汉诺塔综合演示

国06 2252075 刘文飞. 12.4

1. 综合题1-汉诺塔综合演示

将之前的汉诺塔的小程序集成在一个小程序中，能够通过菜单进行选择，满足图形化演示的要求

2. 整体设计思路

三个主体cpp，分别放主函数，菜单函数，以及完成各项汉诺塔任务的函数，通过主函数进行各个函数的调用。

完成main函数与menu函数设计与实现，这两项工作应该是比较轻松的，主要任务是如何将之前的各个程序集成在一个cpp中，如何划分函数，如何通过不同参数实现函数的高效利用是面临的主要问题。

除了之前的小程序外，也要学习如何实现图形化演示，分别完成5，6，7，8，9.

3. 主要功能的实现

3.1 递归函数

Hanoi的递归函数是整个程序逻辑的基础，通过这个递归函数得以有每一步如何进行，我们通过在这个递归函数中再加入其他函数的调用来实现更多功能，并且使得该递归函数精简。

3.2 对应每个任务完成得函数调用

3.2.1 基本解

基本解不需要复杂的函数调用，只需要打印出每一步的移动方向以及盘的编号

3.2.2 基本解（步数记录）

与3.2.1的基本解相比，需要添加一个全局变量num来记录步数

3.2.3 内部数组显示（横向）

在3.2.2基础上添加了全局二维数组的使用，该程序分为两大部分，一是如何在三个数组（代表三根柱子）之间交换数据（实现盘子的移动），二是如何将内部数组显示出来。

一，通过一个chu_ru函数的调用来实现数组之间数据的交换，以src，dst实参传入函数具体实现

二，调用一个横向打印的函数来实现

三，

3.2.4 内部数组显示（纵向+横向）

在3.2.3的基础上添加了纵向数组的显示，第一个关键在于将横向打印转换为纵向打印，实际上是没有什么区别，思路相同，注意一些细节的处理。难点在于如何不在每次移动就清屏（会导致闪烁）的条件下实现纵向数组的打印：通过光标移动函数，用空格抹去原位置，并在现在的位置进行打印。

3.2.5 图形解-预备-画三个圆柱

学会工具函数中cct_showch的使用，计算好基位置进行横向纵向打印

3.2.6 图形解-预备-在起始柱上画n个盘子

在3.2.5的基础上，应用循环与cct_showch函数的结合打印n个盘子，并且用盘子的编号来标记盘子的颜色。

3.2.7 图形解-预备-第一次移动/3.2.8 图形解-自动移动版本

盘子的移动效果实际上是不断打印擦除的过程，那么重点就在于如何实现这个效果：上移与下移可之间通过计算盘高与盘子的大小直接实现，而左移与右移可通过判断src与dst的相对位置来实现，值得注意的一点是，在盘子的移动中基位置显得尤为重要，关乎能不能快速完成这个程序

3.2.9 图形解-游戏版

该程序不存在递归函数，需要调用之前的画柱子，画盘子函数，以及横向纵向显示函数，通过每次不同的输入参数来传入函数实现不同的效果，主要在于如何处理各种错误，大盘不能压小盘，源柱不能为空，以及游戏结束时怎么判断

3.3 通过一个函数输入每个程序需要的参数

用到第六章指针的知识来通过函数形参作为实参指针来改变多个实参值

调试过程碰到的问题

4.1 纵向打印时第一次选择了打印一次清一次屏

直接重新写第四个程序（痛苦面具），推倒重来，耗时耗力，第一次没有注意写完善

解决方法：再写一个

4.2 运行完三后再运行四时出现奇怪的结果（连续运行两次三或四也一样）

没有在每个小程序运行前或者运行后初始化计数的num和三个二维数组和三个一维数组

解决方法：在输入参数时就初始化这些全局变量

4.3 移动盘子时柱子消失

解决方法：在直接用空格缺省色覆盖前一位置时，在原来柱子位置打印一个色块

5. 心得体会

5.1 本次作业的心得体会，经验教训

第一次完成大作业，对我来说是耗时耗力的有以下几点原因，一是之前程序没有应用函数的意识，直接整个写在main函数里面，一坨东西，思路既不清晰也难以去修改。二是耗费大量时间反复横跳，没有很清晰的思路，应该先在纸上或者其他地方列出大纲再进行各个击破。三是第四个程序推倒重来浪费大量时间，第一次完成时没有按要求完成（出来混迟早要还），四是在写程序时要保持思路清晰，时常出现写着写着不知道自己在干什么的情况

5.2 做一些较为复杂的程序时，是分为若干小题还是一道大题

分为若干小题更好，分成若干小题实际上时将一个大问题分解成若干个小问题，一个大问题若是直接拿过来可能没什么思路，但若是将其分解成多个小程序，有利于理顺写程序的思路，知道自己应该去做什么工作，不会发生混乱以及遗漏一些重要的点；

很重要的一点是如果直接上手写一道大题会很难划分好函数，导致代码冗长，利用效率底下，时间成本高，收益少；并且，若是分成若干道小题也方便查出错误，可以找到具体哪一部分出现了问题，不会像无头苍蝇一样不知道到底应该查那一部分，浪费大量时间，还不一定能够纠正

由此看来，若是将复杂程序分解成一个个小题，不仅可以降低整体问题的难度，还可以节省时间成本，提高效率，便于纠错，便于函数的划分，使得能够思路清晰地完成程序。

5.3 总结汉诺塔若干小题之间的关系，如何在后面程序高效利用前面程序的代码，重用代码

基本解的两个程序差别不大，主要区别就是步数记录需要加上一个全局变量来记录步数

内部数组显示的两个程序，横向与纵向的显示差别不大，只是将xy轴做了一个交换，但纵向显示的程序难度的提升在于光标函数的使用，同时结合横向打印的函数，利用前面的代码，来实现这个程序

56789共同构成图形解，是一步一步递进的过程，将画柱子，画盘子的函数反复使用。在写第7个程序时就要考虑到写出来的盘图形化移动的函数也要在第八个第九个程序中能够使用，不能把参数写死，要灵活。

另外，在输入各个程序所需的函数中，要划分好具体的执行语句

总结一下，函数想要能够高效利用，一个是要有普遍性，能够在多个环境下都能使用，这就要考虑到如何设计参数来满足这个目的。二是要有针对性，即在函数内部针对不同的参数要实现相对不同的功能

5.4 以本次作业为例，说明如何才能更好利用函数来编写复杂的程序

函数的参数设计要合理，在形参的名字上可以略加提示，使人能够知道这个参数是用来干什么的，参数个数够用即可，不要在一个函数中搞过多参数，容易混乱，导致函数复用性下降，难以维护，

尽量一个具体的函数来完成一个工作，函数内部逻辑要清晰，每一个函数都不要写得冗长

构成复杂程序中的函数会用到嵌套，不要搞得嵌套层数过多，思维容易混乱

6. 附件：源程序

```
/* 国06 2252075 刘文飞 */
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <iomanip>
#include <tchar.h>
#include <string.h>
#include <conio.h>
#include <Windows.h>
#include "cmd_console_tools.h"
#include"hanoi.h"
using namespace std;
/* ----------------------------------------
----------------------------------------
-

    本文件功能：
    1、存放被 hanoi_main.cpp 中根据菜单返回值调用的各菜单项对应的执行函数

    本文件要求：
    1、不允许定义外部全局变量（const 及 #define 不在限制范围内）
    2、允许定义静态全局变量（具体需要的数量不要超过文档显示，全局变量的使用准则是：少用、慎用、能不用尽量不用）
    3、静态局部变量的数量不限制，但使用准则也是：少用、慎用、能不用尽量不用
    4、按需加入系统头文件、自定义头文件、命名空间等

    ----------------------------------------
    ---------------------------------------
    -- */

int num = 0;
int top[3] = {};
int tower[3][10] = {};
int n_zhi(char src) {
    if (src == 'A')
        return tower[0][top[0]];
    if (src == 'B')
        return tower[1][top[1]];
    if (src == 'C')
        return tower[2][top[2]];
    else
        return 0;
}
void move(char src, char dst, int x, int sol) {
    if (src == 'A') {
        cct_gotoxy(x, 20 + sol * 10 - top[0]);
        cout << "  ";
    }
    if (src == 'B') {
        cct_gotoxy(x + 10, 20 + sol * 10 - top[1]);
        cout << "  ";
    }
    if (src == 'C') {
        cct_gotoxy(x + 20, 20 + sol * 10 - top[2]);
```

```
        cout << "  ";
    }
    if (dst == 'A') {
        cct_gotoxy(x + 1, 20 + sol * 10 -
top[0] + 1);
        cout << tower[0][top[0] - 1];
    }
    if (dst == 'B') {
        cct_gotoxy(x + 10 + 1, 20 + sol *
10 - top[1] + 1);
        cout << tower[1][top[1] - 1];
    }
    if (dst == 'C') {
        cct_gotoxy(x + 20 + 1, 20 + sol *
10 - top[2] + 1);
        cout << tower[2][top[2] - 1];
    }

}
void zong_xiang_dayin(int x, int y, int
T[][10], int top, int Tower) {
    cct_gotoxy(x, y);
    int i;
    for (i = 1; i <= top; i++) {
        cout << setw(2) << T[Tower][i - 1];
        cct_gotoxy(x, y - i);
    }
}
void heng_xiang_dayin(int xianshi, int n,
char src, char dst, int sol) {
    cct_gotoxy(0, 25 + sol * 10);
    cout << "第" << setw(4) << ++num << ' '
<< "步(" << setw(2) << n << ")" << ": " <<
src << "-->" << dst;
    if (xianshi) {
        int i;
        cout << " A:";
        for (i = 0; i < top[0]; i++) {
            cout << setw(2) << tower[0][i];
        }
        for (i = 9; i >= top[0]; i--) {
            cout << "  ";
        }
        cout << " B:";
        for (i = 0; i < top[1]; i++) {
            cout << setw(2) << tower[1][i];
        }
        for (i = 9; i >= top[1]; i--) {
            cout << "  ";
        }
        cout << " C:";
        for (i = 0; i < top[2]; i++) {
            cout << setw(2) << tower[2][i];
        }
        for (i = 9; i >= top[2]; i--) {
            cout << "  ";
        }
        cout << endl;
        return;

    }
    else
        return;
}
void chu_ru_(char src, char dst) {
    int out = 0;
    if (src == 'A')
        out = tower[0][--top[0]];
    if (src == 'B')
        out = tower[1][--top[1]];
    if (src == 'C')
        out = tower[2][--top[2]];
    if (dst == 'A')
        tower[0][top[0]++] = out;
    if (dst == 'B')
        tower[1][top[1]++] = out;
    if (dst == 'C')
        tower[2][top[2]++] = out;
}
void pan_move(char src, char dst, int n,
int t);
void da_yin(int s, int n, char src, char
dst, int xianshi, int t) {
    if (s == 1) {
        cout << setw(3) << n << '#' << ' '
<< src << "-->" << dst << endl;
    }
    if (s == 2) {
        cout << setw(5) << ++num << ':' <<
setw(3) << n << '#' << ' ' << src << "-->"
<< dst << endl;
    }
    if (s == 3) {
        chu_ru_(src, dst);
        cout << "第" << setw(4) << ++num <<
' ' << "步(" << setw(2) << n << ")" << ": "
```

```
<< src << "-->" << dst;
        int i;
        cout << " A:";
        for (i = 0; i < top[0]; i++) {
            cout << setw(2) << tower[0][i];
        }
        for (i = 9; i >= top[0]; i--) {
            cout << "  ";
        }
        cout << " B:";
        for (i = 0; i < top[1]; i++) {
            cout << setw(2) << tower[1][i];
        }
        for (i = 9; i >= top[1]; i--) {
            cout << "  ";
        }
        cout << " C:";
        for (i = 0; i < top[2]; i++) {
            cout << setw(2) << tower[2][i];
        }
        for (i = 9; i >= top[2]; i--) {
            cout << "  ";
        }
        cout << endl;
    }
    if (s == 4) {
        if (t != 0) {
            chu_ru_(src, dst);
            move(src, dst, base_Ax, 0);
            heng_xiang_dayin(xianshi,    n,
src, dst, 0);
            Sleep(1050 - t * 200);
        }
        else {
            while (1) {
                if (getchar() == '\n')
                    break;
            }
            chu_ru_(src, dst);
            move(src, dst, base_Ax, 0);
            heng_xiang_dayin(xianshi,    n,
src, dst, 0);

        }
    }


    if (s == 8) {
        if (t != 0) {
            chu_ru_(src, dst);
            move(src, dst, base_Ax, 1);
           heng_xiang_dayin(1, n, src, dst,
1);
            pan_move(src, dst, n, t);
            cct_setcolor();
            if (n == 1)
                cct_gotoxy(0, 25);
        }
        else {
            while (1) {
                if (getchar() == '\n')
                    break;
            }
            chu_ru_(src, dst);
            move(src, dst, base_Ax, 1);
           heng_xiang_dayin(1, n, src, dst,
1);
            pan_move(src, dst, n, t);
            cct_setcolor();
            if (n == 1)
                cct_gotoxy(0, 25);
        }
    }
}
void hanoi(int n, char src, char tmp, char
dst, int s, int xianshi, int t)
{
    if (n == 1) {
        da_yin(s, n, src, dst, xianshi, t);
    }
    else {
        hanoi(n - 1, src, dst, tmp, s,
xianshi, t);
        da_yin(s, n, src, dst, xianshi, t);
        hanoi(n - 1, tmp, src, dst, s,
xianshi, t);
    }
}
void chu_shi_hua(int* n, char* src, char*
tmp, char* dst, int sol, int* xianshi, int*
t) {
    num = 0;
    int i, j;
    for (i = 0; i < 3; i++)
        top[i] = 0;
    for (i = 0; i < 2; i++) {
```

```
        for (j = 0; j < 10; j++)
            tower[i][j] = 0;
    }

    int t1, t2;
    int n1;
    while (1) {

        cout << "请输入汉诺塔的层数(1-10)"
<< endl;
        cin >> n1;
        if (cin.fail()) {
            cin.clear();
            cin.ignore(10000000, '\n');
            continue;
        }
        if (n1 < 1 || n1>10) {
            cin.ignore(10000000, '\n');
            continue;
        }

        else {
            cin.ignore(10000000, '\n');
            break;
        }
    }
    *n = n1;
    while (1) {
       cout << "请输入起始柱(A-C)" << endl;
        t1 = getchar();
        while ((getchar()) != '\n');
        if (t1 == 65 || t1 == 66 || t1 ==
67 || t1 == 97 || t1 == 98 || t1 == 99)
            break;
        else
            continue;
    }
    while (1) {
       cout << "请输入目标柱(A-C)" << endl;
        t2 = getchar();
        while ((getchar()) != '\n');
        if ((t2 == 65 || t2 == 66 || t2 ==
67 || t2 == 97 || t2 == 98 || t2 == 99) &&
t1 != t2)
            break;
        else if (t1 == t2) {
            if (t1 == 65 || t1 == 97)
                cout << "目标柱(A)不能与起
始柱(A)相同" << endl;
            if (t1 == 66 || t1 == 98)
                cout << "目标柱(B)不能与起
始柱(B)相同" << endl;
            if (t1 == 67 || t1 == 99)
                cout << "目标柱(C)不能与起
始柱(C)相同" << endl;
            continue;
        }
        else
            continue;
    }
    if (t1 == 65 || t1 == 97)
        *src = 'A';
    else if (t1 == 66 || t1 == 98)
        *src = 'B';
    else
        *src = 'C';
    if (t2 == 65 || t2 == 97)
        *dst = 'A';
    else if (t2 == 66 || t2 == 98)
        *dst = 'B';
    else
        *dst = 'C';
    if (*src != 'A' && *dst != 'A')
        *tmp = 'A';
    if (*src != 'B' && *dst != 'B')
        *tmp = 'B';
    if (*src != 'C' && *dst != 'C')
        *tmp = 'C';
    if (sol == 3 || sol == 4 || sol == 7 ||
sol == 8 || sol == 9) {
        for (num = 0; num < *n; num++) {
            if (*src == 'A')
                tower[0][num] = *n - num;
            if (*src == 'B')
                tower[1][num] = *n - num;
            if (*src == 'C')
                tower[2][num] = *n - num;
        }
        if (*src == 'A')
            top[0] = *n;
        if (*src == 'B')
            top[1] = *n;
        if (*src == 'C')
            top[2] = *n;
        if (sol == 4 || sol == 8) {
            cout << "请输入移动速度(0-5: 0-
```

```
按回车单步演示 1-延时最长 5-延时最短)" << endl;
            int t1;
            cin >> t1;
            *t = t1;
            if (sol == 4) {
                cout << "请输入是否显示内部数组值(0-不显示 1-显示)" << endl;
                int xianshi1;
                cin >> xianshi1;
                *xianshi = xianshi1;
            }
        }
    }
}
void hua_ta() {
    cct_setcursor(CURSOR_INVISIBLE);
    cct_showch(base_x,  base_y,  '  ', COLOR_HYELLOW, COLOR_HYELLOW, 23);
    cct_showch(base_x + 30, base_y, ' ', COLOR_HYELLOW, COLOR_HYELLOW, 23);
    cct_showch(base_x + 60, base_y, ' ', COLOR_HYELLOW, COLOR_HYELLOW, 23);
    int i = 1;
    for (i = 1; i <= 13; i++) {
        cct_showch(base_x + 11, base_y - i, ' ', COLOR_HYELLOW, COLOR_HYELLOW, 1);
        Sleep(100);
        cct_showch(base_x + 30 + 11, base_y - i, ' ', COLOR_HYELLOW, COLOR_HYELLOW, 1);
        Sleep(100);
        cct_showch(base_x + 60 + 11, base_y - i, ' ', COLOR_HYELLOW, COLOR_HYELLOW, 1);
        Sleep(100);
    }
    cct_setcolor();
}
void hua_pan(char src, int n) {
    int i;
    for (i = 1; i <= n; i++) {
        cct_showch(base_x + (src - 'A') * 30 + (10 - n + i), base_y - i, ' ', n - i + 1, n - i + 1, 2 * (n - i + 1) + 1);
        Sleep(100);
    }
    cct_setcolor();
}
void pan_move(char src, char dst, int n, int t) {

    int y;
    /*上*/
    for (y = base_y - top[0 + src - 'A'] - 1; y >= MIN_Y; y--) {

        cct_showstr(base_x + 1 + (10 - n) + (src - 'A') * 30, y, "  ", n, n, 2 * n + 1);
        Sleep(170 - t * 30);


        if (y > MIN_Y) {

            cct_showch(base_x + 1 + (10 - n) + (src - 'A') * 30, y, ' ', COLOR_BLACK, COLOR_WHITE, 2 * n + 1);
            if (y > MIN_Y - 1) {
                cct_showch(base_x + 11 + (src - 'A') * 30, y, ' ', COLOR_HYELLOW, COLOR_HYELLOW, 1);
            }
        }
    }
    /*左*/
    if (src > dst) {
        int x;
        for (x = 0; x < (src - dst) * 30; x++) {
            cct_showstr(base_x + 1 + (10 - n) + (src - 'A') * 30 - x, MIN_Y, "  ", n, n, 2 * n + 1);

            Sleep(170 - t * 30);


            if (x < (src - dst) * 30) {

                cct_showch(base_x + 1 + (10 - n) + (src - 'A') * 30 - x, MIN_Y, ' ', COLOR_BLACK, COLOR_WHITE, 2 * n + 1);

            }
        }
    }

    /*右*/
    if (src < dst) {
```

```
        int x;
        for (x = 0; x < (dst - src) * 30;
x++) {
            cct_showstr(base_x + 1 + (10 -
n) + (src - 'A') * 30 + x, MIN_Y, " ", n,
n, 2 * n + 1);

            Sleep(170 - t * 30);


            if (x < (dst - src) * 30) {

                cct_showch(base_x + 1 + (10
- n) + (src - 'A') * 30 + x, MIN_Y, ' ',
COLOR_BLACK, COLOR_WHITE, 2 * n + 1);
            }
        }
    }
    /*下*/
    for (y = MIN_Y; y <= base_y - top[0 +
dst - 'A']; y++) {

        cct_showstr(base_x + 1 + (10 - n) +
(dst - 'A') * 30, y, " ", n, n, 2 * n + 1);



        Sleep(170 - t * 30);

        if (y < base_y - top[0 + dst - 'A'])
{

            cct_showch(base_x + 1 + (10 -
n) + (dst - 'A') * 30, y, ' ', COLOR_BLACK,
COLOR_WHITE, 2 * n + 1);
            if (y != MIN_Y)
                cct_showch(base_x + 11 +
(dst - 'A') * 30, y, ' ', COLOR_HYELLOW,
COLOR_HYELLOW, 1);

        }
    }
}
void pan__move(char tower1, char tower2) {
    chu_ru_(tower1, tower2);
    move(tower1, tower2, base_Ax, 1);
    heng_xiang_dayin(1,    (n_zhi(tower1)),
tower1, tower2, 1);
    pan_move(tower1,                tower2,
(n_zhi(tower1)), 1);
    cct_setcolor();
}

void solution1() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 1,
&xianshi, &t);
    cout << "移动步骤为:" << endl;
    hanoi(n, src, tmp, dst, 1, 0, 0);
    cout << endl;
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
/****************************************
**********************************
  函数名称:
  功    能:
  输入参数:
  返 回 值:
  说    明:
****************************************
**********************************/
void solution2() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 2,
&xianshi, &t);
    hanoi(n, src, tmp, dst, 2, 0, 0);
    cout << endl;
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
/****************************************
**********************************
  函数名称:
  功    能:
```

```
  输入参数：
  返 回 值：
  说    明：
*****************************************
*********************************/
void solution3() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 3,
&xianshi, &t);
    cout << "初始:               ";
    int i;
    cout << " A:";
    for (i = 0; i < top[0]; i++) {
        cout << setw(2) << tower[0][i];
    }
    for (i = 9; i >= top[0]; i--) {
        cout << "  ";
    }
    cout << " B:";
    for (i = 0; i < top[1]; i++) {
        cout << setw(2) << tower[1][i];
    }
    for (i = 9; i >= top[1]; i--) {
        cout << "  ";
    }
    cout << " C:";
    for (i = 0; i < top[2]; i++) {
        cout << setw(2) << tower[2][i];
    }
    for (i = 9; i >= top[2]; i--) {
        cout << "  ";
    }
    cout << endl;
    num = 0;
    hanoi(n, src, tmp, dst, 3, 0, 0);
    cout << endl;
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();

}
/*****************************************
*********************************
  函数名称：
  功    能：
  输入参数：
  返 回 值：
  说    明：
*****************************************
*********************************/
void solution4() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 4,
&xianshi, &t);
    cct_cls();
    num = 0;
    zong_xiang_dayin(base_Ax,      base_Ay,
tower, top[0], 0);
    zong_xiang_dayin(base_Ax + 10, base_Ay,
tower, top[1], 1);
    zong_xiang_dayin(base_Ax + 20, base_Ay,
tower, top[2], 2);
    cct_gotoxy(base_Ax - 5, base_Ay + 1);
    cout                                  <<
"==================================="  <<
endl;
    cout << "               A            B
C" << endl;
    /*打印所有的盘*/
    cct_gotoxy(0, 0);
    cout << "从 " << src << " 移动到 " <<
dst << ", " << "共 " << n << " 层，延时设置
为 " << t << ", ";
    if (xianshi == 0)
        cout << "不";
    cout << "显示内部数组值" << endl;
    while (1) {
        if (getchar() == '\n')
            break;
    }
    hanoi(n, src, tmp, dst, 4, xianshi, t);
    cct_gotoxy(0, 28);
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
```

```
/*************************************************
**********************************
  函数名称:
  功    能:
  输入参数:
  返 回 值:
  说    明:
*************************************************
**********************************/
void solution5() {
    cct_cls();
    hua_ta();

    cct_gotoxy(0, 28);
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
/*************************************************
**********************************
  函数名称:
  功    能:
  输入参数:
  返 回 值:
  说    明:
*************************************************
**********************************/
void solution6() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 6,
&xianshi, &t);
    cct_cls();
    cout << "从 " << src << " 移动到 " <<
dst << ", " << "共 " << n << " 层 " << endl;
    hua_ta();
    hua_pan(src, n);
    cct_gotoxy(0, 28);
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
/*************************************************
**********************************
  函数名称:
  功    能:
  输入参数:
  返 回 值:
  说    明:
*************************************************
**********************************/
void solution7() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 7,
&xianshi, &t);
    num = 0;
    cct_cls();
    cout << "从 " << src << " 移动到 " <<
dst << ", " << "共 " << n << " 层 " << endl;
    hua_ta();
    hua_pan(src, n);

    if (n % 2) {
        chu_ru_(src, dst);
        pan_move(src, dst, 1, 1);
    }
    if (n % 2 - 1) {
        chu_ru_(src, tmp);;
        pan_move(src, tmp, 1, 1);
    }
    cct_setcolor();
    cct_gotoxy(0, 28);
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
/*************************************************
**********************************
  函数名称:
  功    能:
  输入参数:
  返 回 值:
  说    明:
*************************************************
**********************************/
```

```
void solution8() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 8,
&xianshi, &t);
    cct_cls();
    num = 0;
    cct_cls();
    zong_xiang_dayin(base_Ax, base_Ay + 10,
tower, top[0], 0);
    zong_xiang_dayin(base_Ax + 10, base_Ay
+ 10, tower, top[1], 1);
    zong_xiang_dayin(base_Ax + 20, base_Ay
+ 10, tower, top[2], 2);
    cct_gotoxy(base_Ax - 5, base_Ay + 11);
    cout                                  <<
"==================================="  <<
endl;
    cout << "               A           B
C" << endl;
    /*打印所有的盘*/
    cct_gotoxy(0, 0);
    cout << "从 " << src << " 移动到 " <<
dst << ", " << "共 " << n << " 层，延时设置
为 " << t << endl;
    hua_ta();
    hua_pan(src, n);
    Sleep(500);
    while (1) {
        if (getchar() == '\n')
            break;
    }
    hanoi(n, src, tmp, dst, 8, 0, t);
    cct_gotoxy(0, 38);
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
void solution9() {
    int n;
    char src, tmp, dst;
    int t = 0;
    int xianshi = 0;
    chu_shi_hua(&n, &src, &tmp, &dst, 9,
&xianshi, &t);
    cct_cls();
    num = 0;
    zong_xiang_dayin(base_Ax, base_Ay + 10,
tower, top[0], 0);
    zong_xiang_dayin(base_Ax + 10, base_Ay
+ 10, tower, top[1], 1);
    zong_xiang_dayin(base_Ax + 20, base_Ay
+ 10, tower, top[2], 2);
    cct_gotoxy(base_Ax - 5, base_Ay + 11);
    cout                                  <<
"==================================="  <<
endl;
    cout << "               A           B
C" << endl;
    /*打印所有的盘*/
    cct_gotoxy(0, 0);
    cout << "从 " << src << " 移动到 " <<
dst << ", " << "共 " << n << " 层" << endl;
    hua_ta();
    hua_pan(src, n);
    cct_gotoxy(0, 35);
    cout << "初始:                ";
    int i;
    cout << " A:";
    for (i = 0; i < top[0]; i++) {
        cout << setw(2) << tower[0][i];
    }
    for (i = 9; i >= top[0]; i--) {
        cout << "  ";
    }
    cout << " B:";
    for (i = 0; i < top[1]; i++) {
        cout << setw(2) << tower[1][i];
    }
    for (i = 9; i >= top[1]; i--) {
        cout << "  ";
    }
    cout << " C:";
    for (i = 0; i < top[2]; i++) {
        cout << setw(2) << tower[2][i];
    }
    for (i = 9; i >= top[2]; i--) {
        cout << "  ";
    }
    cout << endl;
    cout << "请输入移动的柱号(命令形式：
AC=A 顶端的盘子移动到 C，Q=退出) ：";
```

```
    while (1) {
        char tower1, tower2;
        while (1) {
            cct_gotoxy(60, 36);
            cout << "    ";
            cct_gotoxy(60, 36);
            cin >> tower1;
            if (cin.fail()) {
                cin.clear();
               cin.ignore(10000000, '\n');
                continue;
            }
            if (tower1 == 'q' || tower1 == 'Q')
                break;
            cin>> tower2;
            if (cin.fail()) {
                cin.clear();
               cin.ignore(10000000, '\n');
                continue;
            }
            if (tower1 == tower2) {
               cin.ignore(10000000, '\n');
                continue;
            }
            if (tower1 == 'A' || tower1 == 'B' || tower1 == 'C' || tower1 == 'a' || tower1 == 'b' || tower1 == 'c') {
                if (tower2 == 'A' || tower2 == 'B' || tower2 == 'C' || tower2 == 'a' || tower2 == 'b' || tower2 == 'c') {
                    if (tower1 > 'C')
                        tower1 -= 32;
                    if (tower2 > 'C')
                        tower2 -= 32;
                    if (tower1 == tower2) {

    cin.ignore(10000000, '\n');
                        continue;
                    }
                   if ((getchar()) == '\n')
{
                        if ((top[tower1 - 'A'] == 0)) {
                            cout << "源柱为空!!! " << endl;
                            Sleep(500);
                            cct_gotoxy(0, 37);
                            cout    <<    "
";
                            continue;
                        }
                        if   ((tower[tower1 - 'A'][top[tower1 - 'A']+1] > tower[tower2 - 'A'][top[tower2 - 'A']+1])) {
                            cout << "大盘压小盘，非法移动" << endl;
                            Sleep(500);
                            cct_gotoxy(0, 37);
                            cout    <<    "
";
                            continue;
                        }
                        break;
                    }
                }
            }

            cin.ignore(10000000, '\n');
        }/*输入命令*/
        if (tower1 == 'q' || tower1 == 'Q') {
            cout << "游戏结束!!!" << endl;
            break;
        }

        pan__move(tower1, tower2);
        if (top[dst - 'A'] == n) {
            cct_gotoxy(0, 37);
            cout << "游戏结束!!!" << endl;
            break;
        }
    }
    cct_gotoxy(0, 38);
    cout << "按回车键继续...";
    while (_getch() != '\r')
        ;

    cct_cls();
}
```