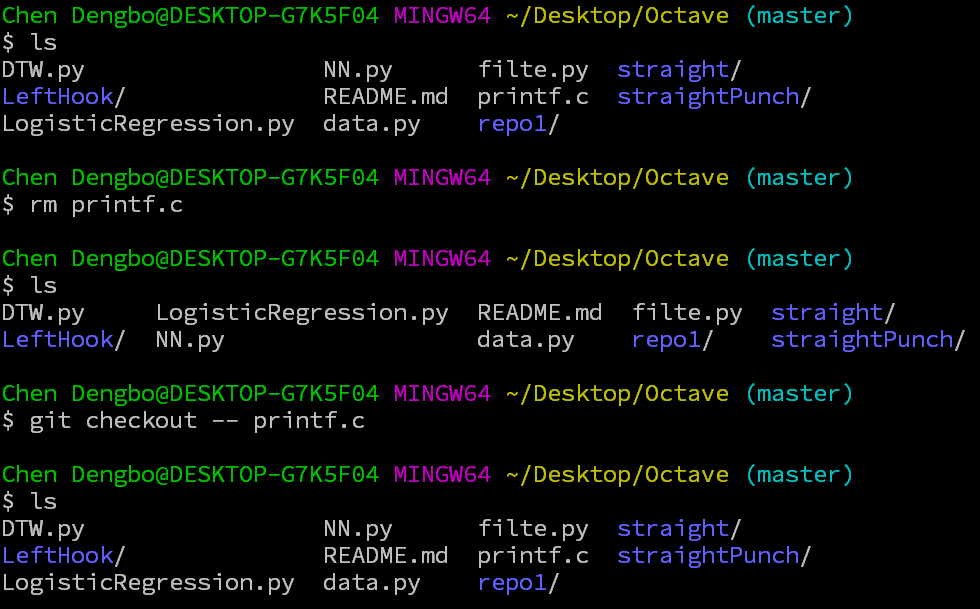
OSLAB1实验报告

150616-14051131-陈登博

思考题

Thinking 1.1

1. git checkout -- printf.c

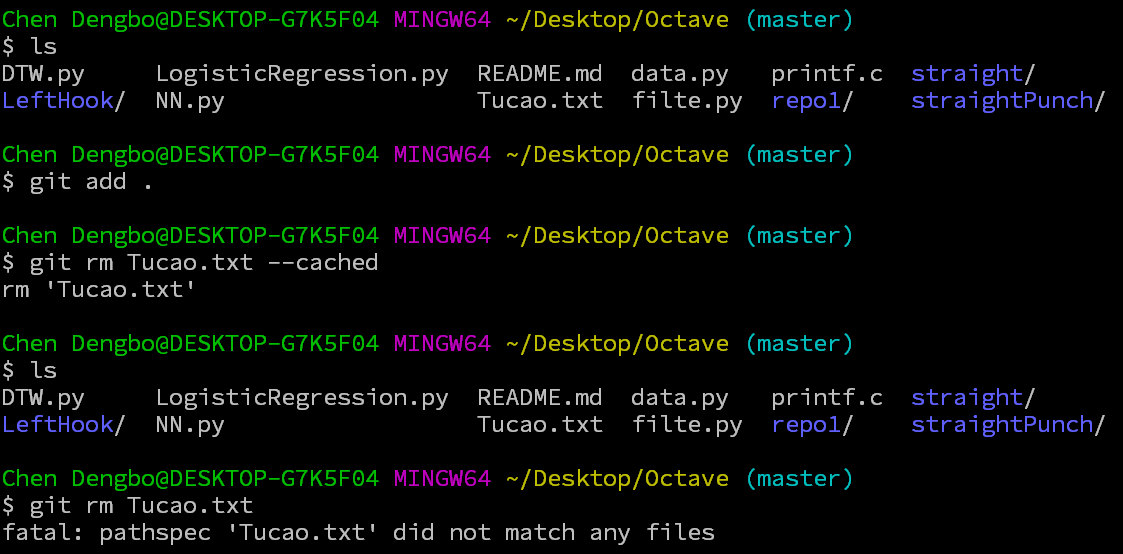


1. git reset HEAD printf.c

git checkout printf.c



1. git rm Tucao.txt –cached



Thinking 1.2

1. 正确，因为我在github上远程建立了一个分支，在本地用克隆命令发现可以克隆所有分支，但是只有HEAD指向的master分支被克隆到本地，其他分支需要用git checkout <branch>的方式才可获得。
2. 正确，因为断网之后，这些命令仍然可以执行。
3. 错误，与1矛盾
4. 错误，克隆后工作区默认处于HEAD指向的分支，而不一定是master分支(参考：廖雪峰的Git教程：创建与合并分支

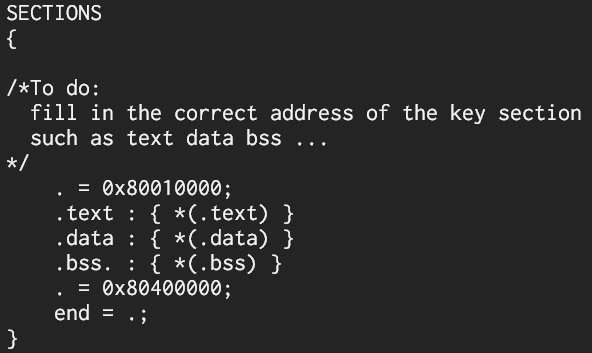
http://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000/001375840038939c291467cc7c747b1810aab2fb8863508000)

实验难点

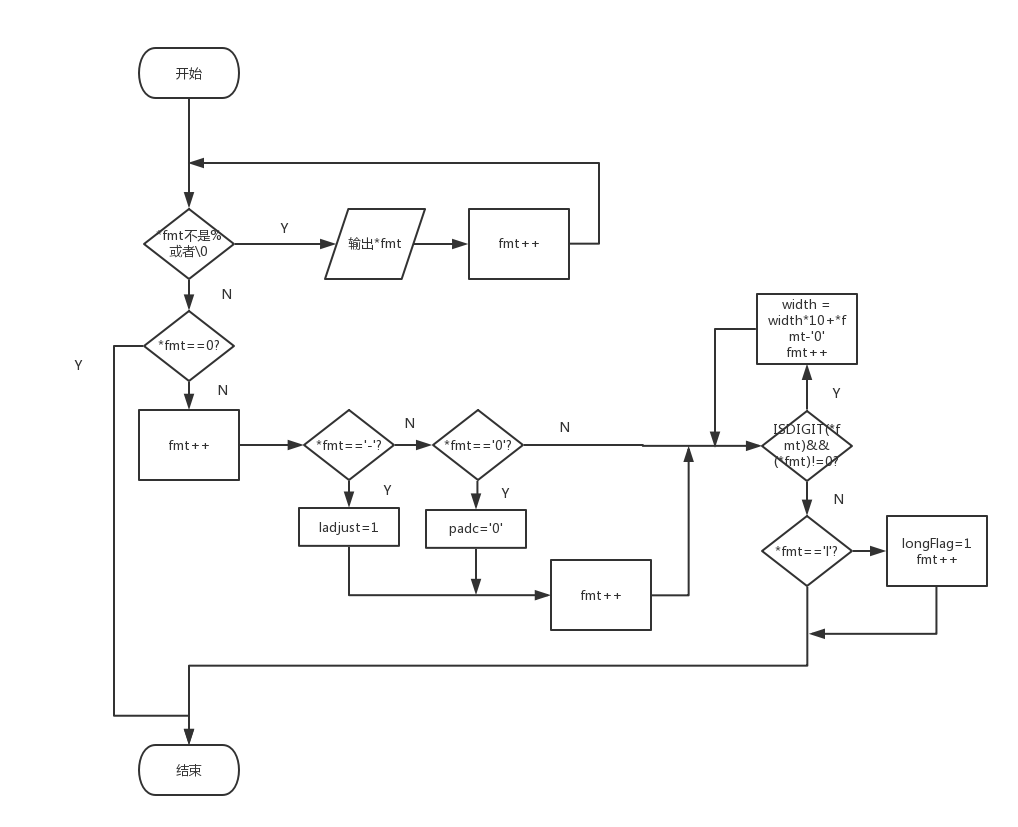
我认为本次实验的难点在于修改scse0\_3.lds文件、补全lp\_Print()函数以及使用git进行代码提交。

首先在修改scse0\_3.lds文件时，由于需要观察操作系统内核完整的内存布局图才能知道.text段，.data段，.bss段的具体位置在哪里，所以一开始看不出正确答案的往往在调试的时候耗时很久，而程序也由于没有执行正确的代码而陷入死循环。在经过多次调试后才得到正确的设置数据。

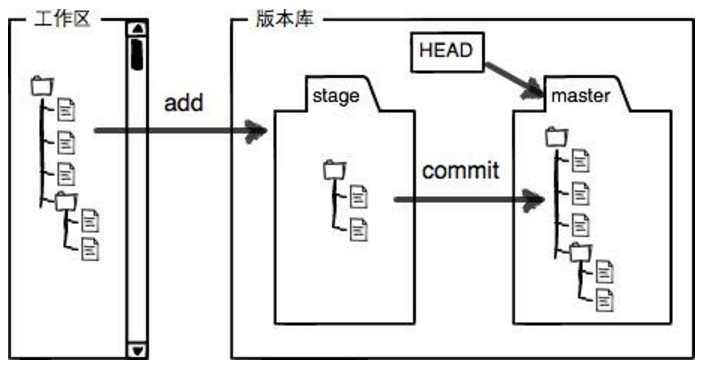
最终得到的代码如图：



在补全lp\_Print()函数时则需要考虑到后面的switch中给出的几个标识符和函数前面定义的若干个变量，例如longFlag,negFlag,ladjust等。通过查阅cplusplus对于这个函数的说明，我最终得到的思路如下：（仅包含补全内容）



git的难点在于如何用git管理和提交自己的代码，主要是涉及到git add,git commit,git push,git branch,git checkout,git pull等命令。仅仅从命令上学习git略为抽象，加上图示则会更好的理解：



Git实际上有着3个区域，离我们最近的是工作区，中间的是暂存区，而远程还有一个版本库。每次在本地创建或者修改文件之后，需要用git add命令将文件的改动上传到暂存区，随后再用git commit命令将暂存区中的改动添加到当前分支。用git push命令则可以将本地文件推送到远程版本库。git pull命令用于从远程版本库中取回文件。而git branch与git checkout则是Git在管理分支方面的命令，git branch可以创建，git checkout可以跳转到制定分支。当然git的有些命令还存在多义性，例如git checkout还可以撤销文件的修改、git branch还可以删除分支等。

感想与体会

总体而言，我认为lab1的难度不算太高，我在lab1上花费的时间大概有4个小时，包括读写lp\_Print函数，使用git进行提交等。本次实验是操作系统的第一个实验，从lab1就可以看出操作系统的课程设计实验中对自学的要求非常高，需要通过多方面查阅资料才能得到正确的解决方案。