1.简述C++的特性

对象的改良，构造函数继承，显示虚函数重载，Lambada表达式，std::function和std::bind，只能指针

2.简述quick中的class是如何进行封装的？

答：当class继承自c++ 类时 （如 class(classname, super)中 super 参数是function 或者super的\_\_ctype == 1）

1.自动创建cls ={} 空表并增加以下成员

cls.\_\_create = super --New 时候要初始化的C++对象

cls.ctor = function () end -- 默认的空构造

cls.\_\_cname = classname --类名字

cls.\_\_ctype = 1 --类型

--然后New的时候做了这些操作，不管什么类，原型都是一个管理表

function cls.new(...) --每个new出来的c++对象

local instance = cls.\_\_create(...)

-- copy fields from class to native object

for k,v in pairs(cls) do instance[k] = v end --把参数传进function 从表中克隆成员

instance.class = cls

instance:ctor(...) --创建完之后执行构造

return instance

end

-----------------------------完成后 类似以下的原型表 ---------------------------------------

--cls = {\_\_\_cname = "GameObject",\_\_ctype = 1,ctor =function () end,\_\_create = function return node end,new = function ()return node end}

--类似 local GameObject = class("GameObject", function() return display.newNode() end) 的操作

--创建基础表之后我们又在其上(GameObject)绑定了一些成员方法,这些都是加入原型表的 New出来的时候 是执行表中的\_\_create之后，再复制表成员。

3.简述背包的实现过程

答：

1.利用cocoStudio设计一个工程，创建一个listview， 完成后，发布资源。

2.设计游戏数据xml，json，excel，csv，table等。

3.创建listview对应的cell，这个自定义的类要继承ccui.Widget。

4.骨骼动画师如何实现换装的？

答：

1. 获取骨骼的名称

2. 创建皮肤

3. 获取当前皮肤的下标

4. 添加皮肤，下标依次递增

5. 替换皮肤

5.简述module的实现过程，input 与require的

答：1.model的实现过程：将所有的数据全部存储在一个table中。model是一个模块,能通过导入在另一个lua文件中访问module中的方法、变量。

2.import是quick里边的，它与lua中的require功能是相同的，都是载入一个模块，但import具有一定程度的自动化特性。

Std关键字：

静态成员矢量：

生命周期与程序相同，指初始化一次，必须在类外初始化

xml解析json