C，C++部分：

static

答:static是静态变量,初始化只运行一次,之后每次赋值都保留值,直到作用域结束.

const 指针常量和常量指针

答:const指针常量是指不可改变的指针，常量指针是指向常量的指针

++i 和i++的区别

答:++i是先对i进行加加操作,之后的表达式用得是进行运算后的I,等于对i进行加操作再应用i.

i++是先应用i,之后再对i进行运算,等于拷贝一个i对拷贝的i进行加操作,先使用原来的i,之后再使用拷贝并且加操作的i.

返回局部变量?原因

答:因为,局部变量是栈存储,出了作用域就自动释放,返回局部变量可以减少内存消耗,提升效率.

指针数组？数组指针

答:用来存放多个指针的数组叫做指针数组，指向数组的指针叫做数组指针

指针函数，函数指针？

答：指针函数是一个返回值为地址的函数，其返回值必须用同类型的指针变量来接收。函数指针是指一个指向函数的指针变量

ifndef/define/endif作用？

答：防止头文件的重复包含和覆盖

c和C++是如何定义常量的，区别？

答：const或define关键字修饰，const是保护数据不被修改，define是统一一个数据

%一个特点？

答：%是取余的运算符

函数传参数方式(值，指针,引用)

构造函数作用，特征

答：作用是初始化实例对象，其特点是函数名和类名相同，没有返回值

数组和指针的区别

答：数组要么在静态存储区被创建（如全局数组），要么在栈上被创建。数组名对应着（而不是指向）一块内存，其地址与容量在生命期内保持不变，只有数组的内容可以改变。指针可以随时指向任意类型的内存块，它的特征是“可变”，所以我们常用指针来操作动态内存。指针远比数组灵活，但也更危险。

内存分配？

答：内存分配是指在程序执行的过程中分配或者回收存储空间的分配内存的方法。内存分配方法有静态内存分配和[动态内存](http://baike.baidu.com/view/3312702.htm)分配两种

空指针，野指针？

答：空指针：空指针是一个特殊的指针值，也是唯一一个对任何指针类型都合法的指针值。指针变量具有空指针值，表示它当时处于闲置状态，没有指向有意义的东西。野指针：野指针也就是指向不可用内存区域的指针。通常对这种指针进行操作的话，将会使程序发生不可预知的错误。

<>和“”区别？

答：< >内引用的是系统的文件，” ”内调用的是自定义的文件

虚函数？

答：在某基类中声明为 virtual 并在一个或多个派生类中被重新定 义的成员函数

构造函数 virtual?

答：构造函数不能是虚函数

引用和指针的区别

答：指针是一个变量，只不过这个变量存储的是一个地址，指向内存的一个存储单元；而引用跟原来的变量实质上是同一个东西，只不过是原变量的一个别名而已。

析构函数 virtual?原因

答：在实现多态时，当用基类操作派生类，在析构时防止只析构基类而不析构派生类的状况发生

菱形继承问题？

答：两个子类[继承](http://baike.baidu.com/view/125322.htm)同一个父类，而又有子类同时继承这两个子类。

构造函数可以virtual?

答：构造函数不能virtual

Extern？

答：extern可置于变量或者函数前，以表示变量或者函数的定义在别的文件中，提示[编译器](http://baike.baidu.com/view/487018.htm)遇到此变量或函数时，在其它模块中寻找其定义。另外，extern也可用来进行链接指定。

全局变量和局部变量的区别？

答：静态全局变量，只本文件可以用。全局变量是没有定义存储类型的外部变量,其作用域是从定义点到程序结束.

Main函数之前会执行什么代码？

答：在可执行文件被加载之后，控制权立即交给由编译器插入的Start函数

拷贝构造函数？

答：

结构体和联合的内存区别？

答：结构体是多个不同类型变量的组合，其占用的内存大小是所有变量大小的总和。联合体是多个不同类型变量的抽取，其占用内存的大小是联合体中最大的那个变量的大小

类型转化方式？

答：static\_cast，const\_cast，dynamic\_cast，reinterpret\_cast

动态绑定？

答：动态绑定是将一个过程调用与相应代码链接起来的行为。是指与给定的过程调用相关联的代码，只有在运行期才可知的一种绑定，他是[多态](http://baike.baidu.com/subview/126521/126521.htm)实现的具体形式

重载重写？

答：重载是让类以统一的方式处理不同类型数据的一种手段，重写是在子类中定义某方法与其父类有相同的名称和参数

空类占用空间？

答：空类占用一个字节的空间

Inline？

答：在函数声明或定义中函数返回类型前加上关键字inline即指定为内联函数。

Malloc/free new/delete?

答：.malloc/free是C/C++语言的标准库函数，new/delete是C++的运算符。new能够自动分配空间大小

纯虚函数？

答：在许多情况下，在[基类](http://baike.baidu.com/view/535539.htm)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，它的实现留给该基类的[派生类](http://baike.baidu.com/view/535532.htm)去做

抽象基类？

答：类里定义了纯虚成员函数的基类

继承的3种方式

答：公有继承，私有继承，保护继承

继承的3种方式？

答：公有继承(public)：公有继承的特点是基类的公有成员和保护成员作为派生类的成员时，它们都保持原有的状态，而基类的私有成员仍然是私有的，不能被这个派生类的子类所访问。私有继承(private)：私有继承的特点是基类的公有成员和保护成员都作为派生类的私有成员，并且不能被这个派生类的子类所访问。 保护继承(protected) ：保护继承的特点是基类的所有公有成员和保护成员都成为派生类的保护成员，并且只能被它的派生类成员函数或友元访问，基类的私有成员仍然是私有的。

C++默认有哪些函数？

答：默认构造函数，拷贝构造函数，默认赋值函数以及析构函数

类中含有一个静态成员空间？

答：

类中含有一个virtual函数空间？

答：

累中有一个virtual函数和char占用空间

答：

stl?Vector list区别 map set区别 mutilemap map区别

答：

函数模版？类模版？

答：

浅拷贝和深拷贝的区别

答：

多态？

答：

冒泡思想

快排思想

选择排序

插入排序

Cocos2d-x部分：

内存机制(OC)

创建精灵的方式?

锚点？

Schedule？

如何理解导演，场景，层，精灵

触摸优先级

适配策略

不同精灵能否同时执行一个动作

节点引用计数什么时候会增加

实现代理

spawn

sequence

searchPath

nullptr 和 NULL区别

runWithScene?

replaceScene?

pushScene?

popScene?

内存管理机制

texture2D

spriteFrameCache

场景跳转的内存变化，应如何避免

异步加载 addAsyncImage(“ImagePath”,”CallBack”)

分析创建的数组为何往往需要retain下

实现一个精灵围绕场景中的某个点转动（该点不是精灵锚点）

Lua部分：

类型(8)

闭合函数

非全局函数

尾调用，好处

协同程序参数问题

元表 元方法

\_\_index

\_\_newindex

rawget

rawest

lua实现面向对象

lua继承

说出以下三种函数中分别可以抛出什么异常

void f();

答：能抛出任何异常

void f() throw()

答：不抛出任何异常

int func( int x ) throw( int, Error\_message)

答：只抛出int类型的异常

运行以下程序会有什么样的结果：

int main()

{

　char \*str = new char[100];

　strcpy( str, "hello" );

　delete []str;

　if(str != NULL)

{

strcpy(str,”world”);

printf(“%s\n”,str);

}

return 0;

}

答：程序崩溃

void GetMemory( char \*\*p, int num )

{

　\*p = new char[num];

}

int main( )

{

　char \*str = NULL;

　GetMemory( &str, 100 );

　strcpy( str, "hello" );

　cout<<str<<endl;

return 0;

}

运行以下程序会有什么样的结果：

char \*GetMemory( void )

{

　char p[] = "hello world";

　return p;

}

int main( )

{

　char \*str = NULL;

　str = GetMemory();

　cout<<str<<endl;

return 0;

}

运行以下程序会有什么样的结果：

void GetMemory( char \*p )

{

　p = new char[100];

}

int main( )

{

　char \*str = NULL;

　GetMemory( str );

　strcpy( str, "hello world" );

　cout<<str<<endl;

return 0;

}

union Person{

short num;

double num1;

int name;

}

sizeof(Person) =8

Struct A{

char s[10];

int num;

short num1;

}

sizeof(A) = 20

Struct B{

Int s;

int num;

short num1;

}

sizeof(B) = 12

struct C{

}

sizeof(C) = 1

class AA{

}

sizeof(AA) = 1

class BB{

int num1;

}

sizeof(BB) = 4

class CC{

static int num;

}

sizeof(CC) = 1

class DD{

virtual ~DD(){}

}

sizeof(DD) =

class EE{

char n;

virtual ~DD(){}

}

sizeof(EE)