# 实验报告

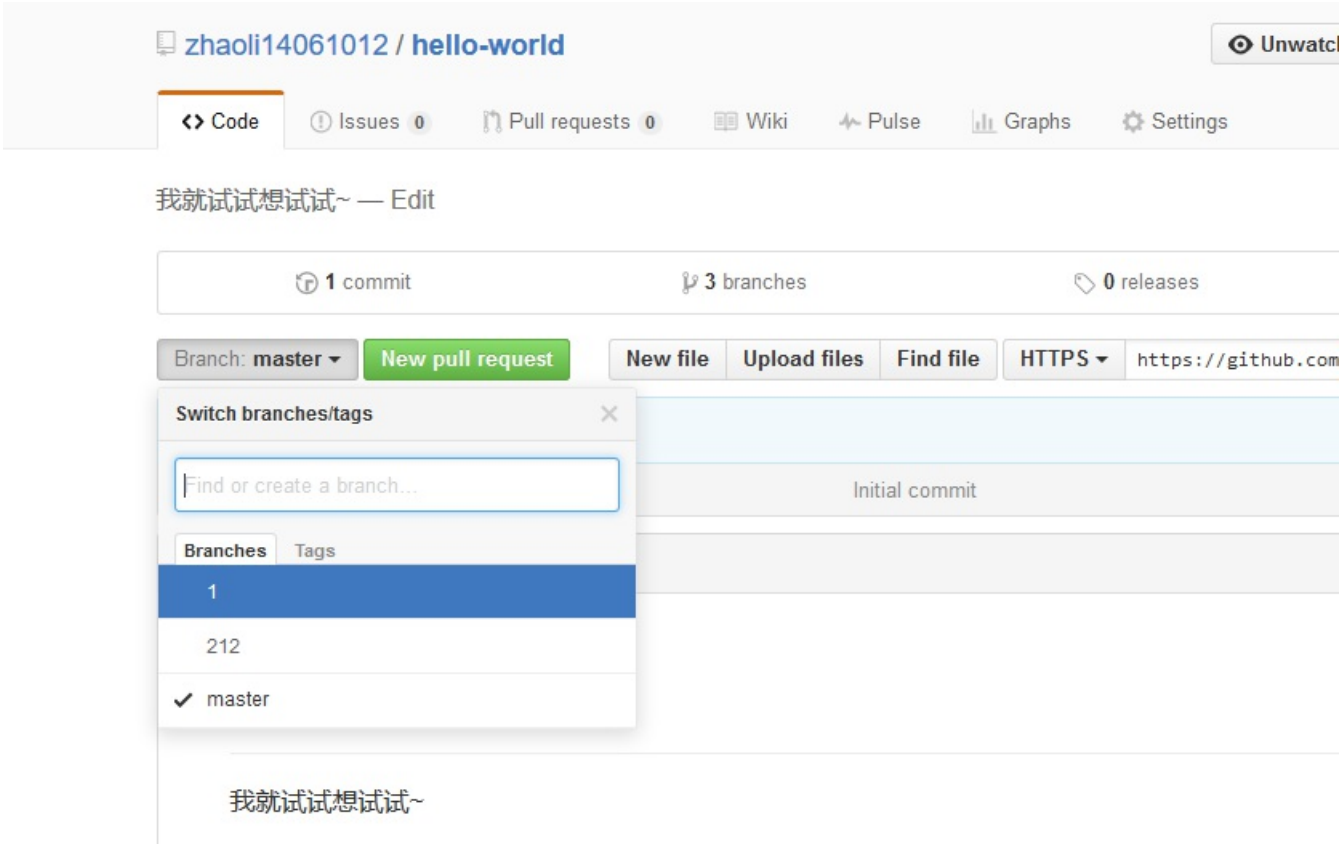
## 一：实验思考

### 1：小明的困境

a:小明把工作区的文件删除后，我们可以使用git checkout-的指令，使工作区恢复。
b:对于小红的行为，可以先使用git reset HEAD指令先恢复暂存区，然后再使用a的指令，使工作区恢复。
c:对于小明的吐槽文件，我们可以执行git rm -cached指令

### 2：克隆

**我在github中申请了一个repositories,**（如图）




a：我认为不正确，我在github里clone的代码到本地之后。不仅所有的分支被克隆，而且所有的分支都被检出。（如图）

```
赵力@DESKTOP-H1R8FIE MINGW64 ~/Desktop/hello-world (master)
$ git checkout 1
Branch 1 set up to track remote branch 1 from origin.
Switched to a new branch '1'

赵力@DESKTOP-H1R8FIE MINGW64 ~/Desktop/hello-world (1)
$ git branch
* 1
  master
```

b:我认为正确，因为这些操作不可以去访问远程版本库，这些指令都只是对本地的版本库进行操作，而不是对于远程仓库的操作。只有git push才能将操作放进远程仓库，如图：

·附：代码修改完成后一次基本的提交

```
gaochao — gaoc@ubuntu: ~/gaoc-lab — ssh gaoc@10.111.1.96 — 80×24
[gaoc@ubuntu:~/gaoc-lab$ ls
 boot  drivers  gxemul  include  include.mk  init  lib  Makefile  tools
[gaoc@ubuntu:~/gaoc-lab$ echo >> Makefile
[gaoc@ubuntu:~/gaoc-lab$ git add .
[gaoc@ubuntu:~/gaoc-lab$ git commit -a -m "first edit"
 [master 37a241a] first edit
  1 file changed, 1 insertion(+)
 gaoc@ubuntu:~/gaoc-lab$ git push
```

在图中，如果不进行push，在远程版本库中没有我们的作业。

c：我认为不正确，因为在执行git branch -a指令之后，我们可以看到存在我在远程的所有分支

```
$ git branch -a
* master
  remotes/origin/1
  remotes/origin/212
  remotes/origin/HEAD -> origin/master
  remotes/origin/master
```

d:我认为正确，在克隆之后，我的默认分支会变到master上，证据有首先，我在ssh客户端中将远程版本库克隆到本地之后，就是默认在master上，因为刚开始不知道这个，这个让我的实验一做了两次。其次在github中我从远程克隆到本地之后，就是默认在master（如图）

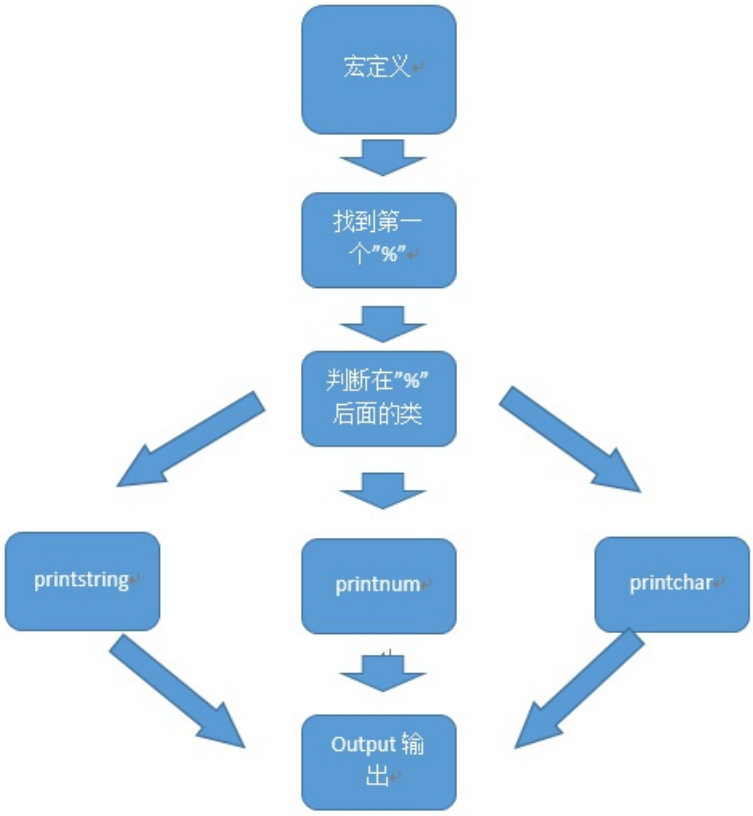
```
赵力@DESKTOP-H1R8FIE MINGW64 ~/Desktop/hello-world (master)
$ git branch
* master
```

# 二：实验难点

## 在本实验中，我认为有以下难点：

print代码部分：多个代码之间存在的逻辑关系


在printf函数中，调用了print.c中的lp_Print函数和console.c中的函数，所以在添加lp_Print中的代码的时候，首先要读懂printf函数。




通过结构图和代码的比较，我们发现在代码中不存在第二个部分。所以我们在lp_print函数中需要添加的部分就是在图示中的找到第一个"%"

## 变参：在此次代码中我第一次遇见了变参。

用法(如图)：
a.首先在函数里定义一具va_list型的变量：

b.然后用va_start宏初始化变量刚定义的va_list变量，这个宏的第二个参数是第一个可变参数的前一个参数，是一个固定的参数。

c.然后用va_arg返回可变的参数，va_arg的第二个参数是你要返回的参数的类型。如果函数有多个可变参数的，依次调用va_arg获取各个参数。

d.最后用va_end宏结束可变参数的获取。

```
[cpp] view plain  copy  print  ?
01. #include <stdarg.h>
```

```
[cpp] view plain  copy  print  ?
01. void va_start(va_list ap, last);
02. type va_arg(va_list ap, type);
03. void va_end(va_list ap);
04. void va_copy(va_list dest, va_list src);
```

# 三：实验感想

本次实验我认为对我来说，难度较大。在本次实验中花费的时间应该至少有一个星期。在这次的实验里，让我印象很深的有以下几点：

首先是字符界面让我很不适应，由于以前从未接触过字符界面，就连在java里也没有使用过命令行，所以使用起来让我觉得很麻烦，而且没有下手的地方。

其次是指导书的难度比较大，虽然我认为指导书已经讲的很清楚了，但是我

还是在读了两遍指导书之后才大致弄清楚我们在实验1里需要做什么，在刚开始连实验要干嘛都不知道。

第三点就是这次的print函数的添加里运用了大量的指针，这并没有什么，但是又运用了变参这一知识点，刚开始的时候让我一点也摸不到头脑

最后我认为也是最难的部分就是对于git的使用，这个让我彻底崩溃了，我刚开始真的一点也不懂。。。

# 四：指导书反馈

希望老师在指导书里，在开始的阶段先简要的说明地第一次实验的目的，在阅读实验指导书的时候，刚开始给人一种云里雾里的感觉。

# 五：残留难点

在实验中，我们最后实现了printf。但是在代码里，最后的状态是这样的：

```
void printcharc(char ch)
{
    *((volatile unsigned char *) PUTCHAR_ADDRESS) = ch;
}
```

我并不明白，为什么这样就能够实现输出的。