实验报告 2
Shell, Vim 与数据整理

洪子翔

2024 年 9 月 12 日

目录

1 实验内容

1.1 Shell

1.1.1 导航

使用pwd查看当前所在目录，ls查看目录下的所有文件。

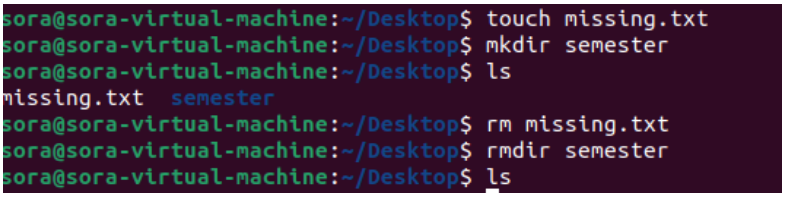
```
sora@sora-virtual-machine:~$ pwd
/home/sora
sora@sora-virtual-machine:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos
```

要移动到其他目录，使用cd ...，后面可以是绝对路径，也可以是相对路径。
其中，相对路径开头是.，代表当前路径。.. 是上一级目录。
使用cd ./Desktop或/cd home/usr/Desktop到桌面。

```
sora@sora-virtual-machine:~$ cd ./Desktop
sora@sora-virtual-machine:~/Desktop$
```

1.1.2 添加和删除文件（夹）

要添加文件touch <name>是最常见的办法，也可以用vim <name>借助 vim 编辑器添加。
删除文件使用rm <name>
添加文件夹使用mkdir <name>，删除则是rmdir <name>
接下来在桌面新建”missing.txt” 和文件夹 semester，再删除。



1.1.3 文件链接

新建文件 test，尝试写入

```
#!/bin/sh
curl --head --silent https://missing.csail.mit.edu
```

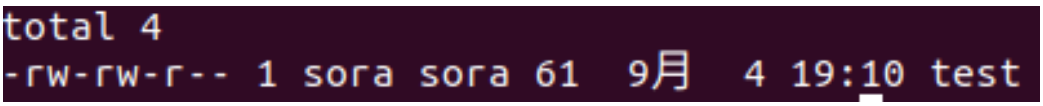
可以使用`< file`和`> file`分别将输入输出写入文件。`>> file`则会在文件末尾追加输出，避免前者覆盖文件中原有内容。

注意上述写入内容含有特殊符号，需要用单引号括起来。

利用`echo <content>`返回内容的性质，可以实现：

1.1.4 查看与更改权限

`ls -l`可以检查当前目录下所有文件的权限：



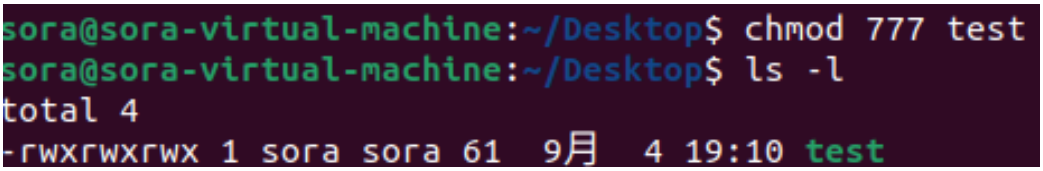
每一列从左到右是：文件类型与权限、硬链接数、所有者、所属组、大小、最后修改时间与文件名。

其中第一部分有 10 个字符，分为四组：

```
-rw-rw-r--
```

第一个字符代表文件类型：目录'd' 或文件 '-'；

之后每三个字符一组，分别代表“读取、写入和执行权限”，由 r,w,x 表示。没有该权限显示为-。此处共有 3 组，代表文件所有者、所属组和其他人的权限。
要修改权限，使用`chmod XXX <file>`，后面的 XXX 是三个 0 7 的数字。将每个数写成三位二进制数，从左到右分别对应三种权限，1 为有 0 为无。
假如让所有人获得该文件的所有权限，只需要输入`chmod 777 test`
此外还有用符号表示的写法。



1.1.5 简单的脚本

编写两个 bash 函数 marco 和 polo 执行下面的操作：
每当执行 marco 时，当前的工作目录应当以某种形式保存，当执行 polo 时，无论现在处在什么目录下，都应当 cd 回到当时执行 marco 的目录
使用 nano 或 vim 编写内容如下：
marco:

```
1    marco(){
2    echo "$(pwd)" > $HOME/marco_history.log
3    echo "save pwd $(pwd)"
4    }
```

polo:

```
1    polo(){
2    cd "$(cat "$HOME/marco_history.log")"
3    }
```

1.2 Vim

1.2.1 启动与基础知识

只需输入`vim <filename>`就可以用 vim 打开文件并编辑。如果没有文件会新建一个。
vim 有多种模式，最常用的是正常模式和插入模式。初始应该在正常模式。在正常模式按 i 进入插入模式，就可以编辑文本了。

1.2.2 退出 vim

编辑完之后退出，需要先按 esc 退出插入模式，再输入`:wq`退出。
其中，: 代表进入命令行模式，w 代表保存，q 代表退出。

1.2.3 移动

上下移动:hjkl 对应 ←↓↑→
以词为单位：b 头 e 尾 w 下一个
行内：0 头 $ 尾 ^ 第一个非空白字符

1.2.4 命令模式下操作

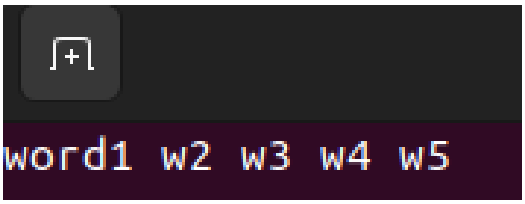
剪切：dd
复制：yy
粘贴：p（光标前）P（后）
撤销：u
插入行：o（上）O（下）
删除：x

1.2.5 删除与改变命令

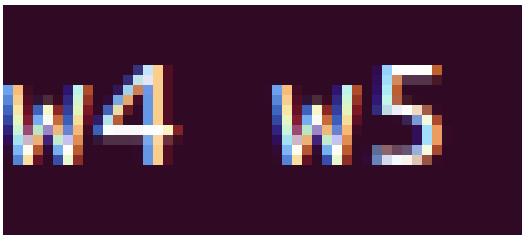
d{移动命令} 删除 {移动命令}
例如 dw 删除词, d$ 删除到行尾, d0 删除到行头。
同理有 c 移动命令 改变移动命令

1.2.6 计数

在命令前加上数字代表执行命令几次。

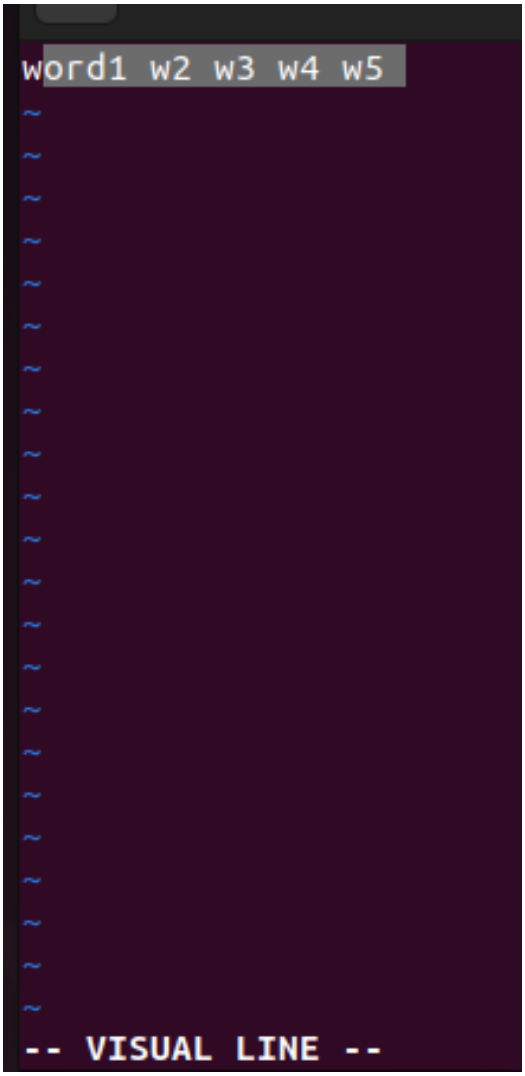


执行 3dw，删除单词 3 次：



1.2.7 可视化模式

按 v 进入可视化模式，V 和 ctrl+v 是行和块可视化。



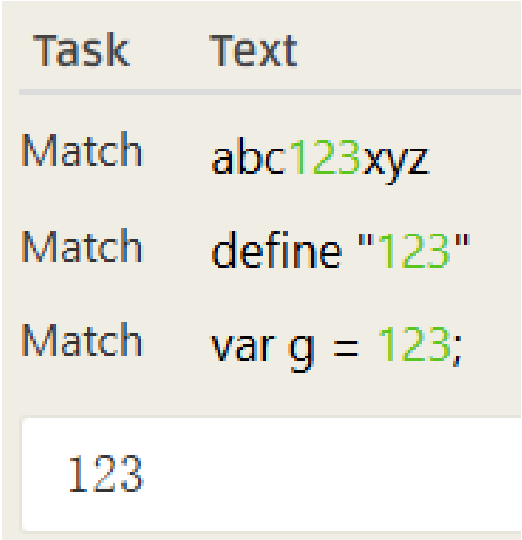
如图，此处用 V 进入行可视化。

1.3 正则表达式

这一段使用 https://regexone.com/lesson/完成习题

1.3.1 基础匹配

直接输入对应的字母和数字，就可以自动匹配
输入代˙ 表匹配任意字符
Lesson1.5：



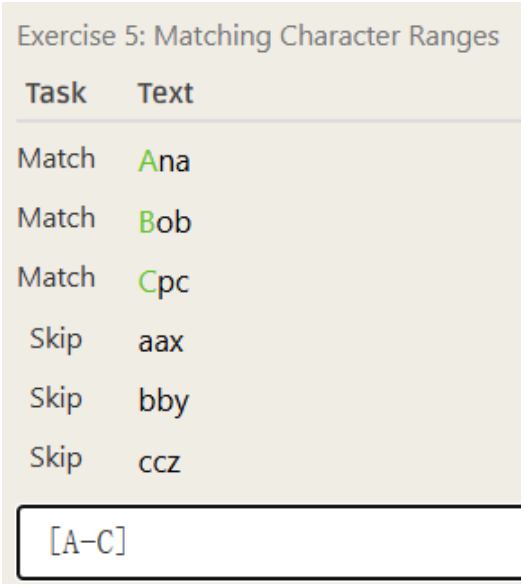
1.3.2 匹配范围

中括号代表范围，包括字母和数字，例如：
dfghj

代表匹配 dfghj
d-z

代表匹配 d-z 的所有字符（含）
-8

代表 0-8 内的所有数字（含）
中括号内开头加 ^ 代表反选
Lesson5:

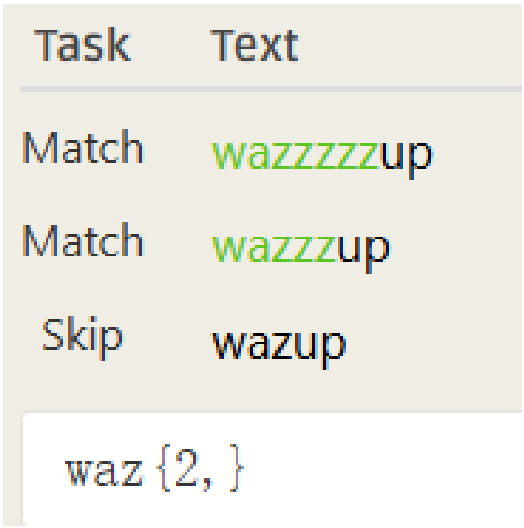


1.3.3 特殊字符

\d：所有数字
\w：所有数字和字母
\s：所有空白字符
上述的字母改成大写，意为“非”

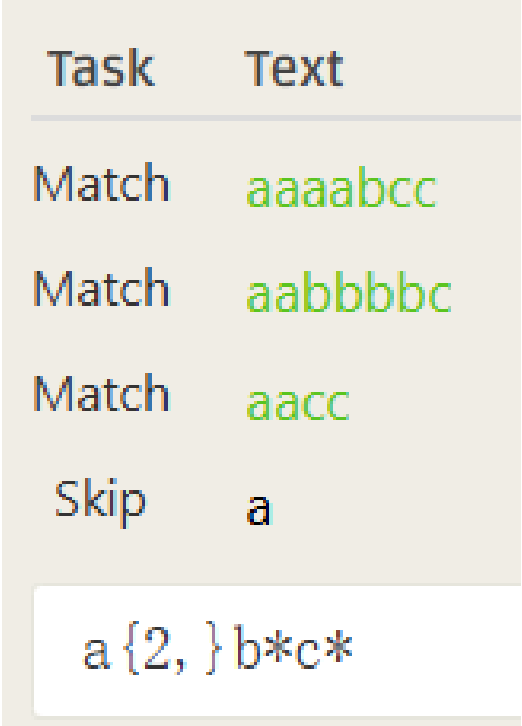
1.3.4 数量限制

在某个字符后加上 {a,b} 代表限制字符出现 a 到 b 次（含），除此之外：
a[3] 代表匹配 a 出现 3 次，a[3,] 匹配 3 次及以上，a[3,5] 匹配 3 5 次
Lesson6:



1.3.5 数量限制符号

* == {0,}
+ == {1,}
? == {0,1}
Lesson7:

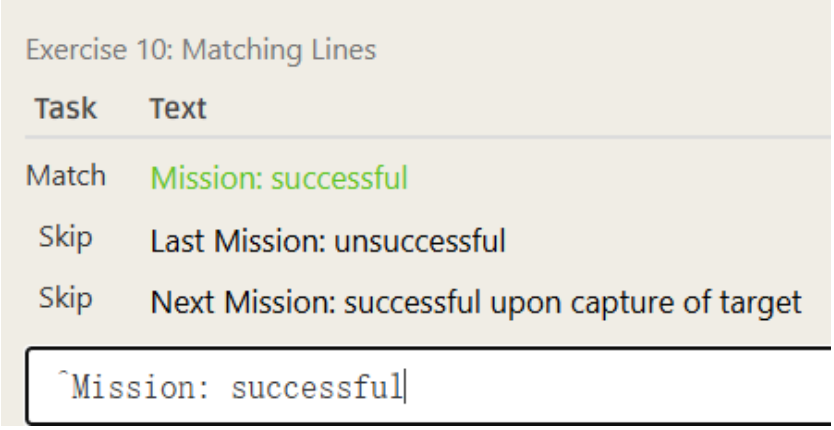


1.3.6 分组

将字符串用小括号括起来可以分组，之后用\<num> 引用第 <num> 个组
小括号内的部分会被抓取
比如匹配”ha-ha haa-haa-haa” 可以写成：
`(ha-\1 (haa)-\2-\2)`

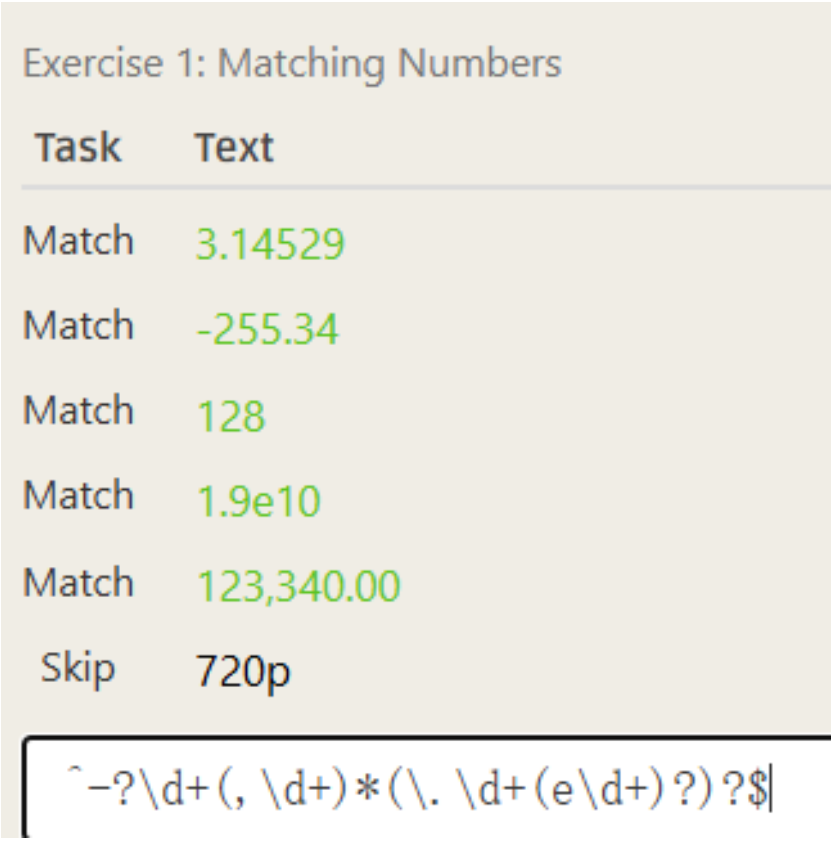
1.3.7 首尾匹配

表达式前加 ^ 代表匹配开头，后加 $ 匹配结尾
Lesson10:

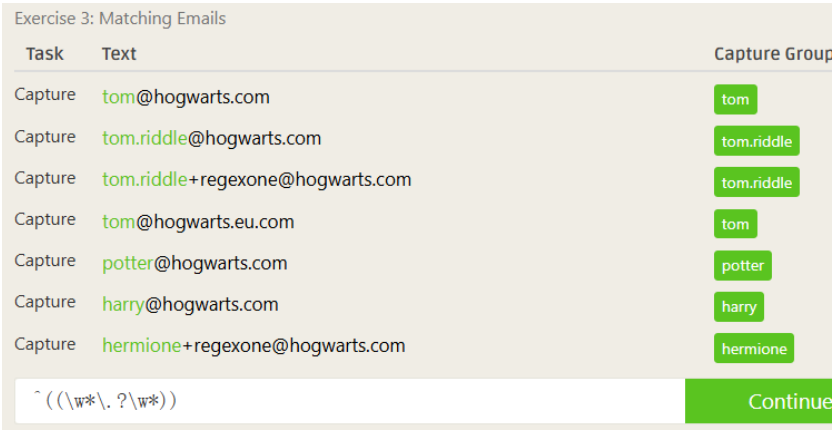

Exercise 10: Matching Lines
Task
Text
Match
Mission: successful
Skip
Last Mission: unsuccessful
Skip
Next Mission: successful upon capture of target
^Mission: successful

1.3.8 练习

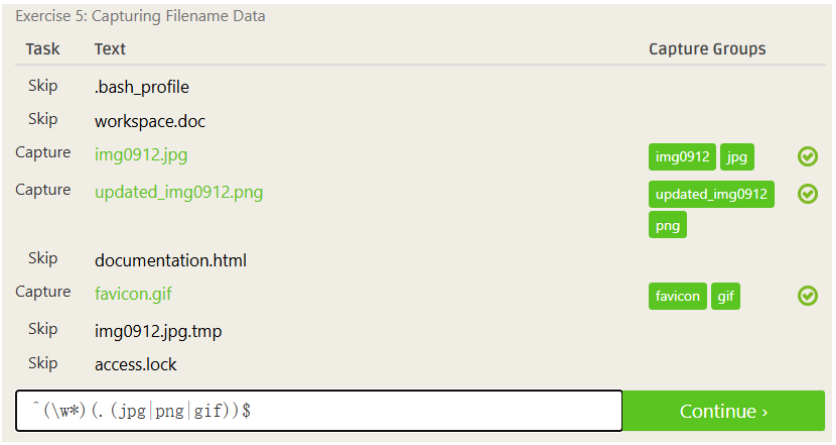
Ex1：


Exercise 1: Matching Numbers
Task
Text
Match
3.14529
Match
-255.34
Match
128
Match
1.9e10
Match
123,340.00
Skip
720p
^-?\d+(,\d+)*(\.\d+(e\d+)?)?$

Ex3:



Ex5:



2 心得

通过这节课的学习，我学到了许多新的东西，相信他们会在之后的学习和工作中发挥大用处。