实验一 Git和Markdown基础

班级： 21计科1

学号： B20210302115

姓名： 文嘉

Github地址：https://github.com/Da-BuLiu/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

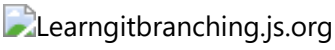
- GitLens
- Git Graph
- Git History
- Markdown All in One
- Markdown Preview Enhanced
- Markdown PDF
- Auto-Open Markdown Preview
- Paste Image
- markdownlint

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。


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。

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：

第一题

git命令

```
git commit
git commit
```

结果

好样的!!! 你用 2 条命令通过了这一关； 我们的答案要用 2 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “Git Branch”?

第二题

git命令

```
git branch bugFix
git checkout bugFix
```

结果

好样的!!! 你用 2 条命令通过了这一关； 我们的答案要用 2 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “Git Merge”?

第三题

git命令

```
git checkout -b bugFix
git commit
git checkout main
git commit
git merge bugFix
```

结果

好样的!!! 你用 5 条命令通过了这一关； 我们的答案要用 5 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “Git Rebase”?

第四题

git命令

```
git checkout -b bugFix
git commit
git checkout main
git commit
git checkout bugFix
git rebase main
```

结果

好样的!!! 你用 6 条命令通过了这一关； 我们的答案要用 6 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “分离 HEAD”?

第五题

git命令

```
git checkout C4
```

结果

好样的!!! 你用 1 条命令通过了这一关； 我们的答案要用 1 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “相对引用（^）”?

第六题

git命令

```
git checkout bugFix^
```

结果 好样的!!! 你用 1 条命令通过了这一关； 我们的答案要用 1 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “相对引用2（~）”?

第七题

git命令

```
git branch -f main C6
git checkout HEAD~1
git branch -f bugFix HEAD~1
```

结果

好样的!!! 你用 3 条命令通过了这一关； 我们的答案要用 3 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “撤销变更”?

第八题

git命令

```
git reset HEAD~1
git checkout pushed
git revert HEAD
```

结果

好样的!!! 你用 3 条命令通过了这一关； 我们的答案要用 3 条命令。

太强了！你的答案符合标准答案甚至更好。

要不要试试下一关 “Git Cherry-pick”?

注意：不要使用截图，Markdown文档转换为Pdf格式后，截图可能会无法显示。

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。

1. 什么是版本控制？使用Git作为版本控制软件有什么优点?

版本控制是一种记录文件或项目在不同时间点的变化，并且能够方便地回溯、恢复、合并和管理这些变化的技术。

Git是一种分布式版本控制系统，它具有以下优点：
快速：Git的设计目标是处理大型项目和速度要求高的操作。
分布式：每个开发者都可以拥有完整的代码仓库，可以离线工作，而不依赖于中央服务器。
强大的分支和合并：Git的分支管理非常强大，可以轻松创建、合并和删除分支。
安全性：Git使用哈希值作为文件的标识，保证数据的完整性。
灵活性：Git允许开发者按照自己的需求配置工作流程和工作方式。

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）

```
git checkout -- <file>
这将撤销对指定文件的修改，并还原到上一次Commit的状态。

git checkout <commit-hash>
这将切换到指定的Commit，并将工作目录和暂存区回溯到该Commit的状态。
```

3. Git中的HEAD是什么？如何让HEAD处于detached HEAD状态？（实际操作）

在Git中，HEAD是一个指向当前所在分支或Commit的指针。

可以使用以下命令将HEAD处于detached HEAD状态：
`git checkout <commit-hash>`
这将切换到指定的Commit，并将HEAD指向该Commit，而不是所在分支。

4. 什么是分支（Branch）？如何创建分支？如何切换分支？（实际操作）

分支是Git中用于并行开发和管理不同版本的功能。创建分支可以使用以下命令：
`git branch <branch-name>`
这将创建一个新的分支。

切换分支可以使用以下命令：
`git checkout <branch-name>`
这将切换到指定的分支。

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

合并分支可以使用以下命令：
`git merge <branch-name>`
这将将指定的分支合并到当前分支。合并后，Git会自动尝试将两个分支的修改合并到一起。

`git merge`和`git rebase`的区别在于合并的方式不同于：
`git merge`会创建一个新的提交，将两个分支的修改合并起来。这样可以保留分支的历史记录，但可能会导致分支历史变得复杂。
`git rebase`会将当前分支的修改“衍合”到目标分支上，创建一个线性的提交历史。这样可以保持历史记录的整洁，但会改变提交的顺序。

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）

标题使用井号(#)来表示，井号的个数表示标题的级别，例如：
```
# 标题1
## 标题2
### 标题3
```
数字列表使用数字后跟一个英文句点来表示，例如：
```
1. 第一项
2. 第二项
3. 第三项
```
无序列表使用减号(-)、加号(+)或星号(*)来表示，例如：
```
- 项目1
- 项目2
- 项目3
```
超链接使用方括号和圆括号来表示，方括号中是链接的显示文本，圆括号中是链接的目标地址，例如：
`[点击这里](http://www.baidu.com)`

实验总结

通过本次实验我学习了如何使用git命令，学到了Markdown的一些基础语法，并且学习了如何使用vscode，同时增加了对git的认识。