while 迴圈與條件測試

丁培毅

實習目標：

1. 練習 while 迴圈的語法, 遵循語法規定, 掌握語法允許的變化, 完全瞭解各部份執行的順序
2. 練習 if 條件測試敘述語法, 練習複合的條件, 練多重選項的 if 敘述
3. 練習**簡化題目**的要求, 針對各個部份設計程式, **逐步增加程式的功能**, **逐步測試**, 善用程式開發環境
4. 練習思考測試資料

while 迴圈與條件測試

請撰寫一個程式

- 不斷地要求使用者由鍵盤鍵入一正整數 n，如果同時滿足
  n 除以 3 的餘數是 2，
  n 除以 5 的餘數是 3 ，且
  n 除以 7 的餘數是 2 ，
  則程式印出恭賀訊息並且結束，否則繼續
- 範例如下：

```
請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數：15
錯！
請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數：28
錯！
請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數：53
錯！
請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數：23
猜對了！
猜了 4 次才猜對，有待加強。
```


- 程式要求：
  1. 輸出提示訊息「請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數：」
  2. 檢查使用者輸入是否符合要求
  3. 不符合的話輸出「錯！」，重複由 1 開始執行
  4. 符合的話輸出「猜對了！」訊息以及依照使用者猜測次數 k 而定的評語：

次數 k	評語
1	哇！一次就猜中！超強！
2~5	猜了 k 次才猜對，有待加強。
>= 6	猜了 k 次才猜對，沒人比你強。

設計的基本方法

先不管每一個程式不同的要求，

- 前面的實習裡 (miles2kms, for loop practice, ASCII arts)，你看到了要設計一個程式，基本的方法是針對程式各個功能**尋找類似功能的範例程式碼**，**瞭解**範例程式碼中每一個部份可以得到的效果，然後稍微**修改**一點點來完成需要的功能

這樣子對於**底層的小功能**應該都有機會完成，可是要寫的程式功能不見得那麼少啊，比較**高層的功能**比較難直接找到範例

- 這個實習裡你會看到另外一種很常用來設計程式的方法，你需要練習 **拆解問題** 和 **簡化問題**：基本上就是 **先設計一小部份的功能，測試完成後，再逐步把需要的功能加進來**，每增加一點程式就進行編譯和測試，不需要把程式所有的功能都一次設計完了才開始編譯測試，避免讓所有的語法問題、邏輯問題通通攪和在一起，使得順利完成的困難度大大升高

分析: 想像你自己是執行這個程式的電腦, 你需要

1. 先不去看各個部份的細節，這個題目可以先**簡化**成
   不斷地重複 "輸出提示，讀入資料，判斷是否答對，輸出結果"
   再稍微拿掉一些功能，可以再**簡化**成
   不斷地重複 "輸出提示，判斷是否答對"
   這樣的要求有點類似下面的迴圈語法所產生的結果

```
while (迴圈繼續執行條件)
{
   迴圈執行內容
}
```

試試看這個迴圈，它會不斷地輸出提示，也會判斷

```
int answer = 0;
while (answer == 0)
{
   printf("請輸入 0 或 1：\n");
}
```

無窮迴圈

但是因爲迴圈內部沒有修改 answer 的值，迴圈開始執行就停不下來了，你要按視窗右上角的 X 關閉視窗或是按 Ctrl-C 把這個程式結束掉

2. 先不要去管列印出來的東西和實際要求的有很大出入，先再加進一點簡化過的功能，讓迴圈能夠適當結束，例如 把問題改成
   不斷地重複 "輸出提示，讀入資料，判斷是否答對"
   在前面的迴圈裡增加一列輸入敘述來修改變數 answer 的值

```
int answer = 0;
while (answer == 0) {
   printf("請輸入 0 或 1：\n");
   scanf("%d", &answer);
}
```

現在程式還是很簡單的，可以**很快測試一下**，確定沒有發生不能預期、不能解釋的結果。
雖然你也許覺得距離程式的完成還很遠，好消息是其實不會，現在這個版本已經有很接近這個程式需要的架構了，同時這樣子一步一步的設計，你會避免掉很多因爲語法不熟悉而發生的錯誤，在練習的過程中你需要**一步一步地確定什麼語法會有什麼效果** (如果你剛開始接觸程式，注意千萬不要貪圖快速，參考同學的範例，一次增加很多功能，發生錯誤的時候自己完全不知道從何處著手修改，就算不小心結果對了，也是碰運氣亂槍打鳥，最後自己什麼概念都沒建立起來，以後程式的複雜度越來越高，就完全沒有辦法跟得上了)

3. 回憶一下這個題目的要求
   不斷地重複 "輸出提示，讀入資料，判斷是否答對，輸出結果"
   你可以先把**輸出提示**部份加強一下，題目要求
   在螢幕上列印提示文字「請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數：」

```
printf("請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數： \n");
```

像上面程式，有的時候你會覺得字串很長，一列敘述在編輯器裡會跑到下一列去，讓你的程式沒有對齊，程式變得不好讀，你可以用下面三種方式，自己把字串切開，印出來的結果和上面是一樣的

```
printf("請猜一個被 3 除餘 2、被 5 除餘 3、\
被 7 除餘 2 的數： \n");
```

```
printf("請猜一個被 3 除餘 2、被 5 除餘 3、"
       "被 7 除餘 2 的數： \n");
```

```
printf("請猜一個被 3 除餘 2、被 5 除餘 3、");
printf("被 7 除餘 2 的數： \n");
```


4. 再回憶一下這個題目的要求
   不斷地重複 "輸出提示，讀入資料，判斷是否答對，輸出結果"
   你可以先把**判斷是否答對**這一部份加強一下，題目要求

從鍵盤讀到的數字被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的話就是正確的，否則迴圈繼續執行的條件成立，程式繼續要求使用者輸入另一個數字

請修改判斷式answer == 1，判斷輸入是否不符合要求：
個別的要求判斷如下

a. (answer % 3) == 2 (判斷answer 除以 3 的餘數等於 2)
b. (answer % 5) == 3
c. (answer % 7) == 2

合併的判斷可以寫成

(... != …) || (…!=…) || (…!=…)

三者之一有一個不等就是不符合，也可以寫成

! ((... == …) && (…==…) && (…==…))

三個都同時相等代表輸入符合要求，前面加上 ! 就代表不符合

這個時候程式改成了

```
int answer = 0;
while ( (answer%3 != 2) ||
        (answer%5 != 3) ||
        (answer%7 != 2) )
{
   printf("請猜一個被 3 除餘 2、被 5 除餘 3、"
          "被 7 除餘 2 的數： \n");
   scanf("%d", &answer);
}
```

你的機器速度很快，編譯測試一下，確定一下現在學到的，程式應該能夠不斷地提示使用者輸入，輸入數值只要不滿足上面的條件，迴圈內部列印提示和讀入資料的兩個步驟就會不斷執行，直到使用者輸入 23 時程式才會停下來 (或是 23 + 105 k，離散數學裡面會教你怎樣用中國餘式定理很快找到這個答案)

5. 接下來可以加強程式在判斷對錯以後的輸出，你希望在
   a. 不符合的話輸出「錯！」
   b. 符合的話輸出「猜對了！」訊息以及依照使用者猜測次數 k 而定的評語

在上面程式架構下，你可以找到兩個地方分別加入輸出訊息的 printf 程式碼

```
int answer = 0;
while ( (answer%3 != 2) || (answer%5 != 3) || (answer%7 != 2) ) {
   輸出「錯！」
   printf("請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數： \n ");
   scanf("%d", &answer);
}
輸出「猜對了！」及其它評語訊息
```

測試一下，會立刻發現一個問題：一開始使用者都還沒有開始回答的時候就先印出一個「錯！」，這是因爲程式一開始把 answer 設定爲 0 所造成的，程式這樣子多印出一個「錯！」的表現是不能接受的，這種問題常常發生在迴圈的第一次或是最後一次，你需要再調整一下，調整的方法有很多種，例如

你可以多運用一個整數變數 k 來記錄使用者已經回答幾次了，第零次不要印出那個「**錯！**」的訊息

```
int k=0, answer = 0;
while ( (answer%3 != 2) || (answer%5 != 3) || (answer%7 != 2) )
{
   if (k>0) 輸出「錯！」
   printf("請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數： \n ");
   scanf("%d", &answer);
   k++;
}
輸出「猜對了！」及其它評語訊息
```

其實你仔細看一下 “其它評語訊息” 的要求，會發現其實也是需要設計一個整數變數 k 來記錄使用者已經回答的次數的

或是你也可以調整測試輸入數字的位置，不要在迴圈的條件的地方測試，而運用 if 敘述來測試，輸出「**錯！**」就可以放在 if 敘述成功的時候才做，當然如果這樣調整的話，就需要再設計一個控制變數 correctAnswer 來控制迴圈是否要繼續執行，例如

```
int k=0, correctAnswer = 0, answer;
while (correctAnswer == 0)
{
   printf("請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數： \n ");
   scanf("%d", &answer);
   k++;
   if ((answer%3 != 2) || (answer%5 != 3) || (answer%7 != 2))
      輸出「錯！」
   else
      correctAnswer = 1;
}
輸出「猜對了！」及其它評語訊息
```

或是

```
int k=0, correctAnswer = 0, answer;
while (correctAnswer == 0) {
   printf("請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數： \n ");
   scanf("%d", &answer);
   k++;
   if ((answer%3 != 2) || (answer%5 != 3) || (answer%7 != 2))
      輸出「錯！」
   else {
      correctAnswer = 1;
      輸出「猜對了！」及其它評語訊息
   }
}
```

或是

```
int k=0, answer;
while (1) {
   printf("請猜一個被 3 除餘 2、被 5 除餘 3、被 7 除餘 2 的數： \n ");
   scanf("%d", &answer);
   k++;
   if ((answer%3 != 2) || (answer%5 != 3) || (answer%7 != 2))
      輸出「錯！」
   else {
      輸出「猜對了！」及其它評語訊息
      break;
   }
}
```


6. 接下來請加強程式需要輸出的 “其它評語訊息”

次數 k	評語
1	哇！一次就猜中！超強！
2~5	猜了 k 次才猜對，有待加強。
>= 6	猜了 k 次才猜對，沒人比你強。

這個地方基本上你可以參照 “多重條件判斷敘述” 的範例來修改，例如:

```
int x;  scanf("%d", &x)
if (x == 1) { // 1
   …
}
else if ((x >= 2) && (x<=4)) { // 2, 3, 4
   …
}
else if (x <= 8) { // 5, …, 8
   …
}
else { // 9, 10, …, 0, -1, -2, …
   …
}
```


- 每一個程式寫好，編譯並且對平常會輸入的測試資料測試完成以後
- 你需要想一下是不是會有一些不太常出現的輸入狀況，例如你現在完成的這個程式如果使用者輸入 0, 輸入一個負數，或是輸入一個很大的數字，或是輸入一個有小數點的數字，或是根本沒有輸入數字而輸入 abcd…

  使用者沒有像我們前面設計的時候所假設的那麼聽話 (他有意或是無義的看不懂你提示的訊息)，一般來說程式設計的人是應該要考慮這些狀況的，反正最後是由電腦在執行，它檢查很多狀況其實不會花很多功夫，但是就不會意外地停下來 (你可以想像飛機上運作的程式突然發生錯誤會怎樣嗎? 太空梭上? 醫院裡病人的維生系統呢?)
- 這些構思其實對有經驗的程式設計者來說會在設計輸入時一次就考量完畢，因爲你開始學寫程式，所以沒有要求你在那個階段思考這個問題，但是你還是應該慢慢練習考量各種可能出現的狀況
- 最後每個練習做完以後，記得回憶一下花了這麼多時間下去，到底學到了什麼，整理一下得到的規則！