```
#include <iostream>
#include<string>
#include<time.h>
#include<conio.h>
using namespace std;
#define NUM 1000
int j = 2;//定义为全局变量
int randNum[NUM];//因为 rand()函数有一定的缺陷，所以在程序中定义了 randNum 数
组来存放随机数
/*即使使用了种子函数 srand(),由于程序运行时间比较短，也不太好设置种子。 因此使
用数组来存放随机数*/
class Poker{
private:
    int poker[53]; //扑克
    int pokerValue[53]; //扑克代表的数值
    string pokerName[53]; //扑克名
    int pokerF[5]; //甲手中的牌
    int pokerL[5]; //乙手中的牌
    int pokerNumF; //甲手中的牌数
    int pokerNumL; //乙手中的牌数
public:
    Poker(); //构造函数，对牌初始化
    void initPokerL();
    void initPokerF();
    //洗牌,在每轮游戏开始前进行
    string getPokerF(); //用字符串的形式返回甲的牌
    string getPokerL(); //用字符串的形式返回乙的牌
    int getSumF(); //返回甲牌的点数，用以判断是否超过 21 点
    int getSumL(); //返回乙牌的点数
    void farmerAsk(); //甲要牌
    void landlordAsk(); //乙要牌
```

```
	void  newGame();  //开始新游戏
	void  thewinner();
};
Poker::Poker()
{
	int  i;
	poker[0]  =  0;
	for  (i  =  1;  i  <=  13;  i++)  //|
	{  //|
		poker[i]  =  i;  //|用构造函数对牌初始化
		poker[i  +  13]  =  i;  //|
		poker[i  +  26]  =  i;  //|
		poker[i  +  39]  =  i;  //|
	}//for 结束

	pokerValue[0]  =  0;
	for  (i  =  1;  i  <=  52;  i++)
	{
		if  (poker[i]  <=  10)  pokerValue[i]  =  poker[i];
		else  pokerValue[i]  =  10;
	}
	pokerName[0]  =  "";
	for  (i  =  0;  i  <  4;  i++)
	{
		pokerName[1  +  13  *  i]  =  "A";
		pokerName[2  +  13  *  i]  =  "2";
		pokerName[3  +  13  *  i]  =  "3";
		pokerName[4  +  13  *  i]  =  "4";
		pokerName[5  +  13  *  i]  =  "5";
		pokerName[6  +  13  *  i]  =  "6";
		pokerName[7  +  13  *  i]  =  "7";
```

```
        pokerName[8 + 13 * i] = "8";
        pokerName[9 + 13 * i] = "9";
        pokerName[10 + 13 * i] = "10";
        pokerName[11 + 13 * i] = "J";
        pokerName[12 + 13 * i] = "Q";
        pokerName[13 + 13 * i] = "K";
    }//for 结束
    for (i = 0; i < 5; i++)
    {
        pokerF[i] = 0; //|对 pokerOfFarmer 初始化
        pokerL[i] = 0; //|对 pokerOfLandlord 初始化
    }
    pokerNumF = 0;//甲手中的牌数初始化为 0
    pokerNumL = 0;//乙手中的牌数初始化为 0

    srand((int)time(0));
    for (i = 0; i < NUM; i++)
    {
        randNum[i] = rand() * 51 / 32767 + 1;//产生随机数数组
    }

}//构造函数 Poker()结束
void Poker::initPokerF()
{
    std::cout << "新一局游戏开始，开始洗牌>>>>>" << endl;

    pokerF[0] = randNum[j++]; //产生 1-52 的随机数
    pokerF[1] = randNum[j++]; //产生 1-52 的随机数
    pokerNumF = 2;
    cout << "洗牌完成,甲的牌为:" << getPokerF() << endl;
}
```

```
void  Poker::initPokerL()
{
    std::cout  <<  "新一局游戏开始，开始洗牌>>>>>"  <<  endl;

    pokerL[0]  =  randNum[j++];  //产生 1-52 的随机数
    pokerL[1]  =  randNum[j++];  //产生 1-52 的随机数
    pokerNumL  =  2;
    cout  <<  "洗牌完成,乙的牌为:"  <<  getPokerL()  <<  endl;
}
//void  Poker::initPoker()结束
string  Poker::getPokerF()//用字符串的形式返回玩家的牌
{
    int  i;
    string  result  =  "";

    for  (i  =  0;  i  <  pokerNumF;  i++)
        result  =  result  +  pokerName[pokerF[i]]  +  "  ";

    return  result;
}//string  Poker::getPokerF()结束

string  Poker::getPokerL()//用字符串的形式返回庄家的牌
{
    int  i;
    string  result  =  "";

    for  (i  =  0;  i  <  pokerNumL;  i++)
        result  =  result  +  pokerName[pokerL[i]]  +  "  ";

    return  result;
}//string  Poker::getPokerL()结束
```

```
int Poker::getSumF() //返回甲的总点数
{
    int result = 0, j = 0;

    for (int i = 0; i < pokerNumF; i++)
        result = result + pokerValue[pokerF[i]];
    if (result < 21) {
        for (int i = 0; i < pokerNumF; i++) {
            if (pokerValue[pokerF[i]] == 1) j++;
        }
        if (j > 0) {
            while (result <= 11 && j > 0) {
                result += 10;
                j--;
            }
        }
    }
    return result;
}

int Poker::getSumL()//返回乙的总点数
{

    int result = 0, j = 0;

    for (int i = 0; i < pokerNumL; i++)
        result = result + pokerValue[pokerL[i]];
    if (result < 21) {
        for (int i = 0; i < pokerNumL; i++) {
            if (pokerValue[pokerL[i]] == 1) j++;
        }
        if (j > 0) {
            while (result <= 11 && j > 0) {
                result += 10;
                j--;
            }
        }
    }
    return result;
}
void Poker::farmerAsk()
{
    if (pokerNumF >= 5)
    {
        std::cout << "甲的牌数已够 5 张，不能再要牌了！" << endl;
    }
    else
    {
        pokerF[pokerNumF++] = randNum[j++]; //产生 1-52 的随机数
        cout << "甲的牌为:" << getPokerF() << endl;
    }
}
void Poker::landlordAsk()
{
    if (pokerNumL >= 5)
    {
        std::cout << "乙的牌数已够 5 张，不能再要牌了！" << endl;
    }
    else
    {
        pokerL[pokerNumL++] = randNum[j++]; //产生 1-52 的随机数
        cout << "乙的牌为:" << getPokerL() << endl;
    }
}
void Poker::thewinner() {

    if (getSumF() > 21)
    {
        if (getSumL() > 21 && getSumL() > getSumF())
            std::cout << "乙撑死了,乙输了" << endl;
        else std::cout << "甲撑死了,甲输了" << endl;
    }
    else if (getSumF() == 21)
    {
        if (getSumF() == 21)
            std::cout << "平局" << endl;
        else std::cout << "乙撑死了,乙输了" << endl;
    }
    else {
        if (getSumF() > getSumL()) {
            cout << "乙输了" << endl;
        }
        else if (getSumF() < getSumL()) cout << "甲输了" << endl;
        else std::cout << "平局" << endl;
    }
    char key;
    cout << "1.重新开始 2.退出 >>请输入数字选择操作:";
    cin >> key;
    if (key == '1') {
        newGame();
        return;
    }
    else exit(0);
}
void Poker::newGame()
{
    int choose = 1;

    initPokerF();
    cout << "甲得到的牌为:" << getPokerF() << endl;
    while (choose == 1 || choose == 2 || choose == 3 || choose == 4)
    {
        cout << "1.要牌 2.不要牌 3.退出 >>请输入数字选择操作:";
        cin >> choose;
        if (choose == 1) farmerAsk();
        else if (choose == 2) break;
        else if (choose == 3) exit(0);
    }
    initPokerL();
    while (choose == 1 || choose == 2 || choose == 3 || choose == 4)
    {
        cout << "1.要牌 2.不要牌 3.退出 >>请输入数字选择操作:";
        cin >> choose;
        if (choose == 1) landlordAsk();
        else if (choose == 2) break;
        else if (choose == 3) exit(0);
    }
    thewinner();
}
#include <iostream>
#include "blackjack.h"
using namespace std;
        int main()
```

```
{
    Poker  poker;
    cout    <<    "*******************************    欢 迎 玩 二 十 一 点 游 戏
*******************************" << endl;
    poker. newGame();
    return 0;
}//main 函数结束
```