九十六學年度高級中學資訊學科能力競賽決賽

上機程式設計題

作答注意事項：

一、 對考題有任何疑義，請於考試開始後二個小時之內填寫「問題單」，交付監考人員轉送命題委員提出問題，逾時不予回覆。

二、 第一題到第四題每題 15 分，第五題和第六題各 20 分，共 100 分。

三、 可選擇指定解題語言中任何一種語言解題。

四、 最後繳交編譯後之執行檔限定在 Windows XP 的命令提示字元下執行。

五、 各題執行檔檔名請設定如下：
考生編號_題號.exe
例如：101_1.exe

六、 各題原始碼檔名請設定如下：
考生編號_題號.解題語言附屬檔名
例如：101_1.c

七、 各題輸入資料檔名如下：
in_題號.txt
例如：in_1.txt

八、 各題輸入方式以讀檔方式為之，請以目前工作目錄（Current Working Directory）下的檔案名稱為讀取路徑。

九、 各題輸出方式為標準輸出（螢幕）。

十、 考試結束後，將不再允許更動及重新編譯程式。

十一、所有發展的程式必須在 10 秒以內於試場內的電腦輸出結果，否則不予計分。

1. 羅馬數字

問題敘述：

強森是一名歷史學家，在閱讀歷史文獻時，有些文獻會以羅馬數字代表年、月、日，有些文獻又會以阿拉伯數字代表年、月、日。因此強森常常需要將歷史文獻中的年份在羅馬數字與阿拉伯數字間做轉換。請寫一個程式幫強森自動的進行數字表示法的轉換。

羅馬數字表示法說明：

羅馬數字只用到七個符號：{I, V, X, L, C, D, M}。這七個符號分別代表 1, 5, 10, 50, 100, 500, 1000。用這七個符號（可重複使用）依不同的順序排列，就可以造出任何十進位阿拉伯數字。羅馬數字轉換到阿拉伯數字的規則很簡單，基本上只需將所有羅馬數字符號所代表的數字加總，但是如果有兩個相鄰的羅馬數字符號其左邊的符號小於右邊的符號時，則這兩個羅馬數字符號所代表的阿拉伯數字則為右邊的羅馬數字減掉左邊的羅馬數字。例如 MMCMXCIX = M + M + CM + XC + IX = 1000 + 1000 + (1000-100) + (100-10) + (10-1) = 2999。羅馬數字不會有連續三個以上的由小到大的情形。例如，IVX 就不是一個合法的羅馬數字。從阿拉伯數字轉換成羅馬數字的時候，請統一從高位數字開始轉換，如 1989，可看成 1000+900+80+9，轉換時由千位數開始先輸出 M，然後接著依序輸出百位數 CM、十位數 LXXX、個位數 IX，得出 MCMLXXXIX。

輸入說明：

第一行有一整數 n ，代表共有 n 個數字需要進行轉換，1<= n <= 5。接下來的 n 行每行有兩個數字。第一個數字可以是「1」或「2」。如果第一個數字是「1」，則代表第二個數字是羅馬數字，如果第一個數字是「2」，則代表第二個數字是阿拉伯數字。測試資料中羅馬數字最長不超過 12 個符號，阿拉伯數字最大亦不超過 2007。第二行起，每行的兩個數字都以一個空白隔開。

輸出說明：

針對所輸入的羅馬數字或阿拉伯數字，依序輸出其對應的阿拉伯數字或羅馬數字，每行輸出一個數字。

輸入範例 1：

```
2
1 MMCMXCIX
1 MMVII
```

輸出範例 1：

```
2999
2007
```

輸入範例 2：

```
3
1 CCC
2 40
2 1502
```

輸出範例 2：

```
300
XL
MDII
```

2. 盤中飧

問題敘述：

大寶跟小寶兩兄弟都不喜歡吃白飯，可是他們的媽媽說，古人有寫詩說：「鋤禾日當午，汗滴禾下土；誰知盤中飧，粒粒皆辛苦。」浪費糧食是不應該的，所以一定要他們兩個人把飯吃完。小寶覺得兄弟倆個人都受苦不如一個人受苦來的好，於是對大寶提出一個賭賽的方式，每餐都來比，誰輸了，誰就得負責把兩兄弟那餐的白飯吃光光。

賭賽的方式是這樣的，首先，由小寶拿出 X 粒白飯，而大寶拿出 Y 粒白飯。接下來兩兄弟由小寶開始輪流吃這些 X+Y 粒白飯，每次可以吃 1 粒或是 k 粒白飯，當剩下的白飯不足 k 粒時，兩兄弟只能一人一粒的輪流吃，而規定吃到最後一粒的人，就得把剩下的白飯通通吃光。小寶還說每次賭賽他會先決定 X 跟 k 兩個數字，之後在由哥哥大寶決定 Y。

開始這種賭賽的前幾天，由於大寶想的太少，已經連吃了好幾天白飯，你能不能幫他寫一個程式，幫他算算該拿出多少粒白飯，才有機會贏？

輸入說明：

輸入的第一個整數 n，即有多少筆測試資料。接下來的 n 行，每行都有兩個正整數 X 跟 k，由空白隔開，其中 X 不超過 10000，k 不超過 1000。

輸出說明：

一筆測試資料輸出一行。當存在一個大於 0 且不超過 10000 的 Y 值能讓大寶不用吃白飯的時候，輸出最小的 Y 值。反之，輸出 0。

輸入範例 1：

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

輸出範例 1：

```
3
2
2
```

輸入範例 2：

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

輸出範例 2：

```
2
1
1
1
```

3. Huffman 編碼中的編碼效能問題

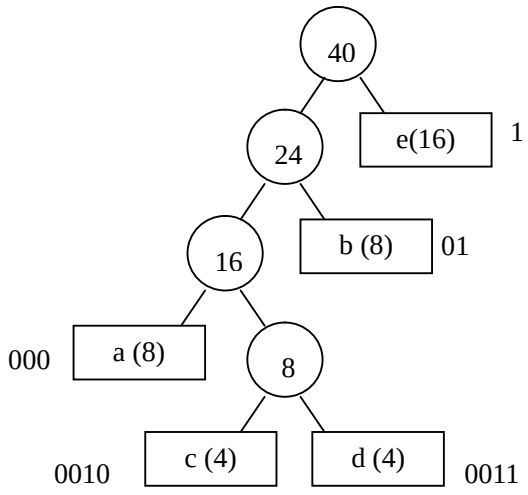
問題敘述：

為了用 0 與 1 的位元串表示電腦中所需使用的符號，常見的編碼法可分為兩大類：固定長度編碼法與非固定長度編碼法。固定長度編碼法指的是，將每個符號用相同長度的位元串表示。例如，要以二進位的 0 與 1 位元串表示{a, b, c, d, e } 5 個符號，每個符號最少需要以 3 個位元來編碼成{000, 001, 010, 011, 100}。
但如果我們已知這些符號在多數資料中出現的頻率不同，就可用非固定長度編碼法更有效地表示這些資料；而 Huffman 編碼法就是非固定長度編碼法中最常使用的方法。例如，已知{a, b, c, d, e}這 5 個符號的出現頻率由大至小如表一所示。

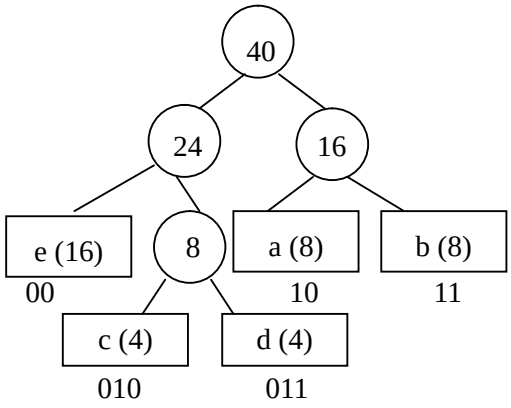
表一

符號	e	a	b	c	d
頻率(出現次數)	16	8	8	4	4

則 Huffman 編碼法利用建立二元樹的方式，首先將每個符號以一個自由節點(free node)表示，這些自由節點的權重(weight)即是符號的頻率。接著，從自由節點中找出權重最小的兩個節點，為這兩個節點做一個父節點，此父節點的權重則為這兩個子節點的權重和。之後，將父節點加入自由節點的行列，並將兩個子節點從自由節點的行列中去除。接著，重覆選取兩個權重最小的節點，造出父節點並更新自由節點的過程，直到最後只剩一個自由節點為止。當得到這棵編碼樹後，我們就可以從二元樹的根結點(root node)開始往下走，每往左子樹(left subtree)走一層就給定一個位元的編碼 0，每往右子樹(right subtree)走一層就給定一個位元的編碼 1；依此方式逐層往下走並同時編碼直到走到葉節點(leaf node)為止。圖一、二所示即為依表一頻率表所建造的不同 Huffman 編碼樹。



圖一



圖二

雖然圖一和圖二的編碼結果不同，但檢視其編碼效能，如表二所示，可發現這兩個編碼所得到的總位元數是相同的：16+24+16+16+16+16=32+16+16+12+12=88。

表二

符號		e	a	b	c	d
頻率(出現次數)		16	8	8	4	4
編碼一	編碼	1	000	01	0011	0011
	位元數	1	3	2	4	4
	頻率×位元數	16	24	16	16	16
編碼二	編碼	00	10	11	011	011
	位元數	2	2	2	3	3
	頻率×位元數	32	16	16	12	12

請撰寫一個程式，根據已知的頻率表(尚未排序過)，計算以 Huffman 編碼後的總位元數。

輸入說明：

第一行為一個整數 n，代表共有幾個符號。($n \le 2^k$，k 為正整數，$1<k\le16$)。
接下來的 n 個數字代表這 n 個符號的頻率，每個數字為介於 1 與 2^{16} 之間的正整數，每兩個數字以一個空格隔開。

輸出說明：

對每一個輸入的頻率表，計算以 Huffman 編碼後的總位元數。

輸入範例 1：

5
16 8 8 4 4

輸出範例 1：

88

輸入範例 2：

5
8 4 4 8 16

輸出範例 2：

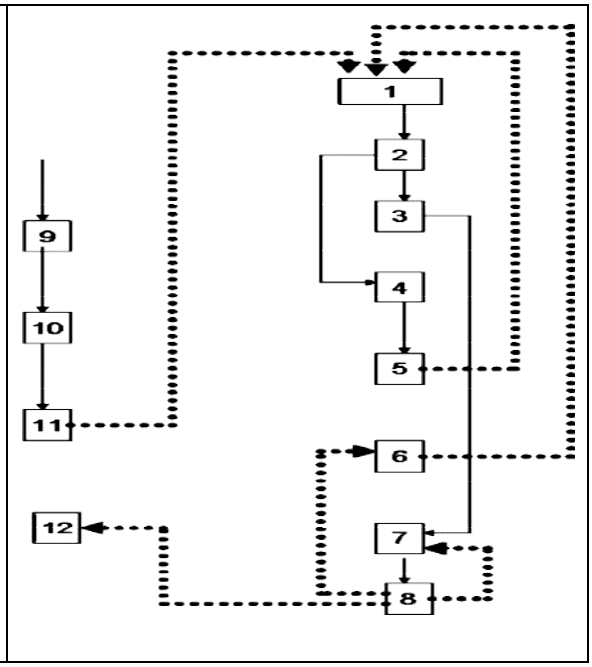
88

4. 合法執行路徑問題

問題敘述：

一個程式的執行與程序呼叫(procedure invocation)關係，可以描述成一個流程圖形(flow graph)，例如以下程式，將計算 Fibonacci 數值，

```
1: void fib(int n, int m, int *v) {
2:   if (n < 2)
3:     *v = m + 1;
4:   else {
5:     fib(n-1,m,v);
6:     fib(n-2,*v,v);
7:  }
8: }
9: main() {
10:   int x;
11:   fib(100,0,&x);
12: } // 計算 Fibonacci 100，而結果存在 x，但右圖的流
//程圖則涵蓋 Fibonacci n, 而 n 是任何大於 0 的正整數
```



觀察上述程式，每一行指令左邊的數字代表行號（行號為自然數 n，若進行函數呼叫時，其返回行號為 n+1），則其流程關係，依照指令的執行先後，可分成程序內部的流程 (intra-procedure flow)與跨越程序的流程(inter-procedure flow)。上述流程圖中，實線表示為程序內部的流程，虛線表示為跨越程序的流程，若遇到程序呼叫 (procedure invocation)、與程序結束(procedure return)，則產生跨越程序流程，而程序結束時將回到程序呼叫的下一行。考慮之路徑包含所有實線與虛線的流程。

根據上例，行號 9 為程式起點，而行號 12 為程式結束，以圖形 (graph)的路徑 (path) 觀點，從端點 9 為起點到終點 12，可以有不同路徑，例如路徑 9 10 11 1 2 3 7 8 12 為一種走法，我們稱為合法路徑 (valid path)，而 9 10 11 1 2 4 5 1 2 3 7 8 12 也是一種走法，但因為不合乎程序呼叫的原則（呼叫程序後，必須返回原呼叫點的下一行），我們稱為不合法路徑 (invalid path)。針對上述程式範例，根據其流程關係所形成之圖形，給予任一由起點到終點的路徑（只考慮程序進入點與離開點相匹配，也就是呼叫函數後，返回點必須是呼叫點的下一行。不考慮遞迴的次數），請你寫一個程式來判斷路徑是否為合法。

輸入說明：

測試檔有許多行，除最後一行外，每行包含一個路徑輸入描述，以序列的行號表

示，行號間以一個或以上的空白隔開，一個路徑描述不會超過 100 個數字，例如：
9 10 11 1 2 3 7 8 12
9 10 11 1 2 4 5 1 2 3 7 8 12
最後一行只有包含 0 一個整數，代表測試檔案的結束。

輸出說明：

根據上述範例程式所形成之圖形，依序判斷所給定之路徑是否合法。若合法，輸出：
valid
否則輸出：
invalid

輸入範例：

9 10 11 1 2 3 7 8 12
9 10 11 1 2 4 5 1 2 3 7 8 12
9 10 11 12
9 10 11 1 2 4 5 1 2 3 7 8 6 1 2 3 7 8 7 8 12
0

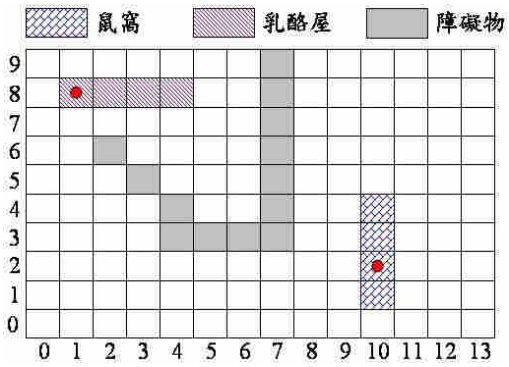
輸出範例：

valid
invalid
invalid
valid

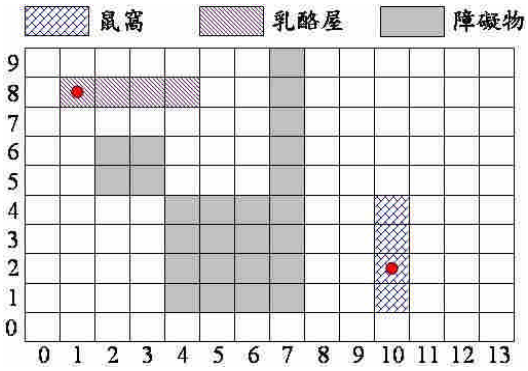
5. 乳酪的誘惑

問題敘述：

這個問題跟傳統的老鼠走迷宮問題不太一樣，給定一個四方形的格點空間，x 座標與 y 座標由左下角原點以 0 開始往右和往上遞增，裡面有一個區域是鼠窩，另外有一個區域是乳酪屋，此外還有很多障礙物是用來阻擋鼠輩的橫行，也就是障礙物是不可穿越的。在此空間裡可能沒有路可以從鼠窩走到乳酪屋，此種情況你必須回報沒有路徑，如果有路可以連接鼠窩跟乳酪屋的話，你必須算出從鼠窩走到乳酪屋的最短路徑。老鼠的每一步只能往四個方向走:上、下、左、右，不能走對角方向。出發點與終點可以是鼠窩與乳酪屋的任何一個位置。以圖一為例，鼠窩到乳酪屋的最短距離為 15，由鼠窩裡圓圈所標示的點(10,2)往左出發走到乳酪屋中圓圈所標示的點(1,8)。鼠窩與乳酪屋的形狀一定是一條直線，障礙物可以是一點可以是一條直線也可以是一個矩形區域，圖一中的障礙物是由三個點與兩條直線所構成，分別為點(2,6)、(3, 5)、與(4, 4)和線(4, 3)~(7, 3)與(7, 4)~(7, 9)。在找尋最短路徑時，不要只派遣一隻老鼠出任務，這樣會累死這隻可憐的倒霉鼠的。



圖一



圖二

輸入說明：

輸入的第一行為兩個整數，描述方形格點空間的大小，第一個表示行的數目，第二個表示列的數目，在此問題中行和列的數目都不會超過 1000 (≤1000)。接下來“.mb”表示後面用兩點來描述鼠窩的位置，每個點先 x 座標再 y 座標，兩個點沒有固定的先後排序。接下來“.cb”表示後面用兩點來描述乳酪屋的位置，兩個點也沒有固定的先後排序。最後“.bb”表示後面開始每一行描述一個障礙物的位置，不管是一點還是一條線都是用兩點來表示，最後以“.be”表示結束。

輸出說明：

沒有路徑的話，輸出“no path”。有路徑的話輸出最短路徑距離。

請注意: 程式未在一分鐘之內跑出結果的視為失敗。

輸入範例 1：以圖一為例，其輸入檔案如下:

```
14  10
.mb  10  4  10  1
.cb  1  8  4  8
.bb
2  6  2  6
3  5  3  5
4  4  4  4
4  3  7  3
7  9  7  4
.be
```

輸出範例 1：

15

輸入範例 2：以圖二為例，其輸入檔案如下:

```
14  10
.mb  10  4  10  1
.cb  1  8  4  8
.bb
2  5  3  6
4  4  7  1
7  5  7  9
.be
```

輸出範例 2：

18

6. $X^2 \equiv 1 \pmod{M}$

問題敘述：

給你一個式子 $X^2 \equiv 1 \pmod{M}$ 和 $M\ (0 < X \leqq M)$，請找出所有符合這個式子的 X。
例如：M=5，則 X 可以等於 1 或 4；M=8 時，X 可以等於 1, 3, 5, 7。

請你寫一個程式，針對每一個 M，輸出能滿足這個式子的 X。

輸入說明：

每一個測試檔裡有一個整數即為 M，你可以假設 M 不會大於 2147483647。

輸出說明：

第一行為一個整數 n，代表共有多少組解。
接下來的 n 行則為所有滿足此式子的 X，並由小到大輸出。

輸入範例 1:

5

輸出範例 1:

2
1
4

輸入範例 2:

8

輸出範例 2:

4
1

3
5
7

輸入範例 3:

15

輸出範例 3:

4
1
4
11
14