

2024 操作系统 Lab0 串讲

2024 操作系统助教组

Linux 基本操作命令

- ls 命令用于列出（list）指定路径下的文件。

```
ls
```

```
用法:ls [选项]... [文件]...
```

```
选项（常用）：
```

```
-a 不隐藏任何以. 开始的项目
```

```
-l 每行只列出一个文件
```

- touch 命令用于创建一个新的文件。

```
touch
```

```
用法:touch [选项]... [文件名]...
```

- mkdir 命令用于创建目录（make directory）。

```
mkdir
```

```
用法:mkdir [选项]... 目录...
```

- cd 命令用于进入指定目录（change directory）。

```
cd
```

```
用法:cd [选项]... 目录...
```

Linux 基本操作命令

- rmdir 删除空的目录（remove directory）。

```
rmdir  
用法:rmdir [选项]... 目录...
```

- rm 命令用于删除（remove）文件，也可以将某个目录及其下属所有文件及其子目录全部删除。

```
rm  
用法:rm [选项]... 文件...  
选项（常用）：  
-r 递归删除目录及其内容，如果不加这个命令，删除一个有内容的文件夹会提示不能删。  
-f 强制删除。忽略不存在的文件，不提示确认
```

- cp 与 mv: 复制（copy）与移动（move）

```
cp  
用法:cp [选项]... 源文件... 目录  
选项（常用）：  
-r 递归复制目录及其子目录内的所有内容
```

```
mv  
用法:mv [选项]... 源文件... 目录
```

Linux 基本操作命令

- cat 命令用于拼接（concatenate）文件并输出到标准输出。也可用于查看单个文件内容。

```
cat  
用法: cat [选项]... [文件]...
```

- head 命令用于输出文件首部内容

```
head  
用法: head [选项]... [文件]...  
选项（常用）：  
-n <n> 显示前 <n> 行内容  
-c <n> 显示前 <n> 个字节内容
```

- tail 命令用于输出文件尾部内容

```
tail  
用法: tail [选项]... [文件]...  
选项（常用）：  
-n <n> 显示末尾 <n> 行内容  
-c <n> 显示末尾 <n> 个字节内容  
-f 当文件增长时，输出后续添加的数据（适用于文件不断变化的情况，如日志文件）
```

Linux 基本操作命令

- ps 命令用于显示当前进程状态（process status）

```
ps
用法:ps [选项]...
选项（常用）：
-e 或 -A 显示所有进程
-f 显示全部信息
```

- kill 命令用于向进程发送信号（并非仅有杀死进程的功能）

```
kill
用法:kill [选项] [pid]...
选项（常用）：
-s <s> 或 -<s> 指定要发送的信息（-9 为 SIGKILL）
```

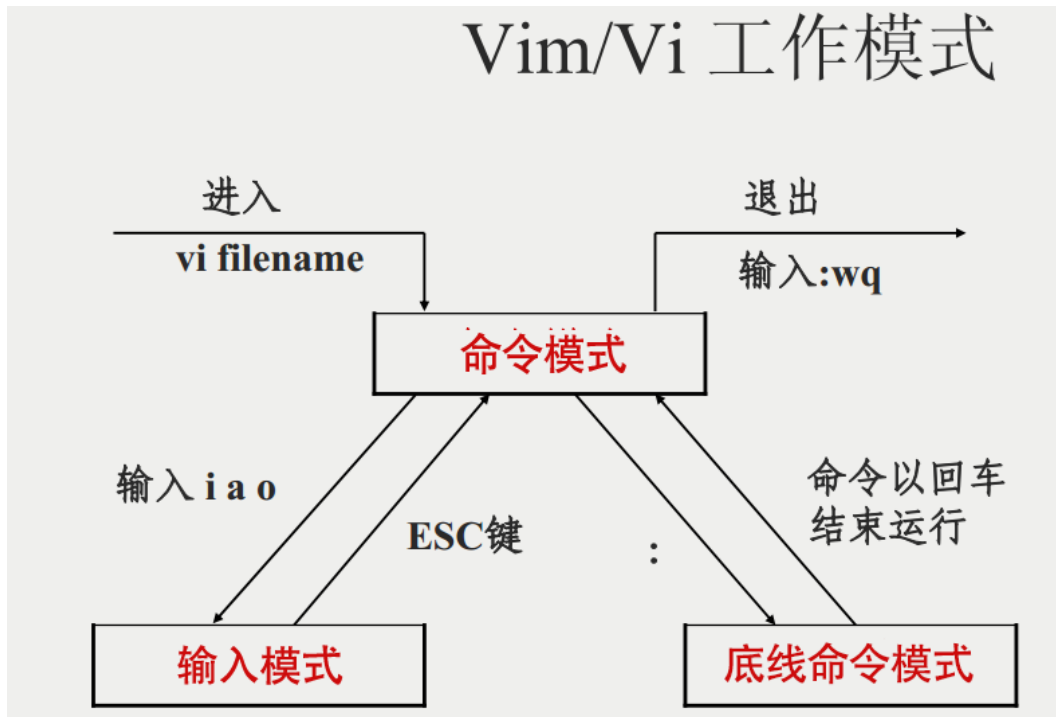
- sudo 命令用于以超级用户（superuser，或称 root）权限执行命令

```
sudo
用法:sudo [命令]
```

Vim 常用功能

编辑器的模式

Vim/Vi 工作模式



Vim 常用功能

光标移动

命令模式下：

操作	含义
<n>G 或 :<n>	n 为数字。移动到这个文件的第 n 行
gg	移动到文件第一行
ctrl+f	向前（forward）翻页
ctrl+b	向后（backward）翻整页

Vim 常用功能

搜索替换

命令模式下：

操作	含义
<code>/<word></code>	文件下寻找名为 <word> 的字符串。
<code>:%s/<word1>/<word2>/g</code>	在全文中寻找 <word1> 字符串，并将该字符串取代为 <word2>

Vim 常用功能

剪切、复制、粘贴

命令模式下：

操作	含义
[n]dd	删除（delete）游标所在的一行或 n 行，用 p/P 可以粘贴（实际更类似于剪切）
[n]yy	复制（yank）游标所在的一行或 n 行，用 p/P 可以粘贴
p/P	p 将已复制的数据粘贴（paste）在光标下一行，P 则为粘贴在光标上一行
u	复原（undo）前一个动作

Vim 常用功能

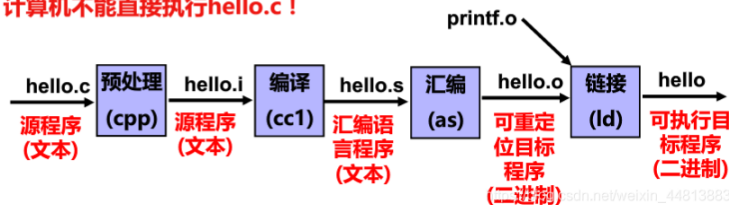
剪切、复制、粘贴

可视模式下（命令模式输入 `v/V`），可以通过鼠标或光标批量选择文本段：

操作	含义
d	剪切选择文本
y	复制选择文本
p	粘贴选择文本
u	复原前一个动作

GCC 编译器的使用

计算机不能直接执行hello.c !



语法: gcc [选项]... [参数]...

选项 (常用) :

- o 指定生成的输出文件
- S 将 C 代码转换为汇编代码
- Wall 显示一些警告信息
- c 仅执行编译操作, 不进行链接操作, 生成可重定位的目标文件
- M 列出依赖
- Ipath 编译时指定头文件目录, 使用标准库时不需要指定目录

- gcc test.c 默认生成名为 a.out 的可执行文件
- gcc test.c -o test 使用 -o 选项编译链接生成 test 的可执行文件
- gcc -c test.c -o test.o 使用 -o -c 选项仅编译生成名为 test.o 的目标文件

Makefile 编写

target、dependencies、command 是 Makefile 的基础

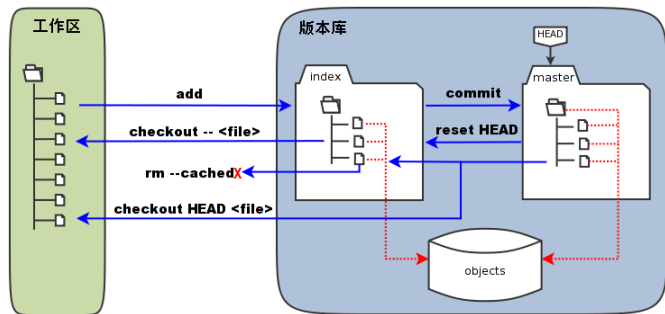
```
target: dependencies
    command 1
    command 2
    ...
    command n
```

- target 是构建 (build) 的目标，可以是目标文件、可执行文件，也可以是一个标签。
- dependencies 是构建该目标所需的其他文件或其他目标。
- command 是构建该目标所需执行的指令。**每一个指令 (command) 之前必须按一次制表符键来控制间隔，而不能是空格**，否则 make 会报错。

如果想要构建 target，那么首先要准备好 dependencies，接着执行 command 中的命令，最终完成构建 target。

Git 使用方法

常用命令



■ `git status`

这个命令可以查看当前分支的状态，以及当前工作区的变动和暂存区的内容，便于我们对工作区的概况进行掌握。

Git 使用方法

常用命令

- `git branch`

`git branch <branch-name>` 本地创建一个基于当前分支产生的分支。

`git branch -d(D) <branch-name>` 删除本地对应分支。

`git branch -a` 查看所有的远程与本地分支。

- `git pull`

这个命令将远程新建的分支下载到本地，并且将远端的更改合并到当前的分支。在利用评测机进行实验分支的初始化之后，可以在开发机中使用这个命令来将新的分支下载到本地。

- `git checkout`

请注意，在切换时，需要保证目前所有文件的状态均为“未修改”（没有修改过，或者已经提交）。

Git 使用方法

常用命令

- `git add`

在你完成一部分实验内容之后，可以使用 `git add .` 将你当前目录下的所有修改加入暂存区，也可以使用 `git add <filename>` 来将指定的文件加入暂存区。

- `git commit`

使用 `git commit -m <message>` 这个命令将暂存区的修改提交到储存库中。在提交时，要求给出一段说明性文字。这段文字可以任意填写，但建议按照提交内容填写，以保证可读性。

- `git push`

这个命令将本地的 commit 推送到一个远程仓库。在课程实验中，这个命令可以将你的 commit 推送到 GitLab。

Git 使用方法

使用流程

- 我们在一次实验结束，新的实验代码下发时，一般是按照以下流程的来开启新的实验之旅。

`git add .` , `git commit -m "xxxx"` 如果当前分支的暂存区还有东西的话，先提交。

`git pull` 这一步很重要！要先确保服务器上的更新全部同步到本地版本库！

`git checkout lab<n>` 检出新实验分支并进行实验。

- 此后我们就可以开始写代码，代码写好后

`git add <modified-file>`

`git commit -m "yyyy"` 提交到本地版本库。

`git push` 将本地版本库推送到服务器。

grep、sed、awk 文本处理三剑客

当你需要在整个项目目录中查找某个函数名、变量名等特定文本的时候，grep 将是你手头一个强有力的工具。

grep

用法：grep [选项] PATTERN FILE

(PATTERN是匹配字符串，FILE是文件或目录的路径)

作用：输出匹配PATTERN的文件和相关的行。

选项（常用）：

- a 不忽略二进制数据进行搜索。
- i 忽略大小写差异。
- r 从目录中递归查找。
- n 显示行号。

grep、sed、awk 文本处理三剑客

sed 是一个文件处理工具，可以将数据行进行替换、删除、新增、选取等特定工作。

sed

sed [选项] '命令' 输入文本

选项（常用）：

-n：安静模式，只显示经过sed处理的内容。否则显示输入文本的所有内容。

-i：直接修改读取的档案内容，而不是输出到屏幕。否则，只输出不编辑。

命令（常用）：

[行号]a[内容]：新增，在行号后新增一行相应内容。行号可以是“数字”，在这一行之后新增，也可以是“起始行，终止行”，在其中的每一行后新增。当不写行号时，在每一行之后新增。使用\$表示最后一行。后面的命令同理。

[行号]c[内容]：取代。用内容取代相应行的文本。

[行号]i[内容]：插入。在当前行的上面插入一行文本。

[行号]d：删除当前行的内容。

[行号]p：输出选择的内容。通常与选项-n一起使用。

s/re（正则表达式）/string：将re匹配的内容替换为string。

grep、sed、awk 文本处理三剑客

awk 是一种处理文本文件的语言，是一个强大的文本分析工具。这里只举几个简单的例子，学有余力的同学可以自行深入学习。

```
1 awk '$1>2 {print $1,$3}' my.txt
```

这个命令的格式为 `awk 'pattern action' file`，pattern 为条件，action 为命令，file 为文件。命令中出项的 `$n` 代表每一行中用空格分隔后的第 `n` 项。所以该命令的意义是文件 `my.txt` 中所有第一项大于 2 的行，输出第一项和第三项。

```
awk -F, '{print $2}' my.txt
```

`-F` 选项用来指定用于分隔的字符，默认是空格。所以该命令的 `$n` 就是用 `,` 分隔的第 `n` 项了。

重定向和管道

- 三种流重定向

```
command <input.txt >output.txt  
command <input.txt >>output.txt  
command <input.txt 1>output.txt 2>err.txt
```

- 管道

```
command1 | command2 | command3 | ...
```

以上内容是将 command1 的 stdout 发给 command2 的 stdin，command2 的 stdout 发给 command3 的 stdin。

shell 脚本编程

变量

定义变量（变量名为 `var_name`，值为 `value`）的方式是：

```
var_name=value
```

请注意，等号两边**不允许有空格**。

使用 `$var_name` 可以获取变量的值。在使用时，建议在变量名的两端加一个花括号（形如 `${var_name}`），以帮助解释器识别变量的边界，避免歧义。

shell 脚本编程

脚本参数——特殊的变量

我们用一个实例来说明参数的传递。我们将 `hello.sh` 的内容修改为：

```
#!/bin/bash
str="Hello, $1 and $2!"
echo $str
```

输入如下命令运行，看看控制台输出了什么：

```
$ ./hello.sh world OS
```

请在双引号中引用变量。如果将上述的双引号改成单引号，则会原文输出引号内的内容，你可以尝试一下。

需要补充的是，对于传递的参数，不仅有 `$1`、`$2` 这样的特殊变量，还提供了其他的特殊变量：

- `$#` 传递的参数个数；
- `$*` 一个字符串，内容是传递的全部参数。

shell 脚本编程

流程控制

`if` 语句块的格式如下，注意在条件和命令间不能缺少 `then` 。

```
if condition1
then
    command11
elif condition2
then
    command21
else
    command31
fi
```

shell 脚本编程

流程控制

`while` 语句块的格式如下：其中循环变量运算赋值可以用 `let i=i+1`、`i=${i+1}`、`i=$((i+1))` 实现。其中 `(())` 是用于整数运算比较的常用运算符：

```
while condition
do
    command1
    ...
done
```

同样注意在条件和命令间不能缺少 `do`。

课下习题提示

■ Exercise 0.1

1. 考察 C 语言
2. 考察 gcc 用法 `-o` 实现指定生成的输出文件
3. 考察 sed 用法：`sed -n '3p;5p' test.c` 可以输出第3行、第5行的内容
4. 考察 cp 用法：`-r` 实现递归复制

■ Exercise 0.2

1. 考察 shell 脚本循环控制语句与 mv 命令

■ Exercise 0.3

1. 考察 grep 查找功能和 awk 分割功能：关注 `grep -n` 显示行号搜索和 `awk -F` 分割并打印行号

课下习题提示

- Exercise 0.4

1. 考察 sed 替换字符串功能：`sed 's/str1/str2/g' test`

2. 考察 Makefile 编写，command 中可以使用 `$(MAKE) -C subdir`