第一章Linux系统知识

1、开发环境？

Ubuntu+XSHELL+XFTP+VScode

XSHELL：Windows端远程操作Linux端工具。需要Linux服务器名称和IP地址（Linux：sudo apt install openssh-server. 默认开启服务。Ifconfig查看ens33IP）。

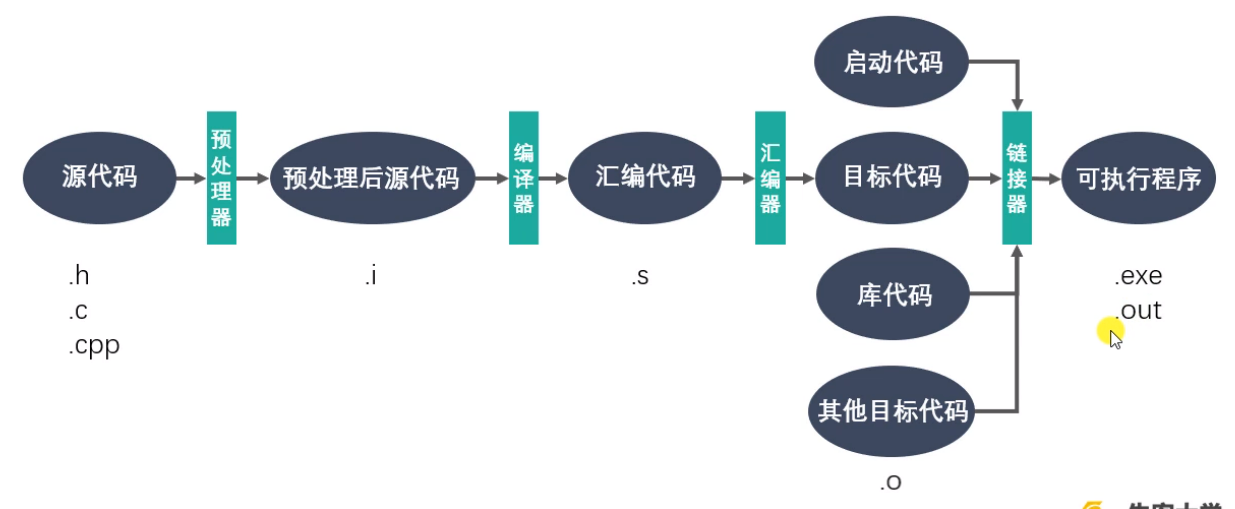
VScode：添加remote development插件，远程链接到Linux；远程资源扩展器中选择SSH，并选择配置文件\config，添加用户名和IP地址（建立远程连接）；管理→命令面板：C\C++编辑配置，在创建的.vscode目录下生成的c\_cpp\_properties.json文件中的includePath字段中添加“/usr/include”（用于解决显示找不到头文件）；在扩展中安装C\C++插件；创建和修改launch.json文件：①运行→创建文档→选择C++→选择默认；②运行→选择gdb启动→鼠标右键配置调试信息。修改miDebuggePath指定调试器，preLaucchTask编译代码生成的可执行文件；创建task.json文件，终端→配置任务→使用模板→Others，修改label和command：gcc，添加args参数。顺序：创建文件-Json-launch.json-task.json（创建的文件不能被包含在./vscode里面！）。

2、GCC（GNU Compiler Collection）？

安装命令：sudo apt install gcc g++

常用命令：mkdir program创建目录；touch test.c创建文件；vim test.c编辑文件；gcc test.c –o app；gcc test.c编译，rm test.c删除文件。

GCC工作流程：



gcc test.c –E –o test.i；gcc test.i –S –o test.s；gcc test.s –s –o test.o；

3、GCC和G++的区别？

.c文件：gcc当作c文件；g++当作c++。

.cpp文件：都是c++。

编译阶段：gcc不能自动和c++程序使用的库联接，所以用g++。

4、库文件？

库是一种不能单独运行的程序，可以提供给用户直接使用的变量函数或类。

静态库和动态库：静态库在程序链接阶段被复制到程序中；动态库在程序运行时由系统动态加载到内存中调用。

库的优点：代码保密；方便部署和开发。

5、静态库？

命名规则：Linux：libxxx.a Windows：libxxx.lib

lib：固定前缀

xxx：自定义库名字

.a：固定后缀

静态库制作（Linux）：

gcc获得 .o文件；将 .o文件打包，使用ar工具（archive）

ar rcs libxxx.a xxx.o xxx.o

r ：将文件插入备存文件中 c：建立备存文件 s：索引。

静态库使用（Linux）：

gcc test.c –o test –I(大写i) ./include/(头文件位置) –l calc(静态库的名字) -L ./lib(库位置)。

6、动态库？

命名规则：Linux：libxxx.so 可执行文件。Windows：libxxx.dll。

动态库制作：

gcc 得到 .o 文件，位置无关！gcc –c –fpic/-fPIC a.c b.c。

gcc –shared a.o b.o –o libcacl.so。

使用报错：找不到动态库文件？

解析：程序运行时，动态库会被加载到内存中，通过ldd（list dynamic dependencies）命令检查动态库依赖关系。 命令：ldd 可运行程序名称。如何定位动态库文件呢？当系统加载可执行代码时，能够知道依赖库的名字，但是还需要知道绝对路径。需要通过动态载入器来获取绝对路径。对于elf格式的可执行程序，由ld-linux.so来完成，先后搜索elf文件的DT\_RPATH段 → 环境变量 LD\_LIBRARY\_PATH → /etc/ld.so.cache文件列表 → /lib/, /usr/lib目录找到库文件后将其加载到内存。

解决：①在终端中临时配置：

env：查看环境变量（shell命令是通过环境变量去实现的）。

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH(获取原来的值):/home/lib（动态库目录地址）：添加环境变量（目的是把动态库文件添加到环境变量中，pwd显示当前目录）。

echo $LD\_LIBRARY\_PATH：打印LD\_L值的目录地址。

②永久配置：

用户级配置：cd进入home目录；vim .bashrc：export；. .bashrc（. ./.bashrc；source .bashrc）。

系统级配置：sudo vim /etc/profile；export；source /etc/profile。

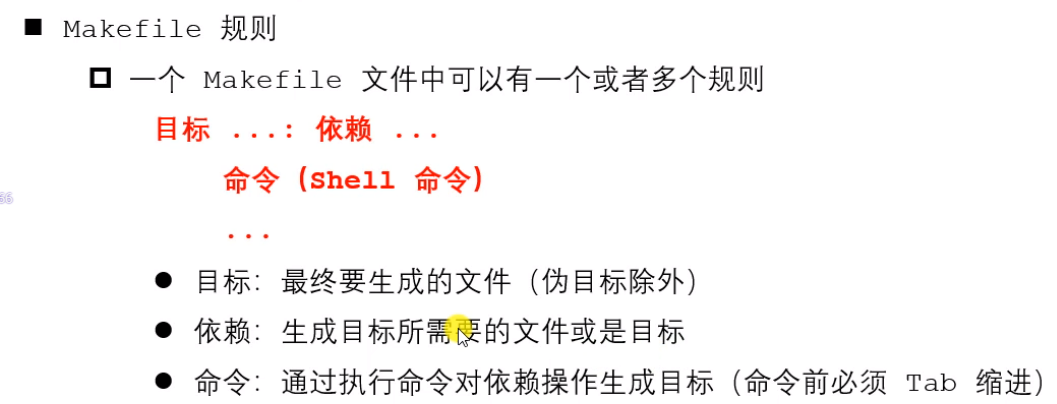
③修改/etc/ld.so.cache文件：sudo vim /etc/ld.so.config；复制路径；sudo ldconfig。

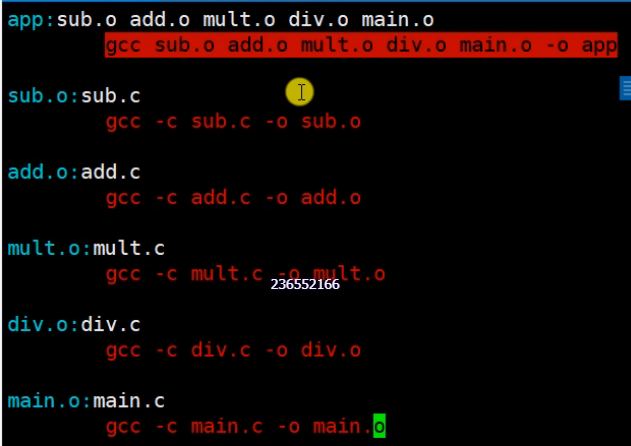
④修改/lib/,/usr/lib/目录：不建议：因为有系统文件。

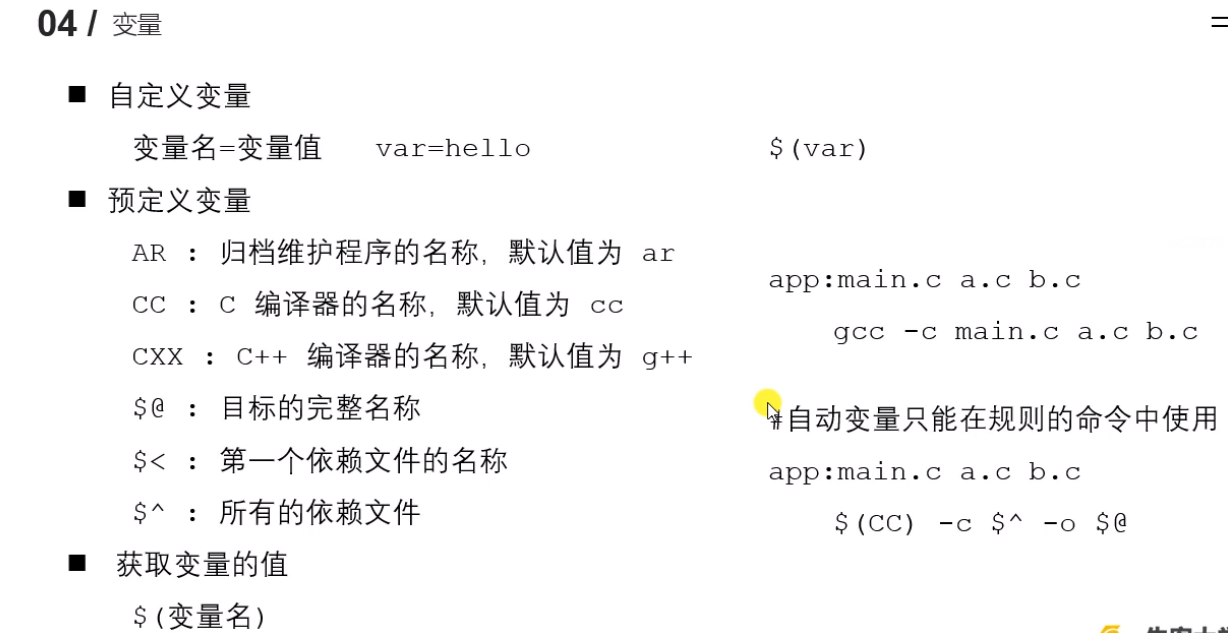
7、Makefile？

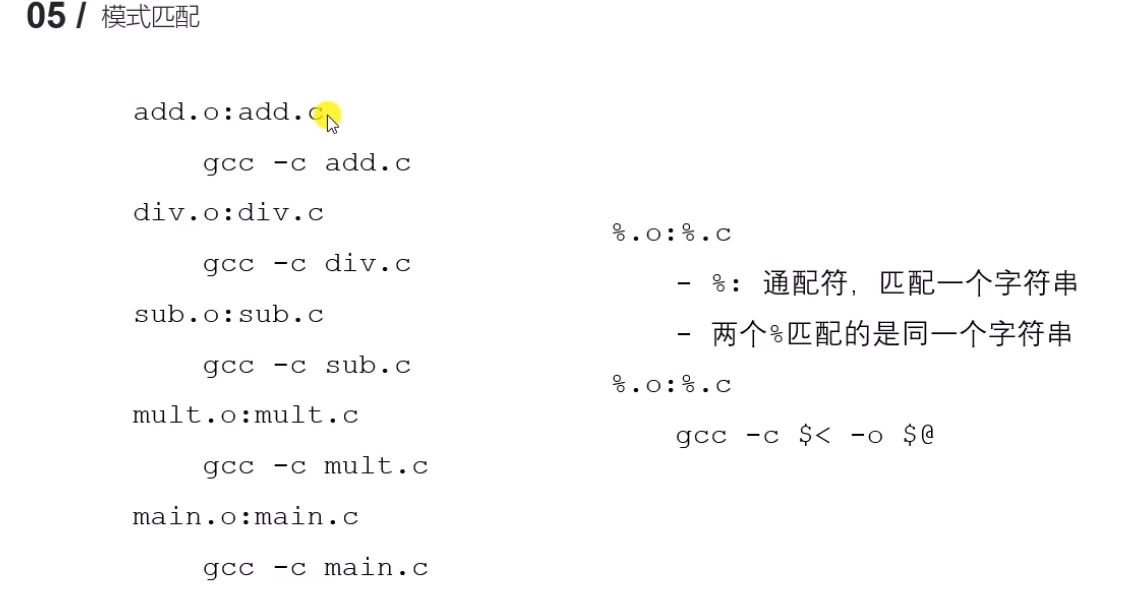
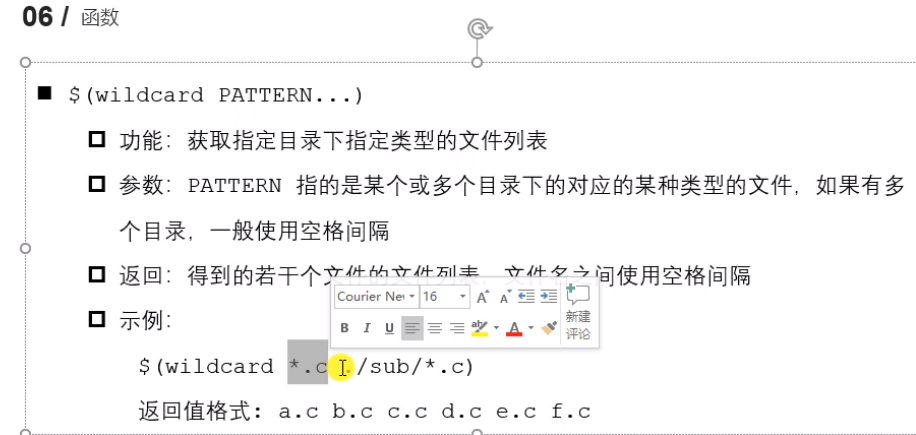
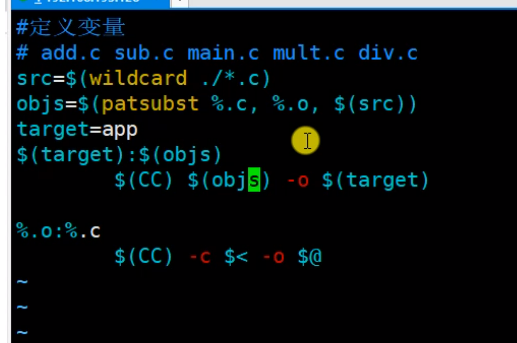
工程的源文件很多，按类型、功能、模块分别放在若干目录中。Makefile文件定义了一系列的规则来指定文件的编译顺序、是否重新编译等功能。好处就是自动化编译，只需要一个make命令，整个工程就可以完全自动化编译。

规则：

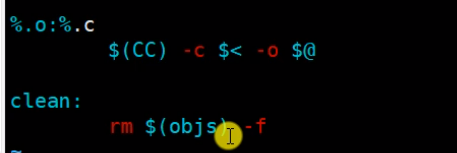






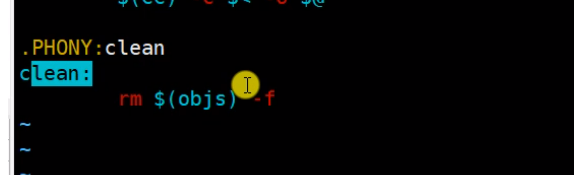
   

Makefile文件删除\*.o文件：



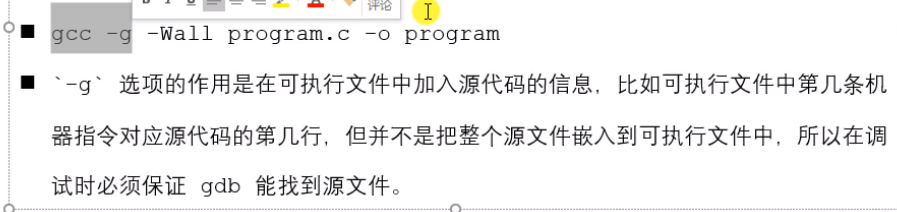
然后执行命令：make clean（直接执行make会提醒可执行程序时最新的，因为没有改写任何的.c文件）。-f：不提示强制删除。

可以将clean设置成伪目标，从而不会生成目标clean文件。

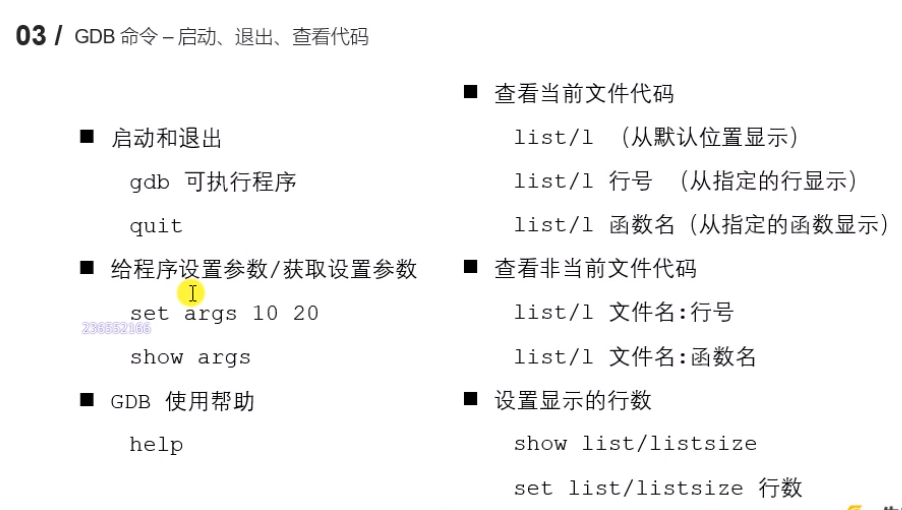
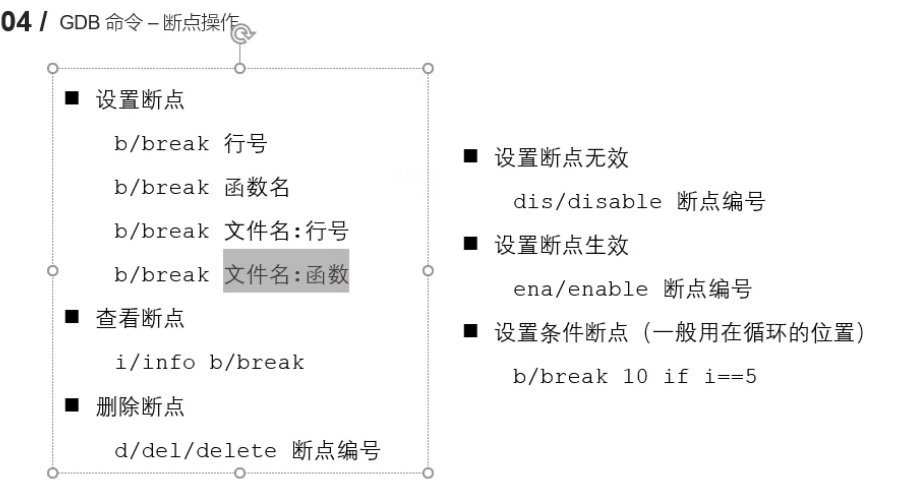


8、GDB？

GDB是GNU提供的调试工具。自定义运行程序；断点处停止。

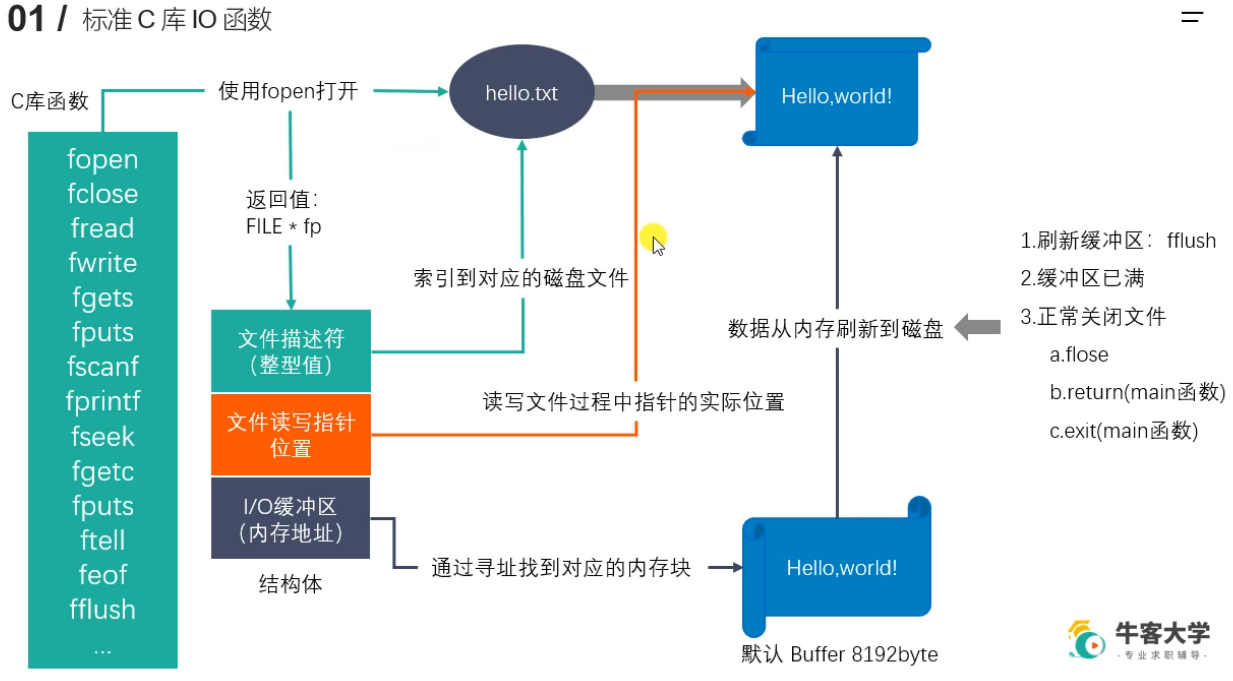
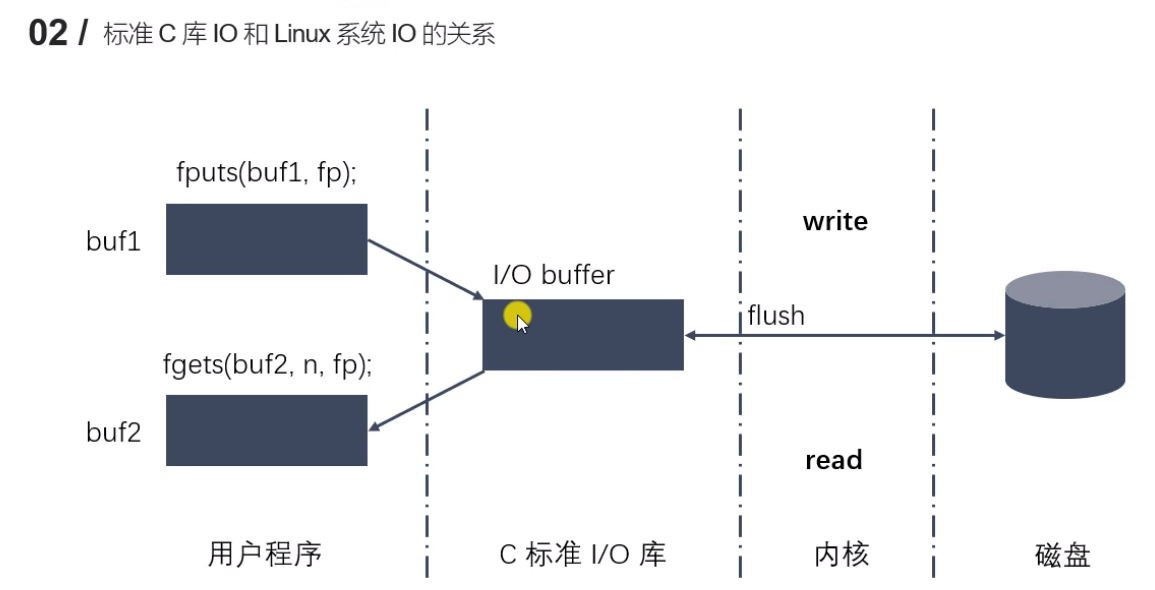


gcc test.c –g –o test。

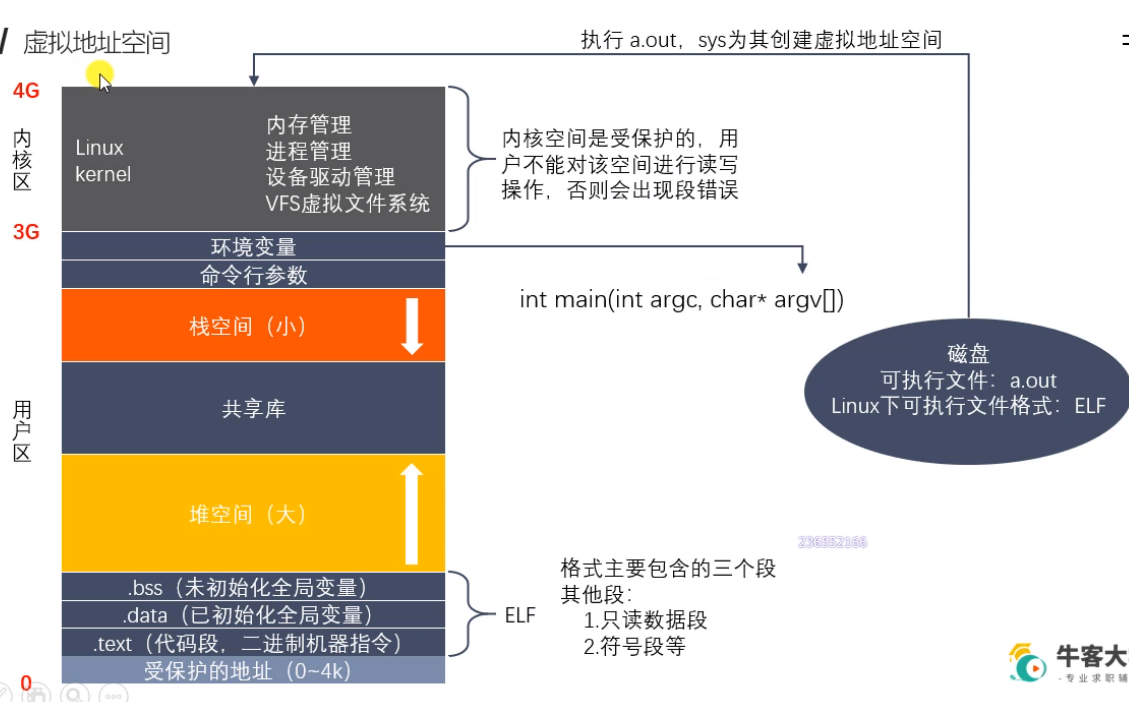
 



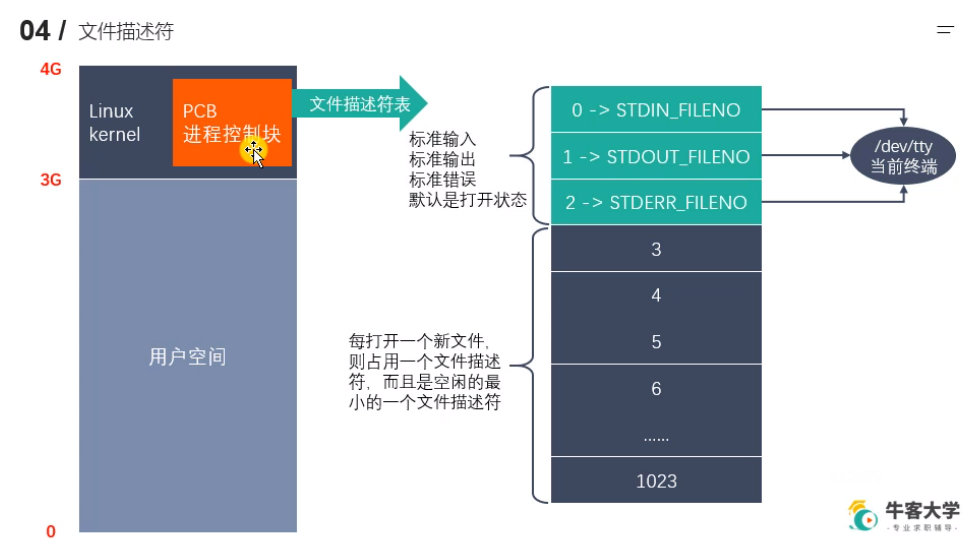
9、IO？

Linux的IO函数没有缓冲区，读写速度要比含有缓冲区的标准C库IO函数要慢，调用一次read函数就要读一次磁盘。在网络通信领域，要追求效率，所以使用Linux的IO函数去读写。在磁盘读写的操作，需要缓冲区提高读写效率，选择标准C库IO函数。 

10、虚拟地址空间？



11、文件描述符？

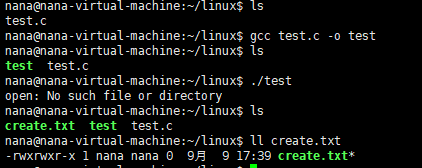


12、Linux IO函数？（man 2查看系统内核常调用函数）

open：(man 2 open)打开文件/创建新文件







详见Linux系统下Linux文件夹。

stat函数：

