**信息科学与工程学院**

**2020－2021学年第一学期**

实 验 报 告

课程名称： 高级编程技术

实验名称： 实验一

专 业 班 级 通信工程 二班

学 生 学 号 201800121050

学 生 姓 名 孟麟芝

实 验 时 间 2020年9月17日

实验报告

【实验目的】

1.掌握C#的基本编程语句

2.理解装箱拆箱的作用

3.学会一些简单程序的编写

【实验要求】

1.进行一些装箱拆箱操作，看地址是否改变

2.用两种方式实现百鸡百钱

3.用两种编码方式实现分扑克牌

【实验具体内容】

【第一个实验】

1.实验源代码：

using System;

namespace PackingAndUnpacking

{

    class Schedule

    {

        static void Main(string[] args)

        {

            int x1 = 111;

            int x2 = x1;

            int x3 = 0;

            Console.WriteLine("尝试查看x1与x2内存地址：x1={0},x2={1}", (object)x1, (object)x2);

            Console.WriteLine("查看x1与x2内存地址是否相等：" + ((object)x1 == (object)x2));

            object obj1 = x1;

            object obj2 = obj1;

            Console.WriteLine("查看obj1与obj2指向的地址是否相等：" + ((object)obj1 == (object)obj2));

            x1 = 2;

            Console.WriteLine("尝试不进行类型转换直接输出obj1：" + obj1);

            Console.WriteLine("进行类型转换再输出obj1：" + (int)obj1);

            x3 = (int)obj1;

            Console.WriteLine("将obj1拆箱给x3，看x3与x1内存地址是否相等：" + ((object)x1 == (object)x3));

            string s1 = "Visual Studio Code";

            string s2 = s1;

            Console.WriteLine("查看s1与s2是否指向同一个地址：" + ((object)s1 == (object)s2));

            s1 = "C#";

            Console.WriteLine("只改变s1，查看s1与s2是否指向同一个地址：" + ((object)s1 == (object)s2));

            s2 = "C#";

            Console.WriteLine("将s1、s2改成相同值，查看s1与s2是否指向同一个地址：" + ((object)s1 == (object)s2));

        }

    }

}

2.实验记录与说明分析：

编写代码后运行，可见如下结果：



可以看到，x1、x2是两个不同的变量，值均为111，尝试查看内存地址时仍输出的是值，可见C#中不可以直接查看变量的地址（但通过unsafe声明使用指针可以查看），但可通过强制类型转换为object类型判断地址是否相等。

装箱后令obj2=obj1，可见两个object类型指向了同一个地址。

不论是否进行类型转换直接用Console.Writeline输出object类型，都只会输出其指向的值，不会输出内存地址。

若定义两个字符串类型，且其值相同，可见指向了同一个地址。只改变其中一个，s1与s2地址不同。再将二者值改为相同的，指向地址又相同。可见string类型的值被修改时，其实是创建了另外一个内存，并由该变量指向新的内存。

【第二个实验】

1.实验源代码：

（1）定义鸡数

using System;

namespace Learn

{

    class Program

    {

        static void Main(string[] args)

        {

            int x = 0, y = 0, z = 0, m = 0;//设x为母鸡数，y为公鸡数，z为小鸡数

            for (x = 0; x <= 25; x++)

            {

                if ((100 - 4 \* x) % 7 == 0)//将两个方程化简得4x+7y=100

                {

                    m++;

                    y = (100 - 4 \* x) / 7;

                    z = 100 - x - y;

                    Console.WriteLine("第{0}组解：公鸡数目={2,-4}  母鸡数目={1,-4}  小鸡数目={3,-4}", m, x, y, z);

                }

            }

        }

    }

}

（2）定义钱数

using System;

namespace Learn

{

    class Program

    {

        static void Main(string[] args)

        {

            int x = 0, y = 0, z = 0, m = 0;//设x为母鸡总钱数，y为公鸡总钱数，z为小鸡总钱数

            for (x = 0; x <= 75; x += 3)

            {

                if ((1500 - 20 \* x) % 7 == 0)//将两个方程化简得20x+21y=1500

                {

                    m++;

                    y = (1500 - 20 \* x) / 21;

                    z = 100 - x - y;

                    Console.WriteLine("第{0}组解：公鸡数目={2,-4}  母鸡数目={1,-4}  小鸡数目={3,-4}", m, x / 3, y / 5, z \* 3);

                }

            }

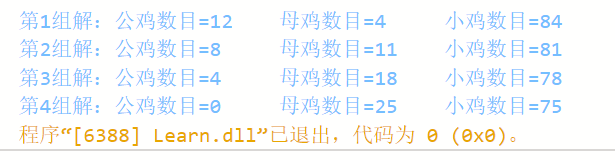
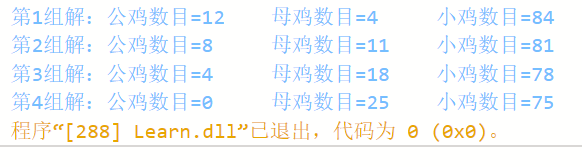
        }

    }

}

2.实验记录与说明分析：

上面两种方法的输出是一样的，左右分别为第一种和第二种：



若使用鸡数考虑，则可写出如下三元线性不定方程组：

化简可得：

则可见x≤25，令x由0到25变化，找到能令y为整数的值，则此时z也必为整数，这样的解即为所需要的解

若使用钱数考虑，则可写出如下三元线性不定方程组：

化简可得：

则可见x≤75，令x由0到75变化，找到能令y为整数的值，则此时z也必为整数，这样的解也即为所需要的解

【第三个实验】

1.实验源代码：

（1）使用101，102……编码

using System;

namespace CardGame

{

    class TestCard

    {

        static void Main(string[] args)

        {

            int i, j, temp;

            Random Rnd = new Random();

            int k;

            int [] Card = new int [52];

            int [,] Player=new int [4,13];

            for (i=0; i<4; i++)                             // 52张牌初始化

                for (j=0; j<13; j++)

                    Card[i\*13+j]=(i+1)\*100+j+1;

            Console.Write ("How many times for card: ");

            string s=Console.ReadLine ();

            int times=Convert.ToInt32 (s);

            for (j=1;j<=times; j++)

                for (i=0;i<52;i++)

                {

                    k=Rnd.Next(51-i+1)+i;                   // 产生i到52之间的随机数

                    temp=Card[i];

                    Card[i]=Card[k];

                    Card[k]=temp;

                }

            k=0;

            for (j=0;j<13;j++)                                  // 52张牌分发给4个玩家

                for (i=0;i<4;i++)

                    Player[i,j]=Card[k++];

            for (i=0;i<4;i++)                               // 显示4个玩家的牌

            {

                Console.WriteLine("玩家{0}的牌：",i+1);

                for (j=0;j<13;j++)

                {

                    k=(int)Player[i,j]/100;                 // 分离出牌的花色

                    switch (k)

                    {

                         case 1:                                // 红桃

                            s="红桃";

                            break;

                        case 2:                                 // 方片

                            s="方片";

                            break;

                        case 3:                                 // 梅花

                            s="梅花";

                            break;

                        case 4:                                 // 黑桃

                            s="黑桃";

                            break;

                 }

                    k=Player[i,j]%100;                      // 分离出牌号

                    switch (k)

                    {

                        case 10:

                            s=s+"A";

                            break;

                        case 11:

                            s=s+"J";

                            break;

                        case 12:

                            s=s+"Q";

                            break;

                        case 13:

                            s=s+"K";

                            break;

                        default:

                            s=s+Convert.ToString (k);//将牌号与花色一同组成一个字符串

                            break;

                    }

                    Console.Write (s);

                    if (j<12)

                        Console.Write (", ");//Console.Write输出后不会换行

                    else

                        Console.WriteLine (" ");//Console.WriteLine输出后会换行

                }

            }

            Console.Read ();

        }

    }

}

（2）使用1~52编码，按区间划分牌色

using System;

namespace CardGame

{

    class TestCard

    {

        static void Main(string[] args)

        {

            int i, j, temp;

            Random Rnd = new Random();

            int k;

            int [] Card = new int [52];

            int [,] Player=new int [4,13];

            for (i=0; i<52; i++)                                // 52张牌初始化

                    Card[i]=i+1;

            Console.Write ("How many times for card: ");

            string s=Console.ReadLine ();

            int times=Convert.ToInt32 (s);

            for (j=1;j<=times; j++)

                for (i=0;i<52;i++)

                {

                    k=Rnd.Next(51-i+1)+i;                   // 产生i到52之间的随机数

                    temp=Card[i];

                    Card[i]=Card[k];

                    Card[k]=temp;

                }

            k=0;

            for (j=0;j<13;j++)                                  // 52张牌分发给4个玩家

                for (i=0;i<4;i++)

                    Player[i,j]=Card[k++];

            for (i=0;i<4;i++)                               // 显示4个玩家的牌

            {

                Console.WriteLine("玩家{0}的牌：",i+1);

                for (j=0;j<13;j++)

                {

                    k=(int)Player[i,j]/13;                  // 分离出牌的花色

                    switch (k)

                    {

                        case 0:                                 // 红桃

                            s="红桃";

                            break;

                        case 1:                                 // 方片

                            s="方片";

                            break;

                        case 2:                                 // 梅花

                            s="梅花";

                            break;

                        case 3:                                 // 黑桃

                            s="黑桃";

                            break;

                    }

                    k=Player[i,j]%13;                       // 分离出牌号

                    switch (k)

                    {

                        case 10:

                            s=s+"A";

                            break;

                        case 11:

                            s=s+"J";

                            break;

                        case 12:

                            s=s+"Q";

                            break;

                        case 13:

                            s=s+"K";

                            break;

                        default:

                            s=s+Convert.ToString (k);//将牌号与花色一同组成一个字符串

                            break;

                    }

                    Console.Write (s);

                    if (j<12)

                        Console.Write (", ");//Console.Write输出后不会换行

                    else

                        Console.WriteLine (" ");//Console.WriteLine输出后会换行

                }

            }

            Console.Read ();

        }

    }

}

2.实验记录与说明分析：

这两种方法其实是相似的，洗牌是交换牌的顺序的操作，在后面再用商求出编号所处区间，即花色，用余数求出其牌号，即可实现分牌。以下两图分别是第一种和第二种方法的结果

