

模擬擲骰子與次數統計

實習目標：

丁培毅

1. 練習使用 `stdlib` 函式庫中的 `srand()` 與 `rand()` 兩個函式, 以及常數 `RAND_MAX`
2. 練習如何產生在指定區間中均勻分佈的亂數
3. 練習使用 `time` 函式庫中的 `time()` 函式
4. 練習運用陣列變數來記錄統計資料
5. 思考有運用虛擬亂數 `rand()` 的程式該如何除錯
6. 思考虛擬亂數序列可以用在什麼樣的應用中

1

模擬擲骰子與次數統計

請撰寫一個程式，模擬執行丟兩顆骰子的機率實驗 6000 次，請紀錄兩個骰子點數和，統計並顯示各種點數和出現的次數
程式輸出範例：

```
2 出現過 187 次
3 出現過 336 次
4 出現過 497 次
....
12 出現過 176 次
```

2

分析

1. 這個題目需要使用 `stdlib` 裡提供的虛擬亂數產生器 `rand()` 以及初始化函式 `srand()`，使用範例如右：

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main()
{
    int i;
    srand(time(0L));
    for (i=0; i<10; i++)
        printf("%f\n", (double) rand() / (RAND_MAX+1.0));
    return 0;
}
```

0.868374
0.713309
0.858882
0.864956
0.566881
0.655721
0.240303
0.127995
0.456038
0.251930

這個範例需要引入 `stdlib.h` 以及 `time.h`，在程式一開始的時候呼叫 `time(0L)` 來得到目前的系統時間 (單位是秒，得到的數值是由 1970/01/01 00:00:00 到現在之間的秒數)，然後以此時間作為亂數產生器的種子，呼叫初始化函式 `srand()` 來初始化亂數產生器 (請注意這個動作只需要做一次)，之後呼叫 10 次 `rand()` 來得到 10 個大小在 `[0, RAND_MAX]` 範圍內的整數，這個整數序列的關聯性很低，同時它們是均勻分佈的，如果程式執行兩次，初始化的種子數值相同的話，呼叫 `rand()` 所得到的就是完全相同的序列。上面範例中呼叫 `rand()` 後，轉換為倍精確浮點數，然後用浮點數除法除以 `RAND_MAX+1.0`，如此可以得到一個分佈在 `[0,1)` 之間的實數亂數序列

3

2. 因為每次呼叫 `rand()` 得到的數值是在 `[0, RAND_MAX]` 範圍內的整數，如果想要模擬一顆公平的骰子的話，需要把輸出限制成六種，例如 `{1,2,3,4,5,6}`，同時需要維持各種輸出出現的機率是相同的，基本上大家常用的有兩種方法：
 - a. `rand() % 6 + 1`：這樣的結果落於 `{1,2,3,4,5,6}` 集合中，至於是不是均勻分佈的，相鄰的數值關聯性高不高，需要看 `rand()` 這個亂數產生器產生出來的數字的低位元的分佈情形
 - b. `(int) (((double) rand()) / (RAND_MAX+1.0) * 6.0) + 1`：這樣子計算的結果也會落於 `{1,2,3,4,5,6}` 這個集合，同時相鄰的數值的關聯性和原來的 `rand()` 序列的關聯性相同，也會是均勻分佈的
3. 有了模擬公平骰子的方法，丟兩顆骰子基本上就是呼叫兩次 `rand()` 函式，然後計算兩個骰子的點數和 (總共有 11 種可能性)，因為要紀錄每一種狀況發生的次數，需要設計整數陣列變數，初始化每一個元素為 0

```
int occurrence[11]={0};
...
occurrence[sum-2] += 1; // sum 是兩個骰子的點數和
```

4

注意:

1. 當一個程式使用 `rand()` 產生虛擬亂數序列時, 程式的表現和所讀到的亂數有關係, 因為這個序列的順序看起來不太規律, 所以在測試程式時, 會感覺不太容易確定程式是否正確執行, 就算有的時候看到程式好像出錯了, 想要重複一樣的錯誤也不是那麼直接的
2. 如果要重複錯誤的表現的話, 需要用完全一樣的亂數序列, 你可以固定 `srand()` 的亂數種子, 只要種子固定下來以後, 其後呼叫 `rand()` 所得到的序列就是同一個序列, 如此重複執行程式才能夠得到一樣的結果, 這在除錯時非常重要
3. 有使用虛擬亂數的程式, 常常每一次執行所看到的表現會有一些不一樣, 要評估程式的表現需要藉由一些統計量, 例如此實習中的出現次數
4. `rand()` 產生的亂數序列的統計特性是不錯的, 可以拿來模擬自然界中很多現象, 程式的表現會很像真實世界。但是這個序列**不是不可預測的**, 請不要拿來做資訊安全的應用, **除了實習之外**, 也不要拿來洗牌, 初始化線上遊戲, 會使得公平性出現問題; 也不要拿來做線上開獎, 不要拿來在線上遊戲中挑選互動的場景, 玩家有機會可以預測程式的表現

5

延伸練習

1. 請撰寫一個程式模擬一個不公平的骰子, 六種點數的機率各為

1	2	3	4	5	6
1/12	1/6	1/4	1/4	1/6	1/12

2. 題 1 中的骰子如果一次丟兩顆, 模擬 6000 次時, 請估計兩顆骰子的點數和的機率分佈
3. 如果有一個兩個人的遊戲, 用丟骰子來決定誰贏, 這個骰子是公平的, 遊戲規則是這樣的: 兩個人先各丟一次骰子, 如果點數和大於 7 就由 A 開始, 小於 7 就由 B 開始, 如果是 7 的話就重新再來一次, 遊戲開始以後, 兩個人輪流丟骰子, 先丟出 6 點的人就贏了, 請模擬一下, 估計先開始丟的人贏這個遊戲的機率是多少

6