**[C++基础知识集锦](http://www.cppblog.com/mzty/archive/2007/07/19/28359.html)**

**一、#include “filename.h”和#include <filename.h>的区别**  
#include “filename.h”是指编译器将从当前工作目录上开始查找此文件  
#include <filename.h>是指编译器将从标准库目录中开始查找此文件  
  
**二、头文件的作用**  
加强安全检测  
通过头文件可能方便地调用库功能，而不必关心其实现方式  
  
**三、\* , &修饰符的位置**int \*i,j; // better for read  
 i = new int(0);  
 j = 0;  
 int \*&y = i; // pointer's reference  
对于\*和&修饰符，为了避免误解，最好将修饰符紧靠变量名  
  
**四、if语句**  
不要将布尔变量与任何值进行比较，那会很容易出错的。  
整形变量必须要有类型相同的值进行比较  
浮点变量最好少比相等，可以通过求差与较小的数比较  
指针变量要和NULL进行比较，不要和布尔型和整形比较  
  
**五、const和#define的比较**  
const有数据类型，#define没有数据类型  
个别编译器中const可以进行调试，#define不可以进行调试  
在类中定义常量有两种方式  
1、 在类中声明常量，但不赋值，在构造函数初始化表中进行赋值；（**常量和引用类型的成员变量必须通过初始化列表来初始化赋值**）  
2、 用枚举代替const常量。  
  
**六、C++函数中值的传递方式**  
有三种方式：值传递(Pass by value)、指针传递(Pass by pointer)、引用传递(Pass by reference)  
void fun(char c) //pass by value  
void fun(char \*str) //pass by pointer  
void fun(char &str) //pass by reference  
如果输入参数是以值传递的话，最好使用引用传递代替，因为引用传递省去了临时对象的构造和析构  
函数的返回类型不能省略，就算没有也要加个void  
  
**七、函数体中的指针或引用常量不能被返回**

Char \*func(void)  
{  
 char str[]=”Hello Word”;  
 //这个是不能被返回的，因为str是个指定变量，不是一般的值，函数结束后会被释放掉  
 return str;   
}

函数体内的指针变量并不会随着函数的消亡而自动释放  
  
**九、内存的分配方式**  
分配方式有三种：  
1、 静态存储区，是在程序编译时就已经分配好的，在整个运行期间都存在，如全局变量、常量。（程序编译后运行时包含code和data两部分，其中data即为静态存储区分配，程序一开始运行便分配整个data的东东）  
2、 栈上分配，函数内的局部变量就是从这分配的，但分配的内存容量有限。  
3、 堆上分配，也称动态分配，如我们用new,malloc分配内存，用delete,free来释放的内存。  
  
**十、内存分配的注意事项**  
用new或malloc分配内存时，必须要对此指针赋初值。  
用delete 或free释放内存后，必须要将指针指向NULL  
不能修改指向常量的指针数据  
  
**十二、sizeof的问题**  
记住一点，C++无法知道指针所指对象的大小，指针的大小在32位计算机上为4字节

char a[]=”Hello World!”  
char \*p=a;  
count<<sizeof(a)<<end; //12字节  
count<<sizeof(p)<<endl; //4字节

而且，在函数调用传参时，**数组参数退化为指针**，所以下面的内容永远输出为4

void fun(char a[1000])  
{  
count<<sizeof(a)<<endl; //输出4而不是1000  
}

**十三、关于指针**  
1、 指针创建时必须被初始化  
2、 指针在free 或delete后必须置为NULL  
3、 32为计算机中指针的长度为4字节  
4、释放内存时，如果是数组指针，必须要释放掉所有的内存，如

char \*p = new char[100];  
strcpy(p,”Hello World”);  
delete **[]**p; //注意前面的［］号  
p = NULL;

**十四、关于malloc/free 和new /delete**  
malloc/free 是C/C+的内存分配符，new /delete是C++的内存分配符。  
注意：malloc/free是库函数，new/delete是运算符  
malloc/free不能执行构造函数与析构函数，而new/delete可以  
两者都必须要成对使用  
C++中可以使用set\_new\_hander函数来定义内存分配异常的处理函数  
  
**十五、Ｃ++的特性**  
Ｃ++新增加有重载(overload)，内联（inline），const，virtual四种机制  
重载和内联：即可用于全局函数，也可用于类的成员函数；  
const和virtual：只可用于类的成员函数；  
重载：在同一类中，函数名相同的函数。由不同的参数决定调用那个函数。函数可要不可要virtual关键字。和全局函数同名的函数不叫重载。如果在类中调用同名的全局函数，必须用全局引用符号**::**引用。  
覆盖是指派生类函数覆盖基类函数：函数名相同；参数相同；基类函数必须有virtual关键字；不同的范围(派生类和基类)。  
隐藏是指派生类屏蔽了基类的同名函数相同  
1、 函数名相同，但参数不同，此时不论基类有无Virtual关键字，基类函数将被隐藏。  
2、 函数名相同，参数也相同，但基类无Virtual关键字(有就是覆盖)，基类函数将被隐藏。  
内联：inline关键字必须与定义体放在一起，而不是单单放在声明中。  
const：const是constant的缩写，“恒定不变”的意思。被const修饰的东西都受到编译器强制保护，可以预防意外的变动，能提高程序的健壮性。  
1、 参数做输入用的指针型参数，加上const可防止被意外改动。  
2、 按值引用的用户类型做输入参数时，最好将按值传递的改为引用传递，并加上const关键字，目的是为了提高效率。数据类型为内部类型的就没必要做这件事情；如：  
将void Func(A a) 改为void Func(const A &a)。  
而void func(int a)就没必要改成void func(const int &a);  
3、 给返回值为指针类型的函数加上const，会使函数返回值不能被修改，赋给的变量也只能是const型变量。如：函数const char\*GetString(void); char \*str=GetString()将会出错。而const char \*str=GetString()将是正确的。  
4、 const成员函数是指该函数不会修改成员变量的值。除非该成员变量用mutable声明。  
virtual：虚函数：派生类可以覆盖掉的函数，纯虚函数：只有函数声明，没有函数实现体；  
  
**十六、extern“C”有什么作用？**  
extern “C”是由Ｃ＋＋提供的一个链接交换指定符号，用于告诉Ｃ＋＋这段代码是Ｃ函数。这是因为C++编译后库中函数名会变得很长，与C生成的不一致，造成Ｃ＋＋不能直接调用C函数，加上extren “c”后，C++就能直接调用C函数了。  
extern “C”主要使用正规DLL函数的引用和导出 和 在C++包含C函数或C头文件时使用。使用时在前面加上extern “c” 关键字即可。  
  
**十七、构造函数与析构函数**  
派生类的构造函数应在构造函数初始化表里调用基类的构造函数；  
派生类和基类的析构函数应加virtual关键字。  
不要小看构造函数和析构函数，其实编起来还是不容易。

**十八、#IFNDEF/#DEFINE/#ENDIF有什么作用**  
可以用来防止该头文件被重复包含