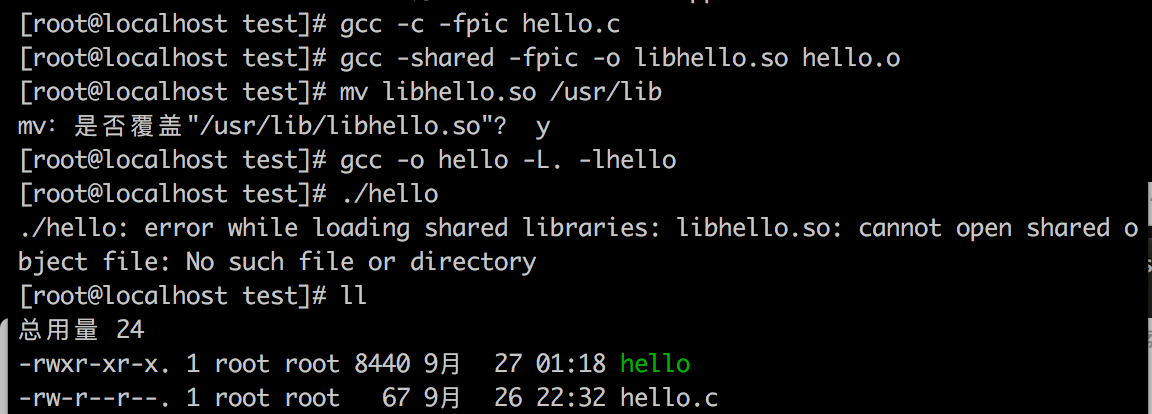
# 程序的静态与动态编译

## 1.Linux 下的动态编译

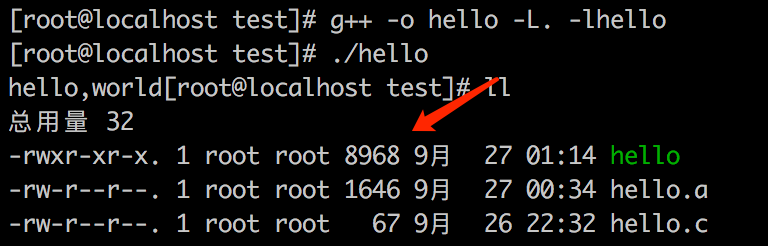
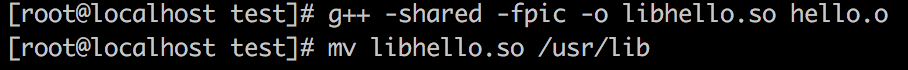
什么叫动态编译?

动态编译是某些[程式](https://zh.wikipedia.org/wiki/%E7%A8%8B%E5%BC%8F%E8%A8%AD%E8%A8%88)[语言](https://zh.wikipedia.org/wiki/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80)在执行时用来增进效能的方法。尽管这技术源于[Self](https://zh.wikipedia.org/wiki/Self)[[来源请求]](https://zh.wikipedia.org/wiki/Wikipedia:%E5%88%97%E6%98%8E%E6%9D%A5%E6%BA%90)，但使用此技术最为人所知的是[Java](https://zh.wikipedia.org/wiki/Java)。此技术可以做到一些只在执行时才能完成的最佳化。使用动态编译的[执行环境](https://zh.wikipedia.org/wiki/%E6%89%A7%E8%A1%8C%E7%8E%AF%E5%A2%83)一开始执行速度较慢，之后，完成大部分的编译和再编译后，会执行得比非动态编译程式快很多。因为初始化时的效能延迟，动态编译不适用于一些情况。在许多实作中，一些可以在[编译时期](https://zh.wikipedia.org/wiki/%E7%B7%A8%E8%AD%AF%E6%99%82%E6%9C%9F)做的最佳化被延到[执行时期](https://zh.wikipedia.org/wiki/%E5%9F%B7%E8%A1%8C%E6%99%82%E6%9C%9F)才编译，导致不必要的效能降低。[即时编译](https://zh.wikipedia.org/wiki/%E5%8D%B3%E6%99%82%E7%B7%A8%E8%AD%AF)是一种动态编译的形式。

给出printf("hello，world");程序的gcc动态编译命令，可执行文件字节数是多少?



给出 cout << "hello，world";程序的 c++/g++动态编译命令，可执行文件字节数是多少?



给出第一周作业中mysql\_demo.cpp的动态编译命令，可执行文件字节数是多少?



如何查找某个可执行文件所依赖的动态链接库?

ldd命令可以查看一个可执行程序依赖的共享库，

例如# ldd /bin/lnlibc.so.6

=> /lib/libc.so.6 (0×40021000)/lib/ld-linux.so.2

=> /lib/ld- linux.so.2 (0×40000000)

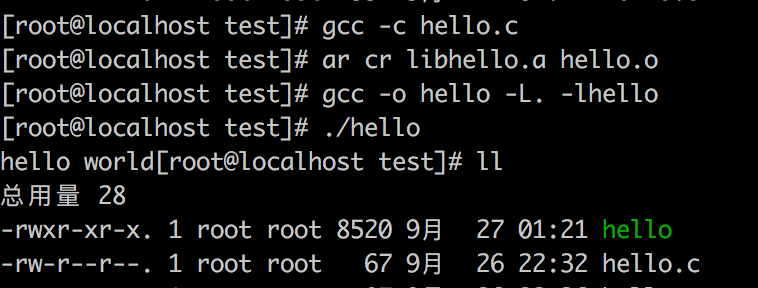
可以看到ln命令依赖于libc库和ld-linux库

## 2.Linux 下的 gcc 静态编译

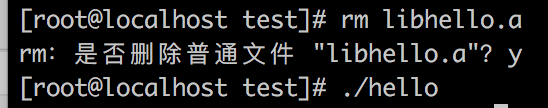
什么叫静态编译?

通常情况下，对函数库的链接是放在编译时期（compile time）完成的。所有相关的对象文件（object file）与牵涉到的函数库（library）被链接合成一个可执行文件 （executable file）。程序 在运行 时，与函数库再无瓜葛，因为所有需要的函数已拷贝到自己门下。所以这些函数库被成为静态库（static libaray），通常文件 名为“libxxx.a”的形式。

给出printf("hello，world");程序的gcc静态编译命令，可执行文件字节数是多少?

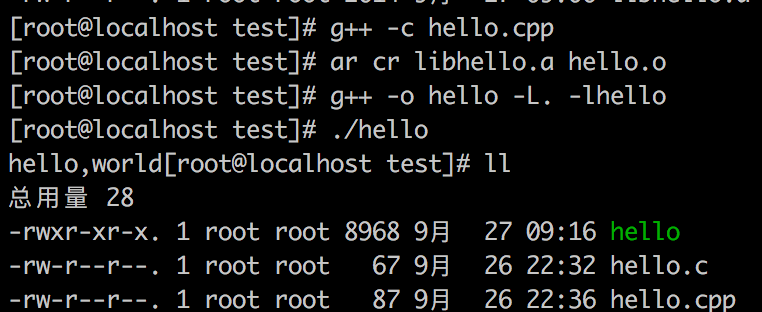


删除静态库仍然能够正常运行



## 3. Linux 下的 c++/g++静态编译

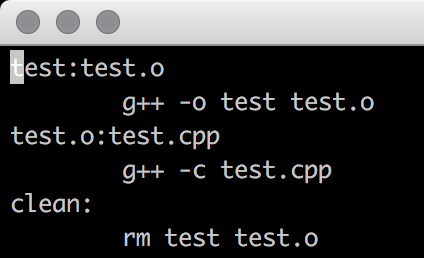
没装相关包好像也运行了



## 4. 按要求写出下列几种常用情况的静态编译测试样例

同makefile的文档 细节不再多写

只是用g++编译的地方稍微改一下



最外层还是那个总的makefile

