# 自动类型转换

自动类型转换：等号两边参与运算的类型必须一致，如果不一致，会发生自动类型转换：

条件1：类型兼容（int,double）

条件2：目标类型大于源类型（范围大小）

强制类型转换：

条件1,：两种类型相兼容

条件2：源类型大于目标类型

Tips：将某一个操作数改变为double类型，整体都会变成double类型。

保留两位小数

Double d = 3.333333333

Console.WriteLine(“{0:0.00}”,d);

Tips：取余操作：除数除以被除数之后剩余的值。

# 程序调试

调试方法：

F11：逐语句调试。

F10：逐过程调试（调试方法）

断点调试：打断点直接启动调试，按F11一步步操作

# 常量

关键词：const （相当于final）

Const 变量类型 变量名 = 值；

# 枚举：一个简单的变量类型

语法：

[public] enum 枚举名{

值1,

值2，

。。。。

}

放在[]中可写可不写，enum表示枚举的关键字。

作用：规范某些地方的开发

# 结构：一次性声明多个不同的变量

语法：

[public] struct 结构名{

[public] 类型 \_字段名1;

[public] 类型 \_字段名2;

[public] 类型 \_字段名3;

}

作用：一次性声明多个不同类型的变量。

Tips：字段和变量的区别，变量在程序运行的时候只能存储一个值，但是字段可以存储多个值。

# 冒泡排序

Int [] arr = {9,8,7,6,5,4,3,2,1};

for(int I = 0;i<arr.Length-1;i++){

for(int j = 0;j<arr.Length-i-1;j++){

if(arr[j]>arr[j+1]){

int temp = arr[j];

arr[j] = arr[j+1];

arr[j+1]=temp;

}

}

}

# 方法：

Out参数：侧重于返回多个不同类型的值。

第一步：定义方法，参数语法为：out 类型 形参名

第二步：在方法中给out参数赋值(必须)

第三步：调用方法之后可以使用out多余传出来的参数

05-函数-out

Tips：必须对传入进来的out参数赋值，out参数一开始可以不用赋初始值。

参数的书写格式 out 数据类型 变量名

填写参数时候的书写格式 out+对应的数据类型的变量名

ref参数

能够将一个变量带入方法中进行改变，然后将改变的值带出方法。（方法不需要写返回值）

Tips:传入的参数必须赋值，方法内部可以不赋值。

Params 可变参数

将实参列表中的值自动转变为形参列表中同类型的值（数组，集合）。

Tips：一个形参列表有且仅有一个params 列表，且要放在最后。

# 方法的重载：

方法的名称相同，参数类型不同，返回值类型不同，参数个数不同。

# 方法的递归

方法自己调用自己。

Tips：注意方法边界，return的时候会一层层的往上return回去。