实验一 Git和Markdown基础

班级： 21计科3班

学号： B20210302318

姓名： 莫扬

Github地址：https://github.com/13428554811yang

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。

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]
- (https://github.com/k88hudson/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：

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：

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？

   用于管理和跟踪软件项目、文档或任何类型文件的系统和方法。它的主要目标是记录文件的更改历史

   优点：分布式架构 高效 分支和合并支持 开源免费

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）

   先查看状态 git status

   撤销修改 git checkout filename

   查看commit历史 git log

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

   HEAD它表示当前所在的分支或提交

   先查看 git log

   进入Detached HEAD状态 git checkout

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：用于开发和管理不同代码线的独立副本。每个分支代表了项目代码的不同状态，允许开发人员同时进行不同的工作而不干扰彼此

   创建分支：git branch

   切换分支：git branch 分支名

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

   git merge和git rebase合并

   git merge：

   ```
   git checkout main

   # 合并功能分支到目标分支
   git merge feature-branch
   ```

   - 是一种将一个分支的更改合并到另一个分支的方法。

- 合并操作会创建一个新的合并提交（Merge Commit），该提交将两个分支的更改合并在一起，并保留了两个分支的完整历史。

git rebase：

```
# 切换到功能分支
git checkout feature-branch

# 对功能分支进行 rebase 到目标分支
git rebase main
```

- 是一种重新应用提交的方法，通常用于将一个分支的更改移动到另一个分支上。
- Rebase 操作会将当前分支上的提交复制到目标分支的末端，然后将当前分支设置为目标分支，使得提交历史线性化。

2. 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）

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。

了解了git的一些指令和功能