



中国海洋大学

# 系统开发工具实验报告



所在学院：信息科学与工程学部 专 业：计算机科学与技术

学 生 姓 名：陈一韬 学 号：23010022003

2024/8/27

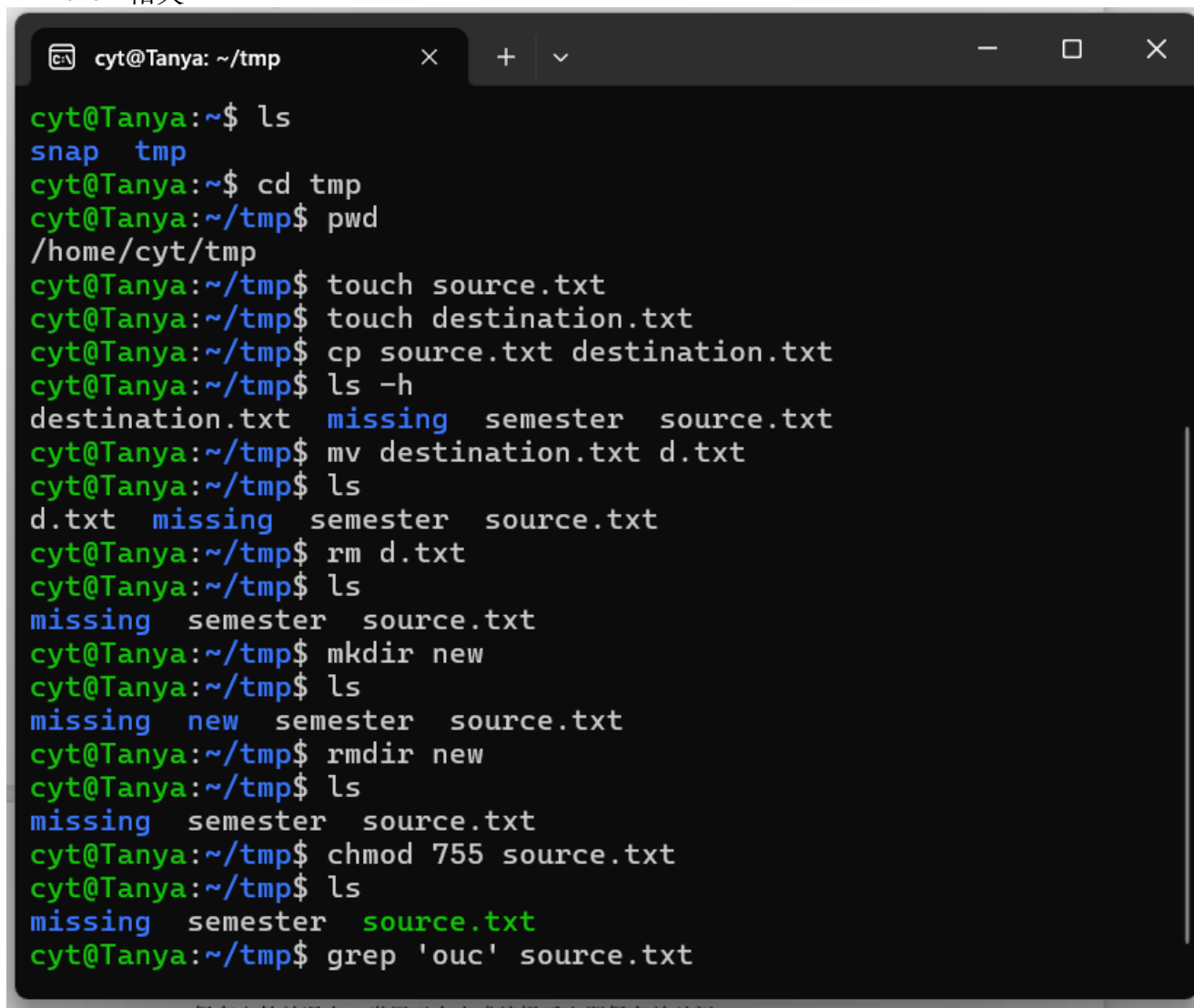
# 主 题：版本控制（Git）与 LaTeX 的使用

## 1 实验目的

1. 学习 shell 相关命令的使用
2. 学习使用正则表达式工具进行文本排版

## 2 课后实例练习

shell 相关



```
cyt@Tanya: ~/tmp
cyt@Tanya:~$ ls
snap tmp
cyt@Tanya:~$ cd tmp
cyt@Tanya:~/tmp$ pwd
/home/cyt/tmp
cyt@Tanya:~/tmp$ touch source.txt
cyt@Tanya:~/tmp$ touch destination.txt
cyt@Tanya:~/tmp$ cp source.txt destination.txt
cyt@Tanya:~/tmp$ ls -h
destination.txt missing semester source.txt
cyt@Tanya:~/tmp$ mv destination.txt d.txt
cyt@Tanya:~/tmp$ ls
d.txt missing semester source.txt
cyt@Tanya:~/tmp$ rm d.txt
cyt@Tanya:~/tmp$ ls
missing semester source.txt
cyt@Tanya:~/tmp$ mkdir new
cyt@Tanya:~/tmp$ ls
missing new semester source.txt
cyt@Tanya:~/tmp$ rmdir new
cyt@Tanya:~/tmp$ ls
missing semester source.txt
cyt@Tanya:~/tmp$ chmod 755 source.txt
cyt@Tanya:~/tmp$ ls
missing semester source.txt
cyt@Tanya:~/tmp$ grep 'ouc' source.txt
```

### 1. ls

ls 命令用于列出指定目录中的文件和子目录。它可以与多种选项结合使用，例如：

ls -l: 以长格式显示文件信息，包括权限、所有者、文件大小和修改时间。

`ls -a`: 显示所有文件，包括以点 (.) 开头的隐藏文件。

`ls -h`: 以人类可读的格式显示文件大小 (例如, KB、MB)。

## 2. `cd`

`cd` (change directory) 命令用于更改当前工作目录。使用方法如下:

`cd /path/to/directory`: 切换到指定路径的目录。

`cd ..`: 返回上一级目录。

`cd` : 切换到当前用户的主目录。

## 3. `pwd`

`pwd` (print working directory) 命令用于显示当前工作目录的完整路径。这对于确认所在的位置非常有用，特别是在进行文件操作的时候。

## 4. `cp`

`cp` 命令用于复制文件或目录。常用选项包括:

`cp source.txt destination.txt`: 将 `source.txt` 复制到 `destination.txt`。

`cp -r /source/directory /destination/directory`: 递归复制整个目录及其内容。

`cp -i`: 在覆盖文件时提示确认。

## 5. `mv`

`mv` 命令用于移动或重命名文件和目录。使用示例:

`mv oldname.txt newname.txt`: 将文件重命名。

`mv file.txt /path/to/directory/`: 将文件移动到指定目录。

`mv -i`: 在覆盖文件时提示确认。

## 6. `rm`

`rm` 命令用于删除文件或目录。常用选项包括:

`rm file.txt`: 删除指定文件。

`rm -r directory`: 递归删除目录及其内容。

`rm -i`: 在删除文件时提示确认。

## 7. `mkdir`

`mkdir` 命令用于创建新目录。使用示例:

`mkdir newdirectory`: 创建一个名为 `newdirectory` 的新目录。

`mkdir -p /path/to/directory`: 创建多级目录，如果父目录不存在则一并创建。

## 8. `rmdir`

`rmdir` 命令用于删除空目录。使用示例:

`rmdir emptydirectory`: 删除名为 `emptydirectory` 的空目录。

`rmdir -p /path/to/directory`: 递归删除空目录，只有在目录为空时才会删除。

## 9. `chmod`

`chmod` 命令用于更改文件或目录的权限。权限可以通过数字或符号表示:

chmod 755 file.txt: 将文件权限设置为所有者可读、可写、可执行, 组和其他用户可读和可执行。

chmod u+x script.sh: 为文件所有者添加执行权限。

## 10. grep

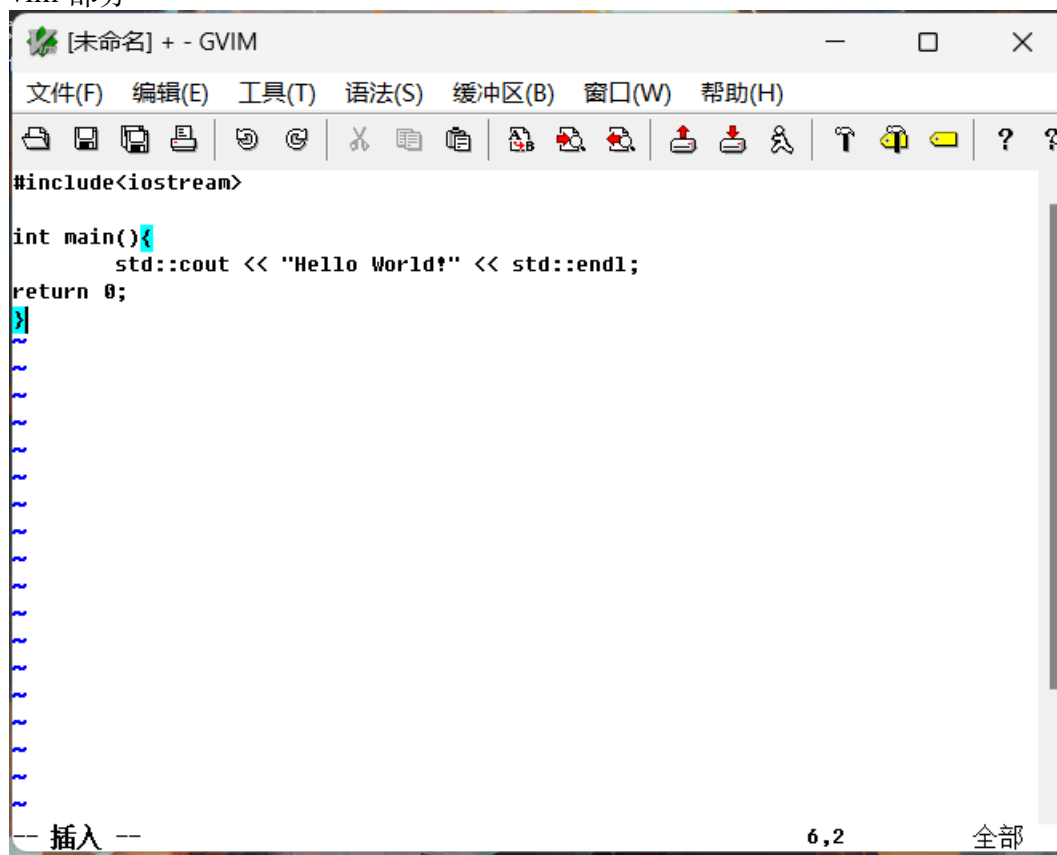
grep 命令用于在文件中搜索特定模式。它支持正则表达式, 常用选项包括:

grep 'pattern' file.txt: 在 file.txt 中搜索匹配 pattern 的行。

grep -i 'pattern' file.txt: 忽略大小写进行搜索。

grep -r 'pattern' /path/to/directory: 递归搜索指定目录中的所有文件。

## vim 部分



### 1. 基本导航命令:

h、j、k、l: 这些是 Vim 的基本导航键, 分别用于向左、向下、向上和向右移动光标。它们允许用户在不离开键盘的情况下快速浏览文本。

gg: 将光标移动到文件的开头, 便于快速访问文件的起始部分。

G: 将光标移动到文件的末尾, 适用于快速跳转到文件的结束位置。

### 2. 插入模式:

i: 在光标前进入插入模式, 允许用户在当前位置插入文本。

I: 在当前行的开头进入插入模式, 适合快速添加文本到行首。

a: 在光标后进入插入模式, 便于在当前字符后添加内容。

- A: 在当前行的末尾进入插入模式，适合快速在行尾添加文本。
- o: 在当前行下方插入新行并进入插入模式，便于快速添加新行。
- O: 在当前行上方插入新行并进入插入模式，适合在上方添加内容。

### 3. 保存和退出：

- :w: 保存当前文件的更改，确保所做的修改不会丢失。
- :q: 退出 Vim 编辑器，适用于完成编辑后关闭文件。
- :wq: 保存文件并退出，常用于在完成编辑后立即保存并关闭。
- :q!: 强制退出而不保存更改，适用于放弃所有未保存的修改。

### 4. 删除和修改：

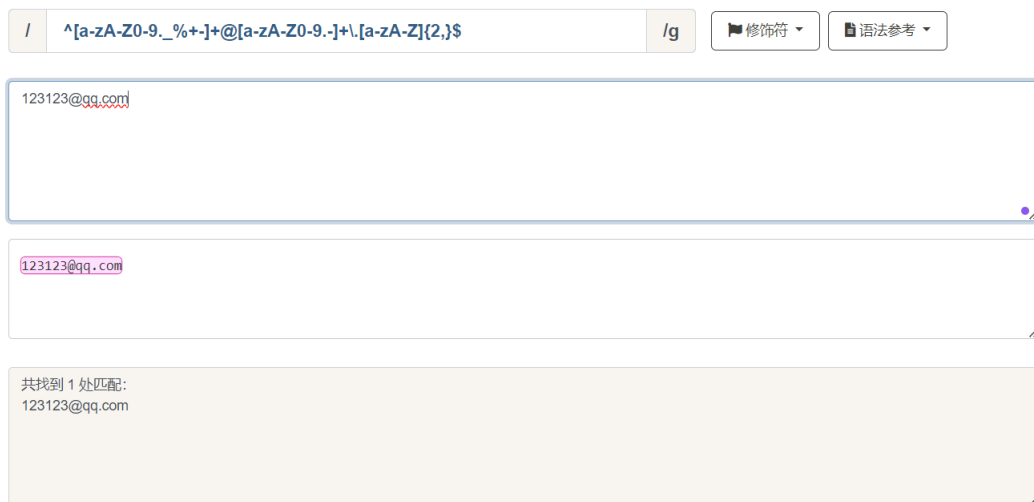
- x: 删除光标所在的字符，适合快速删除单个字符。
- dd: 删除当前行，便于快速移除整行内容。
- u: 撤销上一步操作，允许用户轻松回退到之前的状态。
- Ctrl + r: 重做上一步撤销的操作，便于恢复之前撤销的更改。

### 5. 查找和替换：

- /pattern: 向下查找指定的模式，适合快速定位文本。
- ?pattern: 向上查找指定的模式，便于在文件中反向搜索。
- n: 跳转到下一个匹配项，快速浏览查找结果。
- N: 跳转到上一个匹配项，适用于反向浏览查找结果。

## 数据整理（正则表达式）部分

### 1. 匹配电子邮件地址：



说明：这个正

则表达式用于匹配标准的电子邮件地址格式。它确保电子邮件地址以字母、数字或特定符号开头，后面跟着一个 @ 符号，接着是域名和顶级域名。

### 2. 匹配电话号码：

/
^[\+?[0-9]{1,3}?[-. ]?(?([0-9]{1,4}?)?[-. ]?[0-9]{1,4}[-. ]?[0-9]{1,9})?\$
/g
修饰符
语法参考

|123123123

123123123

共找到 1 处匹配:  
123123123

说明：这个正则表达式用于匹配国际格式的电话号码。它支持可选的国家代码、括号、空格和不同的分隔符（如短横线和点）。

### 3. 匹配 URL：

/
^https?:\V[^\s/\$.?\#\[\]\\*]\*\$
/g
修饰符
语法参考

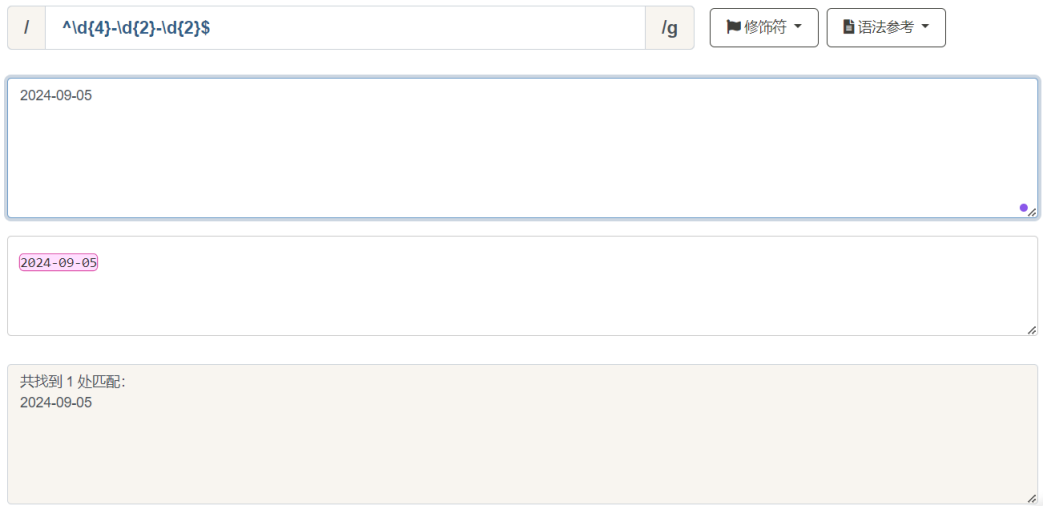
https://baidu.com

https://baidu.com

共找到 1 处匹配:  
https://baidu.com

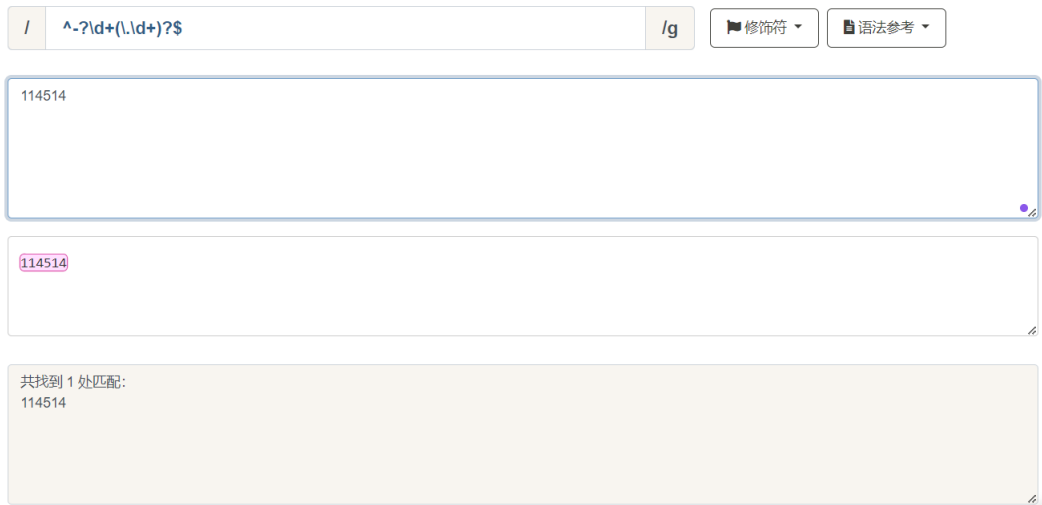
说明：这个正则表达式用于匹配以 http 或 https 开头的 URL。它确保 URL 的格式正确，包含有效的域名和路径。

### 4. 匹配日期（YYYY-MM-DD 格式）：



说明：这个正则表达式用于匹配日期格式为 YYYY-MM-DD。它确保年份为四位数字，月份和日期为两位数字。

#### 5. 匹配数字（整数或小数）：



说明：这个正则表达式用于匹配整数或小数。它支持可选的负号，整数部分可以是一个或多个数字，后面可以跟一个可选的小数部分。

### 3 解题感悟

在完成了这次关于 shell 操作、vim 编辑器使用以及正则表达式的实例操作后，我深感自己在 Linux 环境下的工作效率与技能水平有了显著提升。这次学习不仅填补了我在命令行操作上的知识空白，也让我对文本处理有了更加深入和系统的理解。以下是我对这次学习的一些感悟与体会：

Shell 作为 Linux 系统的核心交互界面，其强大的功能和灵活性让我印象深刻。通过一系列

命令的组合与管道操作，我能够轻松完成文件查找、排序、过滤、统计等复杂任务，这在以前看来几乎是不可想象的。学习过程中，我逐渐习惯了使用 `ls`、`cd`、`grep`、`awk`、`sed` 等常用命令，并学会了如何编写简单的 `shell` 脚本来自动化重复性的工作。这不仅提高了我的工作效率，也让我更加深刻地理解了 `Linux` 系统的运作机制。

`Vim` 作为一款高度可定制的文本编辑器，其强大的编辑能力和独特的操作模式让我大开眼界。`Vim` 的快捷键、模式切换、插件扩展等功能极大地丰富了我的编辑体验，让我在处理文本时更加得心应手。此外，`Vim` 设计的哲学——即用最少的按键完成最多的工作，也对我产生了深远的影响，让我开始反思并优化自己的日常操作习惯。

正则表达式无疑是这次学习中最让我兴奋的部分。它像一把锋利的手术刀，能够精准地切割、匹配和替换文本中的特定内容。通过学习正则表达式的基本语法和高级特性，我学会了如何编写复杂的搜索模式，以实现对文本数据的精确提取和转换。这种能力在数据清洗、日志分析、文本挖掘等领域具有广泛的应用价值，让我对未来可能遇到的文本处理任务充满了信心。

总的来说，这次课程的学习经历对我来说是一次宝贵的成长机会。它不仅让我掌握了 `shell` 操作、`vim` 使用和正则表达式等实用技能，更重要的是培养了我面对复杂问题时的逻辑思维能力和解决问题的能力。我相信，在未来的学习和工作中，这些技能将成为我不可或缺的助手，助我在 `Linux` 世界的探索之路上走得更远、更稳。同时，我也期待能够继续深入学习 `Linux` 系统的其他高级特性，不断提升自己的技术水平。

本次实验报告的 `github` 地址：<https://github.com/Shiny-Magikarp/-git>