* + 计算机专业的本次读书实践周的实施方案

拟：1. git与代码版本管理、学习笔记（必做）；2. 读书笔记（选做）

具体任务：

1、寻找合适的资料进行学习，在本地安装配置git，创建本地仓库；

2、注册远程仓库账号（GitHub、Gitee……），并创建至少一个公开的远程仓库。

2.1 选择可以公开的本地代码推送到远程仓库；

2.2 至少提交三次代码，观察每次提交的变更。

……

1. 建一个公开仓库，写本次实践任务的学习笔记。笔记包括：学习资料来源、链接等；实践流程；遇到的困难及解决方法；心得等部分。欢迎推荐书籍给老师。

成果收集：

各班拟以excel形式收集远程git账号地址，并检查公开仓库、学习笔记、读书笔记可访问性，体现在excel中；

* 大二同学按照计算机网络课程要求完成项目，并更新学习笔记到git。
* 大三的同学按照操作系统课程要求完成项目，并更新学习笔记到git。
* 大四同学将自己的毕设代码更新到git，并分享给导师。如果不适合公开，可以设置为私有仓库。如果不适合上网，请在读书笔记中注明。
* 已经使用过git的同学，可以在上学期的书单中继续选择书目进行阅读，并在git上更新读书笔记。书单地址：<https://docs.qq.com/doc/DWFNlbE5ld3FQVE9v> 欢迎推荐合适的书籍。

本次实践目的：方便评估学生到毕业时整个大学四年的代码工作量。

前期学习资料来源：

1、<https://blog.csdn.net/NAMELZX/article/details/118891788> # **git 简易的命令行入门教程、常用命令**

2、<https://www.cnblogs.com/dc-s/p/18857953> # **Git 完整教程：初学者分步指南**

3、DeepSeek

4、[【Git】Git 教程（三） —— SSH 方式链接 Git 与 GitHub（图文详解）\_git ssh true-CSDN博客](https://blog.csdn.net/weixin_43848614/article/details/107132455)

实践流程：

1. 下载Git。从官网安装Git并按照指引完成勾选。
2. 已经有Github账号，因此只用去创建一个个人仓库。
3. 本地git准备：

（1）、git config --global user.name "Your Name"

git config --global user.email [email@example.com](mailto:email@example.com)

在Git Bash中使用这两个命令创建自己的连接相关配置

（2）、生成SSH key

ssh-keygen -t rsa -C [youremail@example.com](mailto:youremail@example.com)

这里有个问题，使用rsa创建ssh密钥会有问题，可以选择

ssh-keygen -t ed25519 -C [xxxx@xxx.com](mailto:xxxx@xxx.com)区别在于协议

输入这个命令后会弹出“Enter file in which to save the key”这样的，你可以选择给你的密钥文件重新命名，也可以直接回车使用默认的命名（id\_ed25519或者id\_rsa，在于你生成是选择的是、哪个命令）。之后会让我们给密钥添加密码，当然也可以不输入。

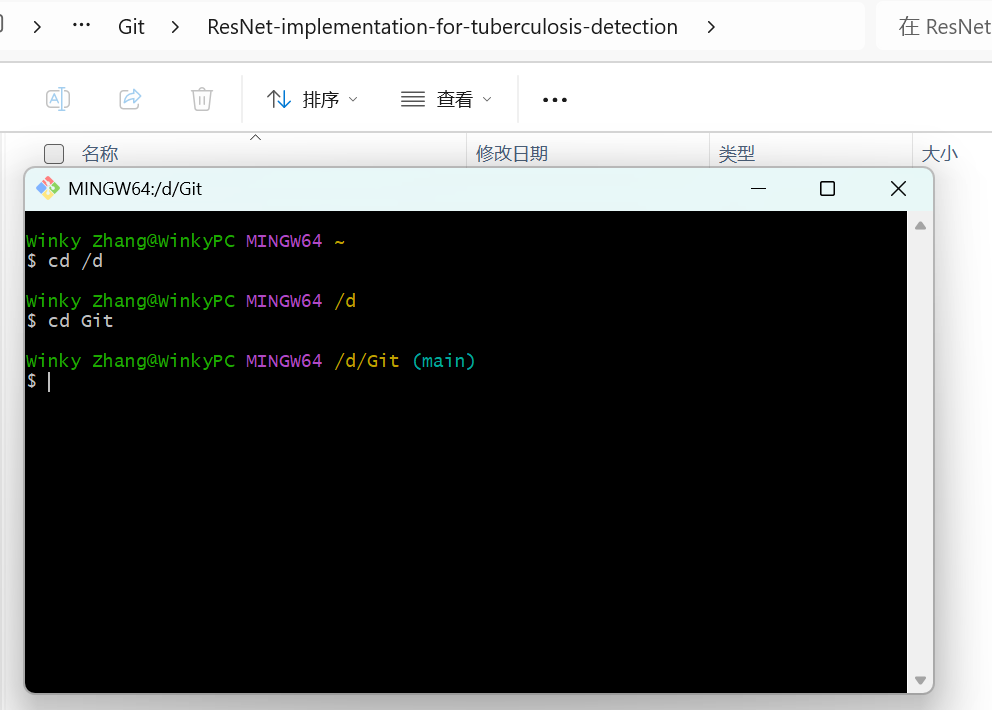
（3）、查找生成的私钥和公钥

前往C>user>username>.ssh文件夹里面找刚刚生成的密钥文件，复制公钥文件内容，私钥不要泄露。

公钥就是带.pub结尾的文件，将.pub文件打开并将里面的内容全部复制。

（4）、打开Github并找到settings，在左侧列表选择SSH and GPG keys，添加新的ssh key。给这个ssh连接命名，并将刚才复制的公钥全文复制到下方文本框内点击创建。创建后会发现shh的钥匙图标是灰色的，是因为本地还没有和github产生链接。

（5）、找到自己想要管理的代码文件夹，并在git bash里面使用cd命令将文件路径指向这个文件夹。



再输入git init命令将该文件夹变成Git可以管理的仓库。输入后文件夹内会出现一个.git隐藏文件夹.

1. 建立本地和远程仓库的连接

（1）、在git bash里面输入ssh -T [git@github.com](mailto:git@github.com)测试能否链接。只有返回“Hi XXX! You've successfully authenticated, but Gitee.com does not provide shell access.”才证明成功连接。

（2）、git remote add origin [https://github.com/WinkyZhang39 /test.git](https://github.com/WinkyZhang39%20/test.git)用于选择远程项目的地址，即GitHub上面你的仓库。

5、根据需要对代码进行修改和提交

git clone git@github.com:用户名/仓库名.git#克隆仓库

git add #添加单个文件

git add #添加多个文件：文件夹1/ 文件夹2/ ……多个文件夹之间空格隔开

git add . #添加此目录下所有的文件

git push -u origin master#第一次验证。（这里还要看bash里面路径之后的标记是main

还是master，这是历史问题，是项目仓库的分支。如果本地分支和远程分支冲突，可以使用：git branch -m master main # 将本地分支重命名为 main

git push -u origin main # 推送本地 main 到远程 main（第一次提交时远程仓库可能是空的，所以要加‘u’，之后不是空的就不用加了）

也可以使用git config --global init.defaultBranch main 修改全局的Git配置）

git push git@github.com:\_\_\_/\_\_\_.git#添加到某个仓库

git add README.md

git commit -m "第一次提交：项目描述"

git push

git add hello.py#在本地文件夹内

git commit -m "第二次提交：添加Python脚本"#注释

git push

git log --oneline # 查看简洁提交历史

git diff HEAD~1 HEAD # 比较最近两次提交差异

方式：

git init 用git初始化当前目录

touch README.md 新建一个README.md文件

git add README.md 添加当前目录下的README.md文件到本地记录

git commit -m "first commit" 提交到本地仓库

git remote add origin https://gitee.com/namelzx/lab.git 填写远程项目的地址，即仓库的https://...地址（这一步可以不使用，如果在前面的git init之后指定了远程仓库，就会记录该文件夹指向的仓库）

git push -u origin master推送到git仓库（此处加了-u 参数,以后即可直接用git push 代替）

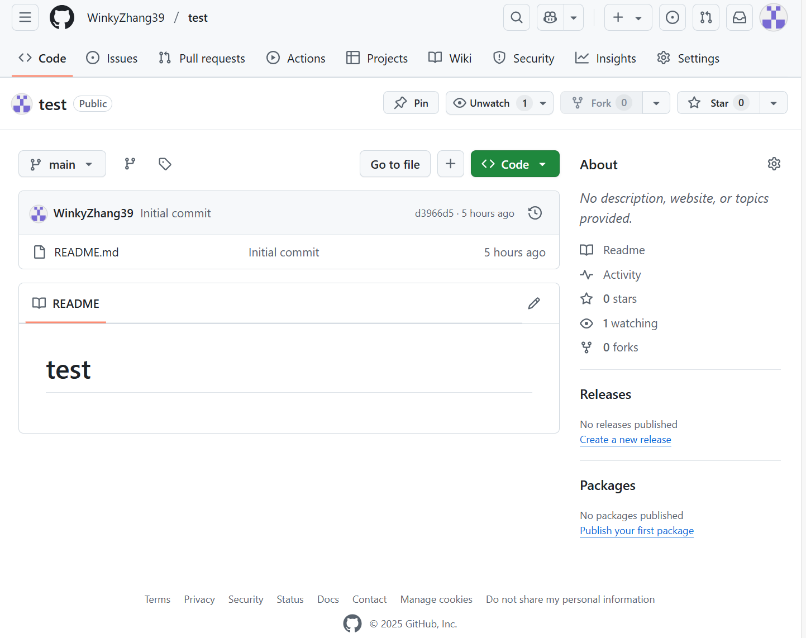
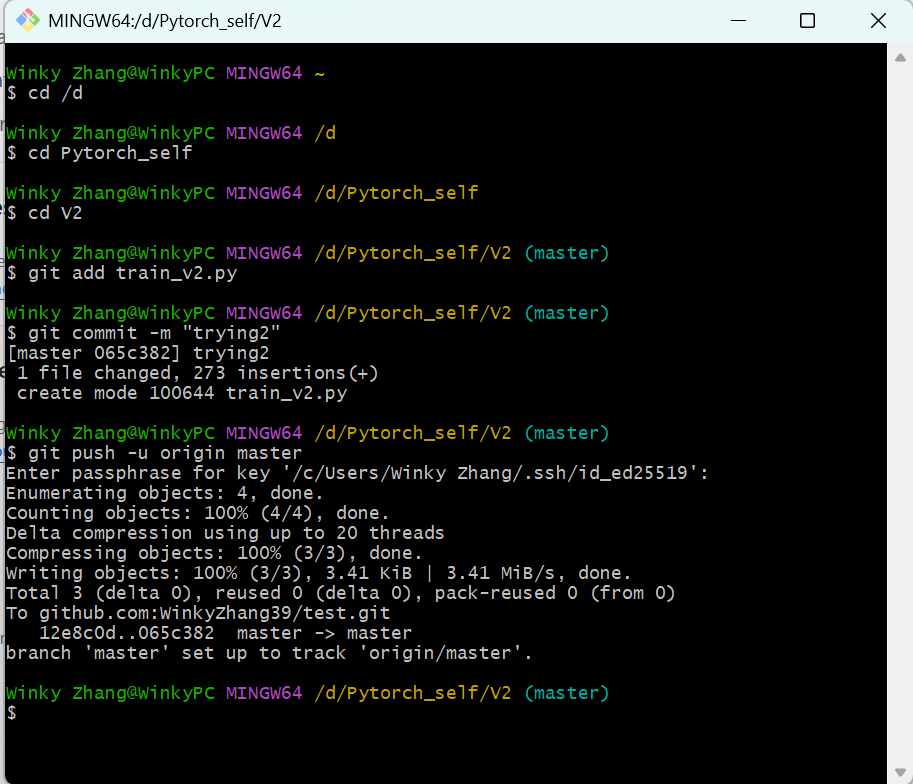
1. 其他git命令
2. git log 命令可以告诉我们历史记录
3. git log --pretty=oneline 命令查看简洁的提交历史记录
4. git reset --hard HEAD^ //回退到上一个版本
5. cat <fileName> //查看文件内容，即查看是否回退到了文件的上个版本
6. git log 　　　　//查看现在版本库的状态，此命令可以帮我们确定要回退到哪个版本
7. git reflog　//用来记录你的每一次提交的命令历史，以便确定要回到未来的哪个版本
8. git文件的目录不算是工作区，而是Git的版本库。

如果修改的文件在工作区，还没放到暂存区：

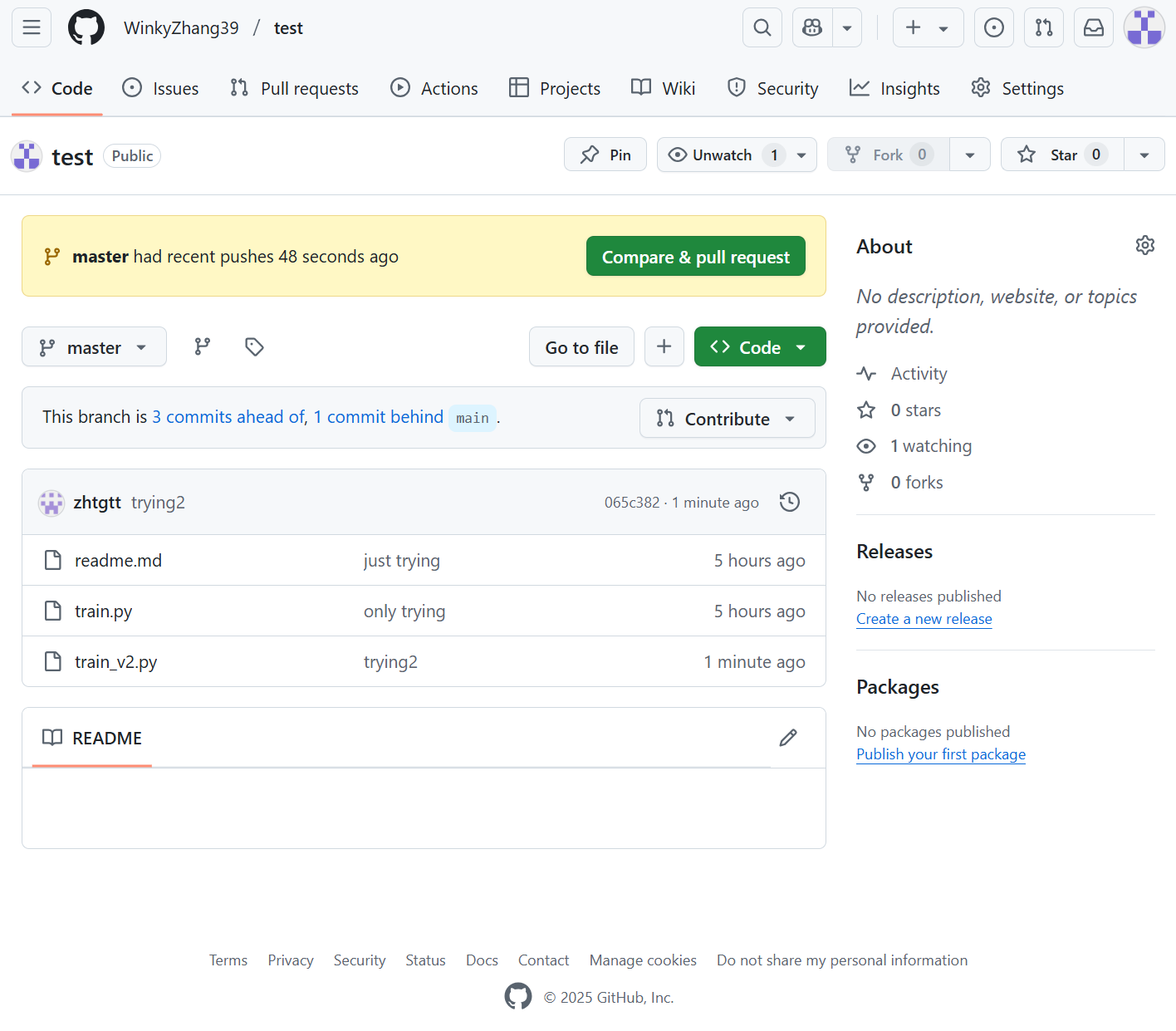
1. git checkout -- readme.txt //丢弃工作区的修改

如果修改的文件在暂存区：

1. git reset HEAD readme.txt　　//把暂存区的修改撤销掉
2. git status　　　//查看一下，暂存区没有修改了，本地工作区有修改
3. git checkout -- readme.txt　　//继续撤销工作区修改（然后 git status 查看）
4. rm test.txt 　　//删除工作区的文件（文件如果没有做commit提交，只在工作区，用此命令删除即可）
5. git status　　//查看下状态，如果版本仓库里面有的话，删除了本地的，仓库里面还存在。
6. git rm test.txt 　　//工作区文件删除，但是版本库里面还有，就用此命令删除。
7. git commit -m "remove test.txt" 　//删除版本仓库以后，再做个提交。

在提交之前的仓库现状

提交之后：



困难和解决方法：

主要在于git通过ssh和GitHub连接的问题，因为GitHub的服务器在外网，所以想要流畅使用是要经过一些方法的。然而ssh对于GitHub端口的连接因为这一原因总是不能实现，报错ssh: connect to host github.com port 22: Connection refused。最后在网上找寻办法和询问DeepSeek无果后自己想到了有可能是这样一层原因。还有就是对于仓库分支的选择，由于main和master不同的分支，导致早一段时间的教学文章并没有提及，还是用的老版本的分支master，最后通过DeepSeek询问分支之间的关系才理解。

心得体会：

Git对我来说确实是一个新鲜的代码版本管理工具，解决了之前我做项目时不断修改模型结构和参数而没有办法对各个版本的代码进行管理的痛点。学习Git的过程就是不断试错和探索，不断地去理解，尚且现在有大模型工具，为我的学习减轻了压力。