

# Lab0实验报告

---

## Thinking 0.1

---

```
位于分支 master

尚无提交

未跟踪的文件：
（使用 "git add <文件>..." 以包含要提交的内容）
  README.txt
  Untracked.txt

提交为空，但是存在尚未跟踪的文件（使用 "git add" 建立跟踪）
```

```
位于分支 master
尚未暂存以备提交的变更：
（使用 "git add <文件>..." 更新要提交的内容）
（使用 "git restore <文件>..." 丢弃工作区的改动）
  修改：      README.txt

未跟踪的文件：
（使用 "git add <文件>..." 以包含要提交的内容）
  Modified.txt
  Stage.txt
  Untracked.txt

修改尚未加入提交（使用 "git add" 和/或 "git commit -a"）
```

Modified.txt和add之前的内容不同，因为README文件已经存在在仓库中，目前状态是被修改但未提交，因此显示修改。

## Thinking 0.2

---

add the file: git add

stage the file: git add

commit: git commit

## Thinking 0.3

---

1. `git checkout -- print.c`
2. `git checkout HEAD print.c`
3. `git rm --cached hello.txt`

## Thinking 0.5

---

```
git@21371469:~/test $ echo first
first
git@21371469:~/test $ echo second > output.txt
git@21371469:~/test $ echo third > output.txt
git@21371469:~/test $ echo forth >> output.txt
git@21371469:~/test $ cat output.txt
third
forth
```

## Thinking 0.6

---

command:

```
touch test
```

result:

```
Shell Start...
set a = 1
set b = 2
set c = a+b
c = 3
save c to ./file1
save b to ./file2
save a to ./file3
save file1 file2 file3 to file 4
save file4 to ./result
3
2
1
```

echo后面的内容被输出到result文件中，除此之外，test中的

```
c=$[a+b]
echo c = $c
```

使得c被赋值为a+b的值，并输出c=3。并且其中的

```
echo $c>file1
```

被判断为重定向，将echo的内容输出到文件中。然后

```
cat file1>file4
cat file2>>file4
cat file3>>file4
cat file4>>result
```

使得file123中的内容被写入result。

echo echo Shell Start输出echo Shell Start

echo `echo Shell Start`输出Shell Start

后者会先处理`中的内容，然后执行echo

echo echo \$c>file将echo c的值 输入到file中

echo `echo \$c>file1`直接输出echo \$c>file1

因为后者处理`中的值没有变化，然后直接当作字符串输出。

## 难点分析

---

第一次实验难点主要在makefile的编写和.sh脚本文件的编写

## 实验体会

---

第一次的os实验感觉并没有课上实验的压力，可能因为还没有进入真正的课上实验。感觉Linux是一个好玩的操作系统，相比于图形化界面的Windows有了更多操作的乐趣。