5_Shell05 正则表达式 sed 基本用法 sed 文本块处理

案例: 编写脚本实现添加用户功能,提示用户输入用户名和密码,用户未输入密码时,默认密码 123456.

```
#!/ban/bash
read -p '请输入用户名:' u
useradd $u &> /dev/null
stty -echo              #关闭用户输入密码时的屏幕显示
read -p '请输入密码:' p
echo                    #换行
stty echo               #开启屏幕显示
echo '用户$u 创建成功'
echo ${p:-123456} | passwd --stdin $u &> /dev/null
#${var:-word}功能实现用户未输入密码时,默认密码 123456
if
       [ -z $p ];then
       echo '您未输入密码,初始密码为:123456'
fi
```

一 正则表达式

1.1 正则表达式:Regular Express

使用”一串符号”来描述有共同属性的数据;对文本进行查找匹配

1.1.1 基本正则列表

正则符号	描述
^	匹配行首
$	匹配行尾,^$匹配空行[空格虽无显示,但不为空或空行]
[]	集合,匹配集合中的任意单个字符; //匹配字符段
[^]	对集合取反
.	匹配任意单个字符[换行符\n 以外的任意单个字符]
*	匹配前一个字符任意次数[不能单独使用],包括 0 次
\{n,m\}	匹配前一个字符出现 n 到 m 次
\{n\}	匹配前一个字符出现 n 次
\{n,\}	匹配前一个字符出现 n 次及以上
\(\)	保留

1.1.2 扩展正则列表　使用 egrep 或 grep -E 与扩展正则配套使用

正则符号	描述	相当于
+	最少匹配一次	\{1,\}
?	最多匹配一次	\{0,1\}
{n,m}	匹配 n 到 m 次	\{n,m\}
()	组合为整体,保留	\(\)
|	或者	
\b	单词边界	

```
[root@server0 opt]# head -n 5 /etc/passwd > user
[root@server0 opt]# grep '^$' user  #匹配空行
[root@server0 opt]# grep '^ ' user  #匹配以空格开头的行
[root@server0 opt]# grep ' $' user  #匹配以空格结尾的行

[root@server0 opt]# grep '[bin]' user  #匹配有字母 b 或 i 或 n 的行
[root@server0 opt]# grep 'roo[tdg]' user
      #匹配有 root 或 rood 或 roog 的行
[root@server0 opt]# grep '[a-z]' user  #匹配有所有小写字母的行
[root@server0 opt]# grep '[A-Z]' user  #匹配有所有大写字母的行
[root@server0 opt]# grep '[a-zA-Z]' user  #匹配有所有字母的行
[root@server0 opt]# grep '[^a-zA-Z]' user  #不匹配字母
[root@server0 opt]# grep '[0-9]' user  #匹配所有数字
[root@server0 opt]# grep '[^0-9]' user  #不匹配数字
[root@server0 opt]# grep '[rot]' user  #匹配 r 或 o 或 t
[root@server0 opt]# grep '[^rot]' user  #匹配 r 或 o 或 t 之外的字符

[root@server0 opt]# grep '.' user  #匹配任意字符
[root@server0 opt]# grep 'roo.' user #匹配 roo 后面追加 1 个字符的行
[root@server0 opt]# grep 'ro..' user #匹配 roo 后面追加 2 个字符的行
```

```
[root@server0 opt]# grep '^.' user  #匹配任意字符开头的行
[root@server0 opt]# grep '.$' user  #匹配任意字符结尾的行
[root@server0 opt]# grep '.*' user  #匹配任意
```

.*相当于正则表达式中的通配符,*表示任意次,可为 0

```
[root@server0 opt]# grep '*' user  #不能单独使用
[root@server0 opt]# grep 'a*' user
```

#匹配有 a 的行,a 可以出现任意次,包括 0 次

```
[root@server0 opt]# grep 'o\{1,2\}' user #匹配 o 出现 1 到 2 次
[root@server0 opt]# grep 'o\{2\}' user  #匹配 o 出现 2 次
[root@server0 opt]# grep 'o\{1,\}' user #匹配 o 出现 1 次及以上
[root@server0 opt]# grep 'o\{2,\}' user #匹配 o 出现 2 次及以上
```

```
[root@server0 opt]# grep '\(0:\)\{2,\}' user
```

#匹配 0:出现 2 次及以上. \(0:\)将 0:变成一个整体

1.2 egrep 过滤工具

1.2.1 文本处理顺序

以行为单位,逐行进行处理

默认只输出与表达式相匹配的文本行

1.2.2 基本用法

格式 1: egrep [选项] ‘正则表达式’ 文件... ...

格式 2: 前置命令 | egrep [选项] ‘正则表达式’

正则表达式 必须加单引号’’

```
[root@server0 opt]# grep 'o\{1,\}' user
[root@server0 opt]# egrep 'o+' user #匹配 o 出现 1 次及以上

[root@server0 opt]# grep 'o\{0,1\}' user
[root@server0 opt]# egrep 'o?' user #匹配 o 出现 0 次或 1 次

[root@server0 opt]# grep '\(0:\)\{2,\}' user
[root@server0 opt]# egrep '(0:){2,}' user
```

#匹配 0:出现 2 次及以上

```
[root@server0 opt]# grep '[ro]' user  #匹配 r 或 o
[root@server0 opt]# egrep 'r|o' user  #匹配 r 或 o

[root@server0 opt]# egrep '\bthe\b' 1.txt
```

#查找前后都无字母数字的 the

1.2.3 常用命令选项

-i	忽略字母大小写
-v	条件取反
-c	统计匹配的行数
-q	静默\无任何输出,一般用于检测
-n	显示出匹配结果所在的行号
--color	标红显示匹配字符串

类 型	含 义	示 例	说 明
( )	组合为整体	ab{1,3}	匹配 ab、abb、abbb
		(ab){1,3}	匹配 ab、abab、ababab
|	或者	root|bin	匹配 root、bin
\b	单词边界	\broot\b	匹配单词root，不匹配 keroot、rooty、brooty等字符串
\<	单词的开头	\<th	匹配以th开头的单词
\>	单词的结束	\<root\>	作用与 \broot\b 相同
\w	字母数字下划线	\wa	匹配xa ,不匹配#a
\s	匹配空白	\sa	匹配 a ,不匹配xa
\d	匹配数字	-P \da	匹配5a，不匹配xa

二 sed

2.1 Stream EDitor 流程式编辑器

非交互式,基于模式匹配过滤及修改文本

逐行处理,并将结果输出到屏幕

可实现对文本的输出\删除\替换\复制\剪切\导入\导出等操作

2.2 格式

格式 1: **前置命令 | sed [选项] ‘定址符 动作指令’**

格式 2: **sed [选项] ‘定址符 动作指令’ 操作的文件**

定址符可以用正则表达式来表示,以;号分隔

2.3 常见选项

-n 屏蔽默认输出,默认 sed 会输出读取文档的全部内容

-i 直接修改文件内容

-r 启用扩展的正则表达式,若与其他选项一起使用,应作为首个选项

2.4 常用动作指令

p	打印行	2,4p 输出第 2 3 4 行	
		2p;4p 输出第 2 行,第 4 行	
d	删除行	2,4d 删除第 2 3 4 行	
s	字符串替换	s/old/new/	将每行的第 1 个 old 替换为 new
		s/old/new/**n**	将每行的第 **n** 个 old 替换为 new
		s/old/new/g	将所有的 old 替换为 new

#分割符 / 可改用其他字符,如# &等,便于修改文件路径

2.5 示例

2.5.1 打印 passwd 第 3 到第 6 行账户的信息：

```
[root@svr5 ~]# sed  -n '3,6p' /etc/passwd
```

2.5.2 sed 命令的 -i 选项

正常情况下，sed 命令所做的处理只是把操作结果（包括打印、删除等）输出到当前终端屏幕，而并不会对原始文件做任何更改：

```
[root@svr5 ~]# sed 'd' /etc/passwd  #删除所有行
[root@svr5 ~]# cat /etc/passwd  #查看原始文本，并未改动
```

若希望直接修改文件内容，应添加选项 -i 。

比如，直接删除 test.txt（自行创建一个任意内容的文件）的第 1~4 行：

```
[root@svr5 ~]# sed -i '1,4d' test.txt   #删除操作
[root@svr5 ~]# cat test.txt  #确认删除结果
```

下文中关于使用 sed 修改文件的示例中，为了避免大家在练习过程中因误操作导致系统故障，命令省略 –i 选项，不再逐一说明。需要时，大家可自行加上此选项。

2.5.3 多个指令可以使用分号隔离

```
[root@svr5 ~]# sed -n '1p;4p' /etc/passwd  #输出第 1 4 行
```

2.5.4 行号案例

```
[root@svr5 ~]# sed -n '3p' /etc/passwd  #打印第 3 行
[root@svr5 ~]# sed -n '3,5p' /etc/passwd  #打印第 3 到 5 行
[root@svr5 ~]# sed -n '3p;5p' /etc/passwd  #打印第 3 和 5 行
[root@svr5 ~]# sed -n '3,+10p' /etc/passwd
#打印第 3 以及后面的 10 行
[root@svr5 ~]# sed -n '1~2p' /etc/passwd  #打印奇数行
[root@svr5 ~]# sed -n '2~2p' /etc/passwd  #打印偶数行
sed -n “=” /etc/passwd  #查看所有行的行号
sed -n “$=” /etc/passwd  #查看最后 1 行的行号
sed -n “$p” /etc/passwd  #输出最后一行
sed “$d” /etc/passwd  #删除最后一行
```

2.5.5 正则案例

```
[root@svr5 ~]# sed -n '/root/p' /etc/passwd  #打印包含 root 的行
[root@svr5 ~]# sed -n '/bash$/p' /etc/passwd  #打印 bash 结尾的行
[root@svr5 ~]# sed -n 'p' /etc/passwd  #没有条件，则表示打印所有行
```

2.5.6 下面看看 sed 工具的 p 指令案例集锦（自己提前生成一个 a.txt 文件）

```
[root@svr5 ~]# sed  -n 'p' a.txt  #输出所有行，等同于 cat a.txt
```

```
[root@svr5 ~]# sed  -n '4p' a.txt  #输出第 4 行
[root@svr5 ~]# sed  -n '4,7p' a.txt  #输出第 4~7 行
[root@svr5 ~]# sed  -n '4,+10p' a.txt  #输出第 4 行及其后的 10 行内容
[root@svr5 ~]# sed  -n '/^bin/p' a.txt  #输出以 bin 开头的行
[root@svr5 ~]# sed  -n '=' a.txt  #输出文件所有行的行数
[root@svr5 ~]# sed  -n '$=' a.txt  #输出文件最后一行的行数
```

2.5.7 下面看看 sed 工具的 d 指令案例集锦（自己提前生成一个 a.txt 文件）

```
[root@svr5 ~]# sed  '3,5d' a.txt  #删除第 3~5 行
[root@svr5 ~]# sed  '/xml/d' a.txt  #删除所有包含 xml 的行
[root@svr5 ~]# sed  '/xml/!d' a.txt
      #删除不包含 xml 的行，!符号表示取反
[root@svr5 ~]# sed  '/^install/d' a.txt  #删除以 install 开头的行
[root@svr5 ~]# sed  '$d' a.txt  #删除文件的最后一行
[root@svr5 ~]# sed  '/^$/d' a.txt   #删除所有空行
```

2.5.8 sed 命令的 s 替换基本功能（s/旧内容/新内容/选项）：

```
[root@svr5 ~]# vim  test.txt                #新建素材
   2017 2011 2018
   2017 2017 2024
```

```
   2017 2017 2017
[root@svr5 ~]# sed 's/2017/xxxx/' test.txt
[root@svr5 ~]# sed 's/2017/xxxx/g' test.txt
[root@svr5 ~]# sed 's/2017/xxxx/2' test.txt
[root@svr5 ~]# sed 's/2017//2' test.txt
[root@svr5 ~]# sed -n 's/2017/xxxx/p' test.txt
```

2.5.9 下面看看 sed 工具的 s 指令案例集锦（自己提前生成一个 a.txt 文件）

注意：**替换操作的分隔“/”可改用其他字符，如#、&等，便于修改文件路径**

```
[root@svr5 ~]# sed 's/xml/XML/'  a.txt  #将每行中第一个 xml 替换为 XML
[root@svr5 ~]# sed 's/xml/XML/3' a.txt
      #将每行中的第 3 个 xml 替换为 XML
[root@svr5 ~]# sed 's/xml/XML/g' a.txt  #将所有的 xml 都替换为 XML
[root@svr5 ~]# sed 's/xml//g' a.txt  #将所有的 xml 都删除（替换为空串）
[root@svr5 ~]# sed 's#/bin/bash#/sbin/sh#' a.txt
      #将/bin/bash 替换为/sbin/sh
      定界符双引号”” ： 允许扩展,以$引用其他变量,替换带路径内容的文档,
[root@svr5 ~]# sed '4,7s/^/#/' a.txt  #将第 4~7 行注释掉（行首加#号）
      # ^代表行首开始位置,$代表行尾结束位置
[root@svr5 ~]# sed 's/^#an/an/' a.txt
```

#解除以#an 开头的行的注释（去除行首的#号）

2.5.10 参考数据文件内容如下：

```
[root@svr5 ~]# cat nssw.txt
Hello the world
ni hao ma beijing
```

本小节的操作使用 nssw.txt 作为测试文件。

删除文件中每行的第二个、最后一个字符

分两次替换操作，第一次替换掉第 2 个字符，第二次替换掉最后一个字符：

```
[root@svr5 ~]# sed 's/.//2;s/.$//p' nssw.txt
```

 # .在正则表达式中代表任意字符,^.表示开头的任意字符,.$表示结尾的任意字符;p 为 sed 的动作指令.

将文件中每行的第一个、倒数第 1 个字符互换

每行文本拆分为“第 1 个字符”、“中间的所有字符”、“倒数第 1 个字符”三个部分，然后通过替换操作重排顺序为“3-2-1”：

```
[root@svr5 ~]# sed -r 's/^(.)(.*)(.)$/\3\2\1/' nssw.txt
```

 # (.)复制,\3 \2 \1 表示粘贴;^(.)复制开头的任意 1 个字符 (.)$复制结尾的任意 1 个字符;.*为正则表达式里的通配符;(.*)复制通配符匹配的字符串

删除文件中所有的数字:

```
[root@svr5 ~]# sed 's/[0-9]//g' nssw.txt
```

以 nssw2.txt 文件为例，删除所有数字、行首空格的操作如下：

```
[root@svr5 ~]# sed -r 's/[0-9]//g;s/^( )+//' nssw2.txt
```

为文件中每个大写字母添加中括号

使用“（）”可实现保留功能，所以可参考下列操作解决：

```
[root@svr5 ~]# sed -r 's/([A-Z])/[\1]/g' nssw.txt
```

#找到所有大写字母并复制,粘贴时添加中括号

2.5.11 使用 sed 修改系统配置

本案例要求熟悉课上的 sed 应用案例，并编写脚本 anonftp.sh，实现以下功能：

通过 yum 安装 vsftpd 软件包

修改 vsftpd 服务配置，开启匿名上传

调整/var/ftp/pub 目录权限，允许写入

启动 vsftpd 服务，并设置开机自运行

步骤

实现此案例需要按照如下步骤进行。

步骤一：编写 anonftp.sh 脚本，用来装配匿名 FTP 服务

1）任务需求及思路分析

vsftpd 服务的安装、改目录权限、起服务等操作可以直接写在脚本中。

修改 vsftpd.conf 配置的工作可以使用 sed 命令，根据默认配置，只需要定位到以

#anon 开头的行，去掉开头的注释即可。

2）根据实现思路编写脚本文件

```
[root@svr5 ~]# vim anonftp.sh
#!/bin/bash
yum -y install vsftpd  #安装 vsftpd 软件
cp /etc/vsftpd/vsftpd.conf{,.bak}  #备份默认的配置文件
sed -i 's/^#anon/anon/' /etc/vsftpd/vsftpd.conf  #修改服务配置
chmod 777 /var/ftp/pub  #调整目录权限
systemctl restart vsftpd  #启动服务
systemctl enable vsftpd  #设为自动运行
systemctl setenforce 0  #关闭 selinux
systemctl stop firewalld  #关闭防火墙
[root@svr5 ~]# chmod +x anonftp.sh
[root@svr5 ~]# ./anonftp.sh
```

三 sed 文本块处理

3.1 sed 单行文本处理

格式: **sed [选项] “定址符(行号) 操作符 内容” 文件...**

定址符可用正则表达式来表示

3.2sed 本文处理操作符

i：在指定的行之前插入文本

a：在指定的行之后追加文本

c：替换指定的行

#sed 以行为单位,行之前与行之后可理解为行上或行下

3.3 单行文本处理案例

注意：系统默认没有 a.txt 文件，需要自己创建一个测试文件！！！

```
[root@svr5 ~]# sed  '2a XX'   a.txt  #在第二行后面，追加 XX
[root@svr5 ~]# sed  '2i XX'   a.txt  #在第二行前面，插入 XX
[root@svr5 ~]# sed  '2c XX'   a.txt  #将第二行替换为 XX
```

3.4 修改主机名案例

3.4.1 确认修改前的配置

```
[root@svr5 ~]# cat /etc/hostname
svr5.tarena.com
```

3.4.2 使用 sed 修改主机名配置所在行的内容（**c 整行替换**）

```
[root@svr5 ~]# sed  '1c mysvr.tarena.com' /etc/hostname
```

3.5 修改 hosts 文件，添加两条映射记录：192.168.4.5 与 svr5.tarena.com、svr5，还有 119.75.217.56 与 www.baidu.com

3.5.1 确认修改前的配置

```
[root@svr5 ~]# cat /etc/hosts
```

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

3.5.2 使用 sed 修改 hosts 文件，添加两行新纪录（a 追加）

[root@svr5 ~]# sed -i '$a 192.168.4.5 svr5.tarena.com svr5' /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.4.5 svr5.tarena.com svr5