const：可以理解为 只读

定义 常量只是 const 的其中一个作用

const用法主要是防止定义的对象再次被修改,定义对象变量时必须初始化变量,因为只读 后续无法再修改。

1.用于定义常量变量,这样这个变量在后面就不可以再被修改

```
const int Val = 10;
Val = 20; //错误,不可被修改
```

2. 节约内存空间,

使用const定义的变量将放于程序的只读数据区

```
#define  PI  3.14 //使用#define宏
const double Pi = 3.14 //使用const,这时候Pi并没有放入内存中
double  a = Pi;  //这时候才为Pi分配内存,不过后面再有这样的定义也不会再分配内存
double  b = PI;  //编译时分配内存
double  c = Pi;  //不会再分配内存,
double  d = PI;  //编译时再分配内存
```

const定义的变量,系统只为它分配一次内存,而使用#define定义的常量宏,能分配好多次,这样const就很节约空间

3. 限制变量不能修改

初级程序员，变量不能修改，我记住不去修改就ok了，没必要使用成 const

但是可能会遗忘，不小心改动了，导致程序出错，应该为常量的对象，没有声明为const，就必须由程序员自己来维护，来记住这个变量不应该被修改，即使你不小心修改导致程序整体混乱了，编译器也不会报错

限制别人修改，参数传参，或者 提供接口，限制不能更改，

属于 编译时限制，如果修改，编译时会报错

用法：

1：对于基础数据类型

const int Val = 10; 和 int const val = 10； 是一样的！int 和 const 是同级的

2：对于指针类型

原则：限右原则（限制右边的数据类型）

例如：

int * const a = xxx;

右边是 a 是指针类型，const 就限制了 指针是只读的，不能改变，而指针指向的内容却可以改变 *a = 10； 是ok的

int const *a = xxx；和 const int *a = xxx ；

右边 是 *a，是 int 类型，const 就限制了 指针指向的值 是只读的，不能修改，而指针是可以改变的 a++； 是ok的，

（实际上 指针改变了，两个指针指向的值 可能会不一样，const 只限制了，不能去改指针指向的值，用处在于遍历。。。）

答疑：

```
1. const int a = 1;
2. int* b = (int*)&a;
3. *b = 31;
```

这样是正常的

编译后const和普通变量没有区别，只是在编译的过程中，编译器会检查代码中是否有对const变量进行修改的代码，如果有则向用户报错。在编译过后，const变量就和普通变量相同了。而且，如果使用memset去修改const变量的内容，也完全没有问题，这就可以看出const修饰是属于编译层面的限制，一般不会涉及到运行层面。在C中，const是用于明确的标识出变量或者函数不能被修改，而且这种限制在编译层面进行约束。