陳奕嘉 Alvin Tan

程式設計作業四
題目：**撲克牌發牌系統**

實現方法：

寫這題的 code 的主要思路是模擬真實世界的發牌方式，和兩兩對調的洗牌方法。

首先我用了 typedef struct 來制定一個叫 card 的資料形態，它同時具有花色和數字的屬性，用法和 int、char 類似。列出花色和數字的兩個子元的 array 后，在空的 deck 陣列裏把花色和數字的組合方式生成 52 張撲克牌，如下：

```
Card 1: A of Spades
Card 2: 2 of Spades
Card 3: 3 of Spades
Card 4: 4 of Spades
Card 5: 5 of Spades
Card 6: 6 of Spades
Card 7: 7 of Spades
Card 8: 8 of Spades
Card 9: 9 of Spades
Card 10: 10 of Spades
Card 11: J of Spades
Card 12: Q of Spades
Card 13: K of Spades
Card 14: A of Hearts
Card 15: 2 of Hearts
Card 16: 3 of Hearts
Card 17: 4 of Hearts
Card 18: 5 of Hearts
Card 19: 6 of Hearts
Card 20: 7 of Hearts
Card 21: 8 of Hearts
Card 22: 9 of Hearts
Card 23: 10 of Hearts
Card 24: J of Hearts
Card 25: Q of Hearts
Card 26: K of Hearts
Card 27: A of Diamonds
Card 28: 2 of Diamonds
Card 29: 3 of Diamonds
Card 30: 4 of Diamonds
Card 31: 5 of Diamonds
Card 32: 6 of Diamonds
Card 33: 7 of Diamonds
Card 34: 8 of Diamonds
Card 35: 9 of Diamonds
Card 36: 10 of Diamonds
Card 37: J of Diamonds
Card 38: Q of Diamonds
Card 39: K of Diamonds
Card 40: A of Clubs
Card 41: 2 of Clubs
Card 42: 3 of Clubs
Card 43: 4 of Clubs
Card 44: 5 of Clubs
Card 45: 6 of Clubs
Card 46: 7 of Clubs
Card 47: 8 of Clubs
Card 48: 9 of Clubs
Card 49: 10 of Clubs
Card 50: J of Clubs
Card 51: Q of Clubs
Card 52: K of Clubs
```

接下來是洗牌階段，在 shuffle fcn 裏從 0-51 之間隨機選擇 x 和 y 兩個號碼，用 temp 的方式把兩張卡片的花色和數字的屬性同時對換，並重複 1000 次，這樣就完成洗牌。

過後要詢問使用者發牌的數量，這裏我想做一個非常完善的防呆系統，所以學了如何用 boolean 來防止奇怪的輸入方法，例如___13___、1abc3。

輸入的防呆方法思路是把使用者的輸入以 char 的形態儲存起來而名爲 ch，代入 parse_int 的函數裏，它可以判定數字前後的空格、小數點的使用、數字之間是否夾雜著子元或空格、或單純只輸入了子元等等。若檢測到輸入并非為有效的數字，函數就回傳 false，讓使用者重新輸入。若檢測後 boolean 仍然為 true，就用 atoi 的方法把子元轉換成有效數字 deal。這個數字會再被檢測一次是否在 1-13 的範圍裏。

發牌是從 Card 1 開始把洗好的牌發給使用者，就是 deck 裏面前 n 個 cards。接下來就用 swapCards 和 permute 兩個函數來對 deal 的 cards 進行排列組合，然後應使用者要求選擇是否羅列。這邊開始就照著流程圖跑，方式是要求使用者輸入 y 或 n 來達成不同的走向。這些都在一個 while loop 裏面，所以使用者輸入錯誤可以重複詢問。羅列卡牌方法特別的部分是用 hand[i].number 和 hand[i].suit 列出，不能直接用 hand[i]列出具有兩種屬性的陣列。程式的結束用語 return 0 設置在程式碼的中間部分，并非末端，4 個 return 0 代表了 4 種結尾。

延伸問題：

Q. 游戲的持續發牌

A. 我認爲其中一種進階的發牌系統可以在發完第一輪，持續進行第二輪、第三輪的發牌。爲了達到這個目的，我想可以把 dealCards 裏面的 i 弄出來重新設置一個特別的 index 標識在函數外面，可以讓 index 停留在剛剛的位置，好讓下一個使用不會抽到相同的牌。

若在發完第一次牌要再次洗牌才發牌，可能就要把第一輪發牌從 deck 裏面去除然後設置 rand() % (num-deal)，這裏的 num=52 要設定為 global，達到持續變更。儅卡牌剩餘數小於 13 要提醒使用者，並要再設置一個防止輸入的 deal 大於剩餘卡牌的機制，還要設定很多 hand 的卡牌堆來存取每一次的發牌。

Q. 游戲的結束條件

A. 卡牌發牌系統其實不是一個完整的游戲，而是任何卡牌游戲的基底，可以從結束的地方延伸下去。譬如説，當持續發牌到發完的時候，是否有另外一副牌來頂替。或者在 21 點的游戲中檔每人確定不再抽牌的時候，對比誰的牌組勝出來判定輸贏，並結束游戲。所以具有除羅列卡牌之外的結束方式。

Q. 是否有其他方法選取 deal 出來的卡牌

A. 還有另外一種在 52 張排當中隨機選取 1-13 張牌的方法，因電腦一次只選一張牌，在被選中的牌不被去除的情況下，電腦在選完 n 張牌后還要對比 n 張牌之間有沒有重複的牌，有的話就重新選擇，這會拉長程式執行時間。隨機選完一張排再把它從 deck 裏去除這種方式雖然寫起來是比較麻煩，但可行性還是很大的。

討論：

這次的題目要把選擇性流程圖以程式碼的方式呈現出來，若一個選擇分裂點代表一個程式代碼塊，則程式碼的下半部分都由代碼塊塞進代碼塊的方式組成。這時候花括號的位置和數量就很重要了，若位置不分明可能造成 bug 的出現，我在寫作業時也有 debug 這部分。所以這能督促我們寫 code 時要注意代碼塊之間的相對位置。

羅列出牌順序方法：

我在程式中用的洗牌方式是讓 deck 中隨機兩張牌作對換，shuffle 的次數設置在 1000 是因爲這個次數能把卡牌打亂得比較有效，隨機性比較高。而當 shuffle 次數只有 10 或 100 次，會有一些卡牌還停留在原本位置的情況，造成兩張或三張左右的排按順序貼在一起，這樣表示洗牌洗得不幹淨。這個關乎 deal 出來的牌。

羅列所有排列組合的方式，是用遞迴函數 permute，它的主要思路是透過交換 hand 牌組中的卡片來生成不同的排列。這個函數首先檢查 start 和 end 這兩個 index 是否相等，若相等表示生成一個新的排列組合，隨之 print 出來該 sequence。若不想等，則執行以下程式。

```
else {
    for (int i=start; i<=end; i++) { // Swap the card in every i with the
        card in start position
        swapCards(&deck[start], &deck[i]); //
        permute(deck, start+1, end); // Permute the remaining cards
            recursively
        swapCards(&deck[start], &deck[i]); // Swapping the cards again to
            restore the original position
    }
}
```

第一個 swapCards 會把每一個 i 処的卡與 start 処的卡交換，這樣每一種卡都有機會在 start 的位置上。接著對 start+1 和之後剩餘的卡牌使用遞迴的方式一直 permute。for 迴圈裏一個 i 完成后，就會用第二個 swapCards 把 start 之後的卡牌調回原本位置，讓下一個 i 卡牌能與 start 卡牌調換。Start==end 就印出該排列組合。

隨著 deal 在 1-13 的數值增大，程式所需的執行時間也指數型地增大，執行速度跟電腦的功率有關。儅 deal=7 時，有 7! = 5040 種排列組合，本人筆電執行時間是 t=1.78 s。儅 deal=13，13!= 6227020800，可以估計 t=2199225.6 s，大約需要 25.45 天才能執行完。所以 deal>=6 才要詢問是否羅列。我認爲普通數字的排列組合執行時間會比較少，因爲對比 1、2、3 這些數字的排列組合，此程式羅列的 sequence 要印出較多的文字。

```
All possible sequences of the cards dealt:

Sequence 1:
5 of Diamonds
K of Spades
7 of Spades
6 of Clubs
4 of Hearts
```

參考資料：

https://youtu.be/W3dtyZr8rcY
https://chat.openai.com/