十月一日

抱着对冰岩的向往以及自己想要学习编程的热情，接到任务的第一天就开始思考选哪个题目。由于没有计算机背景，不了解计算机的架构，四个题目中有三个都不太理解，只对机器学习原理比较清楚，数模搞过这些，但是组长建议我搞一个自己不了解的方向，锻炼一下学习能力，于是就放弃了这个题目，思考后最终决定选“简单版本控制器”这个题目，并决定用Python编程，因为读研也需要用到Python，刚好熟悉一下。

选定题目之后就开始仔细审题了，首先根据组长的建议，先看了一下github，大概知道了版本控制是用来干嘛的，这个对开发程序确实有用，很容易找回以前的版本。刚开始做这个题目，首先是想怎么进行架构设计，用户修改了一个文件之后，用什么来记录该文件被修改过呢？同样，用户删除或增加了一个文件该怎么记录？后来感觉一个可行的方法是把这些修改信息保存在一个文件中，这个文件需要能记录住“修改”、“增加”、“删除”、“移动”、“未托管”5种信息，用户每修改一次文件，就在文件的“修改”记录上，增加该文件名，其他四种同理。在用户输入status命令时，将该文件输出就行了。用户commit的文本都保存在仓库中的svn文件夹下，判断一个文件是否被修改过，就用当前文件与svn文件夹中该文件对比，有不同内容，就说明被修改了。大致思路大概就是这样，由于还不是太清楚github的具体功能，很细节上的还没有考虑。

大体上有个思路之后，就开始学习Python了，只知道思路，不知道怎么用Python代码来实现，然后就看了一下《Python基础教程（第2版）》这本书，学了一下Python的语法，以及对文件的处理方面的知识。之后就按照文档的建议，从第一个比较简单的小题目开始，思考怎么实现两个文本的diff功能，首先想到的是将两个文件内容都读出来后一行行的比较，这个也和组长和群里的成员讨论了 ，后来百度了一下，发现Python竟然有difflib可以直接调用，还可以生成html文件方便查看，这个简直太方便了，然后就开始查一下difflib的具体用法。

十月二日

昨天大致把diff功能实现了一下，今天开始思考Python怎么打包成exe，因为看到文档中是说运行方式是形如“svn.exe create目录名”，不知道怎么把py文件转成exe文件，百度这个问题，说是有py2exe模块，按照教程做了一下，发现不能实现，是因为py2exe是针对Python2的模块，我安装的是Python3，这个和组长以及群里成员讨论了一下， 他们建议我先不管这个了，先直接用Python接收参数，但是Python接收参数我也不清楚怎么做，查了资料也没有解决，只得先搁下，准备先把剩下的功能代码先写一下，测试都放在最后进行。

第二个功能是要在当前目录create一个目录作为仓库，这个查了一下Python怎么创建目录，比较容易实现，但是对要求中“如果当前目录已在一个版本仓库下，创建失败”，有很多疑问，怎么判断当前目录在不在一个版本仓库下呢？又仔细看了一下要求，他是说创建一个目录并初始化一个svn文件夹存储仓库版本数据，如果当前目录的上级目录是svn，不就说明该目录就在一个版本仓库下了吗？我是这么理解的。然后就开始查找Python如何得到上级目录等实现细节上的问题。

十月三日

今天开始思考实现status的功能，这个思路还算清楚。

在维护一个全局变化字典statusRecord的情况下，判断文件有没有修改，首先要判断该文件是否被管理，如果被管理了，还要判断该文件是否commit过，也即在svn文件夹下是否保存有老的版本，如果都有，在调用diff函数，判断两个文件是否相同，如果相同，提交失败，不同就在svn文件夹下的该文件的文件夹内复制一份更新的版本进来，同时要修改名字为版本2。

而判断文件是不是新加的，就比较容易，在调用add时，检查statusRecord里‘？’内是否存在该文件，若是，‘？’移除该文件名，‘+’内添加该文件名，这么做的前提是‘？’内要保证记录了该仓库下的所有未收托管的文件。所有调用status时，首先要遍历仓库，讲不属于statusRecord任何key的value加入到‘？’中。

判断文件是否删除，只要遍历仓库下所有文件名，然后遍历svn下所有文件名，在svn下而不在仓库下的，就说明是被删除了。就在statusRecord的‘D’里添加该文件名。

判断文件是否移动，文件的移动是通过mv命令实现的，只需在执行mv命令时，更新statusRecord的‘MV’值即可。

不过在这个过程中，statusRecord需要精心的维护，比如一个文件a提交一次后转态变为M（被修改），则statusRecord的‘M’中加入a，之后我们把a删除了，这时应该把a加入‘D’里，同时要删除‘M’里的a，也就是说，一个文件的状态改变了，要实时更新statusRecord。

十月四日

* Svn add 文件名

这个命令很好实现，从控制台接收命令后判断是不是add命令，如果是，就判断statusRecord‘？’里是否存在该文件名，如果是就移除，并在‘+’里添加该文件名。

* Svn delete文件名 （--force）

判断文件是够在‘？’里，如果不在，就添加到‘？’里，并刷新statusRecord。判断是够存在--force命令，只需在确定是delete命令时，检查参数个数就知道有没有force命令了，如果存在就删除物理文件。

* Svn mv 文件名 目标目录

将文件移动到指定文件夹，并把文件名和移动路线加入statusRecord的‘MV’，更新statusRecord。

* Svn commit 文件名 (--m ‘提交日志’)

Commit文件时首先调用diff检查是否更新，如果没有变化，就返回‘no modification’，变化了就复制一份新版本到svn下对应的文件夹内，并将文件命名为文件名+版本号，版本号就是该文件夹下所有文件的数目。同时更新文件夹内用以与外界对比的版本。如果存在--m，就将后面的描述写进log文件，否则，写进默认描述“A new commit”。

* Svn git log （文件名）

最简单的命令，输出log文件内容即可，如果有文件名，只检索输出包含该文件名的log内容。

* Svn revert（版本号）(-f 文件名)

版本的回滚比较复杂。

1. 如果不设置版本号和文件名，就将svn下的所有文件名读出，检索仓库中是否存在该文件，如果存在先删除仓库下的存在于svn中的文件，之后将svn中的最新版本文件拷贝到仓库。
2. 如果不设置版本号而设置文件名，就遍历仓库中是否存在该文件，如果存在就先删除，然后复制svn中该文件的文件夹中最新的版本到仓库中。
3. 如果设置版本号而不设置文件名，就将svn中所有文件名取出，并遍历仓库中是否有该文件，如果有先删除该文件， 遍历svn所有文件夹，打开存储各个文件的文件夹，取出对应版本号的文件复制到仓库中，并重命名，将名字的版本号去除。同时更新svn中该文件对应文件夹下用于与外界对比的文件。
4. 如果设置版本号和文件名，就先删除仓库中的该文件（如果有），然后取出svn中该文件对应的文件夹下的对应版本号，复制到仓库中。同时更新svn中该文件对应文件夹下用于与外界对比的文件。

十月五日

整个系统的调试工作，各种小细节，小bug的处理工作。