

資料結構

Data Structure

Lab 04

姓名： 曾致嘉

學號： 113AB0014

Lab04-Q1

Exercise: Shuffling and Dealing Cards (Use Stack)

- 1) Shuffle a standard 52-card deck (excluding jokers).
- 2) Push the shuffled cards into a stack.
- 3) Draw cards from the stack and display the results.

Code

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <random>
#include <ctime>
using namespace std;
// 表示單張撲克牌的類別
class Card {
public:
    string colors; //儲存撲克牌的花色
    string rank; //儲存撲克牌的數值
    Card(string s, string r) : colors(s), rank(r) {} //建立 constructor 來初始化物件，
    當 Card 物件建立時，它會自動執行這個函式，並把 s 和 r 的值存入 colors 和
    rank
    void display() const { //顯示撲克牌的資訊
        cout << rank << " of " << colors << endl;
    }
};

// 實作 Stack
class Stack {
private:
    vector<Card> stack; //表示 stack 是一個能存放 Card 類別物件的 vector

public:
    void push(const Card& card) {
        stack.push_back(card); //將 card 加入 stack 的尾端
    }

    Card pop() {
        Card Last_Card = stack.back(); //取出最後的值
```

```

        stack.pop_back(); //清除最後的值
        return Last_Card; //回傳
    }

    bool isEmpty() const {
        return stack.empty();
    }
};

// 代表一副撲克牌的類別
class Deck {
private:
    vector<Card> cards; // 存放未洗牌的撲克牌
    Stack shuffledDeck; // 存放洗過的牌，用實作的 stack 來管理
public:
    Deck() { //建立 constructor 來初始化物件
        string colors[] = { "Hearts", "Diamonds", "Clubs", "Spades" }; //儲存撲克牌的花色
        string ranks[] = { "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A" }; //儲存撲克牌的數值

        //利用迴圈將 52 張牌加入 cards 這個 vector 裡面
        for (int i = 0; i < 4; i++) { // 四種花色
            for (int j = 0; j < 13; j++) { // 13 種點數
                cards.push_back(Card(colors[i], ranks[j]));
            }
        }
    }

    //洗牌(Hint:使用 shuffle 函數)
    void shuffleDeck() {
        random_shuffle(cards.begin(), cards.end()); //將 cards 裡面的牌打亂
        for (int i = 0; i < 52; i++) {
            shuffledDeck.push(cards[i]); //將打亂的牌加入 shuffledDeck 這個
stack
        }
    }
}

```

```

//發牌
void drawAllCards() {
    while(!shuffledDeck.isEmpty()) {
        shuffledDeck.pop().display(); //將 shuffledDeck 裡面的牌依序取出
        並顯示
    }
}

};

int main() {
    srand(time(0)); //建立隨機種子
    Deck deck; //建立 deck 產生 52 張撲克牌
    deck.shuffleDeck(); //進行洗牌並放入堆疊
    cout << "Shuffled deck:" << endl;
    deck.drawAllCards(); //依序取出堆疊內的牌並顯示
    return 0;
}

```

Discussion Section

First time

The screenshot shows a C++ IDE with two windows. The left window displays the source code for LAB4_Q1.cpp, and the right window shows the output of the program in a command prompt.

Source Code (LAB4_Q1.cpp):

```

1 #include <iostream>
2 using namespace std;
3 // 表示單張撲克牌的類別
4 class Card {
5 public:
6     string colors; //儲存撲克牌的花色
7     string rank; //儲存撲克牌的數字
8     Card(string s, string r) : colors(s), rank(r) {} //建立constructor來初始化物件
9     void display() const { //顯示撲克牌的資訊
10         cout << rank << " of " << colors << endl;
11     }
12 };
13 // 表示撲克牌的堆疊
14 class Stack {
15 private:
16     vector<Card> stack; //表示stack是一個能存放Card類別物件的vector
17 public:
18     void push(const Card& card) {
19         stack.push_back(card); //將card加入stack的尾端
20     }
21     Card pop() {
22         Card Last_Card = stack.back(); //取出最後的值
23         stack.pop_back(); //清除最後的值
24         return Last_Card; //回傳
25     }
26     bool isEmpty() const {
27         return stack.empty();
28     }
29 };
30 // 代表一副撲克牌的類別
31 class Deck {
32 private:
33     vector<Card> cards; //存放撲克牌的撲克牌
34     shuffledDeck: //儲存洗牌的牌，剛建立的stack與管理
35 };
36
37 int main() {
38     Deck deck; //建立deck產生52張撲克牌
39     deck.shuffleDeck(); //進行洗牌並放入堆疊
40     cout << "Shuffled deck:" << endl;
41     deck.drawAllCards(); //依序取出堆疊內的牌並顯示
42     return 0;
43 }

```

Output (Command Prompt):

```

K of Hearts
8 of Hearts
9 of Hearts
6 of Clubs
10 of Clubs
J of Diamonds
10 of Spades
10 of Hearts
6 of Diamonds
J of Spades
4 of Spades
2 of Clubs
A of Spades
Q of Diamonds
3 of Spades
8 of Clubs
9 of Spades
7 of Diamonds
8 of Spades
5 of Clubs
7 of Clubs
Q of Clubs
3 of Clubs
2 of Hearts
Q of Spades
J of Hearts
3 of Hearts
A of Hearts

```

Second time

```
LAB4_Q1.cpp
class Stack {
...
}

// 代表一副撲克牌的類別
class Deck {
private:
    vector<Card> cards; // 存放未洗過的撲克牌
    Stack shuffledDeck; // 存放洗過的牌，用實作的stack來管理
public:
    Deck() { // 建立constructor來初始化物件
        string colors[] = { "Hearts", "Diamonds", "Clubs", "Spades" }; // 儲存撲克牌
        string ranks[] = { "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K" };
        // 利用迴圈將52張牌加入cards這個vector裡面
        for (int i = 0; i < 4; i++) { // 四種花色
            for (int j = 0; j < 13; j++) { // 13 種點數
                cards.push_back(Card(colors[i], ranks[j]));
            }
        }
    }

    // 洗牌(Hint:使用shuffle函數)
    void shuffleDeck() {
        random_shuffle(cards.begin(), cards.end()); // 將cards裡面的牌打亂
        for (int i = 0; i < 52; i++) {
            shuffledDeck.push(cards[i]); // 將打亂的牌加入shuffledDeck這個stack
        }
    }

    // 發牌
    void drawAllCards() {
        while (!shuffledDeck.isEmpty()) {
            shuffledDeck.pop().display(); // 將shuffledDeck裡面的牌依序取出並顯示
        }
    }
};

int main() {
    srand(time(0));
}
```

```
C:\Windows\System32\cmd.exe
5 of Clubs
9 of Clubs
A of Spades
10 of Clubs
A of Hearts
J of Hearts
3 of Hearts
10 of Diamonds
9 of Diamonds
6 of Diamonds
5 of Diamonds
4 of Diamonds
3 of Clubs
7 of Spades
6 of Hearts
K of Hearts
K of Diamonds
A of Diamonds
2 of Hearts
Q of Diamonds
K of Clubs
7 of Hearts
10 of Spades
J of Spades
6 of Clubs
8 of Hearts
5 of Spades
K of Spades

C:\Users\user\Documents\程式碼\113-2-Data_Structure\LAB4>

int main() {
    srand(time(0));
    Deck deck; // 建立deck產生52張撲克牌
    deck.shuffleDeck(); // 進行洗牌並放入堆疊
    cout << "Shuffled deck:" << endl;
    deck.drawAllCards(); // 依序取出堆疊內的牌並顯示
}
```

Third time

```
LAB4_Q1.cpp
class Deck {
...
}

// 洗牌(Hint:使用shuffle函數)
void shuffleDeck() {
    random_shuffle(cards.begin(), cards.end()); // 將cards裡面的牌打亂
    for (int i = 0; i < 52; i++) {
        shuffledDeck.push(cards[i]); // 將打亂的牌加入shuffledDeck這個stack
    }
}

// 發牌
void drawAllCards() {
    while (!shuffledDeck.isEmpty()) {
        shuffledDeck.pop().display(); // 將shuffledDeck裡面的牌依序取出並顯示
    }
}

int main() {
    srand(time(0)); // 建立隨機種子
    Deck deck; // 建立deck產生52張撲克牌
    deck.shuffleDeck(); // 進行洗牌並放入堆疊
    cout << "Shuffled deck:" << endl;
    deck.drawAllCards(); // 依序取出堆疊內的牌並顯示
    return 0;
}
```

```
C:\Windows\System32\cmd.exe
4 of Clubs
J of Hearts
8 of Spades
9 of Clubs
J of Spades
9 of Diamonds
8 of Clubs
2 of Clubs
7 of Spades
10 of Clubs
K of Clubs
J of Clubs
3 of Hearts
K of Hearts
2 of Spades
10 of Diamonds
5 of Spades
4 of Hearts
6 of Diamonds
3 of Clubs
6 of Clubs
7 of Diamonds
A of Clubs
Q of Clubs
Q of Hearts
A of Diamonds
K of Diamonds

C:\Users\user\Documents\程式碼\113-2-Data_Structure\LAB4>

int main() {
    srand(time(0)); // 建立隨機種子
    Deck deck; // 建立deck產生52張撲克牌
    deck.shuffleDeck(); // 進行洗牌並放入堆疊
    cout << "Shuffled deck:" << endl;
    deck.drawAllCards(); // 依序取出堆疊內的牌並顯示
}
```