**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：王敏 学 号：201922080602 指导教师：丁旭阳**

**实验地点：综合楼A2-412 实验时间：2019年11月20日**

**一、实验室名称：**

Linux环境高级编程实验室

**二、实验项目名称：**

对象的序列化

**三、实验学时：**

4学时

**四、实验目的：**

练习linux下对象的序列化，这个将可以用于传输到网络中实现与另外一台计算机的实时通讯。以便于最终课程实验的服务器和服务器的通讯。

**五、实验内容：**

由简单到复杂的将对象进行序列化，从最简单的一个类一个对象的序列化到n个类m个对象的序列化。

**六、实验步骤：**

1.实现版本1：将1个类的1个对象序列化成文件 ：

可以看到在该目录下生产了序列化的data文件，并且输出数据对象b的数据和给对象a的赋值相同。

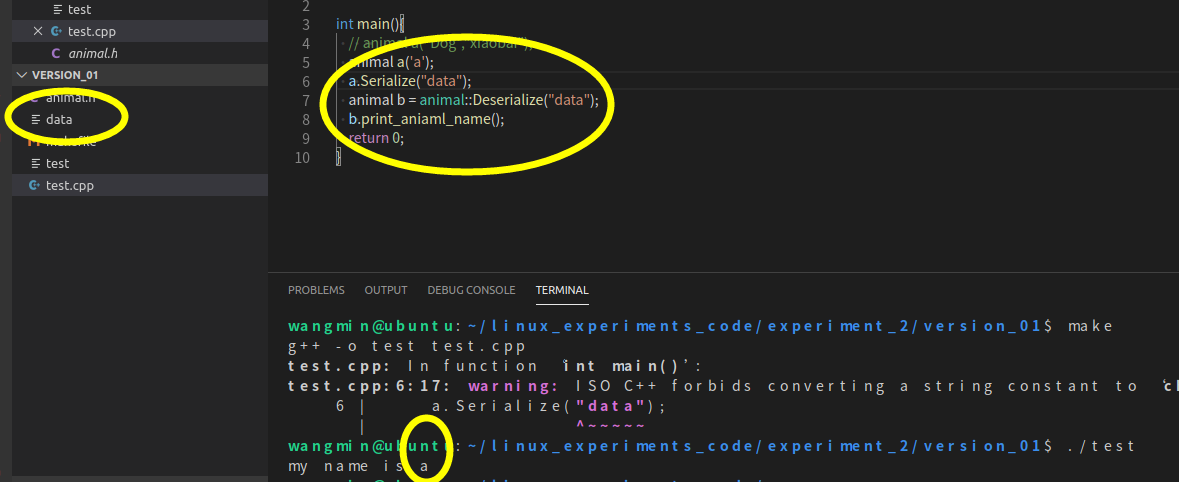


图1 版本1实验结果

2．版本2将1个类的n个对象序列化到文件。使用容器将要序列化的对象装入vector容器，存储的时候再依次取出来存入文件即可。反序列化时便读取文件，并且依照文件大小分割、强制转换。

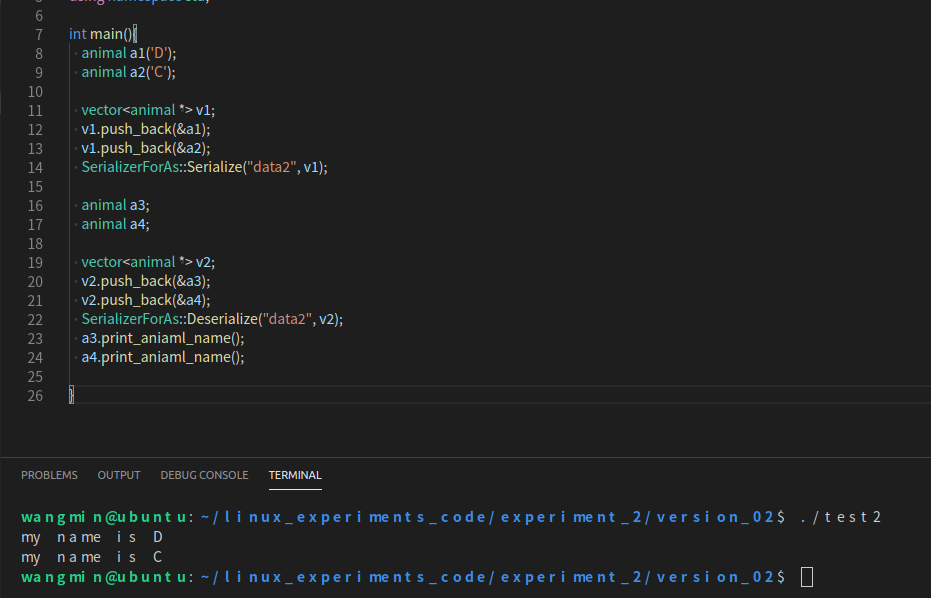


图2 版本2实验结果

3.版本三需要为每个序列化对象附加属性，那么久用struct(当然也可以构造成类，但是由于结构简单，用struct就行了)来存储这些对象，以及对象的类别。这样在强制类型转换时就可以识别出这些对象的长度，用于正确解析这些对象。实验结果如图三所示。



图3 版本3实验结果

4.版本四将三个不同类的多个对象序列化到文件。实现这个的思想是使用面向接口的编程。申明一个接口，每一个需要序列化的对象都是接口的实现。实验结果如图4所示。

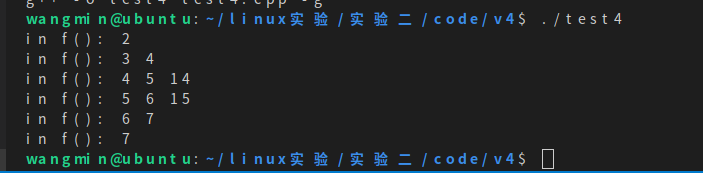


图4 版本4实验结果

5.版本五需要将序列化的文件写到可以指定的地方，添加几行代码使得文件路径从外部加载即可。

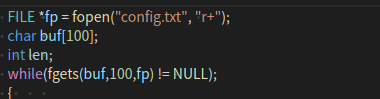


图5 从config中加载文件路径 相关代码

版本5的实验结果如图6所示。

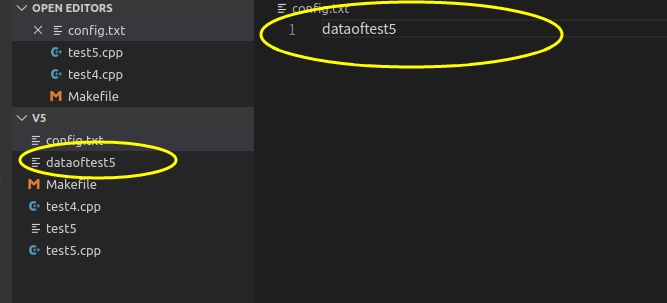


图5 从config中加载文件路径 相关截图

可见生成的序列化文件和配置的文件名相同。

**七、总结及心得体会：**

通过本次实验，我学会了将面向接口的开发思想应用到Linux编程中，学会了用配置文件使得程序更加灵活。

**八、对本实验过程及方法、手段的改进建议：**

老师提供的代码很全面，ppt上说明也很清楚，感觉学的比较懂，。

**报告评分：**

**指导教师签字：**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：王敏 学 号：201922080602 指导教师：丁旭阳**

**实验地点：综合楼A2-412 实验时间：2019年11月27日**

**一、实验室名称：**

Linux环境高级编程实验室

**二、实验项目名称：**

Linux下的网络编程

**三、实验学时：**

4学时

**四、实验目的：**

练习使用Linux下的TCP协议，熟悉一些常用函数，并且应用编程技巧封装这些函数，实现简单的echo服务。.

**五、实验内容：**

分别使用以下几种程序（传统C语言的结构化的编程方法(3.12)

面向对象的编程方法、基于接口的编程方法、静态的面向对象的编程方法、基于方面的编程方法、设计思想，封装基本TCP通信机制，并在封装后的机制上，实现echo服务。

**六、实验步骤：**

1. 程序化的编程即是顺序的使用相关方法建立连接，程序流程如下，类比python的网络编程。

**服务端：**

**Step1：**建立Socket对象**Sk = socket.socket()**

**Step2：**把一个地址族中的特定地址赋给socket **Sk.bind((host,port))**

**Step3：**开启监听进程等待客户端链接服务端**Sk.listen(1)**

**Step4：**接受客户端的申请，获取客户端的IP地址及其端口PORT **clnt, addr = Sk.accept()**

**Step5：**建立好连接之后，等待接受客户端的数据 **data = clnt.recv(1024)**

**Step6：**应答客户机的请求，将对应的数据发回去 **clnt.sendall(data)**

**Step7：**关闭socket对象**Sk.close()**

**TCP客户端的配置步骤：**

**Step1：**建立Socket对象 **S = socket.socket()**

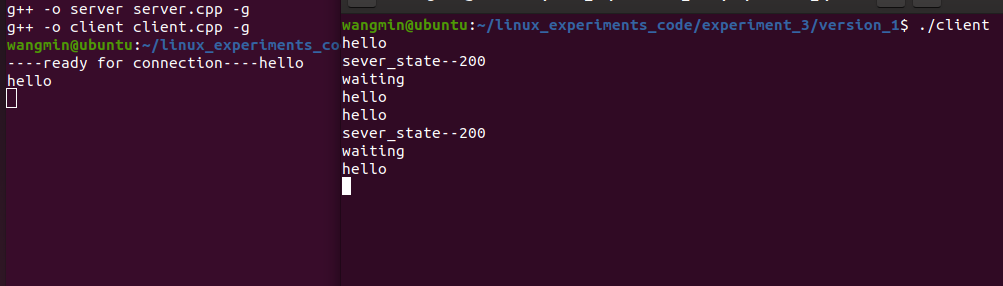
**Step2：**把连接客户端到对应的服务端 **S.connect((host, port))**

**Step3：**连接成功后向服务端发送数据**S.sendall(cmd)**

**Step4：**接等待服务端的回应数据**data = S.recv(1024)**

**Step5：**关闭socket对象 **Sk.close()**

结果如下图。其中输出有写和服务器相关状态的返回。



1. 面向对象
   1. 将需要建立的服务端的变量，函数封装在父类中，并且声明一个虚函数用于子类继承实现。
   2. 将需要建立的客户端的变量，函数封装在父类中，并且声明一个虚函数用于子类继承实现。
   3. 声明自己的服务端、客户端类，并且继承实现服务端的服务和客户端的应用。代码如下：

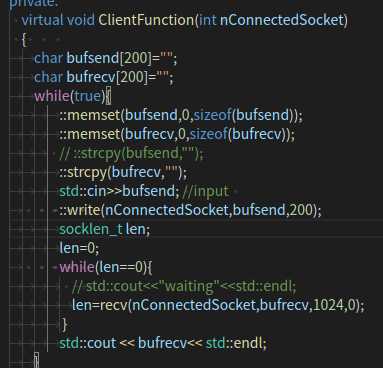


图1：客户端继承的应用程序代码

外层用于循环等待输入，将bufsend和bufrecv置为空，并且将输入字符串写入套接字，第二个while等待服务端返回，当接收缓冲区bufsend长度不为0时，输出得到的服务端字符。

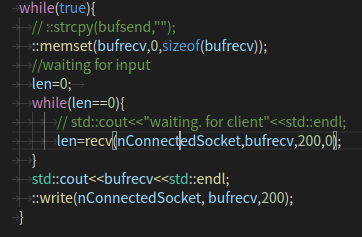


图2：服务端继承的服务程序代码

外层循环用于等待客户端应用发送请求，将接收缓冲区bufrecv置空，第二个while通过判断接收缓冲区长度是否为空等待应用程序发送请求，收到请求后的将收到的字符串发送给客户端应用。

* 1. 一开始我接收缓冲区和发送缓冲区都是用的同一个缓冲区，导致客户应用程序实际上读取的是客户应用程序发送缓冲区自己的东西，后来改变方法使用两个不同的缓冲区。

通过将服务端的服务程序发送改为”Hello, this is server”，可以看出确实是服务端发回的信息；如图3所示：

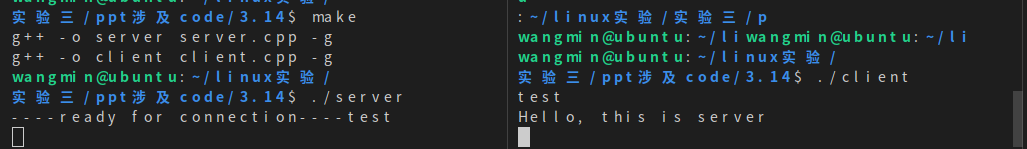


图3：确信是服务端传回的信息

5．将服务端的服务程序改写成发送收到的进程即可实现简单的echo服务。

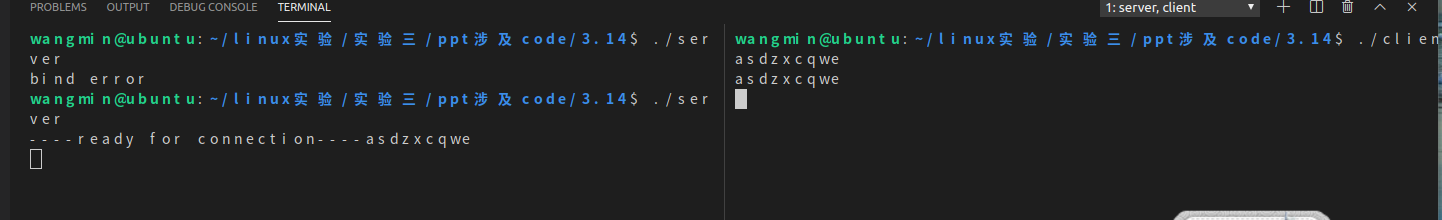


图4：echo服务实现截图

1. 面向接口

后边实现的结果截图都类似，所以截取关键代码。

C++中没有Interface关键字，以客户程序为例子，先申请一个接口类Observer，Observer中定义一个公有的虚方法用于实现**接口的定义**。如图5所示。

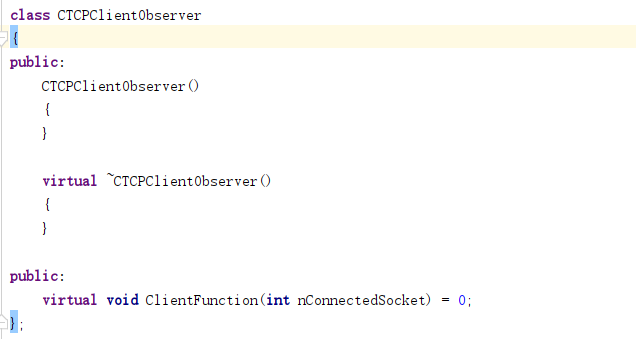


图5：接口的声明（定义）截图

需要**实现这个接口**的方法只需要要继承这个类并在**子类中实现**这个方法。如图6所示。



图6：子类中实现这个方法

需要**调用这个接口**的只需要将声明接口的父类**声明**在类中，调用该方法。这样就实现了**定义与实现**的分离。

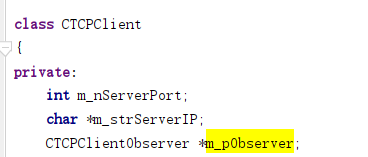
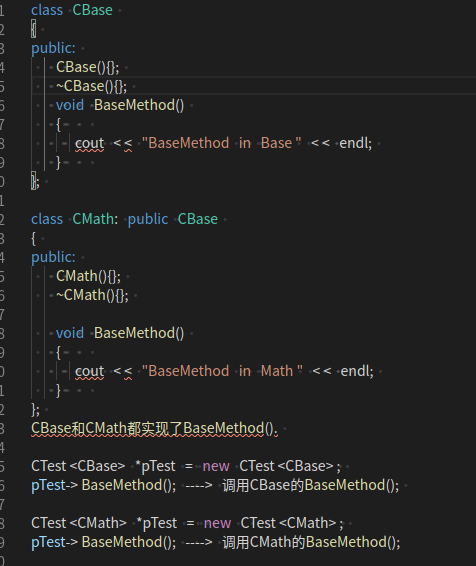


图7：调用这个接口

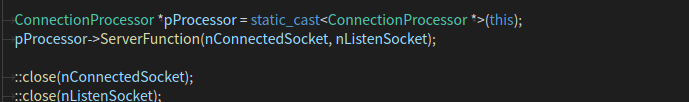
1. 基于静态的面向对象编程，示例如图8所示，实现了多态。



这样，就能根据不同的类来调用不同的方法，实现了多态。

5.基于方面的编程

这个概念我还是第一次听说，手段：分离业务的主逻辑和次逻辑的一种思想。 目的：解决的是逻辑分离问题（主逻辑和次逻辑分开，其实主要是分离业务逻辑和非业务逻辑分开）。实现主要代码如下。



**七、总结及心得体会：**

通过几个简单的例子很好的理解了C++面向过程、面向对象、面向接口、面向静态类、面向方面编程的区别。例子详尽。以及了解了linux下socket网络编程，这对后面的工作可能会有帮助。

**八、对本实验过程及方法、手段的改进建议：**

理解之后编程量不大，主要是网络编程知识比较复杂，PPT做的不太友好。参见博客<https://mp.csdn.net/postedit/103533675>，写了网络编程相关的基础，这样一看就看懂了。内容有点太多了…

**报告评分：**

**指导教师签字：**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：王敏 学 号：201922080602 指导教师：丁旭阳**

**实验地点：综合楼A2-412 实验时间：2019年12月4日**

**一、实验室名称：**

Linux环境高级编程实验室

**二、实验项目名称：**

插件框架实验

**三、实验学时：**

4学时

**四、实验目的：**

学会动态库的编写、使用；掌握插件的编写方法，熟悉一些文件操作函数的参数。

**五、实验内容：**

基础知识，动态库的编写；实例1：插件的更新；实例2：多插件的使用；实例3：多插件的选择与使用；实例4：插件设计优化--减少接口以及综合实验：设计一个面向文件统计功能的插件系统。

**六、实验步骤：**

动态库的编写和插件框架练习1属于是入门比较重要的部分，所以特别记录。对于练习2、3、4则不再赘述。

练习1：动态库的编写

写两个函数名，探究extern C的作用。编译动态库文件testlib.so, 包含f()和g()函数。 f()不使用extern C，g()使用extern C，在test.cpp中调用动态库.

图1为使用dlsym()，FuncName参数分别为”f”和”g”的结果。

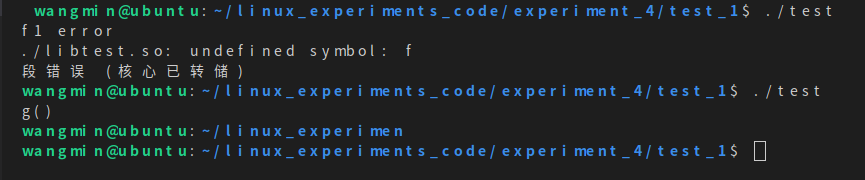


图1 参数分别为”f”和”g”的结果

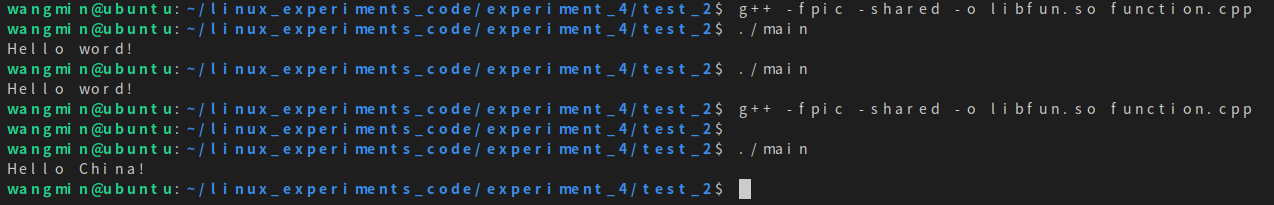
开发一个程序，向屏幕打印“Hello World”；在不重新编译链接原程序的前提下，将打印的文字改为“Hello China”。图2为结果。

图2 只更改生成动态链接库的程序得到的结果

1. 设计一个面向文件统计功能的插件系统

这个的设计思路就是使用IPrintPlugin及派生类，使用插件时生成一个插件对象，插件调用者获得对象指针且仅导出一个接口函数。实验相关结果截图如下

图3是用Makefile生成插件和主程序，图4是随机输入命令时输出的提示，提示怎么用这个程序，图5是调用Help，图6是调用对文件字节数计数的插件，图7是调用对文件计算行数的插件,图8是file1中的字符。

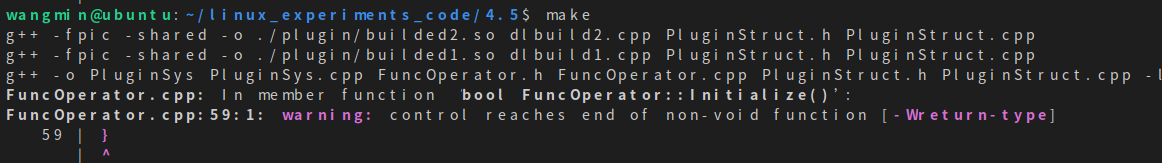


图3：Makefile编译文件



图4：随机输入或者未输入命令，有相应提示

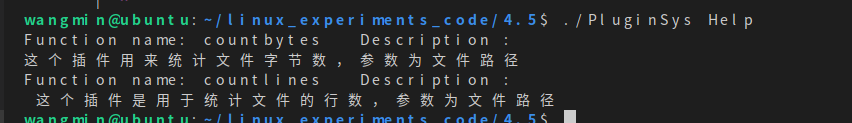


图5：输入Help显示插件信息



图6：对file中文件进行统计字节数

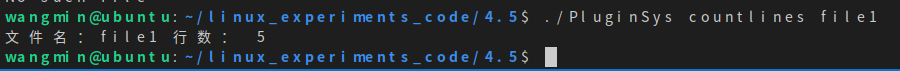


图7：对file中文件进行统计行数

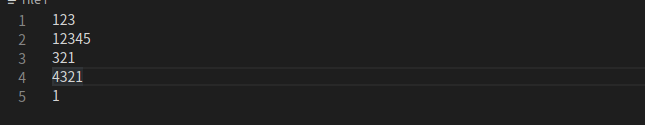


图8：文件中字符串

**七、总结及心得体会：**

通过本次实验我了解了几个linux下面向系统文件路径的方法

**八、对本实验过程及方法、手段的改进建议：**

这个实验教程循序渐进，工作量也不是很大，很适合学习可以适当增加工作量。

**报告评分：**

**指导教师签字：**