計算機程式語言

作　業　六

姓名	潘廣霖
學號	B03203004
原始程式檔名	HW06_002.cpp (_ = [A-B])

評　分　項　目

分數比重	項　目	得　分
40%	程式是否能正確執行?	
40%	程式之使用者介面與輸出結果?	
	是否有繳交原始程式檔與執行檔?	
	程式中的註解是否恰當?	
	程式之結構與邏輯是否正確?	
	程式碼的格式是否合乎要求?	
20%	程式之綜合評分	
總　分		

評語:

A.

```
//=================================================================
//  PROGRAMMER  : 潘廣霖
//  DATE        : 2015-12-17
//  FILENAME    : HW06A002.CPP
//  DESCRIPTION : A usable bridge game card shuffler.
//=================================================================
```

```
#include "stdafx.h"

#include<ctime>
#include<cstdlib>
#include<string>
#include<fstream>
#include<iostream>
#include<iomanip>
#include<cmath>
#include<algorithm>
#include "windows.h"

using namespace System;
using namespace std;

// non-printable characters for Windows
command line
// club, diamond, heart, spade, respectively
static const char POKER_FALLBACK[] = {  'C',
'D',  'H',  'S' };
static const char POKER_PATTERN[]  = { '\5',
'\4', '\3', '\6' };

void printPoker(ostream& os, int k, bool
fallback = false) {
    int sym = k / 13;
    if (sym == 1 || sym == 2)
        Console::ForegroundColor =
ConsoleColor::Red;
    else
        Console::ForegroundColor =
ConsoleColor::White;

    os << (fallback ? POKER_FALLBACK :
POKER_PATTERN)[k / 13]
       << setw(2) << right;

    int n = k % 13 + 1;
         if (n ==  1) { os << 'A'; }
    else if (n == 11) { os << 'J'; }
    else if (n == 12) { os << 'Q'; }
    else if (n == 13) { os << 'K'; }
    else              { os << n;   }

    Console::ResetColor();
}

int rnGen(int max) {
    double r;
    do {
        r = ((double)rand()) / RAND_MAX;
    } while (r == 1.0);

    return (int)(r * max);
}
```

```
// Fisher-Yates shuffle
void shuffle(int* begin, size_t len) {
    int *cur, *dest, tmp;
    cur = begin;
    for (size_t i = 0; i < len; i++) {
        dest = cur + rnGen(len - i);
        tmp = *cur;
        *cur = *dest;
        *dest = tmp;
        ++cur;
    }
}

void printCards(int a[52]) {
    for (int i = 0; i < 13; i++) {
        for (int j = 0; j < 4; j++) {
            int n = i * 4 + j;
            Console::SetCursorPosition(8 + 5 *
i, 7 + 3 * j);
            printPoker(cout, a[i + j * 13]);
            Sleep((int)(10+(n-52)*(n-
52)/24.0));
        }
    }
}

int main(array<System::String ^> ^args) {

    srand((unsigned)time(NULL));

    Console::SetCursorPosition(0, 0);

    int w = Console::BufferWidth;
    for (int l = 0; l < 3; l++) {
        Console::BackgroundColor =
System::ConsoleColor::DarkBlue;
        Console::ForegroundColor =
System::ConsoleColor::White;
        for (int i = 0; i < w; i++)
            Console::Write(" ");
    }

    string title("Bridge Card Game");
    Console::SetCursorPosition((w -
title.length()) / 2, 1);
    cout << title;

    int a[52], p[4];
    for (int i = 0; i < 52; i++)
        a[i] = i;

    Console::BackgroundColor =
System::ConsoleColor::Black;

    Console::SetCursorPosition(0, 3);
    while (true) {
        int q = 3;
        while (q--) {
            Console::SetCursorPosition(2, 5);

            Console::ForegroundColor =
System::ConsoleColor::White;
            cout << "Shuffling cards... ";
            Console::ForegroundColor =
System::ConsoleColor::Green;
            cout << " " << q << " ";
            Console::ResetColor();
            cout << "times remaining." <<
setw(18) << "";

            Console::WriteLine();

            shuffle(a, 52);

            for (int j = 0; j < 4; j++) {
                Console::SetCursorPosition(2,
7 + 3 * j);
                Console::ForegroundColor =
System::ConsoleColor::Green;
                cout << "| #" << (j + 1);
                Console::SetCursorPosition(8,
7 + 3 * j);
                cout << setw(77) << "";
            }

            printCards(a);

            // count for points
            for (int i = 0; i < 4; i++) {
                p[i] = 0;
                for (int j = 0; j < 13; j++) {
                    int n = a[i * 13 + j] % 13
+ 1;
                         if (n ==  1) p[i] +=
4;
                    else if (n == 13) p[i] +=
3;
                    else if (n == 12) p[i] +=
2;
                    else if (n == 11) p[i] +=
1;
                }
            }

            Sleep(200);
```

```
            for (int j = 0; j < 4; j++) {
                Console::SetCursorPosition(73,
7 + 3 * j);
                Console::ForegroundColor =
System::ConsoleColor::Green;
                cout << "PT=" << setw(2) <<
p[j];
            }

            Sleep(1000);
        }


        Console::SetCursorPosition(2, 5);
        Console::BackgroundColor =
System::ConsoleColor::White;
        Console::ForegroundColor =
System::ConsoleColor::DarkGreen;
        cout << "Finished!! [R]eplay / [W]rite
or other char to exit... ";
        Console::ResetColor();

        string c;
        cin >> c;
        if (c[0] == 'W' || c[0] == 'w') {
            ofstream f("BridgeCard.txt");
            f << "Bridge Card Game" << endl;
            for (int i = 0; i < 4; i++) {
                f << "Player #" << (i + 1) <<
"   ";
                for (int j = 0; j < 13; j++) {
                    printPoker(f, a[i * 13 +
j], true);
                    f << "  ";
                }
                f << "  PT=" << setw(2) <<
right << p[i] << endl;
            }
            f << "====== EOF ======" << endl;
            cout << "  OK! Written to
'BridgeCard.txt'. The program will end. ";
            system("pause");
            break;
        }
        else if (c[0] != 'R' && c[0] != 'r') {
            break;
        }
    }
    return 0;
}
```

B.

```cpp
//===============================================================
//  PROGRAMMER  : 潘廣霖
//  DATE        : 2015-12-17
//  FILENAME    : HW06B002.CPP
//  DESCRIPTION : A simple morse code encoder and player.
//===============================================================

#include "stdafx.h"
#include "windows.h"
#include<iostream>
#include<fstream>
#include<sstream>
#include<string>

using namespace System;
using namespace std;

int main(array<System::String ^> ^args) {
    const string MORSE[] = {
        ".-",  "-...", "-.-.", "-..", ".",
        "..-.", "--.", "....", "..", ".---",
        "-.-", ".-..", "--", "-.", "---",
        ".--.", "--.-", ".-.", "...", "-",
        "..-", "...-", ".--", "-..-", "-.--",
        "--.."};
    const string COMMA = "--..--";
    const string FULLSTOP = ".-.-.-";
    const int FREQ = 440; // 4A
    const int DUR = 80;

    ifstream f("MorseCode.txt");
    if (!f) {
        cout << "MorseCode.txt does not
exist!" << endl;
        system("pause");
        return 1;
    }

    char c;
    stringstream ss;
    while (1) {
        c = f.get();
        if (c == -1) break;
        if (c >= 'A' && c <= 'Z') c = c - 'A'
+ 'a';
        if (c >= 'a' && c <= 'z') {
            ss << MORSE[c - 'a'] << " ";
        } else if (c == ' ') {
            ss << "  ";
        } else if (c == ',') {
            // comsume spaces after commas or
periods.
            while (f.peek() == ' ')
f.ignore(1);
            ss << COMMA << " ";
        } else if (c == '.') {
            while (f.peek() == ' ')
f.ignore(1);
            ss << FULLSTOP << " ";
        }
    }

    ofstream of("MorseCode.dat");
    of << ss.rdbuf();
    of.close();

    cout << "Ready to play the Morse code of
MorseCode.txt" << endl;
    cout << "Data had been saved to
MorseCode.dat" << endl;
    cout << "Would you like to have sound?
('Y' for yes, otherwise no): ";
    cin >> c;
    bool snd = (c == 'Y' || c == 'y');
    int spc = 0;

    ss.seekg(0);
    while (1) {
        c = ss.peek();
        if (c == -1) break;
        if (c != ' ') ss.ignore(1);

        spc = 0;
        while (c == ' ') {
            ss.ignore(1);
            spc++;
            c = ss.peek();
        }

        if (spc) {
            if (snd) {
                if (spc == 3)
                    Sleep(DUR * 7);
                else if (spc == 1)
                    Sleep(DUR * 3);
                else return 3;
            }
            while (spc--) cout << ' ';

        } else {
            cout << c;
            if (snd) {
                if (c == '.')
                    Console::Beep(FREQ, DUR);
                else if (c == '-')
                    Console::Beep(FREQ, DUR *
3);

                Sleep(DUR);
            }
        }
    }

    cout << endl;
    system("pause");

    return 0;
}
```