Week1 Git 版本控制以及 LaTex 文档编辑

黄琬晴

September 2025

目录

github 网址：https://github.com/uuukyoo/systools

1 练习：Git 版本控制

1.1 配置与 github 的 ssh 连接

首先生成公钥

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools (master)
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/HWQ/.ssh/id_rsa):
Enter passphrase for "/c/Users/HWQ/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/HWQ/.ssh/id_rsa
Your public key has been saved in /c/Users/HWQ/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:SntdciI3+15D9l8GPiwO8CqnO7bn+GVoUpnP2k6+CpQ HWQ@LAPTOP-UMCJMM4S
The key's randomart image is:
+---[RSA 3072]----+
```

使用 cat 命令来查看公钥内容

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools (master)
$ cat /c/Users/HWQ/.ssh/id_rsa.pub
```

在 github 中添加该公钥

使用 ssh -T git@github.com 命令测试是否配置成功发现无法配置成功

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools (master)
$ ssh -T git@github.com
ssh: connect to host github.com port 22: Connection timed out
```

找到了解决方法在系统用户目录中的.ssh 文件夹中配置 config 文件内容如下

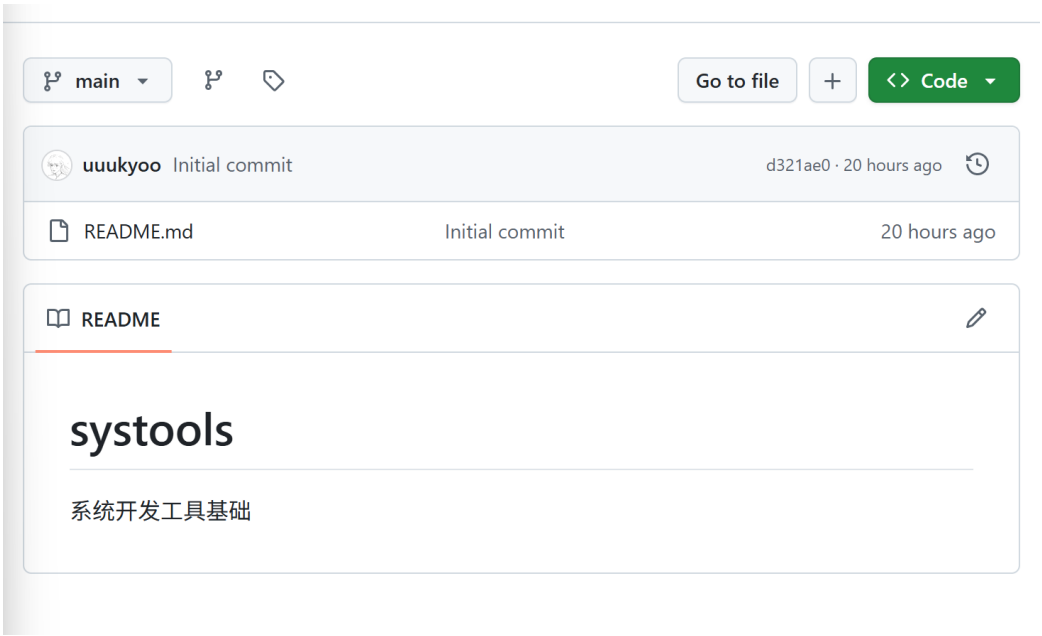
```
Host github.com
User 520002888@qq.com
Hostname ssh.github.com
PreferredAuthentications publickey
IdentityFile ~/.ssh/id_rsa
Port 443
```

再次测试，问题填写 yes，连接配置成功

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools (master)
$  ssh -T git@github.com
The authenticity of host '[ssh.github.com]:443 ([20.205.243.160]:443)' can't be
established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '[ssh.github.com]:443' (ED25519) to the list of known
 hosts.
Hi uuukyoo! You've successfully authenticated, but GitHub does not provide shell
 access.
```

1.2 管理 github 上的仓库

在 github 上新建一个仓库用于存放本课程相关代码及实验报告等



找到该仓库的 ssh 链接，使用 git clone 命令克隆到本地

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools (master)
$ git clone git@github.com:uuukyoo/systools.git
Cloning into 'systools'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```

cd 转到克隆到本地的仓库的命令下修改 README.md 文件之后，使用 git status 命令查看状态，可以看见此时显示已跟踪文件 README.md 文件的内容发生了变化，但还没有放到暂存区

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools (master)
$ cd systools

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools/systools (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

使用 git add 暂存，git commit -m 提交后再查看状态，暂存区中无新增或修改但未提交的文件

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools/systools (main)
$ git add .

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools/systools (main)
$ git commit -m "first. readme changed"
[main 6b2f0cb] first. readme changed
 1 file changed, 3 insertions(+), 2 deletions(-)

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools/systools (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

使用 git log 可以看见刚刚的提交记录

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools/systools (main)
$ git log
commit 6b2f0cbe5cbf9bcd3c8d9cd512081d63aab672d2 (HEAD -> main)
Author: hwq <520002888@qq.com>
Date:   Tue Sep 9 21:30:14 2025 +0800

    first. readme changed

commit d321ae052cbdfa3ea77f12bb4470dd8e30a3a3e9 (origin/main, origin/HEAD)
Author: HSei <134035906+uuukyoo@users.noreply.github.com>
Date:   Tue Sep 9 00:32:37 2025 +0800

    Initial commit
```

最后使用 git push 将修改推送到 github 上

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/systools/systools (main)
$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 32 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 391 bytes | 391.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:uuukyoo/systools.git
   d321ae0..6b2f0cb  main -> main
```

再次查看 github 上的仓库，可以看见 README 文件的变化

1.3 分支的新建，切换

在只有一个分支的仓库中新建分支 feature，并切换到 feature

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (master)
$ git branch feature

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (master)
$ git checkout feature
Switched to branch 'feature'
```

新建文件并在新分支 feature 上提交

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (feature)
$ echo "This is from feature branch" >> hello.txt
git add hello.txt
git commit -m "Update hello.txt in feature branch"
warning: in the working copy of 'hello.txt', LF will be replaced by CRLF the nex
t time Git touches it
[feature 09f14d8] Update hello.txt in feature branch
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
```

查看提交历史

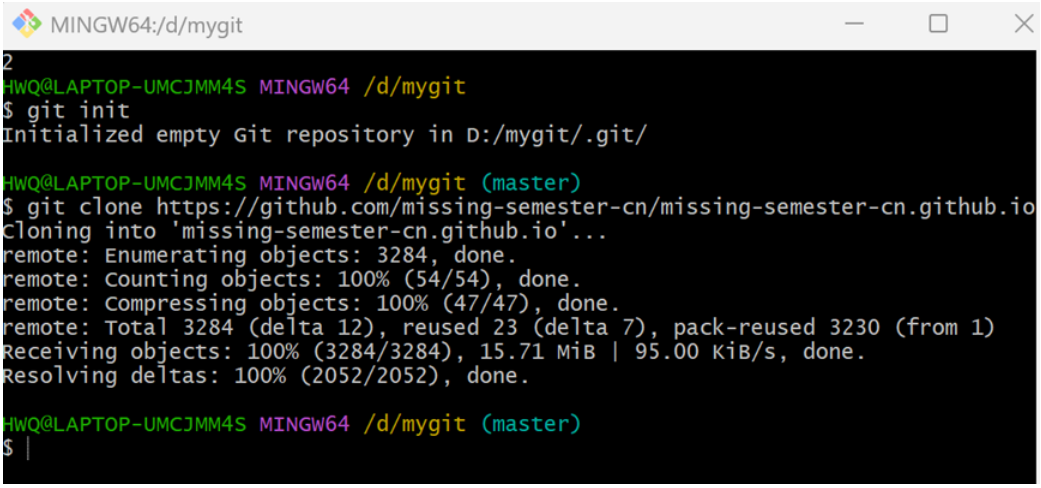
```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (feature)
$ git log
commit 09f14d82d1c47ce01037eb85f29d73069592716f (HEAD -> feature)
Author: hwq <520002888@qq.com>
Date:   Tue Sep 9 22:31:35 2025 +0800

    Update hello.txt in feature branch

commit c39e253c0268ec63569877c644ab5280949329c3 (master)
Author: hwq <520002888@qq.com>
Date:   Tue Sep 9 22:29:45 2025 +0800

    yes

commit 30230ad5931c52f70a2ca75714d84b8cdab07c77
Author: hwq <520002888@qq.com>
Date:   Fri Sep 5 10:38:21 2025 +0800

    first
```

切换到 master 分支，再次擦看历史，发现刚刚在 feature 上的提交消失了

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (feature)
$ git checkout master
Switched to branch 'master'

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (master)
$ git log
commit c39e253c0268ec63569877c644ab5280949329c3 (HEAD -> master)
Author: hwq <520002888@qq.com>
Date:   Tue Sep 9 22:29:45 2025 +0800

    yes

commit 30230ad5931c52f70a2ca75714d84b8cdab07c77
Author: hwq <520002888@qq.com>
Date:   Fri Sep 5 10:38:21 2025 +0800

    first
```

1.4 克隆本课程网站的仓库

使用 git clone 命令来克隆仓库



1.4.1 将版本历史化并进行探索

克隆了课程网站的仓库后，首先先转到克隆后仓库的目录下
然后使用 git log 命令查看版本历史，使用--all 参数来显示所有分支的提交情况，使用--graph 参数图表化，--ecorate 参数让 git log 显示每个 commit 的引用

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit (master)
$ cd missing-semester-cn.github.io

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git log --all --graph --decorate
*   commit 3e31dd9b0f6cc28665e516d76443cc2bde419dd0 (HEAD -> master, origin/mast
er, origin/HEAD)
|\  Merge: d893328 c0d8274
| | Author: Lingfeng_Ai <hanxiaomax@gmail.com>
| | Date:   Wed Aug 20 08:01:52 2025 +0800
| |
| |     Merge pull request #197 from yueneiqi/fix/shell-tools-terminology
| |
| |     Synchronize Chinese translations with English repository updates
| |
| * commit c0d8274f36f767df7fcd098d3d53e753ae6152eb
| | Author: seven_bear <26707756+yueneiqi@users.noreply.github.com>
| | Date:   Sat Aug 16 12:49:27 2025 +0800
| |
| |     Synchronize Chinese translations with English repository updates
| |
| |     This commit consolidates multiple translation synchronization updates:
```

1.4.2 是谁最后修改了 README.md 文件？

使用 git log 命令查看版本历史，参数--README.md 表示查找的是 README.md 文件的历史，-1 表示显示最后一条，pretty=full 表示显示完整的提交信息（包括作者信息）
由输出可以得知最后修改了 REASME.md 文件的人是 Zhenger233

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git log -1 --pretty=full -- README.md
commit fc93d7c0660cee7ac2dfeb23fd85f9ec741ff3a8
Author: Zhenger233 <2042712521@qq.com>
Commit: Zhenger233 <2042712521@qq.com>

    修 改 为 中 文 README
```

1.4.3 最后一次修改 _config.yml 文件中 collections: 行时的提交信息是什么？

使用 git blame 命令找到 _config.yml 文件每行的最后一次提交，并使用管道和 grep 命令找到 collections 行的修改信息，其中可以看见 commitid 为 a88b4eac

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git blame _config.yml | grep collections
a88b4eac (Anish Athalye  2020-01-17 15:26:30 -0500 18) collections:
```

使用 show 命令和刚刚查找到的 commit id，就可找到对应的提交信息为 Redo lectures as a collection

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git show --pretty=format:"%s" a88b4eac | head -1
Redo lectures as a collection
```

1.5 敏感信息处理

使用 Git 时的一个常见错误是提交本不应该由 Git 管理的大文件，或是将含有敏感信息的文件提交给 Git 。尝试向仓库中添加一个文件并添加提交信息，然后将其从历史中删除向仓库中提交文件 secret.txt，并添加提交信息”secret data” 现在使用 git log 命令可以查看到敏感文件 secret.txt 的提交记录

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ echo "this is secret data" > secret.txt

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git add secret.txt
warning: in the working copy of 'secret.txt', LF will be replaced by CRLF the ne
xt time Git touches it

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git commit -m "secret data"
[master db06fcf] secret data
 1 file changed, 1 insertion(+)
 create mode 100644 secret.txt

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git log -1
commit db06fcf6426b29fabc5599b0d1e3d5c209d1f805 (HEAD -> master)
Author: hwq <520002888@qq.com>
Date:   Fri Sep 5 09:18:05 2025 +0800

    secret data
```

使用 ls 命令页可以看见 secret.txt 文件在该仓库中使用命令 git filter-branch –force –index-filter ’git rm –cached –ignore-unmatch ./secret.txt’ –prune-empty –tag-name-filter cat – –all 清楚提交记录

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ ls
404.html      _2019/       about.md              index.md       static/
CNAME         _2020/       apple-touch-icon.png  lectures.html
Gemfile       _config.yml  favicon-16x16.png     license.md
Gemfile.lock  _includes/   favicon-32x32.png     robots.txt
README.md     _layouts/    favicon.ico           secret.txt

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git filter-branch --force --index-filter\
 'git rm --cached --ignore-unmatch ./secret.txt' \
 --prune-empty --tag-name-filter cat -- --all
WARNING: git-filter-branch has a glut of gotchas generating mangled history
         rewrites.  Hit Ctrl-C before proceeding to abort, then use an
         alternative filtering tool such as 'git filter-repo'
         (https://github.com/newren/git-filter-repo/) instead.  See the
```

命令结束后的显示如下

```
Rewrite 3e31dd9b0f6cc28665e516d76443cc2bde419dd0 (855/856) (15884 seconds passed
Rewrite db06fcf6426b29fabc5599b0d1e3d5c209d1f805 (856/856) (15897 seconds passed
, remaining 0 predicted)    rm 'secret.txt'

Ref 'refs/heads/master' was rewritten
Ref 'refs/remotes/origin/master' was rewritten
Ref 'refs/remotes/origin/dependabot/bundler/activesupport-6.0.6.1' was rewritten
Ref 'refs/remotes/origin/dependabot/bundler/addressable-2.8.1' was rewritten
Ref 'refs/remotes/origin/dependabot/bundler/nokogiri-1.14.3' was rewritten
Ref 'refs/remotes/origin/dependabot/bundler/tzinfo-1.2.10' was rewritten
WARNING: Ref 'refs/remotes/origin/master' is unchanged
Ref 'refs/remotes/origin/review' was rewritten

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
```

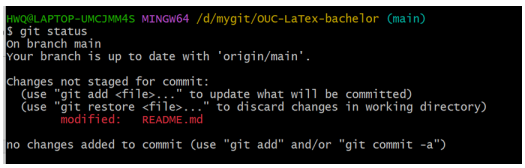
此时再查看提交记录和用 ls 查看文件发现提交记录和文件均已消失

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ git log -1
commit f3c80214b3de36e557721474a7fa0dbdda961e40 (HEAD -> master, origin/maste
origin/HEAD)
Merge: 61897e9 82b6a27
Author: Lingfeng_Ai <hanxiaomax@gmail.com>
Date:   Wed Aug 20 08:01:52 2025 +0800

    Merge pull request #197 from yueneiqi/fix/shell-tools-terminology

    Synchronize Chinese translations with English repository updates

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
$ ls
404.html      _2019/       about.md              index.md
CNAME         _2020/       apple-touch-icon.png  lectures.html
Gemfile       _config.yml  favicon-16x16.png     license.md
Gemfile.lock  _includes/   favicon-32x32.png     robots.txt
README.md     _layouts/    favicon.ico           static/

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/missing-semester-cn.github.io (master)
```

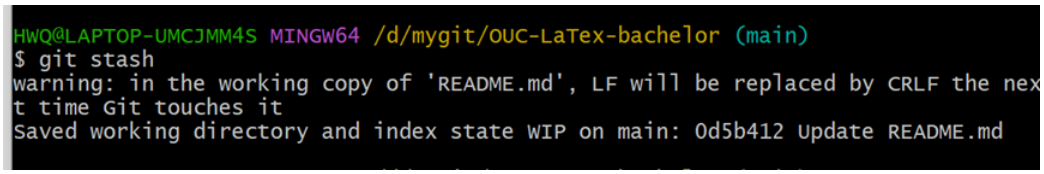
1.6 git stash 与 git stash pop

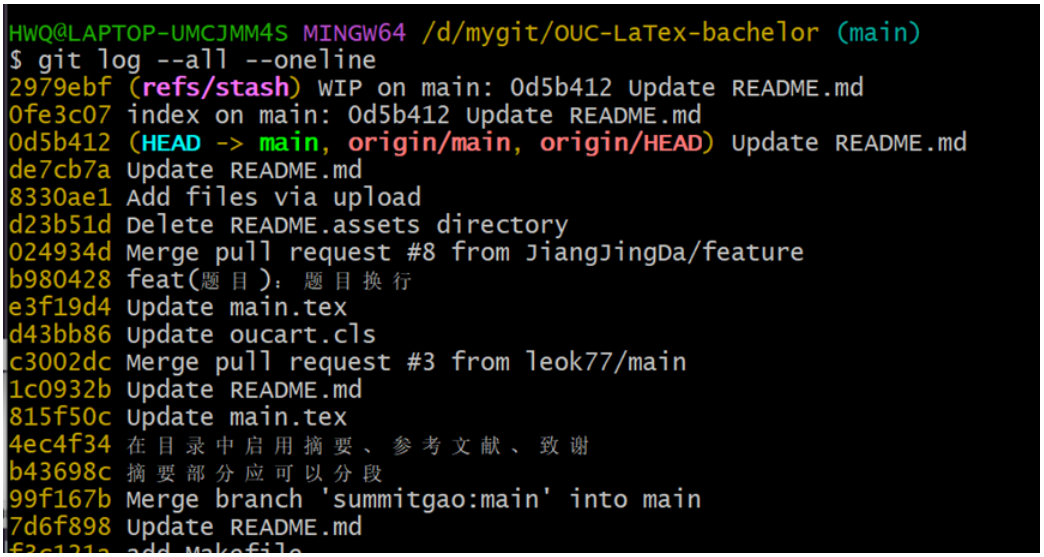
从 GitHub 上克隆某个仓库，修改一些文件。当您使用 git stash 会发生什么？当您执行 `git log --all --oneline` 时会显示什么？通过 git stash pop 命令来撤销 git stash 操作，什么时候会用到这一技巧？在 github 上找到一个仓库克隆

修改 README 后查看状态



git stash 后查看





git stash pop

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/OUC-LaTex-bachelor (main)
$ git stash pop
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (2979ebfdf0bfd28a4f6030beca53886a897451c9)
```

git shash 被撤销

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/mygit/OUC-LaTex-bachelor (main)
$  git log --all --oneline
0d5b412 (HEAD -> main, origin/main, origin/HEAD) Update README.md
de7cb7a Update README.md
8330ae1 Add files via upload
d23b51d Delete README.assets directory
024934d Merge pull request #8 from JiangJingDa/feature
b980428 feat(题目)：题目换行
e3f19d4 Update main.tex
d43bb86 Update oucart.cls
c3002dc Merge pull request #3 from leok77/main
1c0932b Update README.md
815f50c Update main.tex
4ec4f34 在目录中启用摘要、参考文献、致谢
b43698c 摘要部分应可以分段
99f167b Merge branch 'summitgao:main' into main
7d6f898 Update README.md
f3c121a add Makefile
44190dc 使用TeXiFy做格式检查
3192331 Update coveror.sty
1ef9c00 开题报告-引用文献
fce23d0 开题报告
```

你正在修改文件时，但突然要切到别的分支修改文件；你执行 `git stash` 把未完成的修改藏起来，切分支修改文件；修完回来，执行 `git stash pop`，继续原文件的修改。

1.7 使用 `/.gitconfig` 配置文件为命令创建别名

与其他的命令行工具一样，Git 也提供了一个名为 `/.gitconfig` 配置文件 (或 dotfile)。请在 `/.gitconfig` 中创建一个别名，使您在运行 `git graph`

时，您可以得到 `git log --all --graph --decorate --oneline` 的输出结果使用 vim 命令修改 `/.gitconfig` 配置文件

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (master)
$ vim ~/.gitconfig
```

修改如下

```
[user]
        name = hwq
        email = 520002888@qq.com
[alias]
    graph = log --all --graph --decorate --oneline
~
~
~
```

此时 git log –all –graph –decorate –oneline 和 git graph 的输出相同

```
HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (master)
$ git graph
* 5882ae0 secret data
* 30230ad (HEAD -> master) first

HWQ@LAPTOP-UMCJMM4S MINGW64 /d/git1 (master)
$ git log --all --graph --decorate --oneline
* 5882ae0 secret data
* 30230ad (HEAD -> master) first
```

2 练习：LaTex 文档编辑

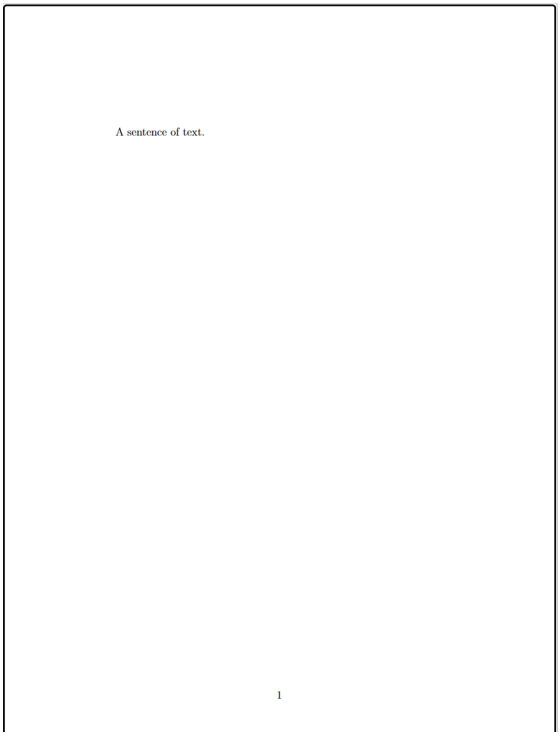
练习代码全部整合放在了文件 learn.tex 中

2.1 基本要素

输入

```
\documentclass[a4paper, 12pt]{article}
\begin{document}
    A sentence of text.
\end{document}
```

编译后展示如下



2.2 添加文档标题

可以使用\maketitle来为文档添加标题
在\begin{document}后输入以下文本

```
\title{My First Document}
\author{HUANG Wanqing}
\date{\today}
\maketitle
```

其中\tytle{}指定文本标题，\author{}指定作者名称，\date{}插入指定的时间，\today表示插入的是当前时间
编译后的显示结果如下

My First Document

HUANG Wanqing

September 9, 2025

A sentence of text.

2.3 章节

可以使用如下命令为文章分章节

```
\section{...}（分节）
\subsection{...}（小节）
\subsubsection{...}（小小节）
\paragraph{...}（段落）
\subparagraph{...}（分段落）
```

在花括号中写入的是章节的标题”A sentence of text.” 替换为：

```
\section{Introduction}
This is the introduction.

\section{Methods}
```

```
\subsection{Stage 1}
The first part of the methods.

\subsection{Stage 2}
The second part of the methods.

\section{Results}
Here are my results.
```

后，就得到了一个分小节的文档

My First Document

HUANG Wanqing

September 9, 2025

1 Introduction

This is the introduction.

2 Methods

2.1 Stage 1

The first part of the methods.

2.2 Stage 2

The second part of the methods.

3 Results

Here are my results.

2.4 创建标签

\label{labelname} 对章节创建标签。然后输入 \ref{labelname} 或者 \pageref{labelname} 来引用对应的章节在 Stag1 下使用\label{labelname}创建标签,在 Results 章节下输入Referring to section \ref{sec1} on page \pageref{sec1}编译结果如下

My First Document

HUANG Wanqing

September 9, 2025

1 Introduction

This is the introduction.

2 Methods

2.1 Stage 1

The first part of the methods.

2.2 Stage 2

The second part of the methods.

3 Results

Here are my results. Referring to section 2.1 on page 1

可以发现`\ref{labelname}`被替换成了标签所在的章节号,`\pageref{labelname}` 被替换成了标签所在的页号，而`\label{labelname}`在编译后是不显示的

2.5 生成目录

使用了分节命令后，再使用`\tableofcontents`就可以生成一个由文章的章节组成的目录在`\maketitle`之后输入

```
\pagenumbering{roman}
\tableofcontents
\newpage
\pagenumbering{arabic}
```

编译后会生成一个目录，目录的内容即为正文章节名，目录页的页号为罗马数字，正文新起一页，页号为阿拉伯数字目录页

My First Document

HUANG Wanqing

September 9, 2025

Contents

增加目录页后的正文页

1 Introduction

This is the introduction.

2 Methods

2.1 Stage 1

The first part of the methods.

2.2 Stage 2

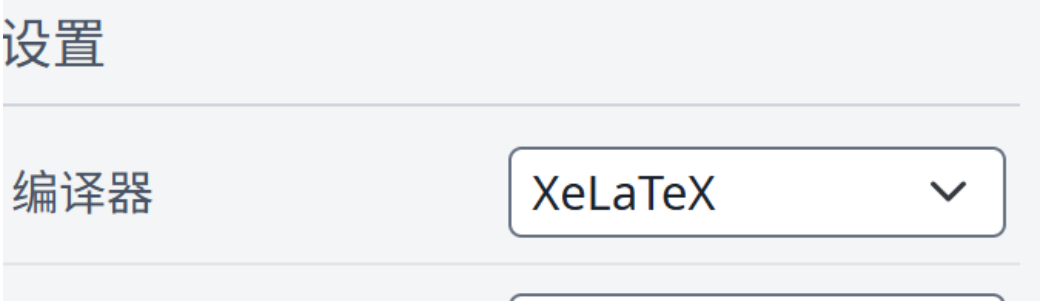
The second part of the methods.

3 Results

Here are my results. Referring to section 2.1 on page 1

2.6 中文字体支持

使用 CTex 宏包 (在文档的前导命令处添加`\usepackage[UTF8]{ctex}`), 并将编译器改为 xelatex



重新编译后，目录部分的字体已变为中文

My First Document

HUANG Wanqing

2025 年 9 月 9 日

目录

若使用 overleaf 默认的 pdflatex 则会产生编译错误

2.7 字体效果

使用如下命令可以为文本添加不同的字体效果

```
\textit{words in italics}
```

```
\textsl{words slanted}
\textsc{words in smallcaps}
\textbf{words in bold}
\texttt{words in teletype}
\textsf{sans serif words}
\textrm{roman words}
\underline{underlined words}
```

words in italics

words slanted

WORDS IN SMALLCAPS

words in bold

`words in teletype`

sans serif words

roman words

underlined words

如下，也可以为中文文本提供不同的字体效果

```
{\kaishu 这里是楷体显示}
{\songti 这里是宋体显示}
{\heiti 这里是黑体显示}
{\fangsong 这里是仿宋显示}
```

这里是楷体显示，
这里是宋体显示，
这里是黑体显示，
这里是仿宋显示。

2.8 彩色字体

使用`\usepackage[options]{package}`后可以使用`{\color{colorname}text}`来指定字体颜色使用`\colorbox{colorname}{text}`来指定文字背景色实践如下

```
{\color{red}红色}
{\color{green}绿色}
{\color{blue}蓝色}
{\color{yellow}黄色}
\colorbox{black}{\color{white}白色}
```

红色绿色蓝色黄色白色

2.9 字体大小

使用命令来调整字体大小:

```
我现在是正常大小，但是，我可以突然{\tiny 变小，然后} {\scriptsize 一点点地}
{\footnotesize 变大} {\small 变大} {\large 变大}
{\Large 再变大} {\LARGE 变得很大} {\huge 变得非常大}
```

我现在是正常大小，但是，我可以突然变小，然后 一点点地 变大 变大 变大 再变大 变得很大 变得非常大

2.10 列表

有序列表用\begin{enumerate}和\end{enumerate}框住,无序列表用\begin{itemize}\end{it住，用\item来表示一个条目
列表之间也可以互相嵌套

```
\begin{enumerate}
    \item 今天吃什么好呢
        \begin{itemize}
            \item 不喜欢吃
                \begin{enumerate}
                \item 茄子
                \item 酸菜
                \end{enumerate}
```

```
            \item 想吃烤冷面
        \end{itemize}
    \item 打个招呼
        \begin{itemize}
            \item 你好
            \item 你也好
        \end{itemize}
    \item 记得早点睡觉
\end{enumerate}
```

1. 今天吃什么好呢
    - 不喜欢吃
        (a) 茄子
        (b) 酸菜
    - 想吃烤冷面
2. 打个招呼
    - 你好
    - 你也好
3. 记得早点睡觉

2.11 表格

尝试画出下列表格：

Item	Quantity	Price($)
Nails	500	0.34
Wooden boards	100	4.00
Bricks	240	11.50

	Year		
City	2006	2007	2008
London	45789	46551	51298
Berlin	34549	32543	29870
Paris	49835	51009	51970

观察题目中给到的表格可以发现，第一个表格的第一列使用的是左对齐 (l)，第二列和第三列是右对齐 (r)，且在每两列之间有竖线，所以花括号中的值为 l|r|r；在第一行和第二行中间有横线 (使用`\hline`)，由观察结果可写出代码如下

```
\begin{tabular}{l|r|r}
    Item &Quantity &Price(\$)\\
    \hline
    Nails &500 &0.34 \\
    Wooden boards &100 &4.00 \\
    Bricks &240 &11.50 \\
\end{tabular}
```

第二个表格的第一列是左对齐 (l)，后三列居中 (c)，只在第一列和第二列的中间有竖线，所以花括号中的值为 l|ccc；后三列的一二行中有横线 (使用`\cline{2-4}`)，三四行有完整的横线 (使用`\hline`)，由观察结果写出代码如下

```
\begin{tabular}{l|ccc}
```

```
 &Year\\
\cline{2-4}
City &2006 &2007 &2008\\
\hline
London &45789 &46551 &51298\\
Berlin &34549 &32543 &29870\\
Paris &49835 &51009 &51970\\

\end{tabular}
```

编译得到的结果如下

Item	Quantity	Price($)
Nails	500	0.34
Wooden boards	100	4.00
Bricks	240	11.50

	Year		
City	2006	2007	2008
London	45789	46551	51298
Berlin	34549	32543	29870
Paris	49835	51009	51970

2.12 图表

将一张图片放在页面中间并添加标题和标签

```
\begin{figure}[h!]
\centering
\includegraphics[width=1\textwidth]{myfigure.jpg}
\caption{NO JUMP NO DIVE}
\label{image}
\end{figure}
图片见\pageref{image}页
```

图片见 26页

图 1: NO JUMP NO DIVE

2.13 显示特殊字符

单个的特殊字符可以使用转义字符反斜线来显示, 如:

```
\# \$ \% \^{} \& \_ \{ \} \~{}
```

$ % ^ & _ { } ~

反斜杠使用\textbackslash展示 \
使用\verb可以展示句间代码
使用
\begin{verbatim}
\end{verbatim} 可以展示段落代码
本实验报告许多地方都用到了这两种方法

3 实验心得

通过本次实验，我学会了基本的使用 git 来进行版本控制的方法，以及初步的使用 latex 进行文本编辑与排版实验报告刚开始是编写源代码编辑器的，写了一部分之后发现还可以用可视化编辑器，比源代码编辑器方便得多，让编写实验报告变得简单得多了