计算机科学与工程学院实验报告（首页）

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **课程名称** | | **Callmysoft** | | | **班级** | | **19网络工程1班** | |
| **实验名称** | | **实验3** | | | **教导教师** | | | **李志远** |
| **姓名** | **罗宏亮** | | **日期** | **2022.5.15** | |
|  |  | |  | |  | |  | | |

**一、实验目的**

1. (简答题)

学习C++primer 4一书5-11章节内容后，完成以下作业：

1、完成一个控制台程序，具体实现的功能为：斗地主发牌器，能随机发出三个人的牌和三张底牌，发出来的全部牌直接展示即可，三个人的牌需要由牌的大小由大到小排序展示;

2、提交内容为：设计思路的文档、项目代码和可执行程序release版本；

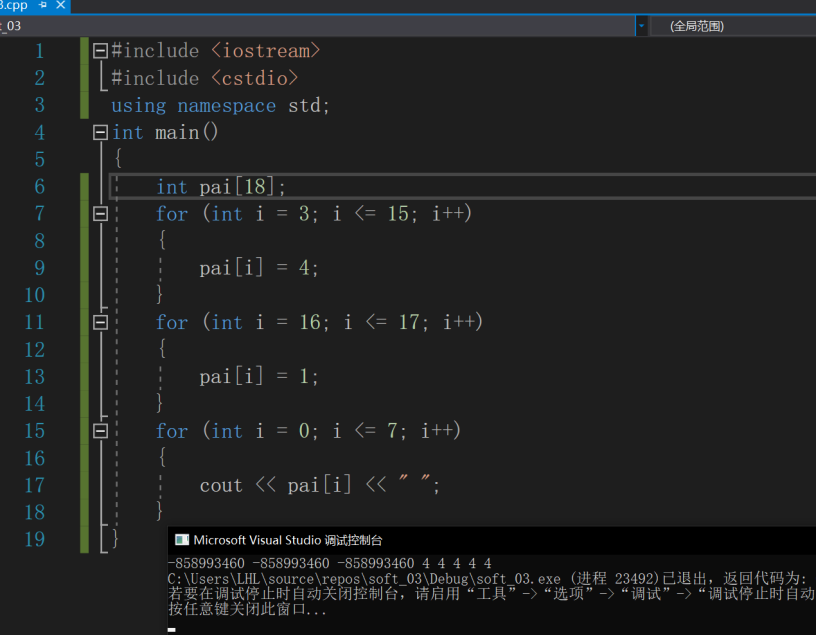
3、反馈学习期间遇到的问题和需要解决的问题、需要公司提供的帮助等。

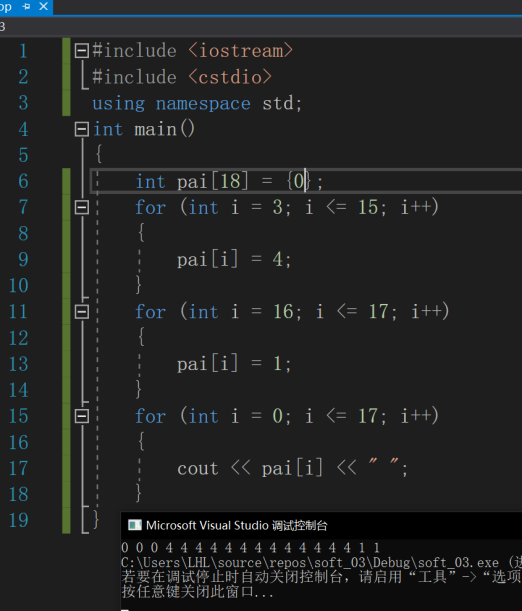
1. **实验过程**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 牌 |  |  |  | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | J | Q | K | A | 2 | X | Y |  |  |  |  |
| 个数 | 0 | 0 | 0 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 1 | 1 |  |  |  | 54 |
| 数组编号 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |  |  |  |  |

创建一个长度为18的数组

要先初始化为0，不然为这样：





rand()产生的是伪随机数，每次执行的结果是相同的

为了避免每次生成固定的随机数，引进srand()函数。该函数的功能是初始化随机数发生器，同样在头文件cstdlib中。

使用srand((unsigned)time(NULL))的方法来产生不同的随机种子。需要引入头文件 time.h

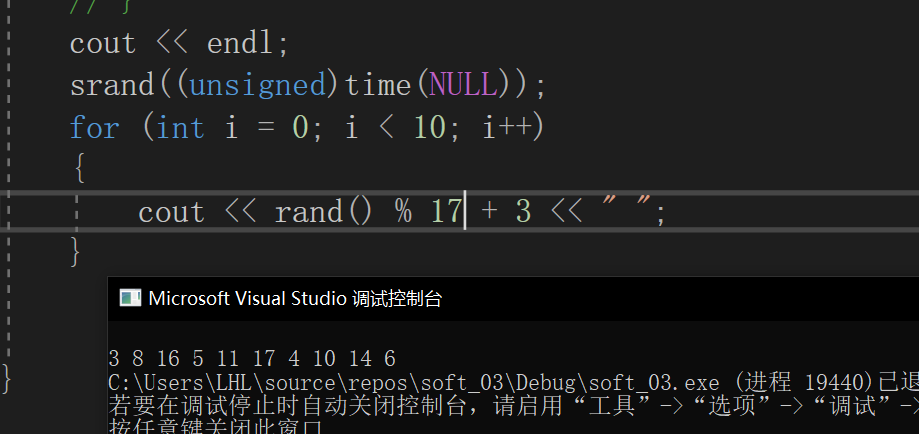
srand((unsigned)time(NULL));

for (int i = 0; i < 10; i++)

{

cout << rand() % 15 + 3<< " ";

}生成3-17中10个随机数



每人17张牌，三个人，3张底牌 \_p指向新位置，0位不用

int play\_1[18] = { 0 };

int play\_1\_p = 1;

int play\_2[18] = { 0 };

int play\_2\_p = 1;

int play\_3[18] = { 0 };

int play\_3\_p = 1;

int cards\_in\_one\_hand[4] = { 0 };

计算剩余牌数：

int Number\_of\_cards\_left(int \*pai)

{

int sum\_num = 0;

for (int i = 3; i <= 17; i++)

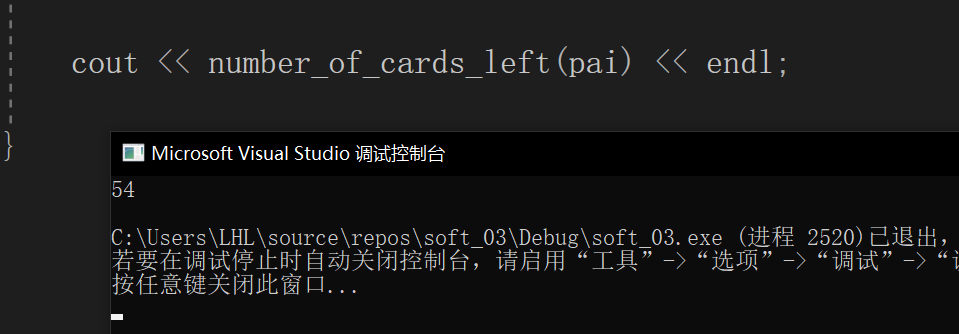
{

sum\_num = sum\_num + pai[i];

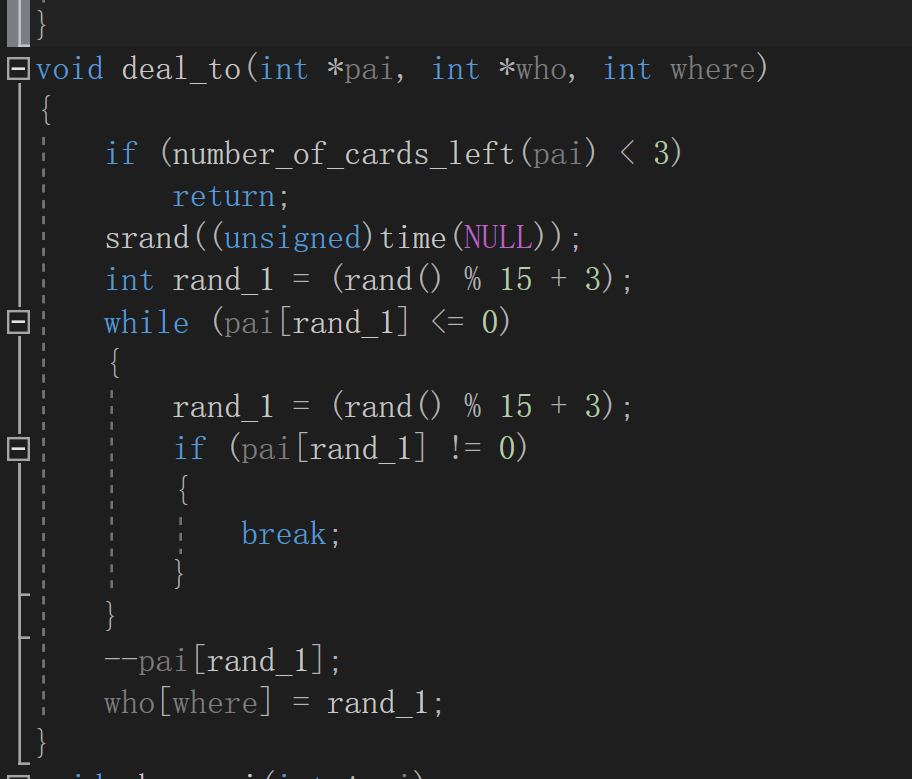
}

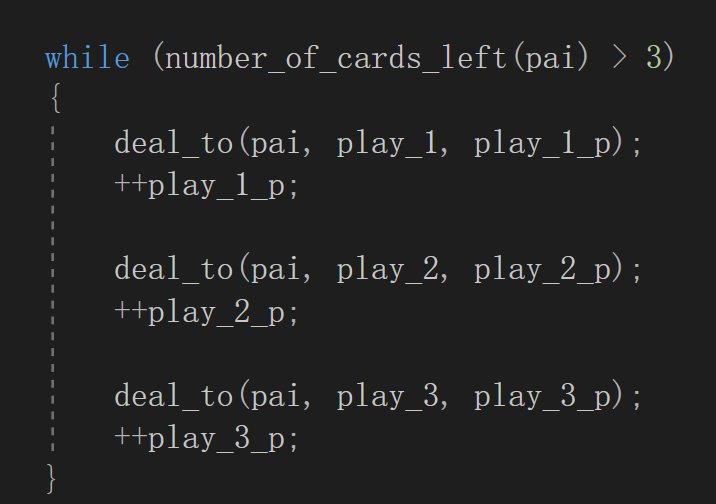
return sum\_num;

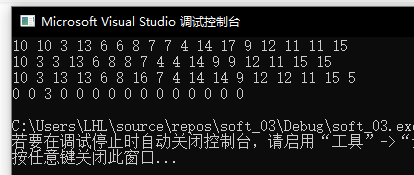
}

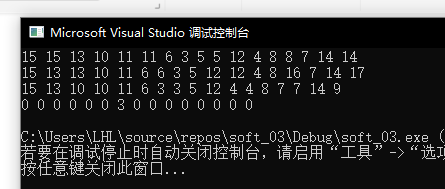


随机发给play123，由于随机数由时间生成，程序运行快每次进入函数的时间几乎一样，达不到随机发牌的目的，









所以增加一个变量记录发牌的次数，用这个变量\*一个很大的倍数+time，让srand改变，达到随机的目的

void deal\_to(int \*pai, int \*who, int where,int deal\_num)

{

if (number\_of\_cards\_left(pai) < 3)

return;

srand((unsigned)time(NULL)+ deal\_num\*1032);

int rand\_1 = (rand() % 15 + 3);

while (pai[rand\_1] <= 0)

{

rand\_1 = (rand() % 15 + 3);

if (pai[rand\_1] != 0)

{

break;

}

}

--pai[rand\_1];

who[where] = rand\_1;

}

----------------------------------

while (number\_of\_cards\_left(pai) > 3)

{

deal\_to(pai, play\_1, play\_1\_p, deal\_num);

++play\_1\_p;

++deal\_num;

deal\_to(pai, play\_2, play\_2\_p, deal\_num);

++play\_2\_p;

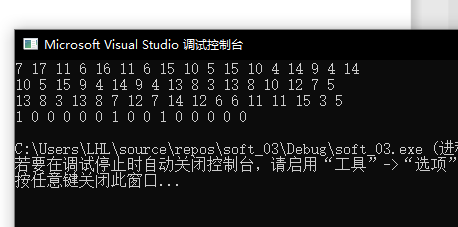
++deal\_num;

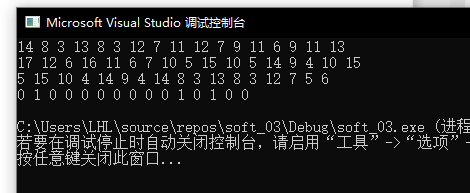
deal\_to(pai, play\_3, play\_3\_p, deal\_num);

++play\_3\_p;

++deal\_num;

}





快速排序对player和底牌进行排序

void myquicksort(int \*arr, int start, int end)

{

if (start >= end)

return;

int pos = arr[start];

int left = start;

int right = end;

while (left < right) {

while (left < right && pos <= arr[right]) right--;

arr[left] = arr[right];

while (left < right && pos >= arr[left]) left++;

arr[right] = arr[left];

}

arr[left] = pos;

myquicksort(arr, start, left - 1);

myquicksort(arr, left + 1, end);

}

**输出结果**

myquicksort(play\_1, 1, 17); //快排player

myquicksort(play\_2, 1, 17); //快排player

myquicksort(play\_3, 1, 17); //快排player

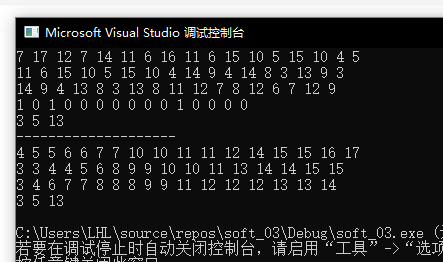
myquicksort(cards\_in\_one\_hand, 1, 3);

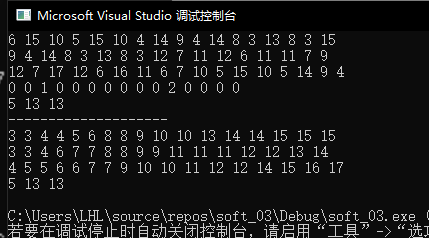
show\_player(play\_1); //展示player1

show\_player(play\_2); //展示player2

show\_player(play\_3); //展示player3

show\_cards\_in\_one\_hand(cards\_in\_one\_hand); //展示底牌





格式化输出得到最终结果：

void show\_me\_player(int \*player)

{

for (int i = 1; i <= 17; i++)

{

switch (player[i])

{

case 11:

cout << "J" << " ";

break;

case 12:

cout << "Q" << " ";

break;

case 13:

cout << "K" << " ";

break;

case 14:

cout << "A" << " ";

break;

case 15:

cout << "2" << " ";

break;

case 16:

cout << "小王" << " ";

break;

case 17:

cout << "大王" << " ";

break;

default:

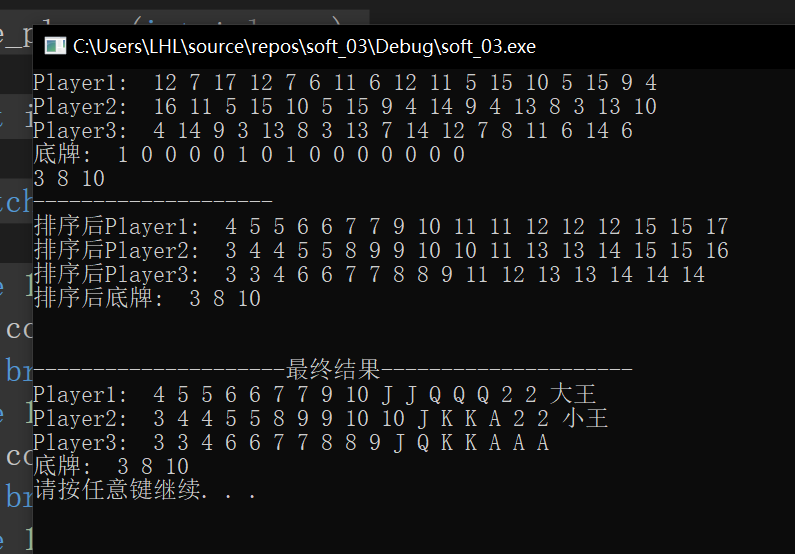
cout << player[i] << " ";

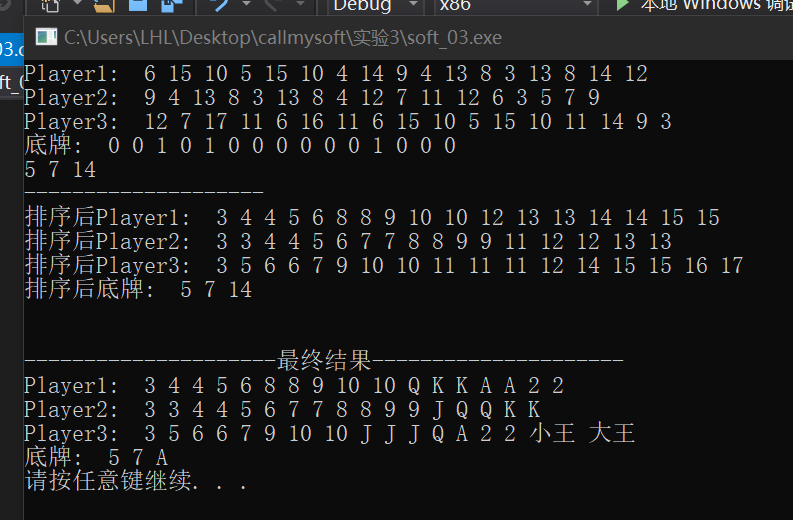
}

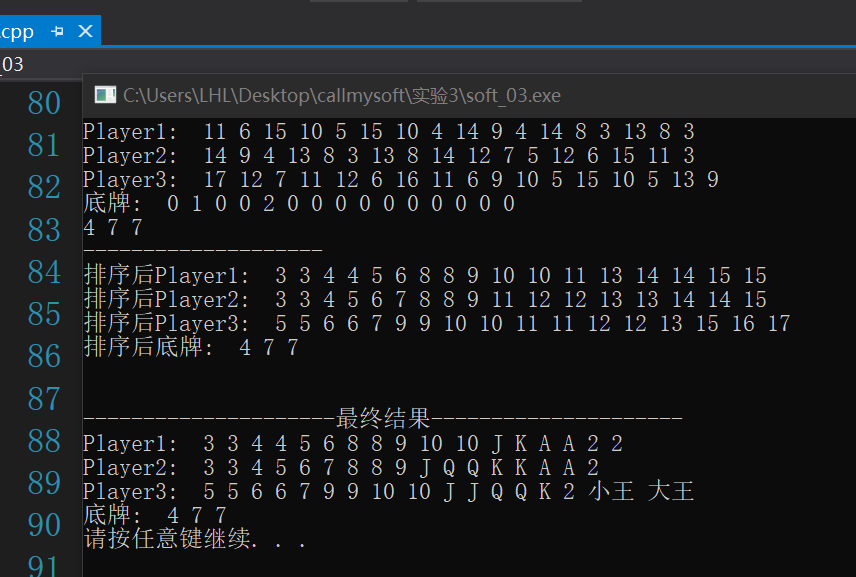
}

cout << endl;

}







输入1再次发牌，输入0退出

Cmd cls清除屏幕信息

