实验一 Git和Markdown基础 班级： 21计科04

学号：B2021302412

姓名： 刘巍

Github地址：https://github.com/smile-lv/smile.git

实验目的 Git基础，使用Git进行版本控制 Markdown基础，使用Markdown进行文档编辑 实验环境 吉特 VSCode VSCode插件 实验内容和步骤 第一部分 实验环境的安装 安装git，从git官网下载后直接点击可以安装：git官网地址 从Github克隆课程的仓库：课程的仓库地址，运行git bash应用（该应用包含在git安装包内），在命令行输入下面的命令（命令运行成功后，课程仓库会存放默认在Windows的用户文件夹下） git clone https://github.com/zhoujing204/python_course.git 如果你在使用命令时遇到SSL错误，请运行下面的git命令（这里假设你的Git使用了默认安装目录）： git clone

git config --global http.sslCAInfo "C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt" 或者运行下面的命令:

git config --global http.sslVerify false 如果遇到错误：，请运行下面的命令重新指定git的安全证书：error setting certificate file

git config --global --unset http.sslCAInfo git config --global http.sslCAInfo "C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt" 该仓库的课程材料后续会有更新，如果需要更新课程材料，可以在本地课程仓库的目录下运行下面的命令：

git pull 在本地的仓库内容有更新后，可以运行下面的命令，将本地仓库的内容和远程仓库的内容同步：

git push origin main 注册Github账号或者Gitee帐号，创建一个新的仓库，使用上面同样的方法将该仓库clone到本地，用于存放实验报告和实验代码，使用和命令保持远程仓库和本地仓库的同步。git pullgit push 安装 VScode，下载地址：Visual Studio Code 安装下列VScode插件 吉特伦斯 Git 图 Git 历史 降价多合一 降价预览增强 降价 PDF 自动打开降价预览 粘贴图像 Markdownlint 第二部分 Git基础 教材《Python编程从入门到实践》P440附录D：使用Git进行版本控制，按照教材的步骤，完成Git基础的学习。

第三部分 learngitbranching.js.org 访问 learngitbranching.js.org，如下图所示完成Main部分的Introduction Sequence和Ramping Up两个小节的学习。

Learngitbranching.js.org

上面你学习到的git命令基本上可以应付百分之九十以上的日常使用，如果你想继续深入学习git，可以：

继续学习learngitbranching.js.org后面的几个小节（包括Main和Remote） 在日常的开发中使用git来管理你的代码和文档，用得越多，记得越牢 在git使用过程中，如果遇到任何问题，例如：错误删除了某个分支、从错误的分支拉取了内容等等，请查询git-flight-rules 第四部分 Markdown基础 查看Markdown cheat-sheet，学习Markdown的基础语法

使用Markdown编辑器（例如VScode）编写本次实验的实验报告，包括实验过程与结果、实验考查和实验总结，并将其导出为 PDF格式 来提交。

如何将markdown文件转换为pdf格式的文件?

安装vscode插件Markdown PDF，安装后重启vscode，打开markdown文件，按下，输入，回车即可导出pdf文件。Ctrl+Shift+PMarkdown PDF: Export (pdf) 使用Google Chrome浏览器，在Github网站或者Gitee网站打开你

的仓库，浏览你的markdown文件，按下，选择，选择为，点击即可导出pdf文件。Ctrl+P打印目标打印机另存为PDF保存 实验过程与结果 请将实验过程中编写的代码和运行结果放在这里，注意代码需要使用markdown的代码块格式化，例如Git命令行语句应该使用下面的格式：

Git命令

显示效果如下：

git commit git checkout git merge git rebase git branch git checkout -f Head~ git reset git revert git cherry-pick git tag

实验考查 请使用自己的语言回答下面的问题，这些问题将在实验检查时用于提问和答辩，并要求进行实际的操作。

什么是版本控制？使用Git作为版本控制软件有什么优点？

版本控制是指对软件开发过程中各种程序代码、配置文件及说明文档等文件变更的管理，是软件配置管理的核心思想之一。 1.高效：Git的版本控制功能非常强大，可以高效地跟踪和管理代码的变更历史。它能够记录每一次代码修改的内容和时间，以及每一次提交的更改，使得开发人员可以轻松地追溯问题的来源，定位修改历史，从而提高开发效率 。 2.并行开发：Git的分支管理功能使得多个开发人员在同一个项目上可以并行开发，然后再合并到主分支，加速了项目进度，提高了协作效率。

3.丰富的托管服务：Git有众多的托管服务，例如GitHub，使得代码托管变得非常方便。这些托管服务也提供了强大的协作功能，例如代码审查、问题跟踪等。

4.提高代码质量：通过Git进行版本控制，可以更好地管理代码的更改历史，使得代码的可读性和可维护性得到极大的提高。同时，在需要回滚代码时，可以轻松地找到之前的版本，并将其还原。

5.安全性：Git的版本控制可以避免代码覆盖和丢失。每个开发者都有自己的本地仓库，可以在本地进行修改和测试，然后再将修改提交到中央仓库。这种分布式版本控制系统提高了代码的安全性和稳定性。

如何使用Git撤销还没有Commit的修改？如何使用Git检出（Checkout）已经以前的Commit？（实际操作） git reset
git revert

Git中的HEAD是什么？如何让HEAD处于detached HEAD状态？（实际操作） git中的HEAD表示当前指向分支的指针，始终指向最新的commmit. git chevkout commit

什么是分支（Branch）？如何创建分支？如何切换分支？（实际操作） 分支（Branch）在Git中指的是一个独立的开发线路，用于进行特定的开发任务或版本控制。在分支中，你可以进行任意次数的提交操作，并且不会影响主分支（通常是master分支）的稳定性。当你在分支中进行了一些开发任务后，可以将分支合并回主分支中，以更新主分支的内容。 创建分支 git branch +name(分支名字) 切换分支 git checkout +name(分支名字)

如何合并分支？git merge和git rebase的区别在哪里？（实际操作） 合并分支 git merge+name

it merge是将两个分支的修改合并在一起，形成一个新的提交，默认情况下会提交合并中的修改内容。而git rebase则是将当前分支的commit复制到目标分支上，然后删除原始分支上的已提取的commit，因此不会形成新的提交。

如何在Markdown格式的文本中使用标题、数字列表、无序列表和超链接？（实际操作）

"#+标题"表示一级标题 "##+标题"表示二级标题，根据#的增加标题的级数不断增加

使用数字来表示列表的顺序。例如

1. 第一项
2. 第二项
3. 第三项

使用“-”或“*”来表示无序列表。例如

- 第一项
- 第二项
- 第三项

使用“链接文本”的方式来创建超链接。例如 "[百度]+(https://www.baidu.com)"

总结一下这次实验你学习和使用到的知识，例如：编程工具的使用、数据结构、程序语言的语法、算法、编程技巧、编程思想。

这此实验在https://learngitbranching.js.org/?locale=zh_CN上学习了关于git版本控制的一些基本的命令，能够初步使用git的相关命令，但是缺乏一些实战，还得去联系应用到实际的操作中去