，棒球就會被瞬間傳送到 $(x \oplus y, |x - y|)$ 的位置。其中 $\oplus$ 是把 x 和 y 寫成二進位以後進行 XOR 運算的意思。

舉個例子來說，一開始的球出現在 $(5, 3)$ 的位置，那麼過了一秒以後就會出現在 $(6, 2)$，再下一秒就會出現在 $(4, 4)$，第四秒就會在 $(0, 0)$ 了。

皮皮和球球經過了詳盡的觀察以後，發現了一件事：只要 $x = y$，下一秒球就會出現在 $(0, 0)$。於是，皮皮和球球打算在 $(0, 0)$ 這個位置守株待球，只要球一出現，就可以把它打出去！

請你幫忙算算，若現在球的初始位置 (a, b) 滿足：$1 \le a \le N$ 且 $1 \le b \le M$，那麼有哪些位置的球經過一段時間之後就會出現在 $(0, 0)$ 呢？

■ 輸入說明

輸入的第一行有一個正整數 $T(T \le 50)$，代表測試資料的組數。

每一筆測試資料，包含兩個正整數 $N, M(1 \le N, M \le 10000)$，中間以一個空白隔開。

■ 輸出說明

對於每一筆測試資料請輸出一行，包含一個整數，表示有幾球會出現在 $(0, 0)$。

■ 範例輸入

```
3
3 4
2 5
514 217
```

■ 範例輸出

```
12
10
111538
```

題目 E
捷運路線

執行時間限制: 2 秒

螞蟻們討厭環狀的東西，所以螞蟻城裡面的捷運不會出現環狀線的捷運。特別的是，螞蟻城裏面的捷運都是沿著東西或南北向前進。也就是說，每次捷運轉彎都是直接轉 90 度的直角。

秉持著一年開通一條線的原則，捷運路線網越來越複雜了。而且每新增一條路線，就會造成更多的負擔。於是，在若干年後的今天，螞蟻城的首長打算好好整治一下現在紛亂的捷運路線。

每一條捷運的路線，都會從某個起點出發，然後沿著軌道**不重複地**經過一些站，抵達終點。為什麼要不重複呢？因為如果有重複經過同一站的話，搭捷運的螞蟻們便會覺得浪費時間，尤其是在這個**沒有環狀路線**的交通路網上。

請你幫忙算一算，至少需要幾條不同的捷運路線，才能使得每一段軌道上都至少有一條捷運路線經過？

■ 輸入說明

輸入的第一行有一個正整數 $T(T \leq 100)$，代表測試資料的組數。

每一組測試資料有兩個正整數 n, m $(1 \leq n, m \leq 100)$，中間以一個空白隔開。接下來有 n 行，每一行有一個長度為 m 的字串。這些字串組成了一張螞蟻城現在的捷運路網圖。

圖上的意義如下，

- **井字號** (‘#’)，表示捷運的軌道。
- **點** (‘.’)，代表這個位置沒有東西。
- 大小寫英文字母，代表這是一個車站。可能有許多不同的車站標示著相同的英文符號。

保證對於任何一個軌道，它緊鄰的四格恰好有兩個字元是軌道或捷運站。

保證不會有兩個捷運站緊緊相鄰，且每一個捷運站周圍的四格之中，至少有一格是軌道。

■ 輸出說明

對於每一組測試資料請輸出一行，包含一個整數表示最少需要幾條不同的捷運線，才能讓任何一段軌道都有列車經過。

■ 範例輸入

```
2
4 15
......##B#....C
.....##..#.####
....##...###...
.A###..........
5 7
A###.D#
...#..#
.F.C##b
.#.#...
.##D...
```

■ 範例輸出

```
1
2
```

題目 F
古可魚語

執行時間限制: 30 秒

可魚國的考古學家們日前發現一處古代遺跡，裡面有以古可魚語書寫的眾多文物，經過語言學家老蚯的努力後，終於破譯出古可魚語的規則。但是文物實在太多了，不知何時才能將所有文物翻完，因此聰明的你想要寫個程式來幫幫他們！

古可魚語中每句話均由一對括號包起來，一句話包含若干個詞，詞跟詞之間會以至少一個空白或換行隔開。空白和換行可能會交錯出現，不過都同樣視為隔開以及排版用，沒有任何意義。

一句話形如 $(w_1\ w_2\ \cdots\ w_n)$，其中 w_1 決定了這句話的句型。每個詞可以是一句話、一個數字或一個識別字。例如 `display` 是一個識別字，`514` 是一個數字，`(display 514)` 則是一句話。

保證數字只由 `0123456789` 組成，範圍在 0 到 2^{30} 之間，且不會有前導 0；識別字的開頭必然不在 `0123456789` 之內，且只會由 ASCII 碼 33 至 39、42 至 126 組成（ASCII 40 跟 41 是括號喲）。

而一個數字翻譯之後 `......` 毫無反應，還是同一個數字；一個識別字翻譯後之結果視當下的情境而定（見如下 `define` 句型），而一句話翻譯之結果則視句型而定。

古可魚語中有下列幾種句型：

1. `(display` w`)`
   印出 w 翻譯之後的結果，保證 w 經過翻譯後是個數字，整句話翻譯後為 `0`。例如

   ```
   (display 514)
   (display (display 514))
   ```

   將會印出

   ```
   514
   514
   0
   ```

2. (begin w_1 w_2 $\cdots$ w_n)
   依序翻譯 w_1 到 w_n，整句話翻譯後為 w_n 翻譯後的結果。例如

```
(display (begin
    (display 1)
    (display 2)
    3))
```

   將會印出

```
1
2
3
```

3. (if w_1 w_2 w_3)
   如果 w_1 翻譯後的結果不為 0，則整句話翻譯後為 w_2 翻譯後的結果，且 w_3 不予翻譯；如果 w_1 翻譯後為 0，則整句話翻譯後為 w_3 翻譯後的結果，且 w_2 不予翻譯。保證 w_1 翻譯後的結果為數字。例如

```
(if (display 514)
    (display 514514)
    (display 514514514))
```

   將會印出

```
514
514514514
```

4. (define w_1 w_2)
   先將 w_2 翻譯為 v_2，之後在當前情境中將 w_1 定義為 v_2，保證 w_1 是個識別字且不在 display、begin、define、lambda、+、-、< 中，重複 define 將會覆蓋之前的定義。整句話翻譯後亦為 w_2 翻譯後的結果。
   例如

```
(if (display (define x 514))
    (display 514514)
    (display x))
(display x)
(define x 50216)
(display x)
```

   將會印出

```
514
514
514
50216
```

5. (+ w_1 w_2)
   依序翻譯 w_1、w_2 為 v_1、v_2，則整句話翻譯後為 $v_1 + v_2$，保證 v_1、v_2 為數字。
   (- w_1 w_2)
   依序翻譯 w_1、w_2 為 v_1、v_2，則整句話翻譯後為 $v_1 - v_2$，保證 v_1、v_2 為數字。
   (< w_1 w_2)
   依序翻譯 w_1、w_2 為 v_1、v_2，如果 $v_1 < v_2$ 則整句話翻譯後為 1，否則為 0，保證 v_1、v_2 為數字。
   例如

```
(if (< 514 50216)
    (display (+ 514 50216))
    (display (- 514 50216)))
```

   將會印出

```
50730
```

6. (lambda (w_1 w_2 $\cdots$ w_n) b)
   保證其中 w_1 到 w_n 均為識別字且兩兩相異。
   注意！這是困惑眾人最久的句型。整句話翻譯後為一個「句型」搭配當下的「情境」，與程式設計中的函數非常類似，若 f 為本句翻譯後的結果，則(f p_1 p_2 $\cdots$ p_n) 將依下述規則翻譯：
   (a) 依序翻譯 p_1 到 p_n，令結果為 v_1 到 v_n。
   (b) 令 b 中的情境為 E_1，其中定義 w_1 到 w_n 為 v_1 到 v_n。
   (c) 令翻譯 (lambda (w_1 w_2 $\cdots$ w_n) b) 時的情境為 E_2。
   (d) 令翻譯 (f p_1 p_2 $\cdots$ p_n) 時的情境為 E_3。
   (e) 翻譯 b，若在過程中遇到 define，則此 define 只作用在 E_1 中；若在過程中遇到識別字需要翻譯，則先嘗試由 E_1 中解讀，若失敗則再嘗試由 E_2 中解讀，若再失敗則嘗試由 E_3 中解讀。

   例如

```
(define add1 (lambda (x) (+ x 1)))
(display (add1 2))
```

   將會印出 3，而

```
(define y 1)
(define magic!! (lambda (x) (+ x (+ y z))))
(define x 100)
(define y 200)
(define z 400)
(display (magic!! 2))
```

   將會印出 403。

■ 輸入說明

輸入檔中僅有一組測試資料，為以古可魚語寫成的若干句話。保證每句話均符合古可魚語之文法。

每句話可能涵蓋一行或多行，但同一行中只會有一句話。如果遇到以 “;” (不含雙引號) 開頭的行，那是考古學家的註記，請不要翻譯。註記不會夾雜在古可魚語中。

保證所有數字在運算過程中絕對值不超過 2^{30}。過程中翻譯詞句的總次數不會超過 10^7 個。

■ 輸出說明

印出所有翻譯過程中會印出的東西，一個一行。遇到考古學家的註記的時候依原樣直接輸出即可。

■ 範例輸入

```
; 3
(display (+ 1 2))

(define a 3)
; 7
(display (+ a 4))

(define add1 (lambda (x) (+ x 1)))
; 6
(display (add1 5))

(define pair (lambda (x y) (lambda (m) (m x y))))
(define first (lambda (x) (x (lambda (a b) a))))
(define second (lambda (x) (x (lambda (a b) b))))

(define a (pair 1 2))
; 1
(display (first a))
; 2
(display (second a))

(define [] (pair 1 1))
(define empty? first)
(define : (lambda (hd tl) (pair 0 (pair hd tl))))
(define head (lambda (xs) (first (second xs))))
(define tail (lambda (xs) (second (second xs))))

(define ++
  (lambda (xs ys)
    (if (empty? xs)
```

```
      ys
      (: (head xs) (++ (tail xs) ys)))))

(define map
  (lambda (func xs)
    (if (empty? xs)
      []
      (: (func (head xs)) (map func (tail xs))))))

(define display-list (lambda (xs) (map display xs)))

(define peter (: 5 (: 0 (: 2 (: 1 (: 6 []))))))
; 5 0 2 1 6
(display-list peter)

(define filter
  (lambda (pred xs)
    (if (empty? xs)
      []
      (begin
        (define xs' (filter pred (tail xs)))
        (if (pred (head xs))
          (: (head xs) xs')
          xs')))))

(define ! (lambda (x) (if x 0 1)))
(define <= (lambda (x y) (! (< y x))))
(define > (lambda (x y) (< y x)))
(define sort
  (lambda (xs)
    (if (empty? xs)
      []
      (begin
        (define x (head xs))
        (define xs' (tail xs))
        (define <=x (filter (lambda (y) (<= y x)) xs'))
        (define  >x (filter (lambda (y) ( > y x)) xs'))
        (++ (sort <=x) (: x (sort >x)))))))
```

```
69
70 ; 0 1 2 5 6
71 (display-list (sort peter))
72
73 (define take
74   (lambda (xs)
75     (lambda (n)
76       (if n
77         (: (head xs) ((take (tail xs)) (- n 1)))
78         [])))))
79
80 ; 5 0 2
81 (display-list ((take peter) 3))
82 (define take-peter (take peter))
83 ; 5 0 2
84 (display-list (take-peter 3))
85
86 (define yin-yang
87   (lambda ()
88     (lambda (m)
89       (m 0 (pair 0 (lambda (m)
90                      (m 0 (pair 1 (yin-yang)))))))))
91
92 ; 0 1 0 1 0
93 (display-list ((take (yin-yang)) 5))
```

■ 範例輸出

```
; 3
3
; 7
7
; 6
6
; 1
1
; 2
2
; 5 0 2 1 6
5
0
2
1
6
; 0 1 2 5 6
0
1
2
5
6
; 5 0 2
5
0
2
; 5 0 2
5
0
2
; 0 1 0 1 0
0
1
0
1
0
```