**this指针**

隐含于每一个非静态成员函数的指针，指向调用该成员函数的那个对象。当对一个对象调用成员函数时，编译程序先将对象的地址赋给this指针，然后调用成员函数，每次成员函数存取数据成员时，都隐式的使用this指针。

**储存类**

auto 让编译期自己翻译

register 定义储存在寄存器中而不是内存中的局部变量 主要为了优化性能

static 编译期在程序生命周期内保持局部变量的存在，不需要每次进入和离开时创建和销毁

extern 告诉编译期这个变量在别的地方定义了

**内联函数**

调用时将函数体直接插入调用处，减少函数调用带来的开销。

优点：效率高；编译期会检查类型，保证调用正确，而宏定义不会；类中声明同时定义的成员函数会自动转化为内联函数，因此内联函数可以访问类的成员变量。

缺点：如果执行函数体内代码的时间，相比于函数调用的开销较大，那么效率上的收获会很小。

**volatile （易变的）**

一种类型修饰符，用它声明的变量可以被某些编译器未知的因素更改。所以告诉编译器不应对这样的对象进行优化。

**assert**

宏，如果它的条件返回错误，则终止程序。

**typedef**

关键字 用来简化复杂名称类型

比如C语言中

typedef struct Student{

int age;

}S;

等价于

struct Student{

int age;

};

typedef struct Student S;

**explicit(显示)关键字**

修饰构造函数时，可以防止隐式转换和赋值初始化

修饰转换函数时，可以防止隐式转换。

**友元**

能访问私有成员

破坏封装性

友元关系不可传递；单向性

友元声明的形势及数量不受限制

**函数重载**

函数重载是一种特殊情况，C++允许在同一作用域中声明几个类似的同名函数，这些同名函数的形参列表（参数个数，类型，顺序）必须不同，常用来处理实现功能类似数据类型不同的问题。

    在C++中不仅函数可以重载，运算符也可以重载。例如：

 运算符<<,>>。既可以做移位运算符，也可以做输出，输入运算符。

注意：重载函数的参数个数，参数类型或参数顺序三者中必须有一个不同

**函数重载是一种静态多态（编译期 早绑定）在调用时就知道函数的地址在哪里**

面试：

1.C语言中为什么不支持函数重载

编译时仅进行简单的重命名

2.C++底层如何实现函数重载

同名函数在函数表中生成的名称不一样

3.c++中能否将一个函数按照C风格来编译

可以 只要在函数名前加上 extern “C”

**虚函数**

虚函数：用virtual修饰成员函数，使其成为虚函数 （派生类自己定义继承的函数）

**虚函数是一种动态多态（运行期 晚绑定）**

虚函数即表明子类中可能重新定义函数

虚函数的实现：vtable指针（内存最底层）指向真正调用的函数

普通函数，静态函数，构造函数不能是虚函数

**虚析构函数**

new delete 希望调用派生类的析构函数 如果没有虚析构函数 只会调用基类的 造成内存泄露 虚析构函数调用顺序为 派生类析构 基类析构

构造函数不需要虚函数 因为已存在基类再派生类的调用顺序

**纯虚函数**

基类中不能对虚函数给出有意义的实现，而把它声明为纯虚函数，它的实现留给子类

virtual int A() = 0;

意义：接口继承

虚函数在子类可以不重写；但纯虚函数必须在子类实现才可以实例化子类；带纯虚函数的类叫做抽象类（接口类：只含有纯虚函数），这种类不能直接生成对象，而只有被继承，并重写其虚函数后，才能使用。

**虚继承**

用于解决多继承下菱形继承的问题 A基类 BC间接基类 D子类 存在ABD和ACD两条路

虚继承使得D中只保留一份A的成员

**虚继承、虚函数**

* 相同之处：都利用了虚指针（均占用类的存储空间）和虚表（均不占用类的存储空间）
* 不同之处：
  + 虚继承
    - 虚基类依旧存在继承类中，只占用存储空间
    - 虚基类表存储的是虚基类相对直接继承类的偏移
  + 虚函数
    - 虚函数不占用存储空间
    - 虚函数表存储的是虚函数地址

**如何定义一个只能在堆上（栈上）生成对象的类？**

**只能在堆上**

方法：将析构函数设置为私有

原因：C++ 是静态绑定语言，编译器管理栈上对象的生命周期，编译器在为类对象分配栈空间时，会先检查类的析构函数的访问性。若析构函数不可访问，则不能在栈上创建对象。

**只能在栈上**

方法：将 new 和 delete 重载为私有

原因：在堆上生成对象，使用 new 关键词操作，其过程分为两阶段：第一阶段，使用 new 在堆上寻找可用内存，分配给对象；第二阶段，调用构造函数生成对象。将 new 操作设置为私有，那么第一阶段就无法完成，就不能够在堆上生成对象。

**智能指针**

#include<memory>

**shared\_ptr** 多个智能指针可以共享一个对象，对象的最末一个拥有者有着责任销毁对象，并清理与该对象相关的所有资源

可防范cross-dll问题（对象在dll中被new创建 却在另一个dll中被delete销毁）

**weak\_ptr** 允许共享但不拥有某对象，一旦最末一个拥有该对象的智能指针失去了所有权，任何弱指针都会自动成空

可打破环状引用（两个已经没有被使用的对象彼此指着，使之看似还在被使用的状态）

**unique\_ptr** 一种在异常时可以帮助避免内存泄露的智能指针。确保一个对象和其相应的资源同一时间只被一个pointer拥有。

**强制类型转换运算符**

**static\_cast<new type> (expression)**

用于非多态类型 不执行运行时检查类型 通常用于转换数值数据类型 子类转化为父类安全 父类转化为子类不安全

**dynamic\_cast<type\*/type&/type&&> (expression)**

用于多态类型 执行运行时类型检查 只适用于指针或引用

**const\_cast<type\*/type&> (expression)**

用于删除const, volatile和\_unaligned特性

**reinterpret\_cast**

允许指针类型和其他类型的互相转换

**bad\_cast**

由于强制转换为引用类型失败，dynamic\_cast运算符引发此异常

C++ 核心理念

分配（allocate）与初始化紧密绑定

**面试常见问题**

5.在类的内部定义成员函数的函数体，这种函数会具备那种属性？  
答：这种函数会自动为内联函数，这种函数在函数调用的地方在编译阶段都会进行代码替换

6.成员函数通过什么来区分不同对象的成员数据？为什么它能够区分？  
答：通过this指针来区分的， 因为它指向的是对象的首地址。

7.C++编译器自动为类产生的四个缺省函数是什么？  
答：默认构造函数，拷贝构造函数，析构函数，赋值函数。

14.静态函数存在的意义？  
答：静态私有成员在类外不能被访问，可通过类的静态成员函数来访问；  
当类的构造函数是私有的时，不像普通类那样实例化自己，只能通过静态成员函数来调用构造函数。

23.对象间是怎样实现数据的共享的？  
答：通过类的静态成员变量来实现的。静态成员变量占有自己独立的空间不为某个对象所私有。

29.是不是一个父类写了一个virtual 函数，如果子类覆盖它的函数不加virtual ,也能实现多态?  
virtual修饰符会被隐形继承的。  
virtual可加可不加,子类覆盖它的函数不加virtual ,也能实现多态。

41.解释堆和栈的区别。  
栈区（stack）— 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。  
堆:一般由程序员分配释放， 若程序员不释放，程序结束时可能由OS回收 。

55.全局变量和局部变量在内存中是否有区别？如果有，是什么区别？  
答 、全局变量储存在静态数据区，局部变量在堆栈中。

66.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。  
2、 栈上分配，函数内的局部变量就是从这分配的，但分配的内存容易有限。  
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。

1. sizeof 和 strlen的区别

答、sizeof返回整个数组的长度 strlen是C风格 返回包含字符的长度 且只返回可见字符的长度

2. malloc free和new delete 的区别

答、new先调用malloc分配内存 然后调用构造函数创建对象

delete先调用析构函数 然后调用free释放空间

3. push\_back和emplace\_back的区别

答、push\_back先调用构造函数创建对象，然后调用拷贝构造或者转移构造将元素移植进容器

emplace\_back直接原地构造，相比push\_back更简洁，利用的资源更少

4. const和static

答、const

修饰变量，说明不可改变；

修饰指针，分自身是常量的指针和指向常量的指针；

修饰引用，用于形参，既避免了拷贝，又避免了函数对值的修改；

修饰成员函数，说明该成员函数内不能修改成员变量。

static

修饰普通变量，使变量储存在静态区域，在main之前就分配了空间；

修饰普通函数，表明函数的作用范围，仅在定义该函数的文件内才能使用；

修饰成员变量，使所有的对象只保存一个该对象，而且不需要生成对象就可以访问该成员；

修饰成员函数，使得不需要生成对象就可以访问该函数，但是在函数内不能访问非静态成员。

5. 左值和右值

答、左值 可以获取其地址

右值 临时值 无法获取其地址

左值引用 int& i1 = 10;

右值引用 int&& i2 = 10; 包含值10的内存位置有了一个名称i2，i2本身变成了一个左值

6. struct和class

答、最本质的区别是默认的访问控制。struct作为数据结构的实现体，它默认的数据访问控制是public，而class作为对象的实现体，它默认的成员变量访问控制是private的。