Lab2　C#基础

学习目标：

1.理解常量与变量，掌握常量的命名规则

2.理解值类型和引用类型的存储区别

3.掌握流程控制语句

4.掌握数组的声明和赋值方法

一、常量与变量

1 常量

常量是指在程序中，值保持不变的量。一个数据在程序内频繁地使用，而且保持不变情况下就可以定义成常量。常量包括整型常量、浮点型常量、布尔型常量、字符型常量等，其中 null 常量只有一个值 null，表示对象的引用为空。

常数常量只能在声明时赋值，声明常量的语法格式如下：

[访问修饰符] const 数据类型 变量名=表达式;

例如：

```
const double PI = 3.1415926;
```

2 变量

在程序运行期间，随时可能产生一些临时数据，这些数据被保存在一些内存单元中，每个内存单元都用一个标识符来标识。这些内存单元就是变量，定义的标识符是变量名，内存单元中存储的数据是变量的值。一般情况下，声明在类中的称为成员变量，声明在方法中的称为局部变量。

变量必须先声明后使用，声明变量就是给变量指定一个类型和一个名称，声明变量后，编译器会给该变量分配一定大小的内存单元。变量可以在声明时赋值，也可以在运行时赋值。声明变量的语法格式如下：

[访问修饰符] 数据类型 变量名[=表达式];

例如：

```
int x=1,y;
y=x+15;
```

上面的代码中，第 1 行代码定义了两个变量 x 与 y，也就相当于分配了两个内存单元，在定义变量的同时为变量 x 赋予了一个初始值 1。第 2 行代码为变量 y 赋值，在执行第 2 行代码时，首先取出变量 x 的值，与 15 相加后，将结果赋值给变量 y。

【注意】

在 C#语言中，要求所有使用的变量遵循“先定义、后使用”的规则。只有定义了的变量，编译系统才会根据变量所属的数据类型分配相应的内存空间，并且编译系统会检查在程序中对该变量进行的运算是否合法。C#的变量名是一种标识符，应该符合标识符的命名规则。

3 变量的作用域

变量的作用域是指变量的使用范围。在程序中，变量一定会被定义在某一对大括号中，该大括号所包含的代码区域就是这个变量的作用域。

例如：在下面代码段中，变量 x 的作用域是从第 2 行到第 10 行，变量 y 的作用域是从第 5 行到第 8 行，变量 i 是在循环语句中声明的变量，只存在于该循环体内，即第 5 行到第 8 行。

```
1    static void Main(string[] args)
2        {
3            int x = 4;
4            for (int i =0;i<5;i++)
5            {
6                int y = 3;
7                ...
8            }
9            ...
10       }
```

二、数据类型

数据类型是表示具有多种相同特征的一组数据。C#是强类型语言，即每个变量和对象都必须声明数据类型，并且为变量赋值时必须赋予与变量同一类型的值，否则程序会报错。

从用户的角度，数据类型分为内置数据类型和用户自定义的数据类型。内置数据类型是.NET Framework 预定义好的类型；用户自定义类型是由用户声明创建的。

从数据存储的角度，数据类型又分为值类型和引用类型。值类型用于存储数据的值，而引用类型用于存储对实际数据的引用地址。这两类又细分为多种数据类型，具体下表所示。

C#数据类型

类型		说明
值类型	简单值类型	有符号整型：sbyte,short,int,long
		无符号整型：byte,ushort,uint,ulong
		浮点型：float,double
		高精度小数：decimal
		布尔型：bool
		Unicode 字符：char
	结构	struct S{…}形式的用户自定义类型
	枚举	enum E{…}形式的用户自定义类型
引用类型	Object	所有其他类型的最终基类型：object
	类类型	class C{…}形式的用户自定义类型
	接口	interface I{…}形式的用户自定义类型
	字符串	Unicode 字符串： string
	数组	一维和多维数组
	委托	delegate TD{…}形式的用户自定义类型

值类型与引用类型的不同之处在于：值类型的变量直接包含数据；而引用类型的变量存储对数据的引用，后者称为对象。对于值类型，每个变量都有自己的数据副本，各变量之间的操作互不影响；对于引用类型，同一个对象可能被多个变量引用，因此一个变量的操作可能影响另一个变量对该对象的引用。在定义一个值类型变量后，将直接为该类型分配空间，可以直接赋值和使用；而引用类型在定义时并不会分配空间，只是在对其实例化时，才真正地分配存储空间。

【注意】

在 C#中，若没有为结构或者类的字段变量初始化，则编译器会自动根据这些数据类型为其赋一个默认值。默认值规则如下：

（1）数值（整数和小数）：0；

（2）字符：’\0’对应的代码点：U+0000；

（3）布尔：false；

（4）枚举：0；

（5）引用类型：null。

三、流程控制语句

1.选择结构：if…else 语句

【实例】判断奇偶数。

```
static void Main(string[] args)
        {
            int num = 45;
            if (num % 2 == 0)
            {
                Console.WriteLine(num + "是一个偶数");
            }
            else
            {
                Console.WriteLine(num + "是一个奇数");
            }
            Console.ReadLine();
        }
```

2.循环结构：for 语句

【实例】求 1~100 的累加和。

程序代码如下：

```
        static void Main(string[] args)
        {
            int sum = 0;
            for (int i = 1; i <= 100; i++)
            {
                sum += i;
            }
            Console.WriteLine("1-100的累加和是："+sum);
            Console.ReadLine();
        }
```

3.选择结构： switch 语句（自学）

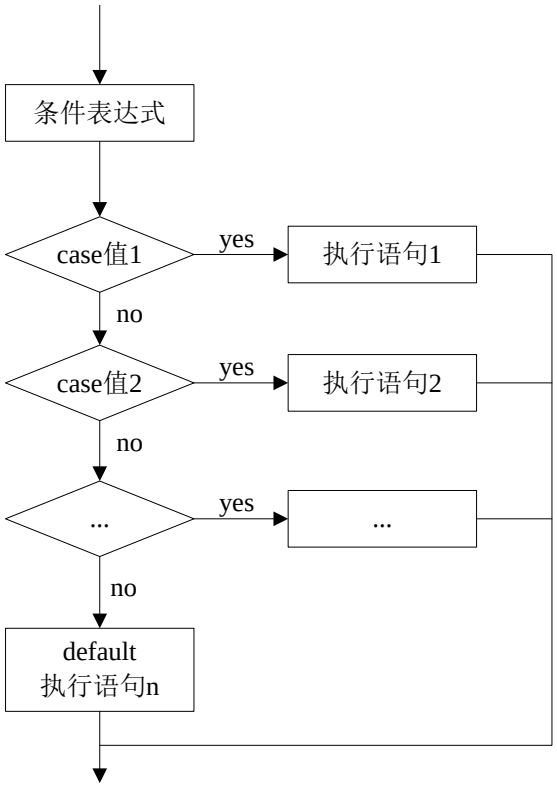
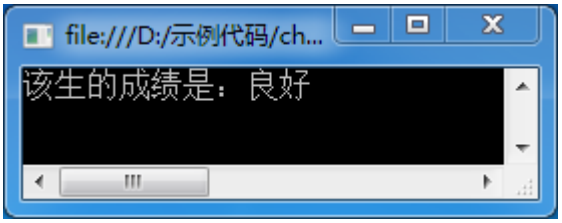


图 2-18 switch 语句执行流程图

【实例】学生成绩等级评定。

```
enum Grade { 优秀, 良好, 中等, 及格, 不及格, none }
class Switch1
{
    static void Main(string[] args)
    {
        Grade grade = Grade.none;
        int x = 86;
        switch ((int)(x / 10))
        {
            case 10:
            case 9:
                grade = Grade.优秀;
                break;
            case 8:
                grade = Grade.良好;
                break;
            case 7:
                grade = Grade.中等;
                break;
            case 6:
                grade = Grade.及格;
                break;
```

```
                default:
                    grade = Grade.不及格;
                    break;
            }
            Console.WriteLine("该生的成绩是：" + grade);
            Console.ReadLine();
        }
```



四、数组

数组是由若干个数据类型相同的数组元素构成的数据结构，索引从 0 开始，每个元素可以通过数组名称和索引进行访问。数组元素可以是任何类型，但因为没有名称，只能通过索引（又称下标）来访问。数组有一个“秩”，它表示和每个数组元素关联的索引的个数。数组的秩又称为数组的维度。“秩”为 1 的数组称为一维数组，“秩”大于 1 的数组称为多维数组，根据维度，多维数组分为二维数组、三维数组等。常用声明数组的语法如下表所示。

常用声明数组语法

数组类型	声明语法
一维数组	数据类型[] 数组名
二维数组	数据类型[,] 数组名
三维数组	数据类型[,,] 数组名

下面分别介绍一维数组和多维数组的创建和初始化。

1.一维数组

（1）一维数组的声明与创建

声明和创建一维数组的一般形式如下：

数组类型[] 数组名 = new 数组类型[数组长度];

例如：

```
int[] arr1 = new int[5];
```

表示声明和创建一个具有 5 个元素的一维数组 arr1。一维数组也可以先声明后创建。

（2）一维数组的初始化

如果在声明和创建数组时，没有初始化数组，则数组元素将自动初始化为该数组类型的默认初始值。初始化数组有以下三种方式。

①创建时初始化

在创建一维数组时，对其初始化的一般形式如下：

数组类型[] 数组名 = new 数组类型[数组长度]{初始值列表};

其中，数组长度可省略。如果省略数组长度，系统将根据初始值的个数来确定一维数组的长度。如果指定了数组长度，则 C#要求初始值的个数必须和数组长度相同，也就是所有数组元素都要初始化，而不允许只对部分元素进行初始化。初始值之间用逗号做间隔。

例如：

```
int[] arr1= new int[5] { 1, 2, 3, 4, 5 };
```

以上代码表示创建一个一维数组 arr1 具有 5 个数组元素，也可以在创建一维数组及初

始化时采用简写形式。

例如：

```
int[] arr1= { 1, 2, 3, 4, 5 };
```

以上代码同样表示创建了数组元素值分别为1、2、3、4、5的一个具有5个数组元素的一维数组。

②先声明后初始化

C#允许先声明一维数组，然后再初始化各数组元素。其一般形式如下：

数组类型[] 数组名;

数组名 = new 数组类型[数组长度]{初始值列表};

例如：

```
int[] arr1;
arr1= new int[5] { 1, 2, 3, 4, 5 };
```

以上代码表示先声明一个一维数组arr1，再用运算符new来创建并初始化。注意，在先声明后初始化数组时，不能采用简写形式。

例如：

```
int[] arr1;
arr1= { 1, 2, 3, 4, 5 };
```

以上代码是错误的。

③先创建后初始化

C#也允许先声明和创建一维数组，然后逐个初始化数组元素。其一般形式如下：

数组类型[] 数组名 = new 数组类型[数组长度];

数组元素 = 值;

例如：

```
int[] arr1= new int[5];
arr1[0] = 1;
arr1[1] = 2;
```

2.多维数组

（1）多维数组的声明和创建

声明和创建多维数组的一般形式如下：

数组类型[逗号列表] 数组名 = new 数组类型[数组长度];

其中，逗号列表的逗号个数加 1 就是维度数，即如果逗号列表为一个逗号，则称为二维数组；如果为两个逗号，则称为三维数组，依此类推。维度长度列表中的每个数字定义维度的长度，数字之间以逗号做间隔。

例如：

```
int[,] arr2 = new int[3, 2];
```

以上代码表示声明和创建一个具有3×2共6个数组元素的二维数组arr2。

（2）多维数组的初始化

多维数组也具有多种初始化方式，但需要注意以下几点：

①以维度为单位组织初始化值，同一维度的初始值放在一对大括号中{}中。

例如：

```
int[,] arr2 = new int[3, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
```

②可以省略维度长度列表，系统能够自动计算维度和维度的长度，但逗号不能省略。

例如：

```
int[,] arr2 = new int[, ] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
```

③多维数组不允许部分初始化。

例如：

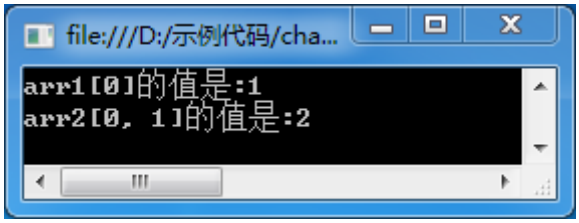
```
int[,] arr2 = new int[3, 2] { { 1, 2 }, { 3, 4 } };
```

以上代码希望只初始化二维数组的前两行元素，这是错误的。

【实例】声明一个一维数组和一个二维数组，并输出指定元素的值。

程序代码如下：

```
class Program
{
    static void Main(string[] args)
    {
        //声明一维数组
        int[] arr1;
        //创建数组对象
        arr1= new int[5] { 1, 2, 3, 4, 5 };
        //声明二维数组并创建数组对象
        int[,] arr2 = new int[3, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
        //输出指定元素的值
        Console.WriteLine("arr1[0]的值是:"+ arr1[0]);
        Console.WriteLine("arr2[0, 1]的值是:"+arr2[0, 1]);
        Console.ReadLine();
    }
}
```



【分析】

本实例中声明了一维数组 arr1 和二维数组 arr2，分别为数组 arr1 和 arr2 赋值，并使用数组名[索引]的形式获得数组中元素值，其中 arr1[0]指 arr1 的第一个值 1，arr2[0, 1]指 arr2 中第一行第二列的值 2。

五、课后练习 ：统计一个长度为 10 的整型数组中 1 出现的次数。（选做）