张金源 76066001

LAB0 实验报告

fibo.c (代码分析)

```
#include<stdio.h>

int fibo(int n){
	if(n == 0)
		return 0;
	if(n == 1)
		return 1;
	return fibo(n-1) + fibo(n-2);
}

int main(){
   int var,j,i;
   scanf("%d", &var);
   j = 1;
   for(i = 1; i <= var; i++)
   {
     printf("%d ", fibo(j));
     j++;
   }
   return 0;
}
```

Makefile 代码

```
all: fibo.c
	gcc -o fibo fibo.c

clean:
	rm fibo
```

makefile 代码当中 gcc –o fibo fibo.c 是拿来编译 fibo.c, 然后他直接生成 fibo.exe (fibo), clean 是拿来删除 fibo.exe （fibo）,如：make clean

hello_os.sh

```
#!/bin/bash
file=${1?Error: no source file}
file2=${2?Error: no target file}
sed '8p;d' $file > $file2
sed '32p;d' $file >> $file2
sed '128p;d' $file >> $file2
sed '512p;d' $file >> $file2
sed '1024p;d' $file >> $file2
```

张金源 76066001

LAB0 实验报告

file 是源文件 file2 目标文件，然后使用 sed 指令把想输出的代码或句子输出到目标文件。

思考题：

思考 0.1

通过你的使用经验，简单分析 CLI Shell，GUI Shell 在你使用过程中的各自优劣（100 字以内

GUI Shell 优点：

- Graphical User Interface – is a type of user interface that allows users to interact with electronic devices using images rather than text commands.
- 使用的时候很方便，让很多人很容易使用电脑。GUI 也让我们系统看起来更好看，优美，对刚开学使用电脑的人 GUI 很有帮助。GUI 的功能也很简单而丰富多彩。

GUI Shell 缺点：

- 虽然让我们系统看起来更好看，但一般 GUI 使用 RAM 比较多。比如如果你在 RAM 2GB 的电脑安装 Win 10, 另一个安装 Ubuntu Linux, 从我经验来说跑 Linux 的电脑不卡，但 Win10 的电脑很卡。The disadvantages of Graphical User Interface are it needs more memory of your computer. It is also difficult when it is not properly installed the program on your computer. GUI 有时候会让使用者的权利不够，因为有些 shell 的功能，如果用 CLI shell 才能实现。

CLI Shell 优点：

- Command Line Interface – is a means of interaction with a computer program where the user issues commands to the program in the form of successive lines of text.
- The advantages of Command Line Interface are easy to integrate with scripting and other programmatic scripts. It only uses a lower memory of the computer. It is also very accurate in piping multiple commands together。CLI 没有那么大使用 RAM，它也是系统最基本的 Shell，对电脑高手使用 CLI 有好多好出比如在 windows 我们可以使用 CMD 删除病毒，他的功能比 GUI 强很多，它查找一个文件也会很快，对我来说如果我们熟悉了 CLI Shell 我们会更了解我们的系统。

张金源 76066001

LAB0 实验报告

CLI Shell 缺点：

- The disadvantages of Command Line Interface are boring to look at. It also has no media. You cannot multitask. It requires hard work because you only use keyboard instead of using mouse through clicking. It is really really hassle for me. CLI Shell 对刚开学使用电脑的人会让他很糊涂，因为 CLI 的命令很多，而它也不太容易使用。我个人学了 3 个月多才感觉比较熟悉了，CLI 其实也没有界面，这个对一般人会让他们很烦，而不是 User Friendly 的。

思考题 0.2

使用你知道的方法（包括重定向）创建下图内容的文件（文件命名为 test），将创建该文件的命令序列保存在 command 文件中，并将 test 文件作为批处理文件运行，将运行结果输出至 result 文件中。给出 command 文件和 result 文件的内容，并对最后的结果进行解释说明（可以从 test 文件的内 容入手）

1. 先创建一个文件用 vim, 命令为“vim test.txt”
2. 在 test.txt 里面插入要进行的命令

```
echo shell start…                    //print shell start
echo set a = 1                       // print set a =1
a=1                                  //取 a 值为 1
echo set b = 2                       //print set b =1
b=2                                  //取 b 值为 2
echo set c = a+b                     //print set c = a+b
c=$[$a+$b]                           //取 c 值为 a+b
echo c = $c                          //print c = $c 的值（a+b）
echo save c to ./file1               // print save c to ./file1
echo $c>file1                        // 插入 c 的值 to file1
echo save b to ./file2               // print save b to ./file2
echo $b>file2                        // 插入 b 的值 to file2
echo save a to ./file3               // print save a to ./file3
echo $a>file3                        //插入 a 的值 to file3
```

张金源 76066001

LAB0 实验报告

echo save file1 file2 file3 to file4 // print

cat file1>file4 //插入 file1 to file4

cat file2>>file4 //插入 file2 to file4 的下一行

cat file3>>file4 //插入 file3 to file4 的下一行

echo save file4 to ./result // print save file4 to ./result

cat file4>>result //插入 file4 to result 的下一行

3. 然后保存，退出 vim 之后我们就可以运行 test.txt 里面的命令。我们使用 “source test.txt” , 结果我们可以在 result 文件里面看见的。我们使用 cat 命令 “cat result” 结果就是这样

   76066001_2018_jac@ubuntu:~$ cat result

思考题 0.3

仔细看看这张图，思考一下箭头中的 add the file 、stage the file 和 commit 分别对应的是 Git 里的哪些命令呢？

add the file/stage the file：$git add filename
commit: $git commit -m “comment of this commit”

思考题 0.4

•深夜，小明在做操作系统实验。困意一阵阵袭来，小明睡倒在了键盘上。等到小明早上醒来的时候，他惊恐地发现，他把一个重要的代码文件 printf.c 删除掉了。苦恼的小明向你求助，你该怎样帮他把代码文件恢复呢？

git checkout -- <file> 例如：git checkout printf.c

•正在小明苦恼的时候，小红主动请缨帮小明解决问题。小红很爽快地在键盘上敲下了 git rm printf.c，这下事情更复杂了，现在你又该如何处理才能弥补小红的过错呢？

git reset HEAD <file> 例如：git reset HEAD printf.c

•处理完代码文件，你正打算去找小明说他的文件已经恢复了，但突然发现小明的仓库里有一个叫 Tucao.txt，你好奇地打开一看，发现是吐槽操作系统实验的，且该文件已经被添加到暂存区了，面对这样的情况，你该如何设置才能使 Tucao.txt 在不从工作区删除的情况下不会被 git commit 指令提交到版本库？

git clean <file> -f 例如：git clean Tucao.txt -f

张金源 76066001

LAB0 实验报告

思考题 0.5

思考下面四个描述，你觉得哪些正确，哪些错误，请给出你参考的资料或实验证据。

1.克隆时所有分支均被克隆，但只有 **HEAD** 指向的分支被检出。正确
https://git-scm.com/docs/git-clone
https://blog.csdn.net/wh_19910525/article/details/7488931

2.克隆出的工作区中执行 **git log、git status、git checkout、git commit** 等操作不会去访问远程版本库。错误

```
76066001_2018_jac@ubuntu:~/76066001-lab$ git log
commit e279d8826cfcf686c33694208e5322759bab1fe4
Author: michael <michael201096@163.com>
Date:   Thu Mar 22 00:39:56 2018 +0800

    lab1-edited

commit 2a15bc074b3735bffc2ed53d8469bb9768db09e4
Author: harry <harry728@126.com>
Date:   Mon Mar 19 01:01:57 2018 +0800

    Initial lab1
76066001_2018_jac@ubuntu:~/76066001-lab$ git status
# On branch lab1
# Your branch is ahead of 'origin/lab1' by 1 commit.
#
nothing to commit (working directory clean)
76066001_2018_jac@ubuntu:~/76066001-lab$ git commit
# On branch lab1
# Your branch is ahead of 'origin/lab1' by 1 commit.
#
nothing to commit (working directory clean)
76066001_2018_jac@ubuntu:~/76066001-lab$ git checkout
Your branch is ahead of 'origin/lab1' by 1 commit.
```

张金源 76066001

LAB0 实验报告

3.克隆时只有远程版本库 **HEAD** 指向的分支被克隆。 错误

Clones a repository into a newly created directory, creates remote-tracking branches for each branch in the cloned repository (visible using `git branch -r`), and creates and checks out an initial branch that is forked from the cloned repository's currently active branch.

https://git-scm.com/docs/git-clone

4.克隆后工作区的默认分支处于 **master** 分支。正确

http://cscore.net.cn/courses/course-v1:BUAA+B3I062140+2017_T2/courseware/153317b39af949f8b9df1c555d922732/cee76ce058174e4e8e1889e6bd7a233b/

我们克隆下来时默认处于 **master** 分支，但很可惜实验的代码是不会在 **master** 分支上测试的，所以我们要先使用检出对应的 **labx** 分支，再进行测试。