系统开发工具基础实验报告 2

姓名：刘浩洋

学号：24040021022

班级：软件工程

实验日期：2025 年 9 月 5 日

一、实验目的

1. 掌握 Linux 系统中 Shell 的基本命令，包括文件、目录和路径操作；
2. 熟悉 Shell 脚本编程基础，包括变量、条件判断、循环等结构；
3. 掌握 Vim 文本编辑器的基本使用方法；
4. 能够独立编写并运行简单的 Shell 脚本；
5. 培养在虚拟机环境下使用 Ubuntu 进行系统开发工具实践的能力。

二、实验环境

- 操作系统：Windows 11
- 虚拟化平台：VMware Workstation Pro 17
- 虚拟机系统：Ubuntu 24.04.3 LTS（64 位）
- Shell 环境：Bash
- 编辑器：Vim

三、练习内容

本次实验主要包括：

- 文件与目录的创建、复制、移动、删除；

- 绝对路径与相对路径的使用；
- Shell 脚本中变量、条件判断、循环的使用；
- 使用 Vim 编辑脚本文件；
- 编写并运行综合 Shell 脚本。

四、20 个实例（shell 与 vim）

实例 1：创建实验目录结构

使用 mkdir 命令可以创建多级目录结构，便于组织和管理实验相关文件。该命令支持 -p 参数，可避免目录已存在时报错。

```
mkdir -p  /lab1/{scripts,data,docs}
cd  /lab1
touch data/file1.txt data/file2.txt
ls
tree
pwd
```

实例 2：文件复制与重命名

cp 命令用于复制文件或目录，mv 命令用于移动或重命名文件。结合通配符 * 可批量操作多个文件。

```
cp data/file1.txt scripts/
mv scripts/file1.txt scripts/test1.bak
cp data/*.txt scripts/
ls scripts/
rm data/file1.txt
ls data/
```

实例 3：切换目录与路径使用

Linux 中路径分为绝对路径和相对路径。cd 命令用于切换当前工作目录。使用.. 可返回上级目录， 代表家目录。

```
echo " 当前目录: $(pwd)"
cd /home/$(whoami)/lab1/docs
cd ..
cd scripts
cd
cd -
```

实例 4：查看与修改文件权限

Linux 文件具有读、写、执行三种权限。chmod 命令用于修改文件权限，可使用符号模式（如 +x）或数字模式（如 755）。

```
ls -l scripts/
chmod +x scripts/test1.bak
chmod 644 scripts/test1.bak
chmod -R 755 scripts/
ls -l scripts/
./scripts/test1.bak
```

实例 5：编写第一个 Shell 脚本

Shell 脚本以 #!/bin/bash 开头，表示使用 Bash 解释器执行。echo 命令用于输出文本信息，$(date) 可嵌入当前系统时间。

```
#!/bin/bash
echo " 欢迎使用 Shell 脚本! "
echo " 当前时间: $(date)"
echo " 实验环境: Ubuntu 24.04"
echo "Shell 版本: $(bash --version | head -1)"
echo " 当前用户: $(whoami)"
echo " 脚本执行完毕"
```

实例 6：Shell 变量定义与使用

Shell 中变量无需声明即可使用，变量名区分大小写，赋值时等号两侧不能有空格。使用 $ 变量名可引用变量值，变量可存储字符串、数字或命令结果。

```
#!/bin/bash
name=" 刘浩洋"
year=2025
school=" 软件工程"
echo " 姓名：$name"
echo " 年份：$year"
echo " 学院：$school"
```

实例 7：使用 read 读取用户输入

read 命令用于从标准输入读取一行内容，并赋值给指定变量。可配合 echo 提示用户输入，实现交互式脚本功能。

```
#!/bin/bash
echo " 请输入姓名："
read name
echo " 请输入学号："
read student_id
```

实例 8：Shell 数组操作

Shell 支持一维数组，使用括号定义。${array[@]} 表示数组所有元素，可通过 for 循环遍历数组内容，实现批量处理数据。

```
#!/bin/bash
fruits=(" 苹果" " 香蕉" " 橙子" " 葡萄")
echo " 水果列表：${fruits[@]}"
for f in "${fruits[@]}"; do
echo " - $f"
done
echo " 共 $fruits[@] 种水果"
```

实例 9：if 条件判断（数字）

if 语句用于根据条件执行不同分支。[ ] 内使用 -gt、-lt、-eq 等比较数字。elif 表示否则如果，fi 表示 if 结束。

```
#!/bin/bash
read num
if [ $num -gt 0 ]; then
echo "$num 是正数"
elif [ $num -lt 0 ]; then
echo "$num 是负数"
else
echo "$num 是零"
fi
```

实例 10：case 多分支选择

case 语句适用于多条件匹配场景。每个分支以) 结尾，;; 表示结束。* 表示默认匹配，esac 表示结束。

```
#!/bin/bash
echo " 请选择功能：1) 日期 2) 磁盘 3) 内存"
read choice
case $choice in
1) echo " 当前时间："; date ;;
2) echo " 磁盘使用："; df -h ;;
3) echo " 内存状态："; free -h ;;
*) echo " 无效选择" ;;
esac
```

实例 11：for 循环遍历文件

for 循环可用于遍历列表或文件。使用通配符 *.sh 可匹配所有 Shell 脚本。basename 命令提取文件名部分，便于格式化输出。

```
#!/bin/bash
for file in scripts/*.sh; do
if [ -f "$file" ]; then
echo " 发现脚本：$(basename $file)"
echo " 路径：$file"
echo " 大小：$(du -h "$file" | cut -f1)"
fi
done
```

实例 12：while 循环计数

while 循环在条件为真时持续执行。常用于计数、监控或等待操作。使用 sleep 可控制循环间隔时间。

```
#!/bin/bash
count=5
while [ $count -gt 0 ]; do
echo "$count..."
sleep 1
count=$((count - 1))
done
echo " 倒计时结束! "
```

实例 13：until 循环使用

until 循环在条件为假时持续执行，当条件变为真时停止。与 while 逻辑相反，适用于倒数等场景。

```
#!/bin/bash
num=10
until [ $num -lt 1 ]; do
echo " 倒数: $num"
num=$((num - 1))
sleep 0.5
done
echo " 发射! "
```

实例 14：函数定义与调用

函数用于封装可重复使用的代码块。定义格式为函数名 () { ... }。使用 $1、$2 等引用传入参数。

```
#!/bin/bash
greet() {
echo " 你好, $1"
echo " 欢迎来到 $2"
echo " 当前时间: $(date +"%H:%M:%S")"
}
greet " 张三" "Shell 实验"
greet " 李四" "Linux 世界"
```

实例 15：Vim 的三种基本模式

Vim 编辑器分为三种基本模式：命令模式、插入模式和底线命令模式。启动 Vim 后默认进入命令模式，此时按键被视为命令而非输入。按 i、a、o 等键可进入插入模式进行文本编辑。按: 进入底线命令模式可执行保存、退出等操作。

```
vim test.txt
--- 启动后为命令模式 ---
i
输入一些文本内容...
Esc
:
:wq
```

实例 16：进入与退出插入模式

在命令模式下，按 i 在光标当前位置进入插入模式。按 a 在光标后一个字符处开始输入。按 o 在当前行下方插入新行并进入插入模式。按 Esc 可随时返回命令模式。

```
vim edit.txt
i
这是第一行文本。
a
继续在同一行输入。
o
这是新插入的一行。
Esc
```

实例 17：替换模式与文本覆盖

Vim 支持两种替换方式：单字符替换和连续替换。按 r 可替换光标所在的一个字符，无需进入插入模式。按 R 进入连续替换模式，后续输入将逐个覆盖原有字符。此模式适合修改固定长度的文本内容。

```
vim replace.txt
i
hello world
Esc
f w
r W
0
R
HELLO
Esc
```

实例 18：保存与退出操作

在底线命令模式中，:w 用于保存文件。:q 用于退出，若已修改则需加! 强制退出。:wq 或 ZZ 可保存并退出。:q! 可放弃修改并强制退出。

```
vim save.txt
i
这是一段测试文本。
Esc
:w
:q
--- 若未保存可 ---
:wq
:q!
```

实例 19：光标移动与文本操作

Vim 提供丰富的光标移动快捷键。h、j、k、l 分别代表左、下、上、右。0 移动到行首，dd yy p u

```
vim move.txt
i
第一行
第二行
第三行
Esc
2G
dd
G
O
新行
yykP
```

实例 20：查找与全局替换

在命令模式下，/keyword 可向下搜索文本。?keyword 向上搜索，n 跳转下一个，N 跳转上一个。在底线命令模式中，:%s/old/new/g 可全局替换。加 c 可确认每次替换。

```
vim search.txt
i
apple banana apple
apple orange apple
Esc
/apple
n
N
:%s/apple/fruit/g
:%s/banana/mango/
:wq
```

五、实验结果

- 成功完成 20 个 Shell 与 Vim 操作实例；
- 所有脚本均能正确运行；
- 掌握了基本文件操作与 Shell 编程结构；

- 熟练使用 Vim 进行脚本编辑；
- 实验代码已整理并提交。

六、解题感悟

通过本次实验，我系统地学习了 Linux Shell 和 Vim 的基本使用。从创建目录到编写脚本，每一步都增强了我对命令行操作的理解。Shell 脚本的自动化能力令人印象深刻，而 Vim 作为经典编辑器，掌握其基本操作对后续学习至关重要。实验过程中遇到问题时，通过查阅资料和反复练习得以解决，提升了我的自主学习能力。今后将继续深入学习 Shell 高级特性，为系统开发打下坚实基础。

七、GitHub 链接

`https://github.com/ouc-lhy/for-lesson/tree/master/lesson2`