附件二：Linux 管道与重定向简明指南

一、概述

1.1 什么是重定向和管道?

在 Linux 中，命令的输入输出可以像水流一样连接、转向。这种“流”的管理方式主要有：

- **重定向（Redirection）**：把输入/输出“转向”文件；
- **管道（Pipe）**：把一个命令的输出作为下一个命令的输入。

1.2 标准输入输出流

Linux 中有三个标准数据流：

- **标准输入(stdin)**：文件描述符 `0`，键盘或输入文件（默认来自键盘）
- **标准输出(stdout)**：文件描述符 `1`，默认输出到终端
- **标准错误(stderr)**：文件描述符 `2`，错误信息输出通道，默认输出到终端

重定向本质上是对这些“文件描述符”的操作。

二、重定向基础语法

2.1 输出重定向（将输出写入文件）

命令	说明
`command > file`	将 stdout 重定向到文件（覆盖，清空原文件）
`command >> file`	将 stdout 追加到文件末尾
`command 2> file`	将 stderr 重定向到文件（覆盖）
`command 2>> file`	将 stderr 追加到文件末尾
`command &> file`	将 stdout 和 stderr 都重定向到文件
`command > file 2>&1`	同上，另一种写法

示例：

```
echo "hello" > file.txt       # 创建 file.txt 并写入 hello
echo "world" >> file.txt      # 向 file.txt 追加 world
```

2.2 输入重定向（从文件读取内容）

命令形式	说明
`< 文件名`	把文件内容作为输入

示例：

```
cat < file.txt               # 相当于 cat file.txt
```

2.3 错误重定向

命令形式	说明
`2> 文件名`	将错误输出重定向到文件
`2>> 文件名`	追加错误输出到文件
`1> 文件.txt 2>&1`	输出与错误都写入同一文件

示例：

```
ls /abc > out.txt 2> err.txt    # 把正常输出和错误分开存文件
```

三、特殊重定向技巧

3.1 丢弃输出（黑洞设备）

```
command > /dev/null    # 丢弃 stdout
command 2> /dev/null   # 丢弃 stderr
command &> /dev/null   # 丢弃所有输出
```

3.2 同时输出到文件和屏幕

```
command | tee file.txt   # 输出到文件同时显示在屏幕
command | tee -a file.txt  # 追加模式
```

3.3 合并 标准错误（stderr） 到 标准输出（stdout）

```
command 2>&1 | another_command
```

3.4 进程替换

```
diff <(command1) <(command2)  # 比较两个命令的输出
```

四、管道符 `|` 的使用

4.1 基础用法

- 管道 `|` 把前一条命令的“标准输出”传给后一条命令作为“标准输入”；
- 支持“命令接力”，形成数据流；
- 用于**过滤、分析、组合多个命令**。

```
ps aux | grep nginx          # 查找 nginx 进程
ls -l | less                 # 分页显示目录内容
cat file.txt | wc -l         # 统计行数
```

4.2 复杂管道实用案例

🌟 **示例1**：查看当前系统中消耗 CPU 最多的前五个进程

```
ps aux | sort -nrk 3,3 | head -n 5
```

说明：

- `ps aux`：列出所有进程；
- `sort -nrk 3,3`：按第3列（CPU%）排序，`-n` 数值，`-r` 降序；
- `head -n 5`：取前五行。

🌟 **示例2**：统计 `/etc/passwd` 文件中使用 bash 作为登录 shell 的用户数量

```
cat /etc/passwd | grep '/bin/bash' | wc -l
```

可简化为：

```
grep '/bin/bash' /etc/passwd | wc -l
```

🌟 **示例3**：找出当前目录下最大 3 个文件

```
find . -type f -exec du -h {} + | sort -hr | head -n 3
```

说明：

- `find . -type f`：查找所有文件；
- `du -h`：显示大小（人类可读）；
- `sort -hr`：按大小倒序；
- `head -n 3`：只取前三个。

🌟 **示例4**：实时监控 SSH 登录日志（筛选成功登录）

```
tail -f /var/log/auth.log | grep 'Accepted'
```

🌟 **示例5**：找出当前登录用户并统计数量

```
who | awk '{print $1}' | sort | uniq | wc -l
```

说明：

- `who`：列出当前登录用户；
- `awk '{print $1}'`：取出用户名字段；
- `sort | uniq`：去重；
- `wc -l`：统计唯一用户数。

🌟 **示例6**：查看某个目录下文件类型统计（比如文本 vs 二进制）

```
find . -type f | xargs file | cut -d: -f2 | sort | uniq -c | sort -nr
```

说明：

- `file` 判断类型；
- `cut` 提取类型字段；
- `uniq -c` 计数；
- `sort -nr` 按数量排序。

🌟 **示例7**：列出当前目录中以 `.log` 结尾的文件中包含“error”的行，并去重统计

```
cat *.log | grep -i error | sort | uniq -c | sort -nr
```

4.3 组合流程口诀

```
ps aux            ← 提供数据
| sort            ← 排序分析
| head/tail       ← 选取结果
| grep / awk      ← 筛选字段
| wc / uniq       ← 统计输出
```

或记为：

生产数据 → 筛选数据 → 格式转换 → 统计汇总

六、推荐练习任务

6.1 重定向练习

1. 保存文本到文件

```
echo "Linux is great." > demo.txt
cat demo.txt
```

2. 保存命令结果到文件

```
df -h > disk.txt              # 保存磁盘信息
cat disk.txt
```

3. 追加写入

```
echo "Learn it well." >> demo.txt
cat demo.txt
```

4. 错误输出重定向

```
ls /nonexistent 2> error.txt
cat error.txt
```

6.2 进阶管道

1. 查看内存占用前五名进程：

```
ps aux | sort -nrk 4,4 | head -n 5
```

2. 统计 `/var/log/syslog` 中出现“error”的次数（忽略大小写）：

```
grep -i 'error' /var/log/syslog | wc -l
```

3. 统计当前用户家目录下的文件总数：

```
find ~ -type f | wc -l
```

4. 显示磁盘使用前五个子目录：

```
du -h --max-depth=1 | sort -hr | head -n 5
```

5. 查看使用 bash 的用户

```
cat /etc/passwd | grep bash
```

6.3 同时查看并保存输出

使用 `tee` 命令：

```
ls -l | tee list.txt          # 输出到屏幕，同时保存到文件
```

七、小结口诀

```
>  是覆盖写文件，
>> 是追加再补充，
<  让文件做输入，
2> 单独存错误；
|  把前面输给后面，
组合命令威力大！
```