# 动态链接库的编译与使用

## 1.Linux 下的动态链接库

动态链接库的基本概念

动态链接库（英语：Dynamic-link library，缩写为DLL）是[微软公司](https://zh.wikipedia.org/wiki/%E5%BE%AE%E8%BD%AF%E5%85%AC%E5%8F%B8)在[微软视窗](https://zh.wikipedia.org/wiki/%E5%BE%AE%E8%BD%AF%E8%A7%86%E7%AA%97)[操作系统](https://zh.wikipedia.org/wiki/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F)中实现共享[函数库](https://zh.wikipedia.org/wiki/%E5%87%BD%E6%95%B0%E5%BA%93)概念的一种实现方式。这些库函数的[扩展名](https://zh.wikipedia.org/wiki/%E6%89%A9%E5%B1%95%E5%90%8D)是.DLL、.OCX（包含[ActiveX](https://zh.wikipedia.org/wiki/ActiveX)控制的库）或者.DRV（旧式的系统[驱动程序](https://zh.wikipedia.org/wiki/%E9%A9%B1%E5%8A%A8%E7%A8%8B%E5%BA%8F))。

所谓动态链接，就是把一些经常会共用的代码（静态链接的[OBJ](https://zh.wikipedia.org/w/index.php?title=OBJ&action=edit&redlink=1)程序库）制作成DLL档，当可执行文件调用到DLL档内的函数时，Windows操作系统才会把DLL档加载内存内，DLL档本身的结构就是可执行文件，当程序有需求时函数才进行链接。通过动态链接方式，内存浪费的情形将可大幅降低。[静态链接库](https://zh.wikipedia.org/wiki/%E9%9D%9C%E6%85%8B%E9%80%A3%E7%B5%90%E5%87%BD%E5%BC%8F%E5%BA%AB)则是直接[链接](https://zh.wikipedia.org/wiki/%E9%93%BE%E6%8E%A5%E5%99%A8)到可执行文件.

动态链接库的好处

* 节省内存和减少交换操作。 很多进程可以同时使用一个 DLL，在内存中共享该 DLL 的一个副本。 相反，对于每个用静态链接库生成的应用程序，Windows 必须在内存中加载库代码的一个副本。
* 节省磁盘空间。 许多应用程序可在磁盘上共享 DLL 的一个副本。 相反，每个用静态链接库生成的应用程序均具有作为单独的副本链接到其可执行图像中的库代码。
* 升级到 DLL 更为容易。 当 DLL 中的函数发生更改时，只要函数的参数和返回值没有更改，就不需重新编译或重新链接使用它们的应用程序。 相反，静态链接的对象代码要求在函数更改时重新链接应用程序。

动态链接库的使用实例

实例:用 vs2015 等开发工具建立工程时，可以在应用程序设置中的应用程序类型设置 DLL 表示建立动态链接库，之后可以在 DLL 项目中添加导出函数以便在编程时在附加的动态库中调用。

## 2. 按要求写出下列几种常用情况的动态链接库的测试样例

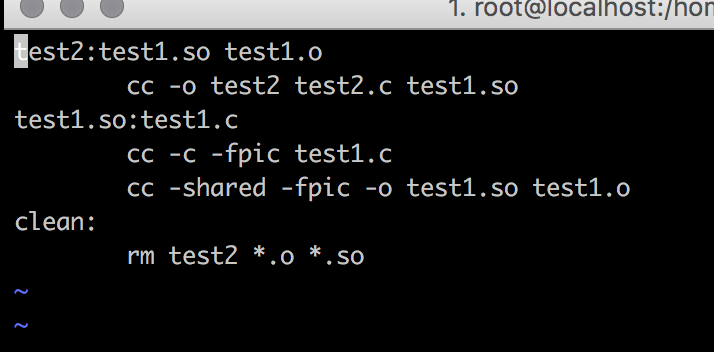
很多步骤和之前makefile中类似 相同的在此不再赘述

建立子目录和.c文件在此略过

主要说下makefile文件

### 2.2

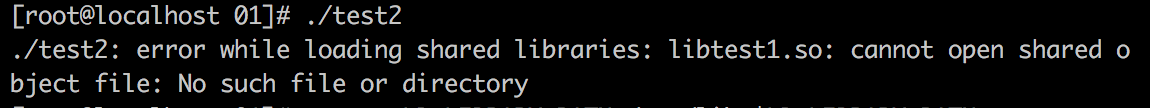
makefile



在这里我把文件名改成了libtest1.so

这样-ltest1就可以找到

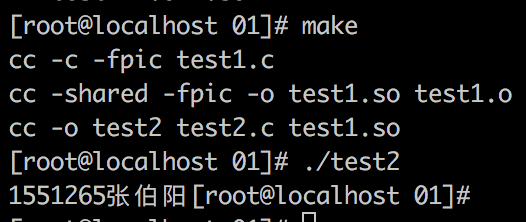
有可能连接的文件找不到



此时暂时给一个export

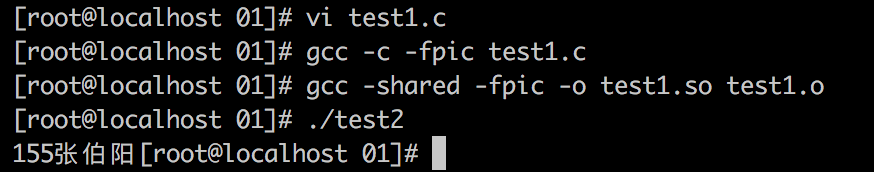


就可以成功运行出预期结果



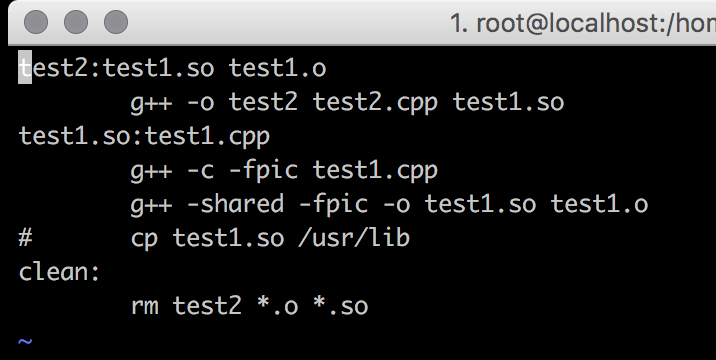
之后我改了test1.c文件 改了其输出 并使其重新输出.so文件覆盖之前的文件

运行test2可执行文件达到了预期效果

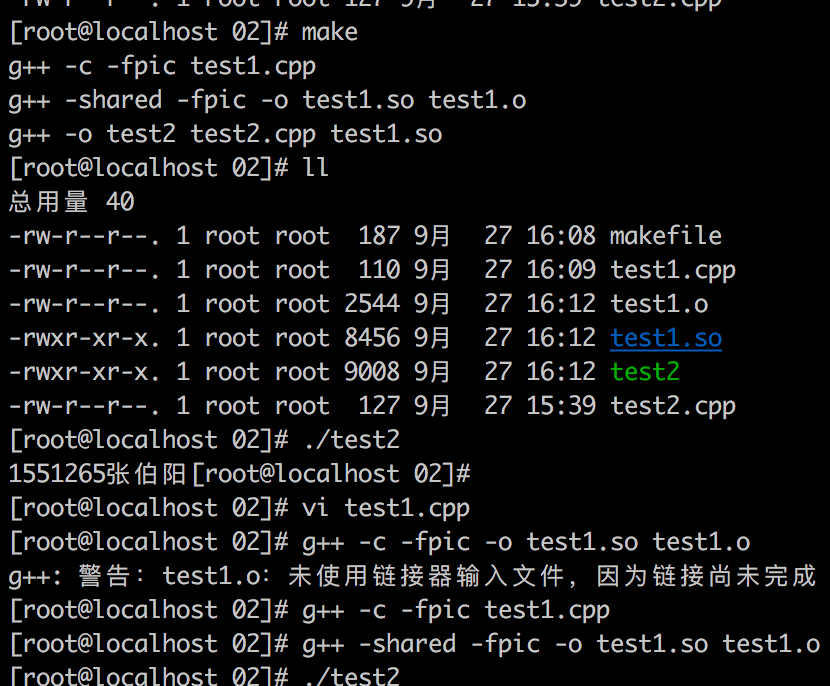


### 2.3

将上一题的makefile文件中的gcc改为g++ .c改成.cpp即可



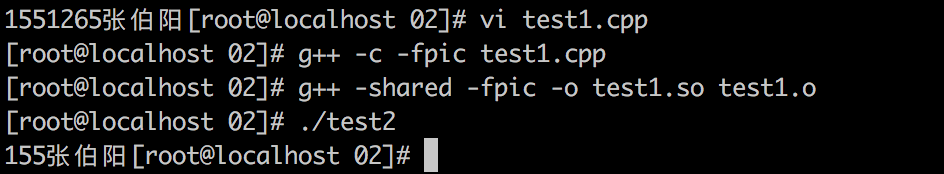
运行可执行文件test2 达到预期效果



修改test1.cpp的学号

重新编译并修改动态库文件

运行test2达到预期效果



### 2.4

还是之前的总makefile文件

