

Shell工具与Vim编辑器的学习

23020007073 刘畅

2024 年 9 月 13 日

1 实验目的

本次课程主要了解了Shell工具，脚本，编辑器Vim以及数据整理。

2 练习内容

2.1 Shell学习样例10个

1.echo "Hello, World!"用Shell打印hello world echo 是一个在命令行界面（shell）中常用的命令，它能够输出一段文本或者变量的值到标准输出

```
91558@1123c123 MINGW64 ~/Desktop (master)
$ echo "hello, world!"
hello, world!
```

图 1: echo 用来显示文本或变量

2.变量赋值及调用

```
91558@1123c123 MINGW64 ~/Desktop (master)
$ sentence="Good morning!"

91558@1123c123 MINGW64 ~/Desktop (master)
$ echo $sentence
Good morning!
```

图 2: 变量的赋值和使用

3.加法运算

```
91558@1123c123 MINGW64 ~/Desktop (master)
$ a=1

91558@1123c123 MINGW64 ~/Desktop (master)
$ b=2

91558@1123c123 MINGW64 ~/Desktop (master)
$ sum=$((a+b))

91558@1123c123 MINGW64 ~/Desktop (master)
$ echo"The sum of $a and $b is: $sum"
bash: echoThe sum of 1 and 2 is: 3: comma
```

图 3: 进行加法运算

4. 条件语句的判断

```
91558@1123c123 MINGW64 ~  
$ a=10  
  
91558@1123c123 MINGW64 ~  
$ if [ a=10 ]; then  
> echo "yes"  
> else  
> echo "no"  
> fi  
yes
```

图 4: 条件语句的判断

5. 循环的使用

```
91558@1123c123 MINGW64 ~  
$ for i in {1..5}  
> do  
> echo "shuzi: $i"  
> done  
shuzi: 1  
shuzi: 2  
shuzi: 3  
shuzi: 4  
shuzi: 5
```

图 5: 循环语句

6. 函数的创建和使用

```
91558@1123c123 MINGW64 ~  
$ greet() {  
> echo "Hello, world!"  
> }  
  
91558@1123c123 MINGW64 ~  
$ greet  
Hello, world!
```

图 6: 定义函数与使用

7. 文件的创立与写入

```

$ touch filename.txt

91558@1123c123 MINGW64 ~
$ cat > filename.txt <<EOF
> 123asd
> 456zxc
> EOF

91558@1123c123 MINGW64 ~
$ cat filename.txt
123asd
456zxc

```

图 7: 创建写入读取文件

8. 创建目录与切换

```

91558@1123c123 MINGW64 ~
$ mkdir new_directory

91558@1123c123 MINGW64 ~
$ cd new_directory

91558@1123c123 MINGW64 ~/new_directory
$ pwd
/c/Users/91558/new_directory

```

图 8: 创建与切换新目录

9. 获取工作目录，并列出工作目录

```

91558@1123c123 MINGW64 ~
$ mkdir new_directory

91558@1123c123 MINGW64 ~
$ cd new_directory

91558@1123c123 MINGW64 ~/new_directory
$ pwd
/c/Users/91558/new_directory

```

图 9: 获取目录并列出当前目录

10. 查看当前目录文件数目

```
91558@1123c123 MINGW64 ~/new_directory
$ file_count=$(ls -l | grep ^- |wc -l)

91558@1123c123 MINGW64 ~/new_directory
$ echo"$file_count"
bash: echo0: command not found

91558@1123c123 MINGW64 ~/new_directory
$ echo "$file_count"
0
```

图 10: 查看当前目录文件数目

2.2 Vim学习例子10个

1. Vim文件的创立，在终端中输入 `vim example.txt`

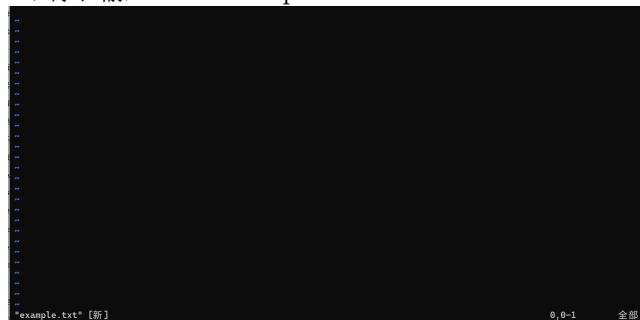


图 11: Vim文件的创立

2. `i`键为插入，开始输入内容hello, vim!

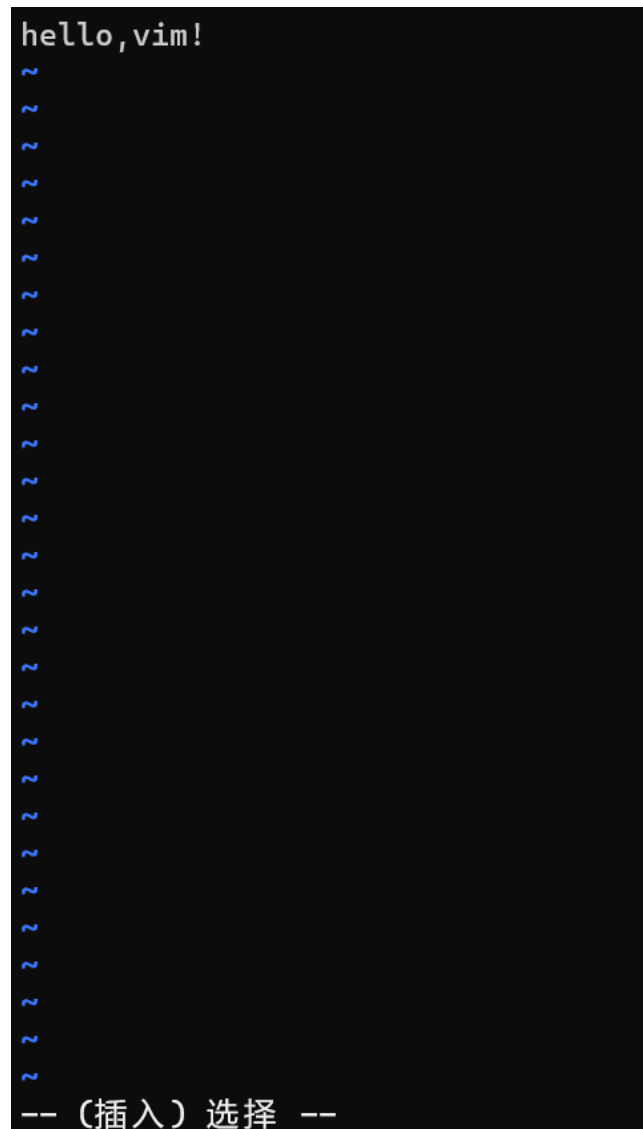


图 12: i插入内容

3.编辑的退出与保存

Esc是回到正常模式

: **w**带编者保存

: **q**代表退出

如果不按**w**的话则文件为**SWP**文件，用于临时储存数据

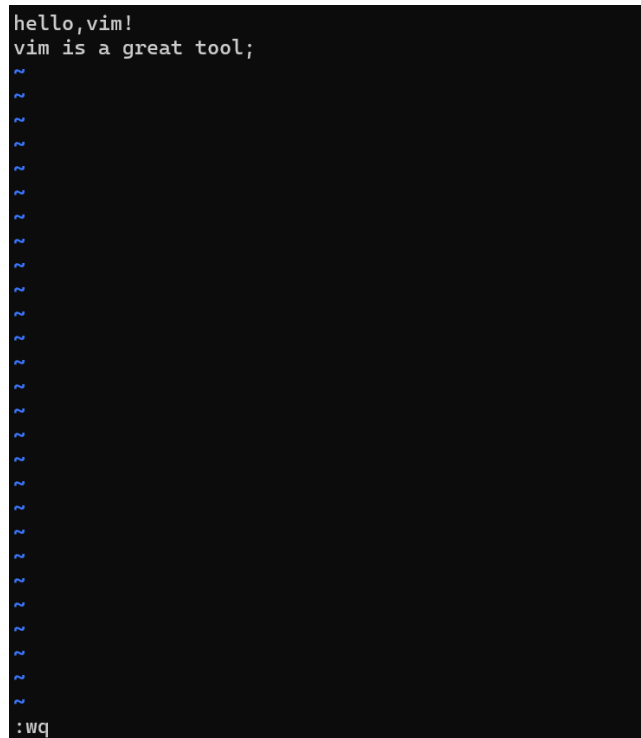


图 13: 编辑的退出与保存

4.dd表示删除当前行

在vim is a greet tool前输入dd删除后的结果

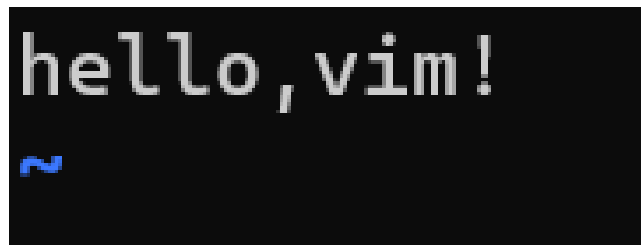


图 14: dd删除当前行后的结果

5.dw: 删除光标后的单词。

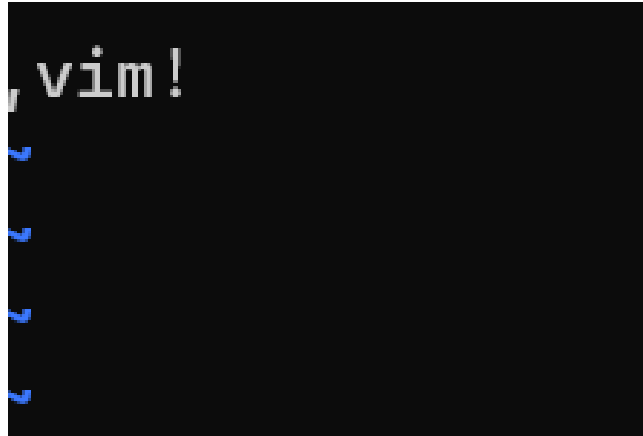


图 15: dw删除光标后的单词的结果

6.x: 删除光标下的字符。

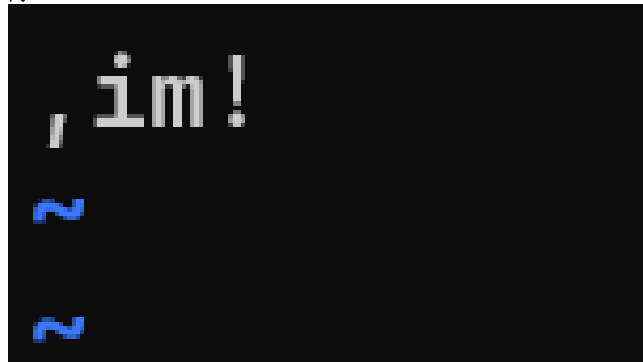


图 16: 删除光标下的字符的结果

7.r: 替换光标下的单个字符。R: 进入替换模式，连续替换多个字符直到按下 Esc。

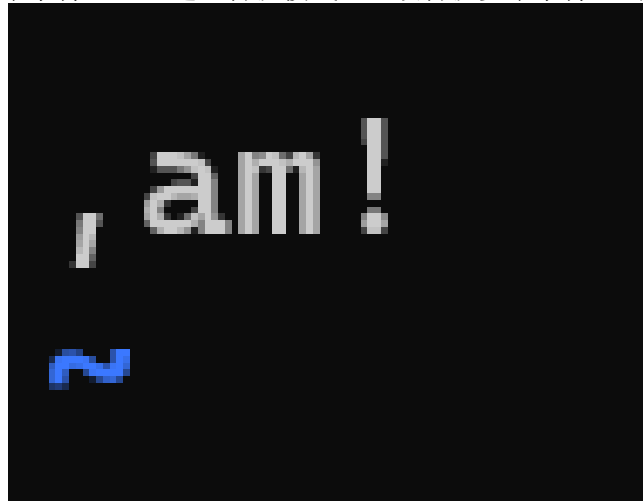


图 17: 用r替换光标下的单个字符

~~~~~

—

222

8. u: 撤销最后一次更改。



Ctrl + r: 重做最后一次撤销, 即反撤销。

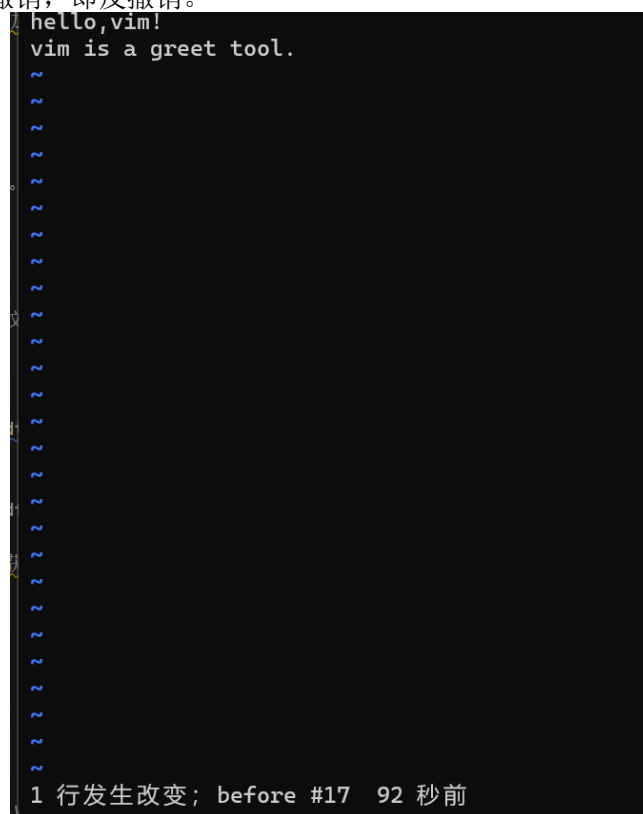


图 20: 撤销

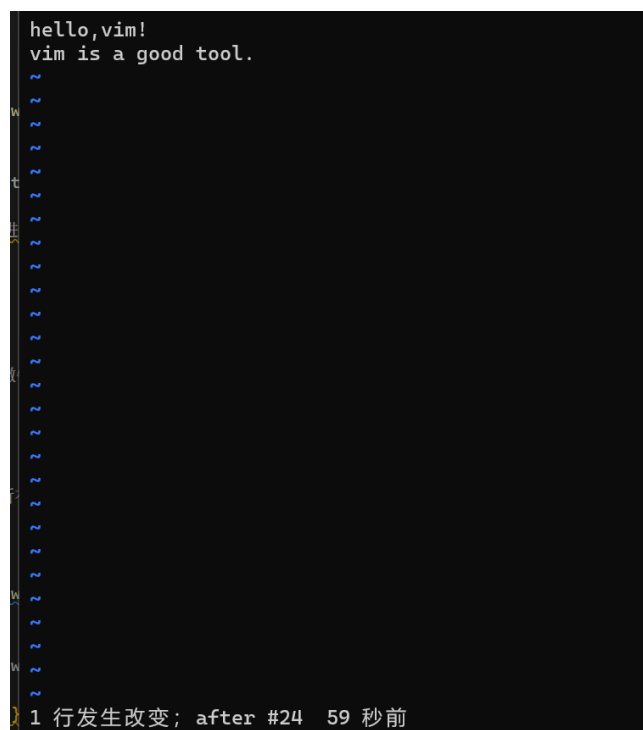
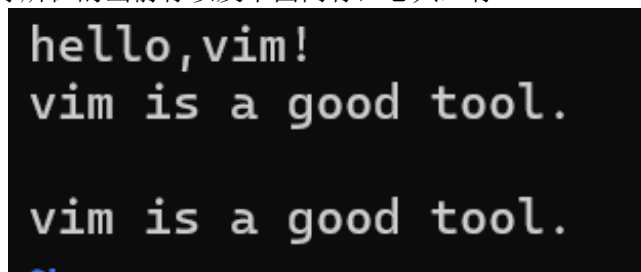


图 21: 反撤销

9.复制的学习：单行：将光标移动到要复制的行的任意位置。输入 yy 来复制整行。

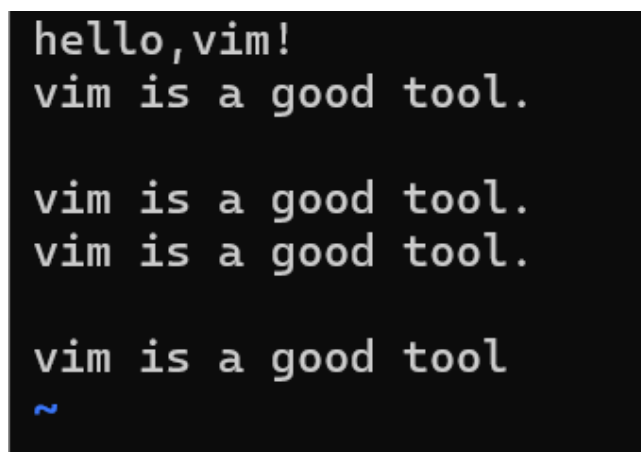
部分行：将光标移动到要开始复制的位置。进入可视化模式，可以按 v（用于字符模式）或 V（用于行模式）。使用光标键hjkl选择要复制的文本。一旦选择了文本，按 y 来复制选中的文本。

多个行：将光标移动到要复制的第一行的任意位置。输入数字yy，其中“数字”是您想要复制的行数。例如，3yy 会复制光标所在的当前行以及下面两行，总共三行。



```
hello,vim!  
vim is a good tool.  
vim is a good tool.  
~
```

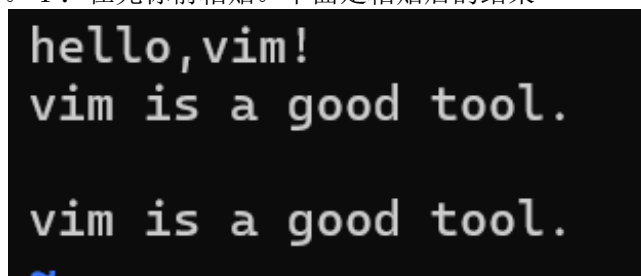
图 22: 复制一行后的结果



```
hello,vim!  
vim is a good tool.  
  
vim is a good tool.  
vim is a good tool.  
  
vim is a good tool  
~
```

图 23: 选择性复制后的结果

10. p: 在光标后粘贴。 P: 在光标前粘贴。下面是粘贴后的结果



```
hello,vim!  
vim is a good tool.  
vim is a good tool.  
~
```

图 24: 粘贴后的结果

### 3 解题感悟

Shell是一种强大的自动化工具。通过编写脚本,日常繁琐的任务会变得简单高效, Vim的也是一种更高效的工具。虽然学习过程有艰辛,但是新的习惯一旦形成,它将带来巨大的收益。

github路径您可以在查看项目的源代码:

[https://github.com/L-c-hang/homework\\_two/](https://github.com/L-c-hang/homework_two/)