实验一 Git和Markdown基础

班级： 21计科1

学号： 20190202222

姓名： 陈乐

Github地址：https://github.com/lechen20/python_course

实验目的

1. Git基础，使用Git进行版本控制
2. Markdown基础，使用Markdown进行文档编辑

实验环境

1. Git
2. VSCode
3. VSCode插件

实验内容和步骤

第一部分 实验环境的安装

1. 安装git，从git官网下载后直接点击可以安装：git官网地址
2. 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，课程仓库会默认存放在Windows的用户文件夹下）

```
git clone https://github.com/zhoujing204/python_course.git
```

如果你在使用`git clone`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 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
```

3. 注册Github账号，创建一个新的仓库，用于存放实验报告和实验代码。
4. 安装VScode，下载地址：Visual Studio Code
5. 安装下列VScode插件

- GitLens
- Git Graph
- Git History
- Markdown All in One
- Markdown Preview Enhanced
- Markdown PDF
- Auto-Open Markdown Preview
- Paste Image
- 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。

第四部分 Markdown基础

查看Markdown cheat-sheet，学习Markdown的基础语法

实验过程与结果

Introduction Sequence

1. Git Commit

```
git commit          #创建新的提交记录
git commit
```

2. Git Branch

```
git branch bugFix    #创建新的分支
git checkout bugFix  #切换到这个分支
```

3. Git Merge

```
git branch bugFix    #创建一个新的名字叫bugFix分支
git checkout bugFix  #切换到这个分支
git commit           #提交一次
git checkout main    #切换回到main分支
git commit           #提交一次
git merge bugFix     #合并分支
```

4. Git rebase

```
git branch bugFix    #创建一个新的名字叫bugFix分支
git checkout bugFix  #切换到这个分支
git commit           #提交一次
git checkout main    #切换回到main分支
git commit           #提交一次
git checkout bugFix  #切换到这个分支
git rebase main      #合并分支
```

Ramping Up

1. 分离HEAD

```
git checkout C4     #由HEAD->main->C4，变成HEAD->C4
```

2. 相对引用^

```
git checkout C4^     #向上移动一个提交记录
```

3. 相对引用2~（可强制移动分支）

```
git branch -f bugFix HEAD~2  #将bugFix分支强制指向HEAD的第2级父提交
git branch -f main C6        #将main分支强制指向C6
git checkout HEAD^           #切换到HEAD的上一个提交记录
```

4. 撤销变更

```
git reset HEAD~1       #退回上一个(删去C2，保留C1)—本地分支
git checkout pushed    #切换到这个分支
git revert HEAD        #引入C2'相当于C1的状态—远程分支
```

实验考查

1. 什么是版本控制？使用Git作为版本控制软件有什么优点？
   版本控制是一种可以记录文件或代码在不同时间点的变化，并且能够追踪、管理和恢复这些变化的系统。Git作为一种分布式版本控制系统，具有强大的分支管理、高效的性能、完整的历史记录和强大的冲突解决功能。
2. 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）

要撤销还没有 commit 的修改，可以使用 git checkout 命令 a. 查看所有的Commit记录：git log b. 检出某个Commit：git checkout

3. Git中的HEAD是什么？如何让HEAD处于detached HEAD状态？（实际操作）
   HEAD 是一个指向当前所在分支或提交的指针。它指示当前工作目录中的内容来自于哪个分支或提交。要将 HEAD 置于 "detached HEAD" 状态，可以使用以下命令：git checkout <commit号或分支名>
4. 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）是用于在代码库中并行开发和管理不同版本的功能的一种机制。每个分支都代表了代码库的一个独立的版本线，可以在不影响其他分支的情况下进行修改和提交。 要创建分支，可以使用以下命令：git branch <分支名> 要切换到某个分支，可以使用以下命令：git checkout <分支名>
5. 如何合并分支？git merge和git rebase的区别在哪里？（实际操作）
   合并分支可以使用git merge命令和git rebase命令 区别： git merge 会创建一个新的合并提交，将两个分支的修改合并到一起。这样可以保留原始分支的完整历史记录，并且合并后的结果可以在两个分支之间进行更好的对比和回溯。但是，合并提交会增加提交历史的复杂性。 git rebase 会将一个分支的修改应用到另一个分支上，并创建一个新的提交历史。这样可以保持提交历史的线性，并减少合并提交的数量。但是，由于修改是应用到目标分支上的，所以会改变目标分支的提交历史，并可能导致冲突。
6. 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：使用 # 符号来表示标题，# 的数量表示标题的级别。例如，要创建一个一级标题，可以在文本前面添加一个 # 符号，如 # 标题。要创建一个二级标题，可以添加两个 # 符号，如 ## 标题。 数字列表：使用数字和 . 符号来表示数字列表。在每个列表项前面添加一个数字和 . 符号，并在列表项之间使用空行分隔。例如： 1. 列表项1 2. 列表项2 3. 列表项3 无序列表：使用 -、+ 或 * 符号来表示无序列表。在每个列表项前面添加一个符号，并在列表项之间使用空行分隔。例如： - 列表项1 - 列表项2 - 列表项3 超链接：使用 链接文本 的格式来表示超链接。将要显示的链接文本放在方括号中，链接地址放在圆括号中。例如： GitHub 这将会创建一个指向 GitHub 的超链接，链接文本显示为 "GitHub"。

实验总结

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。

1. Git 基础 初步学习使用git进行版本控制，了解到版本控制是一种可以记录文件或代码在不同时间点的变化，并且能够追踪、管理和恢复这些变化的系统，还了 解到Git作为一种分布式版本控制系统，具有强大的分支管理、高效的性能、完整的历史记录和强大的冲突解决功能。 通过学习和闯关： 基础：git commit、git branch、git merge、git rebase 高级：分离HEAD、相对引用^、相对引用2~（可强制移动分支）、撤销变更 实操学习，使我更加形象的理解一些基础命令，为我接下来熟练使用git和代码管理打下良好的基础。例如分支管理：Git的分支管理功能非常强大，学习Git基础命令后，您可以使用git branch命令创建和查看分支，使用git checkout命令切换分支，使用git merge命令合并分支；版本回退：可以使用git reset命令回退到之前的版本，这样可以轻松地查看代码的演变过程，比较不同版本之间的差异，并且可以回滚到之前的版本。除此之外还有git revert用于撤销某次提交的修改，它会创建一个新的提交，将之前提交的修改撤销掉，这个新的提交会保留之前提交的历史记录，因此可以避免历史记录的混乱等等还有非常多且能够提供效率的命令，是需要我接下来继续学习的。

2. Markdown基础
   虽然是初次学习和使用Markdown，但已经感受到它的简洁易读易写的特点了，在初次使用过程中，因为不够熟练，还是需要查使用资料，不过我相信经 过持续的使用，Markdown会成为我之后常用的文本编辑语言，我会继续探寻它的优点和功能。