

列印指定月份月曆

丁培毅

實習目標：

1. 練習簡化問題, 由不同角度看問題, 轉換已經知道的語法和範例來達成要求
2. 迴圈練習 (迴圈控制變數的意義, 迴圈結束條件設計)
3. 螢幕上列印資料需要用適當的 **stdio** 函式, 有的時候一整段印出, 有的時候一個數字一個數字印出, 有的時候一個字元一個字元印出, 需要靈活運用
4. 練習整數求餘數的運算 **%**, 在迴圈控制或是以後的陣列存取常常可以作出很簡潔的程式碼

列印指定月份月曆

請撰寫一個程式

1. 要求使用者由鍵盤鍵入一整數代表這個月有幾天
 2. 要求使用者由鍵盤鍵入一整數代表這個月一號星期幾
- 在螢幕上印出這個月的月曆

範例執行程式：

```
請問這個月有多少天？31  
請問這個月第一天是星期幾？(0=星期天, 1=星期一, ...) 2  
S   M   T   W   T   F   S  
    1   2   3   4   5  
6   7   8   9  10  11  12  
13  14  15  16  17  18  19  
20  21  22  23  24  25  26  
27  28  29  30  31
```

分析

1. 這個題目的輸入部份很單純，直接列印提示之後讀入整數變數 `nDays` 和 `startWeekDay` 中
2. 這個題目最主要需要練習的是輸出：前面我們有做過 **ASCII 藝術圖**，但是那時候一系列一系列的資料事先就設計好，直接寫在 `printf` 敘述裡面，現在這個程式要求
 - a. 在螢幕上印出 **S M T W T F S**，這可以用先前的方法
 - b. 在螢幕上印出 **1 2 ... nDays**，這比較不一樣
 - i. 要印出來的資料 **1 ~ nDays** 是由程式計算出來的，這應該也不是問題，可以寫一個 `for` 迴圈，迴圈控制變數 `i` 的數值就可以由 **1, 2**，一直加到 `nDays`
 - ii. 每一列開始的數字都不太一樣，個數不太一樣，前面的空格數目不太一樣，如果對某一系列來說，這三個資料都指定好，列印一系列的程式並不困難，例如：
 - “空 6 格，印出 1 到 5，每個數字 3 格”
 - “空 0 格，印出 6 到 12，每個數字 3 格”
 - “空 `nSpace` 格，印出 `start` 到 `end`，每個數字 3 格”

拆解題目

```
int i;
for (i=0; i<nSpace; i++) printf(" ");
for (i=start; i<=end; i++) printf("%2d ", i);
printf("\n");
```

列印個數可變的空格
需要運用迴圈來完成

3. 接下來的問題就是怎麼算出 `nSpace`, `start`, 以及 `end`? 還有總共要列印幾列(迴圈結束條件)?

a. 除了第一列 `nSpace` 為 $3 * \text{startWeekDay}$ 之外，其它列都是 0

b. 第一列 `start` 為 1，`end` 為 $7 - \text{startWeekDay}$

第二列以後 `start` 為前一列的 `end + 1`，`end` 為 `start + 6` 或是 `nDays` 兩者比較小的

c. 結束條件：`start` 會是前一列的 `end + 1`，如果前一列已經到達 `nDays`，`start` 就會超過 `nDays`

```
int i, nSpace = 3 * startWeekDay;
int start = 1, end = 7 - startWeekDay;
while (start <= nDays)
{
    for (i=0; i<nSpace; i++) printf(" ");
    for (i=start; i<=end; i++) printf("%2d ", i);
    printf("\n");
    nSpace = 0;
    start = end + 1;
    if (start + 6 > nDays)
        end = nDays;
    else
        end = start + 6;
}
```

4. 請注意，前面我們用的分析方法是把問題逐步拆解，最後得到比較簡化的問題，能夠以比較直覺的方式寫出程式；平常不管問題是什麼，撰寫程式的方法都會有很多種，有些問題的解法需要腦筋稍微轉個彎，寫出來的程式會比較簡潔，不過因為需要稍微轉換一下思考的方式，這個轉換幾乎都是和問題相關的，沒有一般性的作法，所以有的時候又不見得一定想得到。

不過不用擔心，不管用哪一種方法，只要能夠完成題目的要求，就是可以接受的方法，記得不要太要求自己一定要以最簡潔的方法完成，有的時候也因為那樣的要求，會卡在一些不需要的問題上，反而得不償失，對於程式玩家來說，當然是很好的自我要求，但是對於資訊系訓練出來的軟體工程師來說，這不見得是必要的要求。常常最基本、最直接的解法，所寫出來的程式也是最容易讓人看得懂、最容易擴充、最容易維護的。

5. 接下來我們針對這個問題，看一個比較簡潔的作法，前面的方法中我們看到月曆的輸出，把它看成是一列一列獨立的輸出，每一列需要印出來的由三個參數決定：nSpace, start, end

5. 現在我們嘗試把列印月曆的問題簡化成列印單一遞增數列：

請問這個月有多少天？**31**

請問這個月第一天是星期幾？(0=星期天, 1=星期一, ...) **2**

S M T W T F S

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

這個問題應該很容易完成，空格的個數 `nSpace` 是 `startWeekDay*3`，然後用一個迴圈就可以列印 1 到 `nDays`

```
int i;
for (i=0; i<nSpace; i++) printf(" ");
for (i=1; i<=nDays; i++) printf("%2d ", i);
printf("\n");
```

和原先的月曆差別在哪裡呢？少印了五個換列字元 `'\n'`，分別在 5, 12, 19, 26, 31 之後，當然這些位置 `i` 和使用者的 `startWeekDay` 有關係，計算一下 `i` 應該滿足下列運算式

$$(\text{startWeekDay} + i) \% 7 == 0$$

所以我們可以在列印 1 到 `nDays` 的迴圈內部增加一列測試是否該印出換列字元的敘述如右

```
int i;
for (i=0; i<nSpace; i++) printf(" ");
for (i=1; i<=nDays; i++) {
    printf("%2d ", i);
    if ((startWeekDay+i)%7 == 0)&&
        (i<nDays))
        printf("\n");
}
printf("\n");
```

請注意

前面的說明盡量由各個角度來描述程式該怎樣設計，也不可避免地把一些程式碼直接寫出來，希望你能夠完整地了解程式開發的邏輯思考過程。但是你也就看過了那些程式碼，有的時候其實看過了某一種設計方法以後，很難跳脫那個窠臼，很難要求你獨立寫一個完全不一樣的程式，就算你沒有完全瞭解，常常也可以依照部份的記憶來完成程式，那樣子其實練習的目的就大打折扣了，如果可以的話，還是希望你

1. 看完題目以後，先嘗試構思程式
2. 如果覺得有困難才開始看這些說明，一開始覺得有頭緒，就應該繼續嘗試用自己的想法寫下去
3. 如果你一次就把這份資料讀完了，希望你不要馬上寫，稍微沈澱一下，過個 **24** 小時再開始寫，不要邊看編寫，才能夠擺脫記憶模式，回歸你自己的思考邏輯

延伸練習

1. 已知公元元年 1/1 日為星期一，請寫一個程式要求使用者輸入年月 (例如 2015/10)，印出指定月份的月曆