实验一 Git和Markdown基础

班级： 21计科04

学号： B20210302406

姓名： 陈健

Github地址：https://github.com/ruoyu88/git_practice.git

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"
```

或者运行下面的命令:

```
git config --global http.sslVerify false
```

如果遇到错误：`error setting certificate file`，请运行下面的命令重新指定git的安全证书：

```
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
```

3. 注册Github账号或者Gitee帐号，创建一个新的仓库，例如：https://gitee.com/zj204/python_task.git，使用下面的命令将新建的仓库clone到本地：

```
git clone https://gitee.com/zj204/python_task.git
```

如果已经关联了远程仓库，显示结果如下：

```
origin  https://github.com/zhoujing204/python_course.git (fetch)
origin  https://github.com/zhoujing204/python_course.git (push)
```

如果还没有关联远程仓库，可以使用你创建的远程仓库的地址和下面的命令，添加你要关联的远程仓库：

```
git remote add gitee https://gitee.com/zj204/python_task.git
```

接下来准备好你的远程仓库账号的邮箱地址和密码，使用下面的命令下载远程仓库的内容更新本地仓库：

```
git pull gitee main
```

运行下面的命令，将本地仓库的内容同步到远程仓库：

```
git push gitee main
```

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。

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`Ctrl+Shift+P`，输入`Markdown PDF: Export (pdf)`，回车即可导出pdf文件。
- 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`Ctrl+P`，选择`打印`，选择`目标打印机`为`另存为PDF`，点击`保存`即可导出pdf文件。

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 init
git add .
git status
git commit -m "first commit"

显示效果如下：

```
git init
git add .
git status
git commit -m "first commit"
```

如果是Python代码，应该使用下面代码块格式，例如：

def add_binary(a,b):
return bin(a+b)[2:]

显示效果如下：

```
def add_binary(a,b):
    return bin(a+b)[2:]
```

代码运行结果的文本可以直接粘贴在这里。

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？
   版本控制是一种在开发的过程中用于管理我们对文件、目录或工程等内容的修改历史，方便查看更改历史记录，备份以便恢复以前的版本的软件工程技术。简单来说就是用于管理多人协同开发项目的技术。
   优点：1.克隆库比较方便，容易获取资料和方便在不同地点或主机上进行代码管理
   2.支持离线修改和提交代码
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）
   首先可以使用git status命令，然后系统会给出提交和撤销的方法，一般使用git restore（可以在后面在文件名撤销对指定文件的修改）命令即可撤销还没有commit的修改。
   使用git checkout+已经commit的分支名即可回到该分支。
3. Git中的HEAD是什么？如何让HEAD处于detached HEAD状态？（实际操作）
   HEAD相当于指针，HEAD指向当前分支上最近一次提交记录。
   使用命令切换分支即可让HEAD处于分离状态，
   如git checkout <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）

分支实际上是指向更改快照的指针。
创建分支的命令：git branch <分支名>
切换分支的命令：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）
   本地分支的合并：
   在分支一完成操作后，先用git checkout master命令，切换到本地的master分支，然后再用git merge <分支一名>命令即可完成合并。
   区别：git merge是将分支的所有parent节点及其所有祖先合并起来，也就是说包含了对代码库的所有修改记录。
   而git rebase是取出一系列提交记录，“复制”它们，然后在另外一个地方逐个放下去，此时被“复制”的分支还存在，而合并的是分支的副本。
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）
   "="表示一级标题，"-"表示二级标题。使用"#"可以表示1-6级标题，如
   #一级标题
   ##二级标题
   以此类推。
   数字列表使用数字加上"."号来表示，如"1.第一项""2.第二项".
   无序列表使用"*","+"或"-"作为标记，这些标记后要加一个空格，然后再填写内容，如

```
* 第一项
* 第二项

+ 第一项
+ 第二项

- 第一项
- 第二项
```

   超链接：

```
[链接名称](链接地址)或者<链接地址>
```

   也可以通过变量设置一个链接，变量赋值在文档末尾进行。如

```
这是bus链接

[bus]: 网址
```

实验总结

本次实验我学会了初步使用markdown进行python语言和git命令的使用，掌握了git的一些基础命令，然后对GitHub这个平台也有了一定的了解，能够更好的管理代码和运用GitHub平台。