

天津医科大学实验课教案首页

(共 4 页、第 1 页)

课程名称：Linux 系统概论		实验名称：实验 5 Linux 高级命令的操作
教师姓名：伊现富	职称：讲师	教学日期：2016 年 3 月 30 日 13:30-15:30
授课对象：生物医学工程学院 2014 级生信班（本）		实验人数：30
实验类型（验证型、综合型、设计型、创新型）：验证型		实验分组：一人一机
学时数：2		教材版本：Linux 系统概论上机指南（自编教材）

实验目的与要求：

- 了解 datamash 的使用方法。
- 了解 Linux 命令在生物信息学文本处理中的应用。
- 掌握 find、grep、sed 和 AWK 的使用方法。

实验内容及学时分配：

- (10') 元字符和正则表达式：回顾元字符和正则表达式的基础知识。
- (10') 文本处理命令：通过实例回顾 find、grep、sed 和 AWK 的使用方法，总结常用的文本处理命令。
- (80') 实验操作：以 CentOS 发行版为例，练习 find、grep、sed 和 AWK 等高级命令的使用，掌握它们的基本语法。

主要仪器和实验材料：

- 主要仪器：一台安装有 CentOS 的计算机。

实验重点、难点及解决策略：

- 重点难点：正则表达式的构建，find、grep、sed 和 AWK 的基本语法。
- 解决策略：通过实例进行讲解，通过演示进行学习，通过练习熟练掌握。

思考题：

- 列举常见的元字符并解释其含义。
- 根据要求构建正则表达式。
- 使用 find、grep、sed 和 AWK 完成指定的任务。
- 根据要求组合使用文本处理命令。
- 根据要求使用 Linux 命令处理生物信息学文本数据。

参考资料：

- Linux 基础及应用习题解析与实验指导（第二版），谢蓉 编著。中国铁道出版社，2014。

主任签字：

年 月 日

教务处制

一、元字符和正则表达式 (10 分钟)

1. 简介

- 元字符：代表一组字符或命令的字符，用最小的字符集表示多组文本。
- 正则表达式：具有一定句法的集合或短语，包含元字符或普通字符，表示某类文本或字符串。
- 相关命令：less, more, grep, sed, AWK, vim, ……

2. 元字符 (结合实例与完整的正则表达式讲解每一个元字符)

- 字符：一般字符, ., [], [a-z], [0-9], [^], \
- 字符集：\d, \D, \s, \S, w, \W
- 量词：?, *, +, {m}, {m,n}
- 边界：^, \$
- 其他：(), |

正则表达式	描 述	示 例
^	行起始标记	^tux 匹配以tux起始的行
\$	行尾标记	tux\$ 匹配以tux结尾的行
.	匹配任意一个字符	Hack. 匹配Hackl和Hacki, 但是不能匹配Hackl2和Hackil, 它只能匹配单个字符
[]	匹配包含在[字符]之中的任意一个字符	coo[kl] 匹配cook或cool
[^]	匹配除[^字符]之外的任意一个字符	9[^01] 匹配92、93, 但是不匹配91或90
[-]	匹配[]中指定范围内的任意一个字符	[1-5] 匹配从1~5的任意一个数字
?	匹配之前的项1次或0次	colou?r 匹配color或colour, 但是不能匹配colourr
+	匹配之前的项1次或多次	Rollno-9+ 匹配Rollno-99、Rollno-9, 但是不能匹配Rollno-
*	匹配之前的项0次或多次	co*l 匹配cl、col、cool等
()	创建一个用于匹配的子串	ma(tri)? 匹配max或maxtrix
{n}	匹配之前的项n次	[0-9]{3} 匹配任意一个三位数, [0-9]{3}可以扩展为[0-9][0-9][0-9]
{n,}	之前的项至少需要匹配n次	[0-9]{2,} 匹配任意一个两位或更多位的数字
{n,m}	指定之前的项所必需匹配的最小次数和最大次数	[0-9]{2,5} 匹配从两位数到五位数之间的任意一个数字
	交替——匹配 两边的任意一项	Oct (1st 2nd) 匹配Oct 1st或Oct 2nd
\	转义符可以将上面介绍的特殊字符进行转义	a\.b 匹配a.b, 但不能匹配ajb。通过在. 之间加上前缀\, 从而忽略了. 的特殊意义

3. 正则表达式

a. /^[A-Z]..\$/ 查找文本中所有以大写字母开头、后跟两个任意字符，再跟一个换行符的行。查找结果是第 5 行的 Dan。

b. /^[A-Z][a-z]*3[0-5]/ 查找所有以大写字母开头、后跟零个或多个小写字母或空格，再跟数字 3 和一个 0~5 之间的数字的行。查找结果是第 2 行。

c. /[a-z]*\./ 查找包含跟在零个或多个小写字母后的句点的行。结果是第 1、2、7、8 行。

d. /^[^]*[A-Z][a-z][a-z]\$/ 查找以零个或多个空格开头(注意：制表符不算空格)，后跟一个大写字母、两个小写字母和一个换行符的行。结果将是第 4 行的 Tom 和第 5 行的 Dan。

e. /^[A-Za-z]*^[^,][A-Za-z]*\$/ 查找以零个或多个大/小写字母开头，后跟一个非逗号的字符，再跟零个或多个大/小写字母和一个换行符的行。结果是第 5 行。

二、文本处理命令 (10 分钟)

1. find (通过实例讲解 find 的语法和使用)

`find . -name "*.jpg" -exec rm -fv {} \;`

找出以".jpg"结尾的文件

rm -fv 进行删除操作

{} 代表find过滤出来的以".jpg"结尾的文件

": -exec参数指定的command结束符; "\"对后面的分号进行转义

2. grep (通过实例讲解 grep 的语法和使用, 注意正则表达式在其中的应用)

命令	显示
<code>grep '[A-Z]' list</code>	list 中包含一个大写字母的行
<code>grep '[0-9]' data</code>	data 中包含数字的行
<code>grep '[A-Z]...[0-9]' list</code>	list 中包含以大写字母开始、数字结尾的 5 个字符组合的行
<code>grep '\.pic\$' filelist</code>	filelist 中以.pic 结尾的行

3. sed (通过实例讲解 sed 的语法和使用)

- `sed -n '1,3!p' 123.txt`
- `sed -n -e '/the/p' -e '/the/=' 123.txt`
- `sed 's/this/that/g' 123.txt`
- `echo "hello" | sed 's/$/.txt/g'`
- `ls -l | sed -n '1,3p'`

4. AWK (通过实例讲解 AWK 的语法和使用)

- `awk '{print $0}' 123.txt`
- `ls -l | awk '{if($1 !~ /^d/) {print $0}}'`
- `awk '{printf("%03d %s\n",NR,$0)}' ori.txt > dst.txt`
- `awk 'BEGIN{FS=" ";OFS="\t"}{print $1,$2} ori.txt > dst.txt`

5. 文本处理命令

(1) 常用文本处理命令

cut, sort, uniq, split, paste, join, comm, tr,

(2) 使用实例

`cat ~/.bash_history | cut -d " " -f1 | sort | uniq -c | sort -nr | head`

空格

为了使uniq起作用, 所有的重复行必须是相邻的

delimiter

用空格分隔出的第一部分

count

number+reverse

三、实验操作 (80 分钟)

1. find 的使用

- (1) 根据文件名查找文件: `-name`, `-iname`
- (2) 限定搜索目录的深度: `-maxdepth`, `-mindepth`
- (3) 在找到的文件上执行命令 (理解基本的语法, `-exec` vs. `-ok`)
- (4) 相反匹配: `-not`
- (5) 根据 inode 查找文件: `-inum`
- (6) 根据文件权限查找文件: `-perm`, `-type`
- (7) 查找空文件: `-empty`
- (8) 查找最大最小的文件 (组合使用 `find`、`sort` 和 `head`)
- (9) 查找指定类型的文件: `-type`
- (10) 根据文件大小查找文件: `-size`, `+`, `-`
- (11) 根据时间戳查找文件: `-mmin`, `-mtime`, `-amin`, `-atime`, `-cmin`, `-ctime`
- (12) 基于文件比较进行查找: `-newer`, `-anewer`, `-cnewer`

2. grep 的使用 (常用选项: `-i`, `-v`, `-A`, `-B`, `-C`, `-c`, `-n`, `-w`, `-f`)

3. sed 的使用 (常用选项与命令: `-e`, `-f`; `s`, `d`)

4. AWK 的使用 (基本语法结构, 字段的含义)

5. datamash 的使用 (学生课后自学, 与 AWK、Perl、R 等进行比较)