**C++里是怎么定义常量的？常量存放在内存的哪个位置？**

A：常量在C++里的定义就是一个const加上对象类型，常量定义必须初始化。对于局部对象，常量存放在栈区，对于全局对象，常量存放在全局/静态存储区。对于字面值常量，常量存放在常量存储区。

**你刚刚说到了const，const修饰成员函数的目的是什么？**

A：const修饰的成员函数表明函数调用不会对对象做出任何更改，事实上，如果确认不会对对象做更改，就应该为函数加上const限定，这样无论const对象还是普通对象都可以调用该函数。

**new/delete与malloc/free的区别是什么？**

**说说reinterpret\_cast.？**

A: 这种是不常用的，可以用于任意类型的指针之间的转换，对转换的结果不做任何保证

**说说dynamic\_cast**

A: 这种其实也是不被推荐使用的，更多使用static\_cast，dynamic本身只能用于存在虚函数的父子关系的强制类型转换，对于指针，转换失败则返回nullptr，对于引用，转换失败会抛出异常。

**Q：你刚刚提到虚函数表，具体是怎样实现运行时多态的**。

A：简单来讲，子类若重写父类虚函数，虚函数表中，该函数的地址会被替换，对于存在虚函数的类的对象，在VS中，对象的对象模型的头部存放指向虚函数表的指针，通过该机制实现多态。

Q：extern “C” ？

A：C++调用C函数需要extern C，因为C语言没有函数重载。

Q：C++函数栈空间的最大值 ？

A：默认是1M，不过可以调整

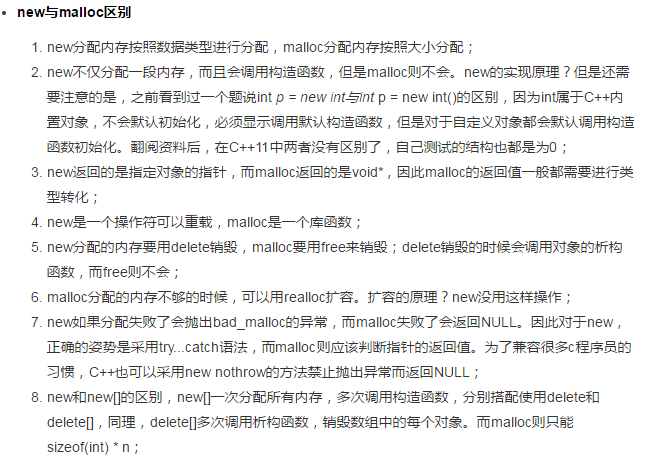
[extern关键字作用](http://www.cnblogs.com/yc_sunniwell/archive/2010/07/14/1777431.html)

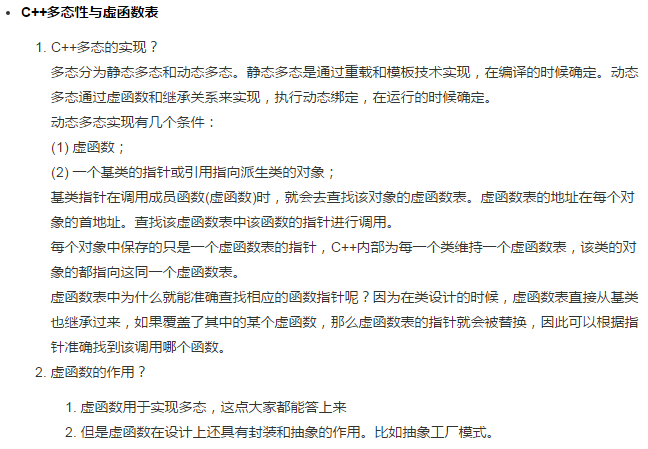
extern声明变量在在外部定义？

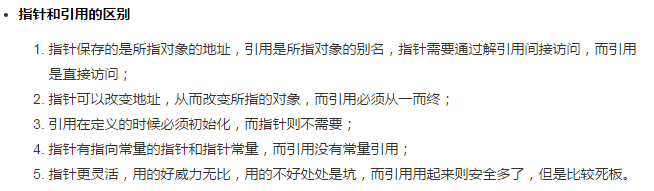
extern修饰函数？

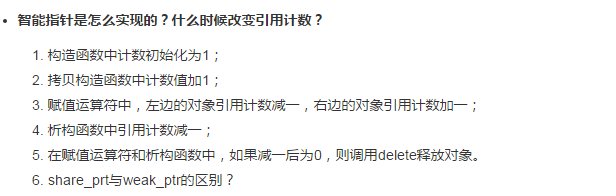
extern C的作用？用法？

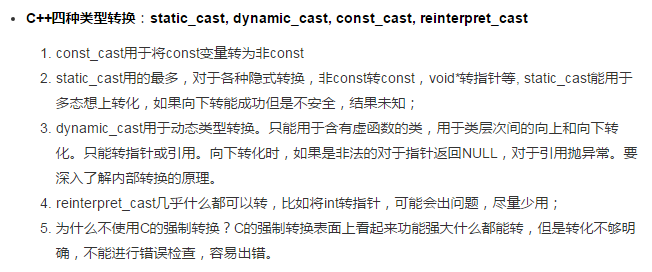
**说说const的作用，越多越好**

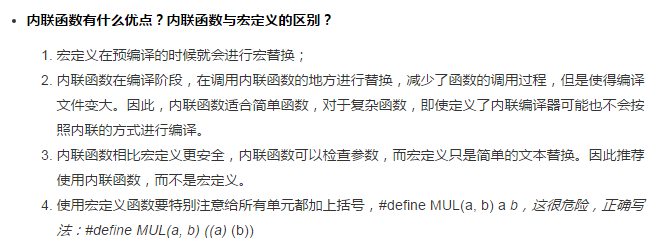












**c++中的内存分布，堆栈，常量区，静态区**

栈区（局部变量区）：由编译器自动分配释放 ，存放函数的参数值，局部变量的值等。其操作方式类似于数据结构中的栈

堆区（动态存储区）：一般由程序员分配释放(new)， 若程序员不释放，程序结束时可能由OS回收 。注意它与数据结构中的堆是两回事，分配方式倒是类似于链表。

自由存储区：和堆区类似，对应malloc

全局变量与静态变量区：全局变量和静态变量的存储是放在一块的，初始化的全局变量和静态变量在一块区域， 未初始化的全局变量和未初始化的静态变量在相邻的另一块区域。 - 程序结束后有系统释放

代码区：存放函数体的二进制代码

PS: 栈由系统自动分配，速度较快。但程序员是无法控制的。   
堆是由new分配的内存，一般速度比较慢，而且容易产生内存碎片,不过用起来最方便.

**C++/C的内存分配，栈和堆的区别，为什么栈要快**

对堆来说，频繁new/delete会造成内存空间不连续，碎片化，是程序效率降低。

栈是机器系统提供的数据结构，计算机会在底层对栈提供支持：分配专门的寄存器存放栈的地址，压栈出栈都有专门的指令执行，这就决定了栈的效率比较高。堆则是C/C++函数库提供的，它的机制是很复杂的。例如为了分配一块内存，库函数会按照一定的算法在堆>内存中搜索可用的足够大小的空间，如果没有足够大小的空间（可能是由于内存碎片太多），就有可能调用系统功能去增加程序数据段的内存空间，这样就有机会>分到足够大小的内存，然后进行返回。显然，堆的效率比栈要低得多。

管理方式：对于栈来讲，是由编译器自动管理，无需我们手工控制；对于堆来说，释放工作由程序员控制，容易产生memoryleak

生长方向：对于堆来讲，生长方向是向上的，也就是向着内存地址增加的方向；对于栈来讲，它的生长方向是向下的，是向着内存地址减小的方向增长。

**static关键字的作用，定义的变量存储在进程空间的哪个区域**

静态全局变量，首先是全局变量，文件内部各个函数可见,由于static关键字的修饰,对其它cpp文件不可见

函数内部定义的静态局部变量，仍然是局部变量，函数外部不可见

存储在静态数据区的变量会在程序刚开始运行时就完成初始化，也是唯一的一次初始化

**static的第三个作用是默认初始化为0（static变量）**

[**C++中explicit关键字的作用**](https://www.cnblogs.com/winnersun/archive/2011/07/16/2108440.html)

Explicit用来防止由构造函数定义的隐式转换

**宏定义#define与const常量有什么区别**

**讲一下面向对象编程**

一个世界，可以完全由对象组成，而将算法所基于的世界只用对象表现出来，再进行后续代码的编写，这种编程方法就被称作面向对象的编程思想

面向对象编程基于 数据抽象-类和对象 （抽象，封装，继承与多态）

多态 [多态性](https://baike.baidu.com/item/%E5%A4%9A%E6%80%81%E6%80%A7)是指相同的操作或函数、过程可作用于多种类型的[对象](https://baike.baidu.com/item/%E5%AF%B9%E8%B1%A1)上并获得不同的结果。不同的[对象](https://baike.baidu.com/item/%E5%AF%B9%E8%B1%A1)，收到同一消息可以产生不同的结果，这种现象称为[多态性](https://baike.baidu.com/item/%E5%A4%9A%E6%80%81%E6%80%A7)

从系统实现的角度看:  多态性分为两类：**静态多态性**和**动态多态性**。函数重载和运算符重载实现的多态性属于静态多态性。（静态联编，早捆绑）动态多态性则是通过虚函数（Virtual Function）实现的。（动态联编，晚捆绑，是根据对象的类型）

虚函数的作用：**允许子类中重新定义与基类同名的函数，并且可以通过父类的指针或引用访问父类和子类中的同名函数**

虚函数用法：

1. 在父类中用  virtual 关键字声明成员函数为虚函数。这样就可以在子类中重新定义此函数，赋予新的功能。
2. 在子类中重新定义此函数：**要求函数名，函数返回值，函数的参数个数，类型**全部与父类的虚函数相同。
3. C++编译器规定，当一个成员函数被声明为虚函数后，其子类中的同名函数自动成为虚函数。（即当一个函数在父类
4. 中被声明为virtual，那么在所有的子类中它都是virtual。）因此在子类重新声明该虚函数时，可加virtual，可不加但习惯上一般加上，使程序更清晰。
5. 在子类中的重定义通常称为重写。

**虚函数 纯虚函数 虚函数表**

抽象类是指包括至少一个纯虚函数的类。

**定义一个函数为纯虚函数，才代表函数没有被实现，定义他是为了实现一个接口，起到一个规范的作用，规范继承这个类的程序员必须实现这个函数。**

**多态**用虚函数来实现，结合动态绑定。：C++的多态性用一句话概括就是：在基类的函数前加上virtual关键字，在派生类中重写该函数，运行时将会根据对象的实际类型来调用相应的函数。如果对象类型是派生类，就调用派生类的函数；如果对象类型是基类，就调用基类的函数。

**类什么时候会析构**

对象被删除之前

**C++解决内存泄漏的方法，如何判断**

**手写atoi**

**std::string实现**

**protobuf 3版本相较于protobuf 2，有哪些改进**