# Linux基础知识

## Linux根目录下主要目录

/bin

/boot

/lib

/usr unix system resource 系统资源 头库文件放在此目录下

一般软件安装在/usr/local下

/sbin 可执行文件 ifconfig 在此目录下。管理员的可执行目录下

/proc

/etc 系统启动的配置

/etc/password 用户存放文件的位置 man 5 password 查看文件格式

/dev 一切皆文件

/home /user 用户家目录

## 操作目录相关

Cd 命令 进入目录

cd.. 回到上级目录

cd 回到家目录

cd- 回到上次访问目录

cd ~回到家目录

cd $HOME回到家目录

$ HOME环境变量代表家目录

Cd /home/max回到家目录

Ls命令 查看目录中的文件

–l 显示详情信息

–a 显示隐藏信息

–R递归显示 当前及子目录文件

–lrt 按照时间显示全部文件

Ls -lrt

通配符

\*任意多个字符

？任意一个字符

相对路径和绝对路径

绝对：/

相对：不是以/开头

Pwd 显示当前工作目录

Mkdir 创建目录 mkdir dirname1 name 2

–p 递归创建目录

Mkdir -p dirname1/ name 2

Rmdir

删除目录 不可删除非空目录 性能较弱（不常用）

Tree目录乘树状结构显示 如果不存在，则需要安装tree命令

Tree 目录

Tree aa

Which 查找命令的位置路径

Which 指令名

Which ls 查找ls的路径

Which cd不是任何一个目录，shell本身带着

## 文件操作相关

Touch 创建文件 对于存在的文件，不修改文件内容，只改访问时间,不存在，创建

Touch hello hello2创建多个文件

Rm 删除文件

rm hello.c

Rm –r 删除文件夹 rm –r test

Rm –rf\* 强制删除目录下所有文件，风险高，慎用

Cp 拷贝文件或者目录

Cp [option] srcpath decpath 拷贝文件

Decpath 为目录 将源文件拷贝到目标目录下

Decpath不是目录或者不存在，则在decpath平级目录中创建一个名为decpth的文件，并将src中的内容拷贝进来

Cp –r srcdir decdir 拷贝目录及其子目录

Decdir 存在，将目录放在decdir目录下

Decdir不存在，将源文件考到decdir 平级下名为decdir的目录中

Mv 移动文件或者目录 也需要注意最后的是否为一个目录，还是一个文件

Mv 移动文件

Mv移动目录

## 文件内容相关

Cat 查看文件内容

Cat filename 直接将文件内容显示在屏幕中

More 分屏显示内容 回车一行显示 空格一页显示（常用）

More filename

Less 分屏显示内容 可以上下翻看

Head 查看文件头信息 默认显示10行

Head –n 指定显示的行数

Tail 显示文件尾部 默认10行

Tail –n 指定显示的行数

Tail –f 追踪文件变化

## 统计信息相关

Tree 树状显示列表

Wc 统计文件信息

-l 统计行数

- w 统计单词数

-c 统计字节

Du 查看目录占用的空间

Df 显示磁盘空间信息

## 文件属性和用户组相关

Whoami 查看当前登录人

Chmod 改变文件权限

Chmod [u|g|o|a] +-w|x|r filename

Chmod 777 filename

注意：目录需要有可执行权限

Chown 改变用户或者分组（需要用管理员root去改变文件权限）

Chowm 用户名：组 文件名|目录

Chgro 组 文件名|目录（鸡肋）

Ln 软硬链接

Ln a a.hard 建立硬连接 可以创建多个 实质是指向磁盘中的同一个空间，修改其中一个文件，其他文件也变化

Ln –s a a.soft 相当于快捷方式

删除软硬链接 (在linux中硬连接数为0时，文件被系统清除)

Unlink 解除链接

Unlink a.hard

## 查找和索引

Find 查找文件 find dir [option] 内容

-name 按照名字查找

find ./ -name “\*.c” 查找当前目录中.c文件

-type 按照文件类型查找文件

Find ./ -type f 查找当前目录中的普通文件（注意：普通文件用f，不用-）

Find ./ -type d查找当前目录中的目录（注意：目录用d）

-size 按照文件大小查找文件

Find ./ -size +1M 查找当前目录中大于1M的文件

Find ./ -size +1M –size -5M 查找当前目录中大于1M小于5M的文件

不写+-的话相当于等于，大约，不精确

Find ./ -maxdepth 1 –size +1M 查找当前目录中深度为1 的文件中大小大于1M的文件

Find ./ -maxdepth 1 –size +1M –exec ls {} \; 查找当前目录中深度为1 的文件中大小大于1M的文件，并把他们以列表形式显示出来

Find ./ -maxdepth 1 –size +1M –ok ls {} \; 安全模式

Find ./ -maxdepth 1 –size +1M |xargs ls –l 查找当前目录中深度为1 的文件中大小大于1M的文件，并把他们以列表形式显示出来

(find 和xargs是好伴侣)将find查找的结果，xargs按照块传递给后面的指令

Xargs 将前一个目录分成小块输入给后面的指令

Grep命令按照文件内容过滤

Grep [option] 内容 dir|filename

-r 递归查找

-n 显示行号

-v +内容 表示排除内容

-m 内容 文件或者目录

Find ./ -name “\*.c”| Grep 内容。通过管道将find和grep结合起来使用。

与find tail 等命令结合使用

## 压缩管理相关

Zip 生成.zip文件

Zip [option] 压缩包名字 文件1，文件2

-r 递归压缩

Unzip

Unzip 文件名

Tar

Tar [option] 压缩后文件名 压缩文件1 压缩文件2

-c 压缩

-v 显示信息

-f 指定压缩包名字

-z 按照gz格式压缩

- x解压缩

Tar –zcvf a.tar.gz a

Tar –zxvf a.tar.gz

Rar压缩包

Rar a –r 压缩包名（无后缀） 原材料

a 压缩

x 解压缩

-r 递归压缩子目录

Rar a –r b b

Rar x b.rar

## 软件的安装和卸载

apt-get 安装卸载软件

sudo apt-get update 更新源 更新/etc/apt/source.list文件

sudo apt-get install tree 安装tree命令

sudo apt-get remove tree 移除tree命令

sudo apt-get clean 清理包

源码安装

1. 解压源码包
2. Cd dir
3. ./config 检测文件是否缺失，创建makefile,检测编译环境
4. Make 编译源码，生产可执行程序
5. Sudo make install 把库和可执行程序安装到系统目录下
6. Sudo make distclean 删除和卸载软件

## 用户管理

创建用户

Sudo useradd –s /bin/bash -g itcast -d /home/itcast –m itcast

-s 指定shell

-g 指定组

-d 指定家目录

-m 家目录不存在，自动创建

设置用户组

Sudo groupadd itcast

设置密码

Sudo password 新密码

切换用户

Su itcast 切换用户

Su –itcast切换到itcast用户，同时带有环境变量

Root用户

Sudo su

删除用户

Sudo userdel itcast 删除用户

Sudo userdel –r itcast 删除用户的同时也删除该用户的家目录

删除组

Sudo grouprdel itcast 删除用户

## 其他常用指令

Echo

输出变量或者字符串

Date

查看日期

Alias

给指令起别名

Man命令

Man 章节 命令名

Man ls

## 关机相关（需要root权限）

立即关机 shutdown -h now

Init 0 关机

Power off 关机

Reboot 重启

## ---------------------------------------------------------

# Vim

## Vim命令模式下的使用

Vim 是vi的增强

Vimtutor 查看系统自带的vim教程

光标移动

使用hjkl四个键控制上下左右的移动

k

h lmax

j

0 行首

$行尾

gg 文件头

G 文件尾部

nG 指定光标到n行

vim删除内容

x 删除光标所在的字母

X 删除光标前的字母

dw 删除单词（从光标处开始删除）

d0 删除光标到行首

d$ (D) 删除光标到行尾

dd 删除光标所在行（内容放在剪切板中）

ndd 删除光标行开始n行（内容放在剪切板中）

u 撤销上部操作

ctrl +r 反撤销

vim复制粘贴

yy 复制一行内容

dd 剪切一行内容

nyy 复制多行内容（从光标所在行开始）

p/P 粘贴（注意：p是在光标下一行开始，P在光标所在行开始）

r 替换 输入r后再输入一个字母，可以将光标所在位置的字母替换

## vim可视模式

v进入可视模式

移动光标选中内容

按y复制内容

移动光标到目的地

按p/P粘贴内容

Vim查找

/str n/N遍历 n往下遍历 N往上遍历

?str N/n遍历 N往下遍历 n往上遍历

光标移动到查找的字符串上，按#可以查找光标选中的内容，按n/N上下移动

Vim调整文件格式

gg=G 整体调整格式

>>向右缩进tab

<<向左缩进tab

n>>当前行开始n行向右缩进一个tab

n<<当前行开始n行向左缩进一个tab

vim中man查看函数实现

光标放在函数上，按K进入man 1页

按n+K进入man n 页

一般的库函数都在2、3页

## Vim命令模式和编辑模式的转换

小I 在光标前插入

小a在光标后插入kkk

I在行首插入

A在行尾插入

小o 在光标上一行开始插入模式

O在光标下一行开始插入模式

小s删除当前光标所在的字母变成插入模式

S 删除当前行，进行插入模式

## Vim末行模式

末行模式中在屏幕最下方出现NORMAL字样

可以执行命令

:!ls –l

代码替换

一般对于函数名、变量等批量替换操作

:s/src/des 只替换光标所在行的第一个匹配的src为des

:s/src/des/g 只替换光标所在行的所有匹配的src为des

:%s/src/des 只替换所有行的第一个匹配的src为des

:%s/src/des/g替换文件内容中所有匹配的src为des

保存退出

:wq保存退出

:w 只保存

:q 只退出，文件不能修改

:q! 强制退出

:x 保存退出

ZZ 保存退出（在命令模式下）

分屏

:sp filename横向分屏

:vsp filename竖向分屏

:qall 全退不保存

:wqall全保存

多屏切换：ctrl +w+w

Vim安装插件

自行百度学习  
Vim配置

Cd



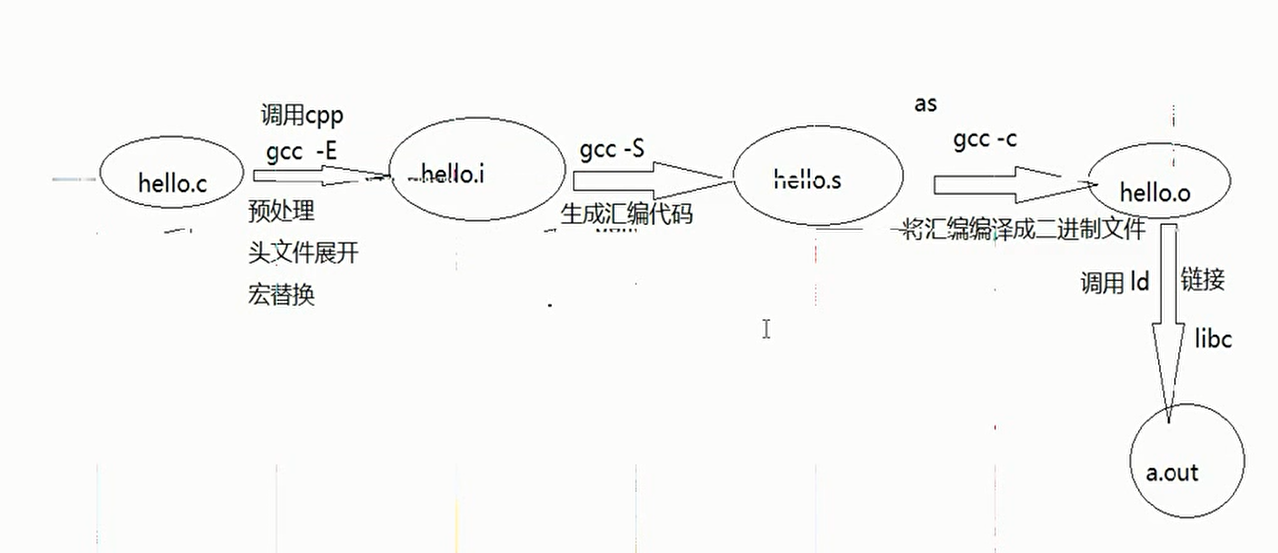
指令中使用vi的快捷键

设置 ~/.bashrc

添加 set -o vi ------可以直接使用vi的各种快捷键

# GCC

## Gcc的编译流程



## Gcc 参数相关

-I 指令文件中包含的头文件 gcc a.c –I包含的头文件的目录路径

-o 指定编译后的文件的名称 gcc a.c –I filename –o a2.c

-O 优化选项 ，1-3越高优先级越高，提高代码运行效率

-D 指定宏编译 DEBUG

-lstdc++ 编译c++代码

-g 用于gbd调试不加此项不能gdb调试

-w 显示警告 all 显示更多 -wall

-c 编译成二进制文件

-E 输出到标准输出，宏替换，头文件展开

-S 编译成汇编

-L 包含库路径

-l指定库名

编译C++文件的方式

G++ a.cpp

Gcc –lstdc a.pp –o b

## 库的制作

### 静态库

静态库的优缺点：

优点：

对运行环境的依赖性较小,具有较好的兼容性，执行快，发布应用时不需要发布库

缺点:

生成的程序比较大，需要更多的系统资源，在装入内存时会消耗更多的时间

库函数有了更新，必须重新编译应用程序

静态库的命名：libxxx.a

静态库制作

生成.o文件 gcc –c a.cpp –I include –o app.o

再生成libxxx.a ar crvs libxxx.a app.o

用nm命令查看静态库内容

Nm xxx(libxxx.a中的xxx)

静态库的发布

发布库文件和头文件

### 动态库

动态库的优缺点：

优点：

在需要的时候才会调入对应的资源函数

简化程序的升级；有着较小的程序体积

实现进程之间的资源共享（避免重复拷贝）

缺点：

依赖动态库，不能独立运行

动态库依赖版本问题严重

动态库命名

Libxxx.so

动态库制作

编译与位置无关的代码，生成.o文件，关键参数 –fPIC gcc –fPIC -c \*.c

将.o文件打包：关键参数 –shared gcc –shared libxxx.so \*.o

以上两步可以同时完成：gcc –fPIC –shared –o libxxx.so \*.o

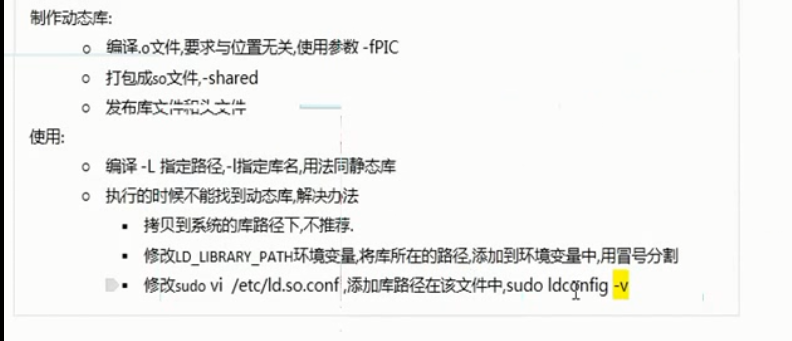
将库与头文件一起发布

动态库发布

将头文件和库文件放在同一目录中

编译使用了自定义动态库的文件

Gcc main.c –o newapp –I include/ -L lib/ -l xxx



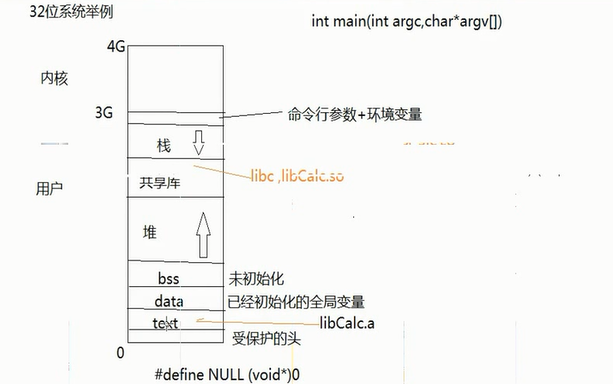
### 解决不能加载动态库的问题

注意在编译和运行过程中，都需要找到动态库，如果想在运行过程中找到动态库,这时需要对动态库进行配置，需要将动态库放在系统或者用户的lib文件夹下。/lib /usr/lib（不推荐）

1. 拷贝到/lib下：复制、硬连接方式。（不推荐）
2. 将库路径增加到环境变量中LD\_LIBRARY\_PATH（不推荐）
3. 配置/etc/ld.so.conf文件，增加库路径（/home/itcatt/cmd/calc/lib路径，执行sudo ldconfig –v）

式

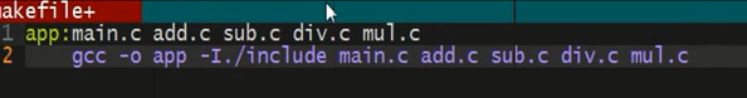
### 程序在内存中的存放顺序



# Makefile

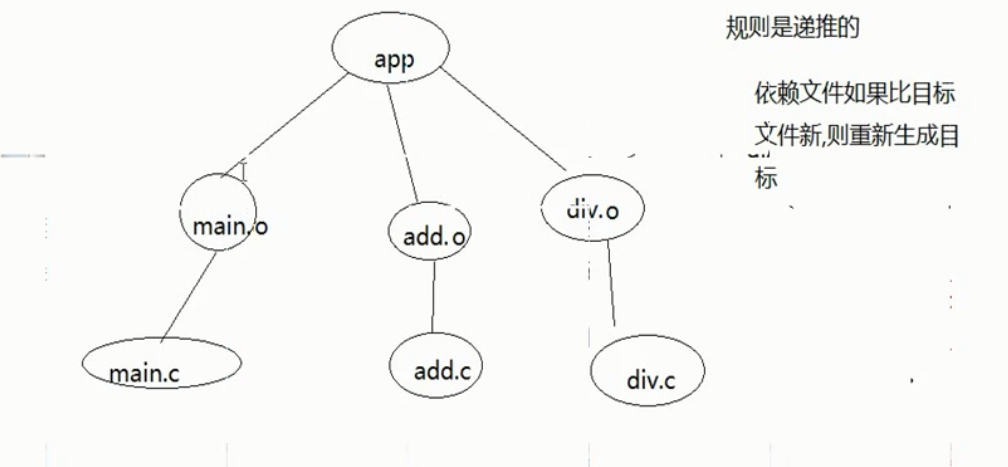


Makefile文件编写案例—第一版

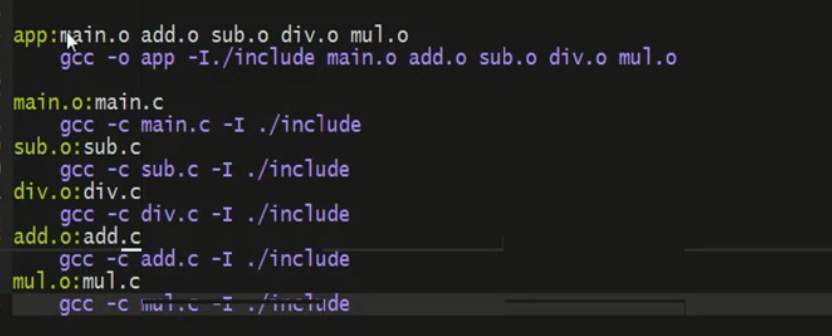


上图中存在的不足：更改一个文件的源码，所有的文件都重新编译

Makefile 的执行流程分析



Makefile文件编写案例—第二版



Makefile文件编写案例—第三版

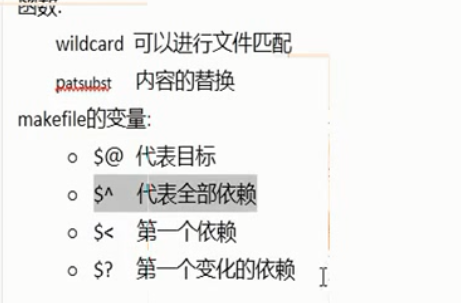
Makefile中可以定义变量

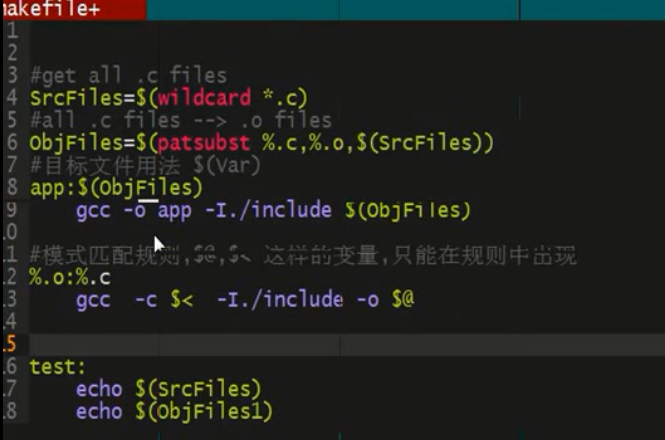




Makefile文件编写案例—第四版

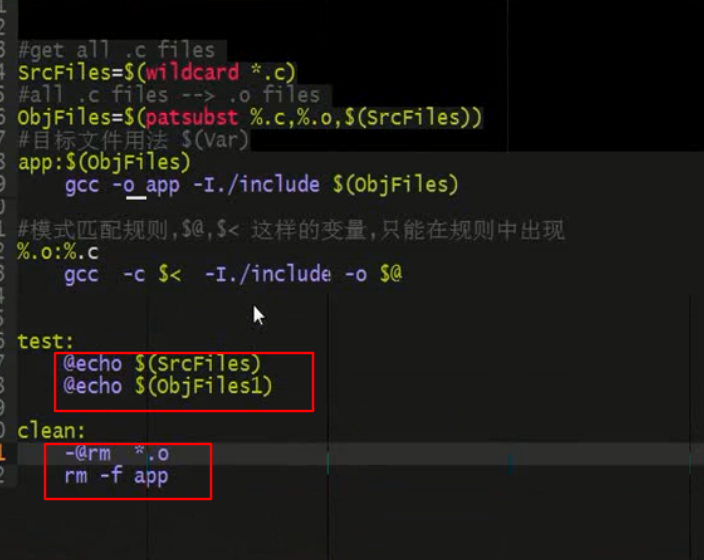
通过变量和函数得方式写makefile

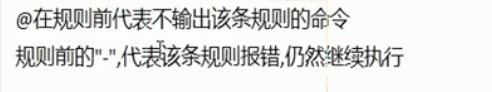




Makefile文件编写案例—第五版

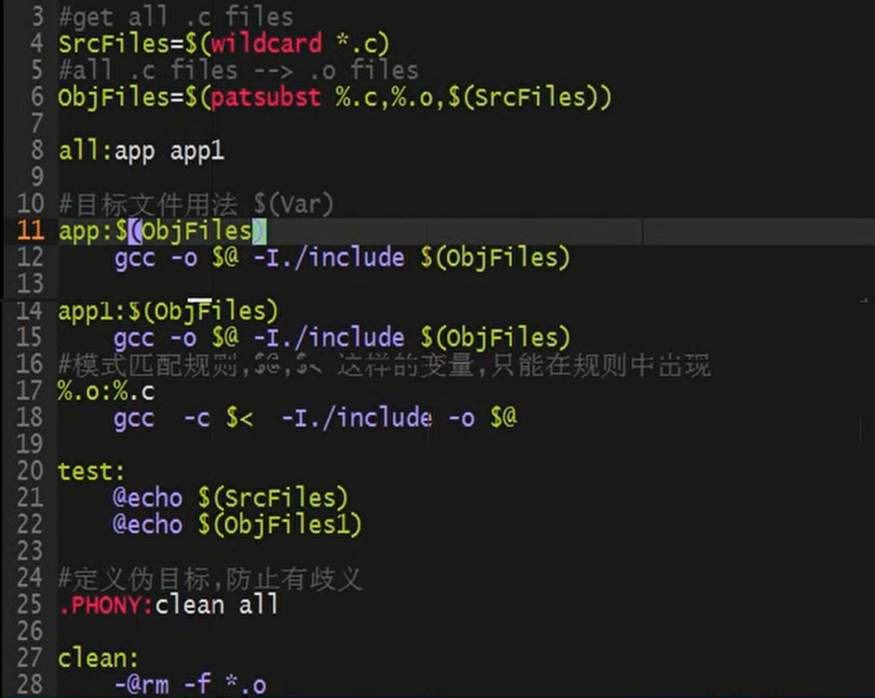
在makefile文件中定义clean目标，同时通过@-控制输出

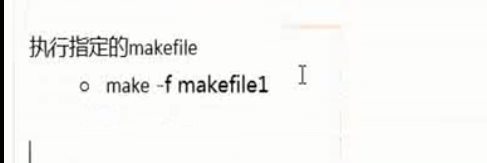




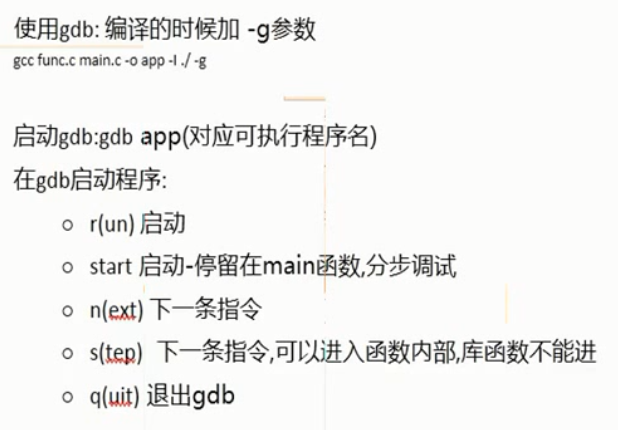
Makefile文件编写案例—第六版

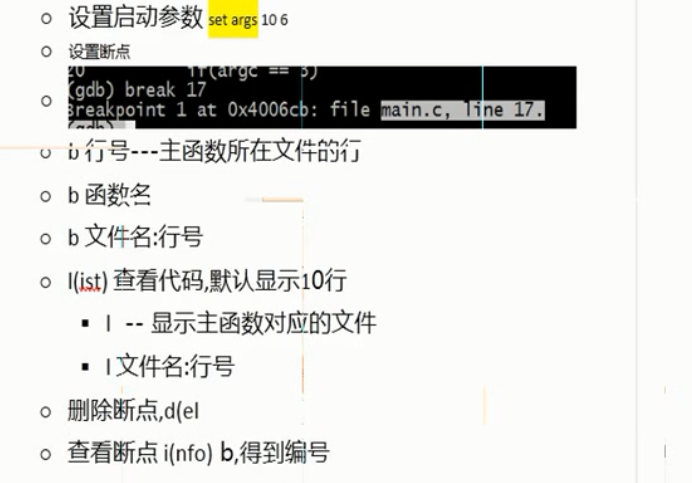
定义伪目标，防止有歧义

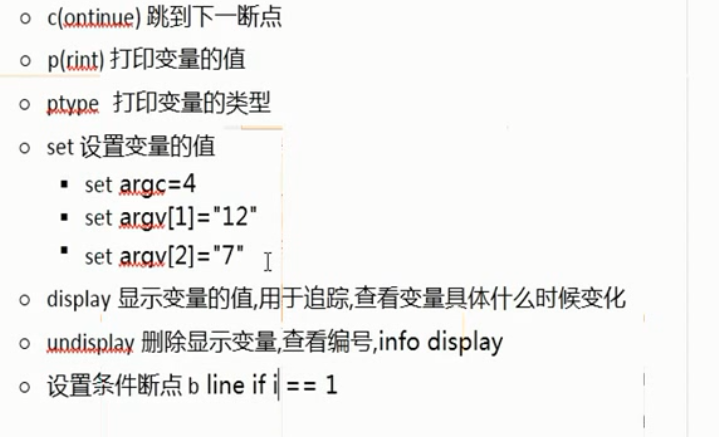




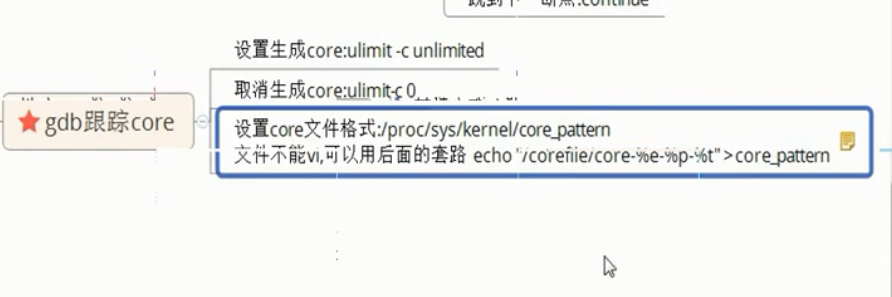
# GDB调试



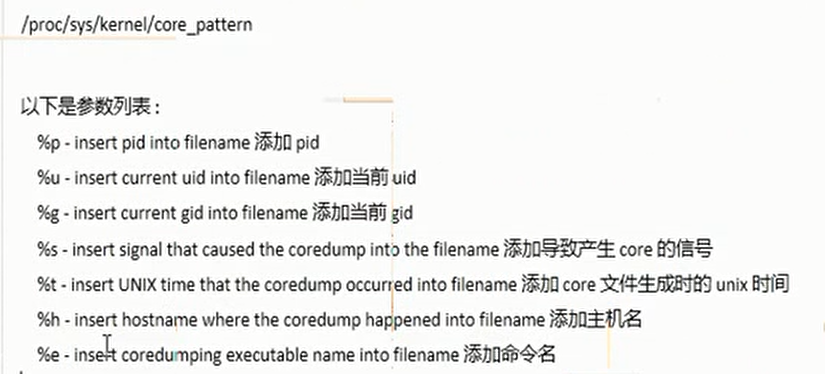




# GDB调试core文件



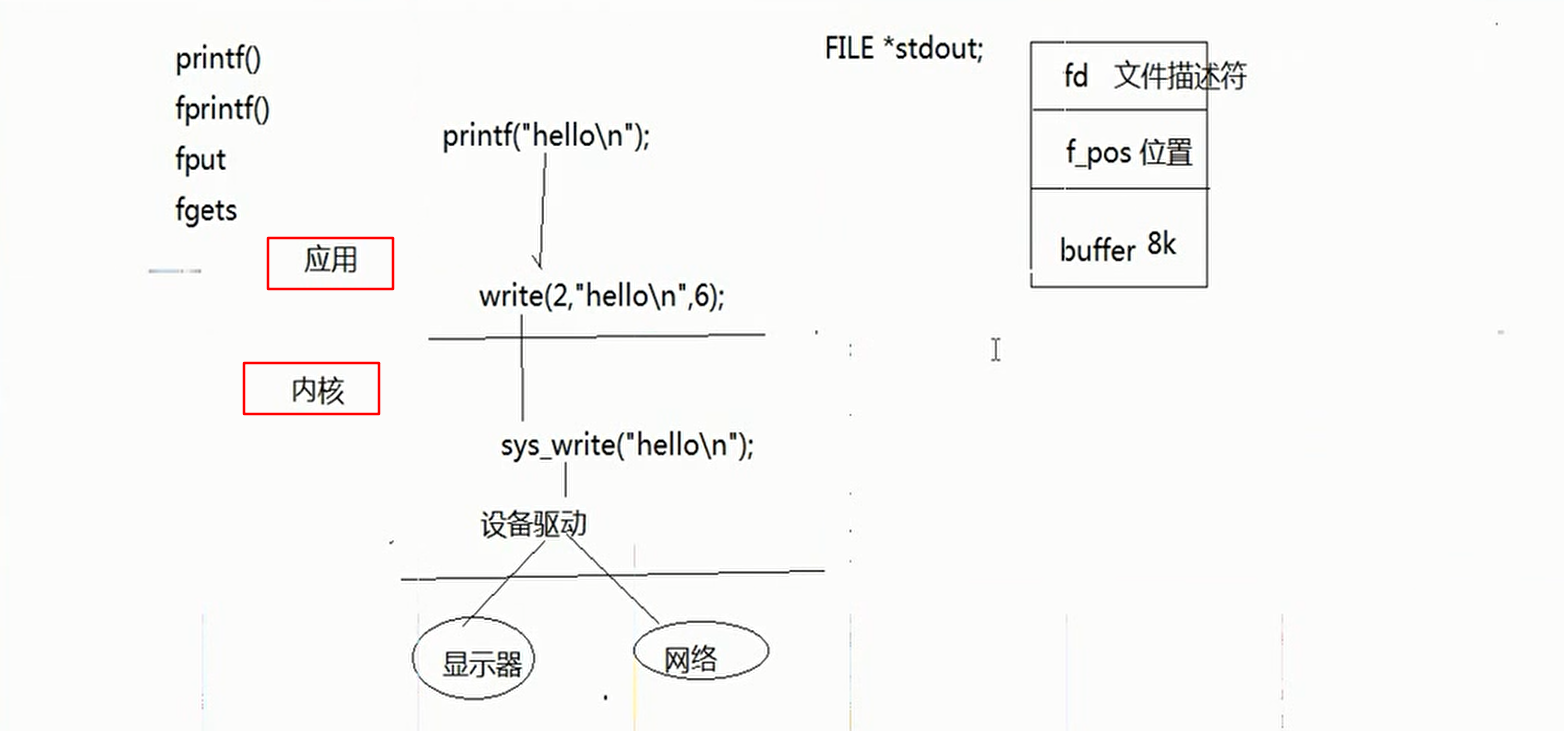
Core文件设置



（注意：需要切换到root 下，通过echo 指令写入）

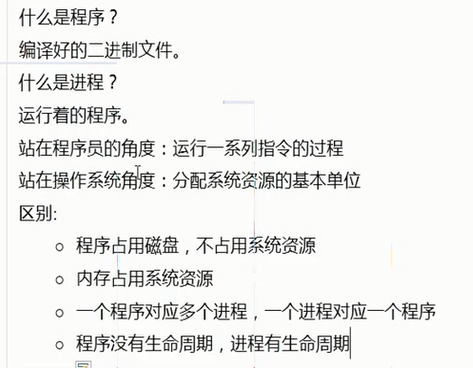
# 系统API与库函数

系统函数调用流程

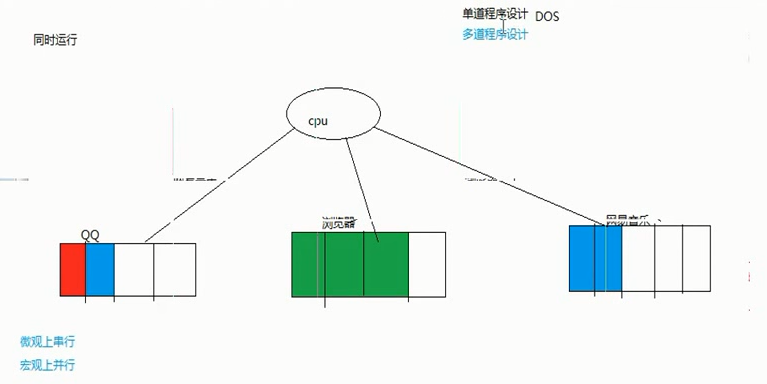


# 进程

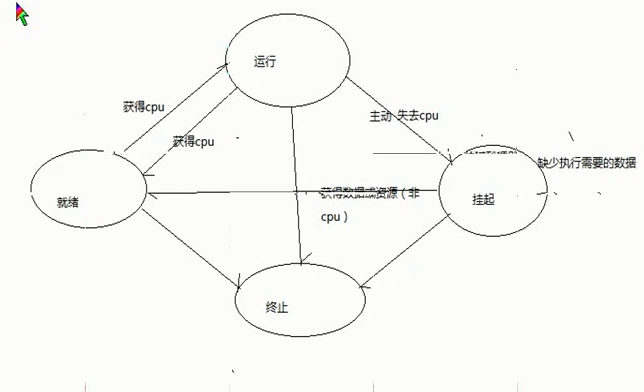
## 相关概念



## 单道和多道



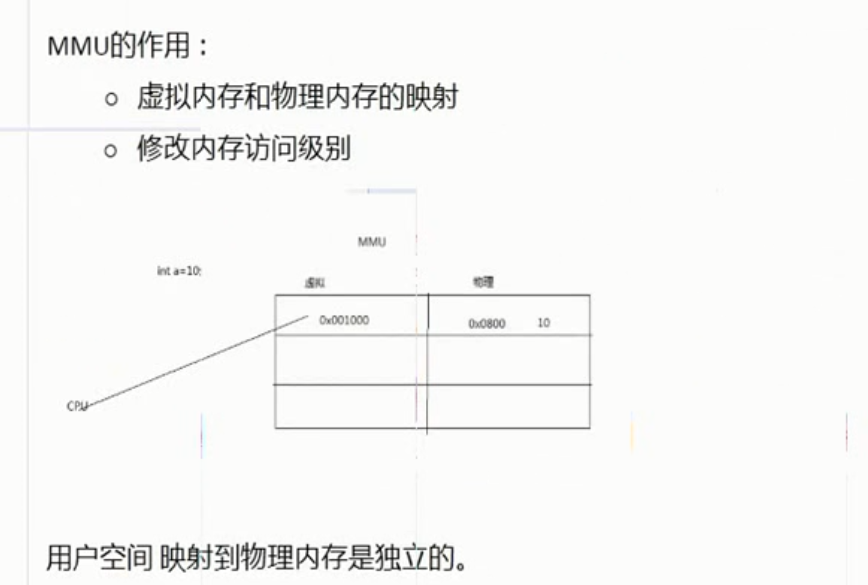
## 进程状态



## CPU和MMU(内存管理单元)

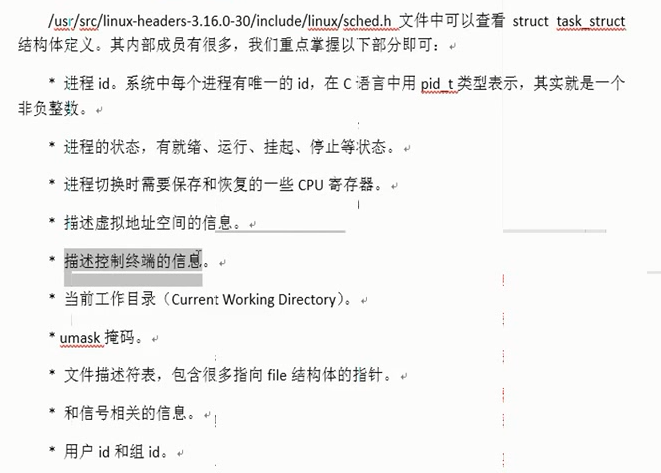






## PCB（进程控制块）

在linux的核心代码中PCB是一个结构体



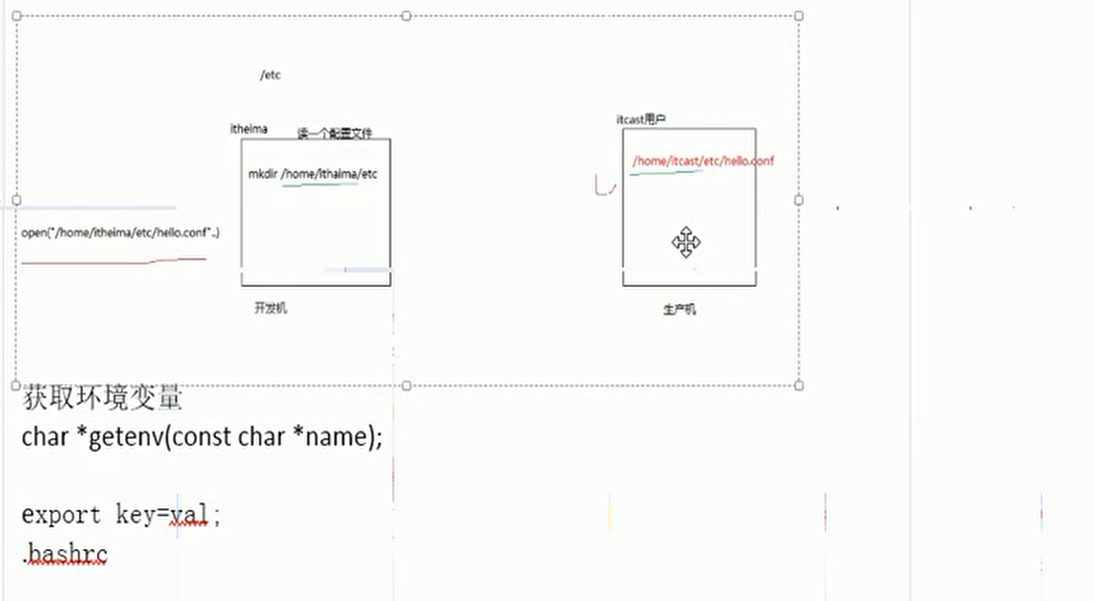


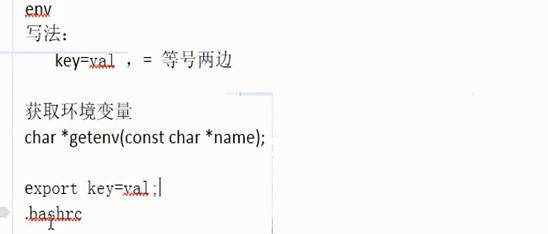
## 环境变量

Linux下常见环境变量

HOME

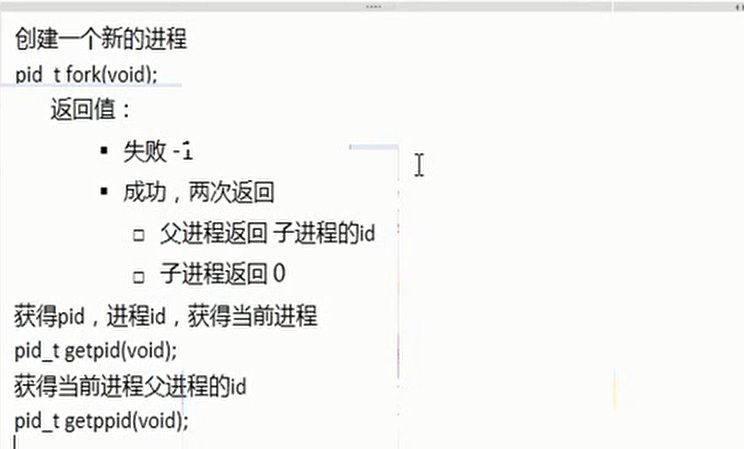
SHELL PATH



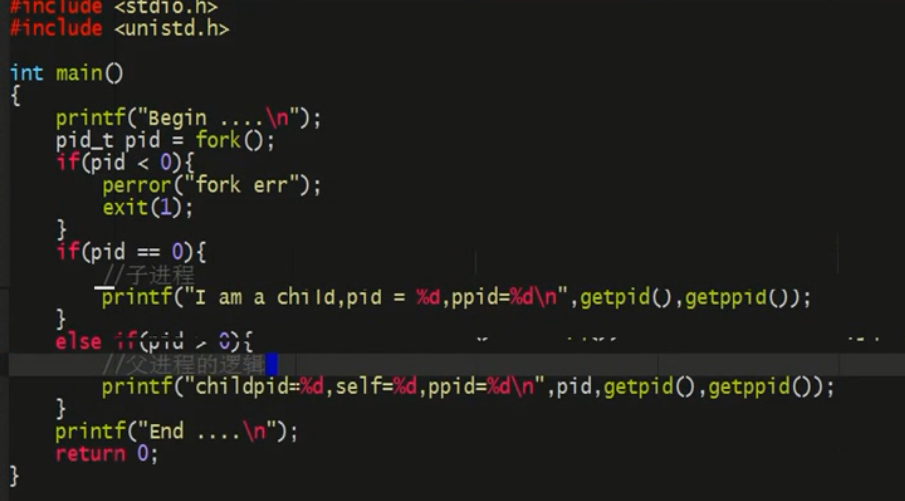


## 进程API和控制

### 1 创建子进程



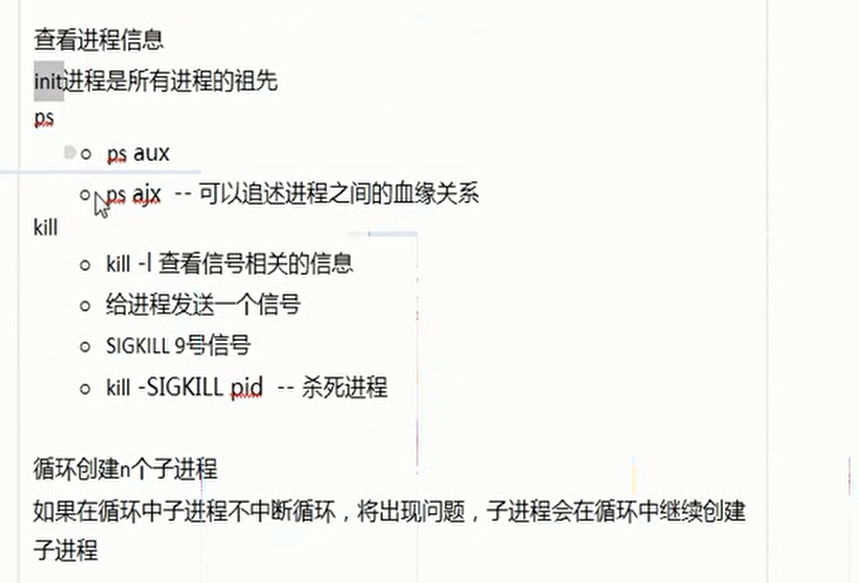
Fork(),返回子进程id

上诉代码执行后会发现父进程先死，子进程变成孤儿进程。

可以通过让父进程sleep的方式，解决

可以通过上诉的方式判断是父还是子进程

### 2 查看和杀死进程



Ps aux 查看当前用户下的进程

Ps ajx 查看当前用户下的用户组进程信息（血缘关系）

Kill -9 pid 杀死进程

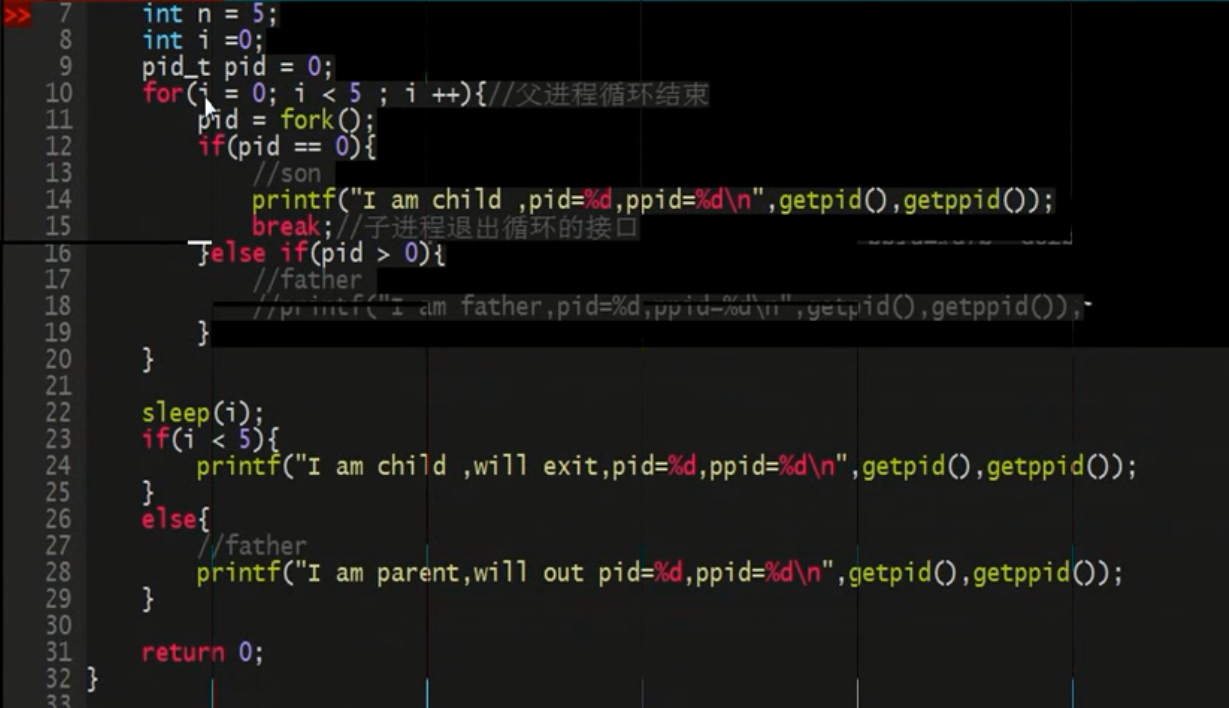
### 3 创建多个子进程

同时创建多个进程

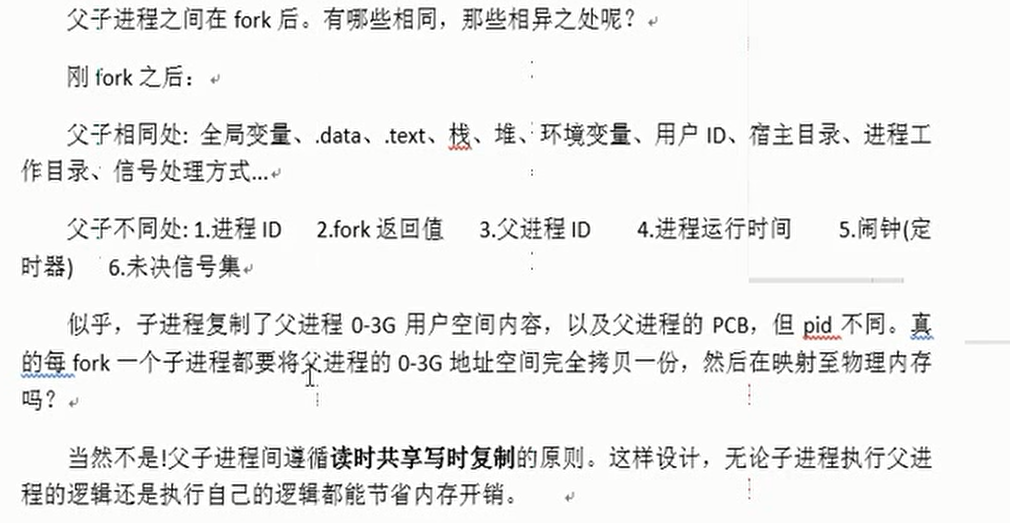


为了防止在循环时每个进程都创建子进程，在子进程执行完逻辑后，退出循环。

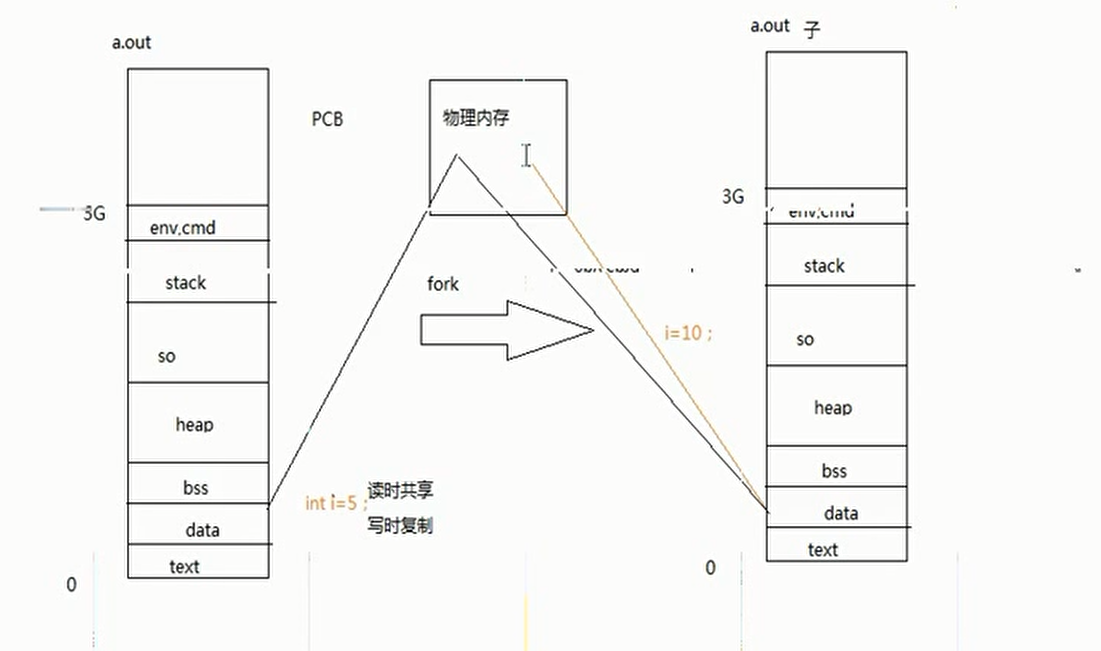
创建多个进程，按照每个进程的创建顺序退出进程

通过让创建的子进程sleep 的方式让进程按照创建顺序退出。

### 4父子进程异同



读时共享，写时复制



### 5 代码在内存中的存储

