



中国海洋大学

实验报告

——Git与Latex的学习与应用

学号： 23060021010

姓名： 郭晓伟

班级： 23级软件工程五八班

1 实验要求

1.1 学习Git和Late的使用

1.2 完成4个课堂练习与20个与Git和Latex有关的实例

2 实验内容

2.1 Git的学习

2.1.1 Git是一个开源的分布式版本控制系统，用于高效地管理和跟踪代码的变化。它由Linus Torvalds于2005年开发，最初是为了更好地管理Linux内核的开发过程。Git具有许多强大的功能，适用于各种规模的项目，尤其在开源社区和企业级开发中广泛使用。

2.1.2 Git的关键特点和功能：

- 1.分布式版本控制
- 2.高效的分支管理
- 3.数据完整性
- 4.高效处理大项目
- 5.开源与社区支持

2.1.3 常用Git命令：

- git init：初始化一个新的Git仓库
- git clone：克隆一个远程仓库到本地
- git add：将文件的更改添加到暂存区
- git commit：将暂存区的更改提交到本地仓库
- git pull：从远程仓库拉取最新的更改并合并到本地
- git push：将本地提交推送到远程仓库
- git branch：列出、创建或删除分支
- git checkout：切换到指定的分支或提交

2.2 Latex的学习

2.2.1 LaTeX 是一种基于排版系统的文档编写工具，广泛应用于科学、工程、数学、计算机科学等领域的学术论文、技术文档、书籍和报告的撰写。

2.2.2 LaTeX的特点和优势：

- 1.高质量排版
- 2.结构化文档
- 3.可扩展性
- 4.跨平台
- 5.免费开源

3 实验中遇到的问题与解决方法

3.1 使用Latex撰写实验报告时，发现编译出来的PDF文档中出现部分文字无法显示或显示不全的问题

原因是内容太多而 LaTeX 无法自动分页，使用newpag命令手动添加分页，即可显示全部文字。

3.2 在向仓库中提交敏感信息时，无法提交并显示who you are?

这是由于没有登记个人信息，在命令行终端输入指令git config --global user.email "邮箱"、git config --global user.name "名称"，登记个人名称和邮箱之后即解决。

3.3 在使用Latex进行文本编辑时，无法显示中文

查阅资料后得知，Latex基础字体不包含中文，有两种方法，一为导入含有中文字体的宏包，二可以设置文章类型，采用经过国人开发的ctexbook、ctexart和ctexbeamer类型，这些类型自带了对中文的支持，且编译器要换成XeLaTeX。

3.4 即使加入了graphicx宏包也无法插入图片

想要插入图片需要将图片放在与tex文件同一目录中，或者通过语句graphicspath引入图片所在的文件夹，用includegraphics语句来表示要插入的图片、设置图片的长宽。

3.5 完成文本写作后,结果第一页是空白页，文本内容从第二页开始

第一页默认为封面页，需要添加封面。

4 实例练习

4.1 创建Git仓库

输入git status命令，查看是否为仓库，并输入git init命令，初始化git仓库

```
23241@welkin MINGW64 /e/development tool/lab1
$ git status
fatal: not a git repository (or any of the parent directories): .git

23241@welkin MINGW64 /e/development tool/lab1
$ git init
Initialized empty Git repository in E:/development tool/lab1/.git/
```

4.2 查看当前状态

输入git status指令，显示当前工作目录和暂存区的状态，提示仓库中有文件未追踪

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    report/

nothing added to commit but untracked files present (use "git add" to track)
```

4.3 添加文件

git add "文件名"，将文件添加到git仓库

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git add report/

23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   report/lab1.aux
    new file:   report/lab1.log
    new file:   report/lab1.pdf
    new file:   report/lab1.synctex.gz
    new file:   report/lab1.tex
    new file:   report/picture/1.png
```

4.4 提交文件

git commit -m "提交信息"，将文件提交到仓库中，并附上提交信息

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git commit -m "report commit"
[master (root-commit) a18ff74] report commit
35 files changed, 997 insertions(+)
create mode 100644 report/lab1.aux
create mode 100644 report/lab1.log
create mode 100644 report/lab1.pdf
create mode 100644 report/lab1.synctex.gz
create mode 100644 report/lab1.tex
create mode 100644 report/picture/1.png
create mode 100644 report/picture/11.png
create mode 100644 report/picture/12.png
create mode 100644 report/picture/131.png
create mode 100644 report/picture/132.png
create mode 100644 report/picture/14.png
create mode 100644 report/picture/15.png
create mode 100644 report/picture/16.png
create mode 100644 report/picture/17.png
create mode 100644 report/picture/18.png
create mode 100644 report/picture/19.png
create mode 100644 report/picture/2.png
```

4.5 查看提交日志

git log, 显示当前分支的提交历史, 包括提交ID、作者、日期和提交信息

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git log
commit a18ff7414bc1c367349bd0618e966dfdfac3798b (HEAD -> master)
Author: xwelkin <18373570985@163.com>
Date: Thu Aug 29 20:59:03 2024 +0800

    report commit

23241@welkin MINGW64 /e/development tool/lab1 (master)
$
```

4.6 创建仓库分支并查看

git branch命令, 查看 Git 仓库的分支情况, git branch a 则为创建一个名为a的分支, 当前分支仍为主分支

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git branch a

23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git branch
a
* master
```

4.7 切换分支

输入git checkout a命令，切换到a分支，也可以在创建分支的同时，直接切换到新分支，命令为git checkout -b

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git checkout a
Switched to branch 'a'

23241@welkin MINGW64 /e/development tool/lab1 (a)
$ git branch
* a
  master

23241@welkin MINGW64 /e/development tool/lab1 (a)
$
```

4.8 合并分支

切换到master分支，然后输入git merge a命令，将a分支合并到master分支

```
23241@welkin MINGW64 /e/development tool/lab1 (a)
$ git checkout master
Switched to branch 'master'

23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git merge a
Already up to date.

23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git branch
  a
* master
```

4.9 删除其他分支

输入git branch -d a命令，删除a分支。有时通过git branch -d命令出现删除不了的现象，例如分支a的代码没有合并到主分支等，这时可以通过命令git branch -D进行强制删除

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git branch -d a
Deleted branch a (was a18ff74).
```

4.10 关联远程仓库

git remote add origin "远程仓库地址" 命令，关联远程仓库，其中origin为远程仓库的名字

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git remote add origin https://github.com/xwelkin/lab1.git
```

4.11 本地代码推到远程仓库

如果我们本地的代码有了更新，为了保持本地与远程的代码同步，我们就需要把本地的代码推到远程的仓库，使用git push origin master命令

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git push origin master
Enumerating objects: 39, done.
Counting objects: 100% (39/39), done.
Delta compression using up to 32 threads
Compressing objects: 100% (38/38), done.
Writing objects: 100% (39/39), 3.82 MiB | 1.52 MiB/s, done.
Total 39 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/xwelkin/lab1.git
 * [new branch]      master -> master
```

4.12 远程代码拉到本地

如果我们远程仓库的代码有了更新，同样为了保持本地与远程的代码同步，我们就需要把远程的代码拉到本地，使用git pull origin master命令

```
23241@welkin MINGW64 /e/development tool/lab1 (master)
$ git pull origin master
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 961 bytes | 73.00 KiB/s, done.
From https://github.com/xwelkin/lab1
 * branch                master      -> FETCH_HEAD
  a18ff74..eb090d7  master      -> origin/master
Updating a18ff74..eb090d7
Fast-forward
 README.md | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
```

4.13 Latex实现中文

采用文章类型如经过国人开发的ctexbook、ctexart和ctexbeamer类型，自带中文支持，并且编译器换为XeLaTeX

```
\documentclass{ctexart}

\begin{document}

你好LaTeX

\end{document}
```

你好 LaTeX

4.14 实现标题与作者等

可以用title、author、date来设置，日期中选择today会自动输出编译当天的日期。它们只能放在导言区，然后通过maketitle展现出来

```
\documentclass[utf8]{ctexart}

\title{你好LaTeX}
\author{郭晓伟}
\date{\today}

\begin{document}

\maketitle

你好LaTeX

\end{document}
```

你好 LaTeX

郭晓伟

2024 年 8 月 29 日

你好 LaTeX

4.15 实现文章段落

章节分三阶，section、subsection、subsubsection，在它们后面的花括号中填写题目，加入*则可以消去前面对应的序数

```
\documentclass[utf8]{ctexart}

\title{你好LaTeX}
\author{郭晓伟}
\date{\today}

\begin{document}

\maketitle

你好LaTeX

\section{第一章}

\section*{第一节}

\section{第一小节}

\end{document}
```

你好 LaTeX

郭晓伟

2024 年 8 月 29 日

你好 LaTeX

1 第一章

第一节

2 第一小节

4.16 插入图片

插入图片需要使用graphicx宏包，使用includegraphics，可以在width=后设置宽度

```

1 \documentclass[utf8]{ctexart}
2 \usepackage{graphicx}
3
4 \title{你好LaTeX}
5 \author{郭晓伟}
6 \date{\today}
7
8
9 \begin{document}
10 \maketitle
11 你好LaTeX
12
13 \section{第一章}
14
15 \section*{第一节}
16
17 \section{第一小节}
18
19 \includegraphics[width = 6cm]{s1}
20
21 \end{document}

```

你好 LaTeX

郭晓伟

2024 年 8 月 29 日

你好 LaTeX

1 第一章

第一节

2 第一小节



4.17 插入代码

调用listings宏包，并且在正文中使用lstlisting指令框住所想插入的代码，后面的中括号框内的语言选项帮助关键字高亮

```

\documentclass[utf8]{ctexart}
\usepackage{graphicx}
\usepackage{listings}

\begin{document}

\includegraphics[width = 6cm]{s1}

\begin{lstlisting}[language=python]
print('Hello,World!')
\end{lstlisting}

\end{document}

```



print('Hello , World!')

4.18 注释

Latex可以实现单行注释和多行注释

```
1 \documentclass[utf8]{ctexa
2 \usepackage{graphicx}
3 \usepackage{listings}
4
5 %这是注释
6
7 \iffalse
8 这是注释
9 注释
10 \fi
11
12 \begin{document}
13
14 \includegraphics[width = 6
15
16 \begin{lstlisting}[languag
17
18 print('Hello,World!')
19
```

4.19 插入超链接

导入宏包url，使用url+链接

```

4 \usepackage{url}
5
6 %这是注释
7
8 \iffalse
9 这是注释
10 注释
11 \fi
12
13
14
15 \begin{document}
16
17 \includegraphics[width = 6cm]{s1}
18
19
20
21 \begin{lstlisting}[language=python]
22
23 print('Hello,World!')
24
25 \end{lstlisting}
26
27 \url{https://github.com/xwelkin/lab1.git}
28

```



```

print('Hello,World!')
https://github.com/xwelkin/lab1.git

```

4.20 数学公式

输入数学公式需要导入宏包amsmath，使用equation

```

\usepackage{amsmath}

\begin{document}

\includegraphics[width = 6cm]{s1}

\begin{equation}
a^2+b^2=c^2
\end{equation}

```



$$a^2 + b^2 = c^2$$

(1)

5 实验收获与感悟

通过学习Git和LaTeX，我深刻体会到版本控制在项目管理中的重要性。Git不仅能够有效记录和更改项目版本，还能帮助解决冲突。我掌握了Git的分支管理和合并技巧，明白了它在团队协作中作为沟通桥梁的重要作用，这能够大大减少沟通成本并提升工作效率。这次学习让我进一步适应了Git的独特工作方式和思维模式。

LaTeX凭借其卓越的排版能力，可以生成美观、专业的文档，尤其是其强大的数学公式编辑功能，能够轻松呈现复杂的数学表达式，提升论文的可读性并确保符合学术规范。此外，我还学会了如何在网上查找合适的论文模板，并将其tex文件移植过来，极大地提高了写论文的效率。这为我未来撰写数学建模比赛论文奠定了坚实基础。

虽然学习Git和LaTeX的过程充满挑战，但它们带给我的收获和启发却是巨大的。Git让我掌握了版本控制和团队协作的核心要素，而LaTeX则让我能够以更专业和高效的方式编写文档。这两种工具不仅提升了我的技术能力，也让我更深入地理解了项目管理和学术研究的本质。我相信在未来的学习和工作

中，它们将继续发挥重要作用，并为我提供有力的支持。

Github仓库链接: <https://github.com/xwelkin/lab1.git>