软件工程实验报告：项目管理

2112524 张梓杰

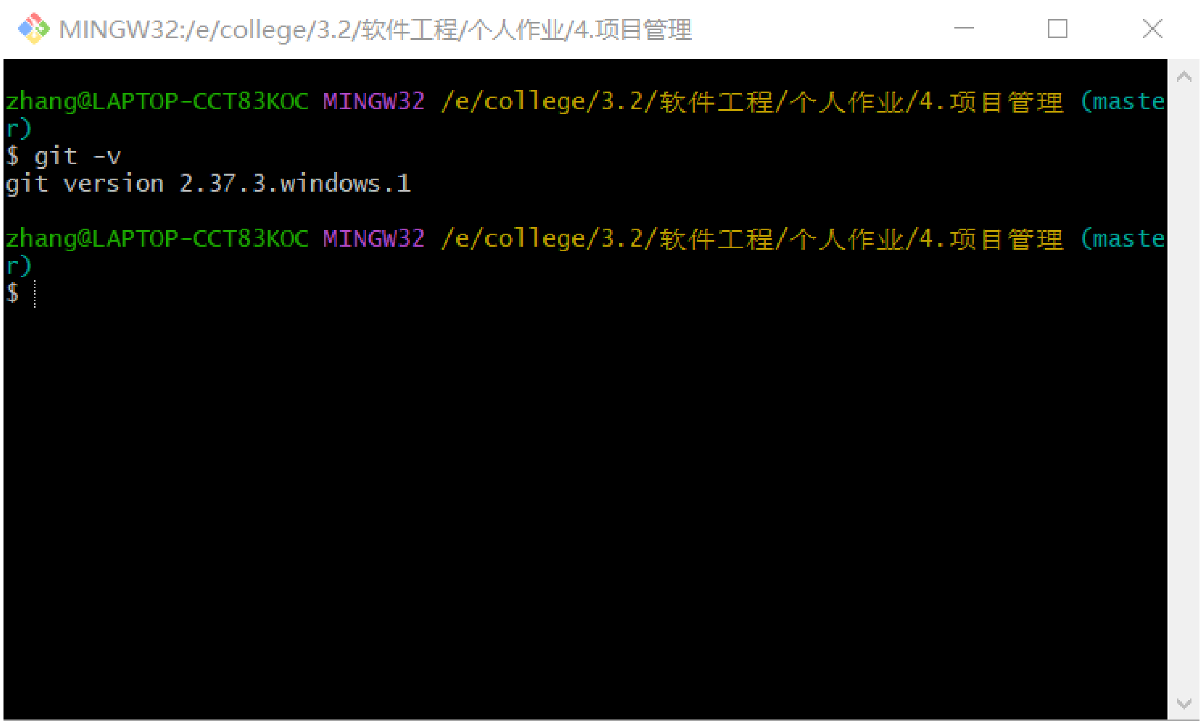
实验要求

- 熟练掌握git的基本指令和分支管理指令
- 掌握git支持软件配置管理的核心机理
- 在实践项目中使用 git / $github$ 管理自己的项目源代码
- 本次实验由**个人单独完成**（使用团队作业的文件，但**仓库需要个人新建**）

安装Git

本地机器安装Git

Git在本机已安装，版本号为2.37.3.windows.1


MINGW32:/e/college/3.2/软件工程/个人作业/4.项目管理

```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/软件工程/个人作业/4.项目管理 (maste
r)
$ git -v
git version 2.37.3.windows.1

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/软件工程/个人作业/4.项目管理 (maste
r)
$
```

申请Github账号

实验步骤

仓库创建和提交

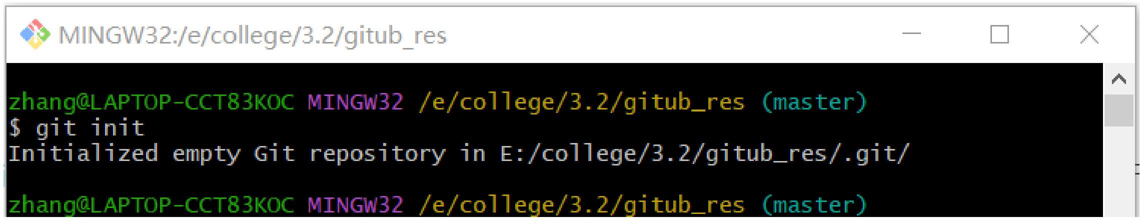
R0：监测

- 针对R1和R7，在进行每次git操作之前，随时查看工作区、暂存区、git仓库的状态，确认项目里的各文件当前处于什么状态；

R1：初始化

- 本地初始化一个git仓库，将自己在团队作业中所创建项目的全部源文件加入进去，纳入git管理；

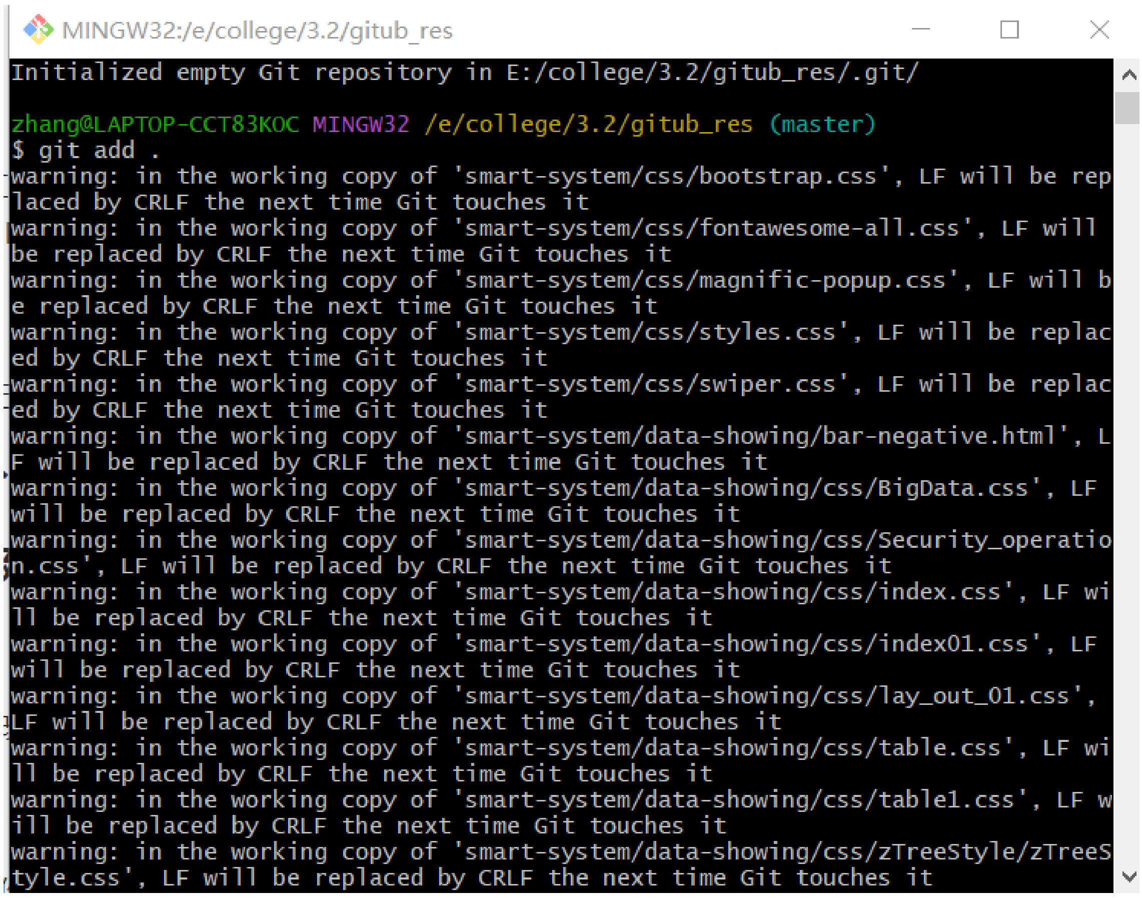
  在Git Bash上使用 `git init` 指令在本机指定路径下初始化仓库：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git init
Initialized empty Git repository in E:/college/3.2/gitub_res/.git/

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
```

R2：提交

- 将文件拖入本地仓库文件夹，使用 `git add .` 指令将文件从本地存入暂存区：（warning略过）

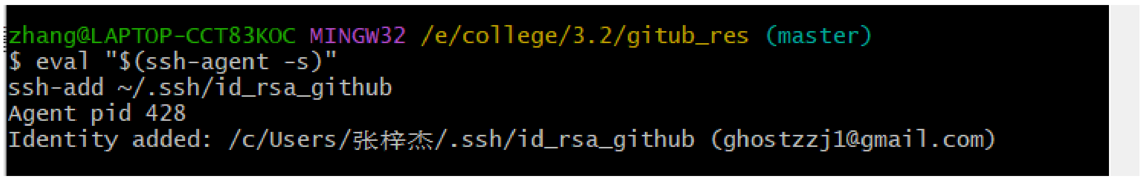


```
Initialized empty Git repository in E:/college/3.2/gitub_res/.git/

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git add .
warning: in the working copy of 'smart-system/css/bootstrap.css', LF will be rep
laced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/css/fontawesome-all.css', LF will
be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/css/magnific-popup.css', LF will b
e replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/css/styles.css', LF will be replac
ed by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/css/swiper.css', LF will be replac
ed by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/bar-negative.html', L
F will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/BigData.css', LF
will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/Security_operatio
n.css', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/index.css', LF wi
ll be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/index01.css', LF
will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/lay_out_01.css',
LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/table.css', LF wi
ll be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/table1.css', LF w
ill be replaced by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/data-showing/css/zTreeStyle/zTreeS
tyle.css', LF will be replaced by CRLF the next time Git touches it
```

提交的时候显示错误需要配置SSH秘钥。秘钥已经生成，使用

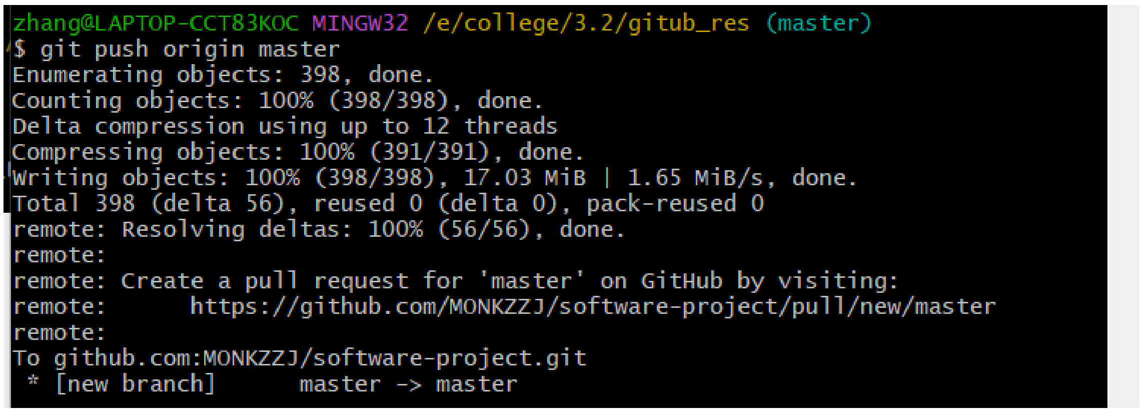
```
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_rsa_github
```

将秘钥添加到当前`ssh-agent`



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_rsa_github
Agent pid 428
Identity added: /c/Users/张梓杰/.ssh/id_rsa_github (ghostzzj1@gmail.com)
```

再次执行`git push`即可正常提交：



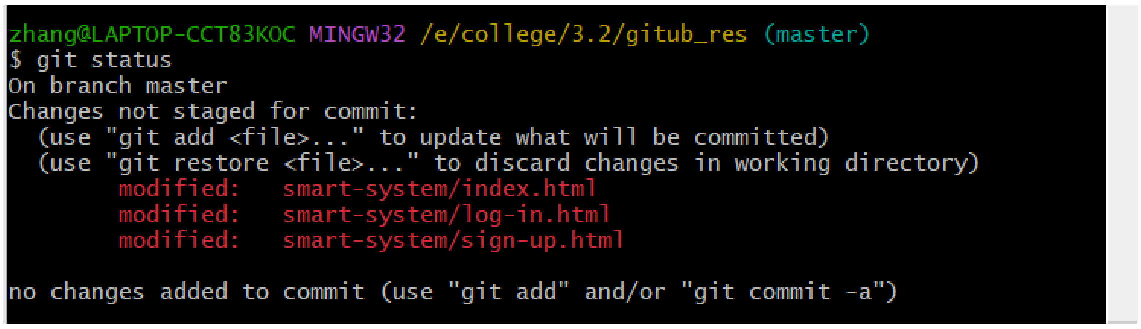
```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git push origin master
Enumerating objects: 398, done.
Counting objects: 100% (398/398), done.
Delta compression using up to 12 threads
Compressing objects: 100% (391/391), done.
Writing objects: 100% (398/398), 17.03 MiB | 1.65 MiB/s, done.
Total 398 (delta 56), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (56/56), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/MONKZZJ/software-project/pull/new/master
remote:
To github.com:MONKZZJ/software-project.git
 * [new branch]      master -> master
```

这里出现新分支创建提示是因为本地仓库默认根分支是main()。

R3：差异化查看

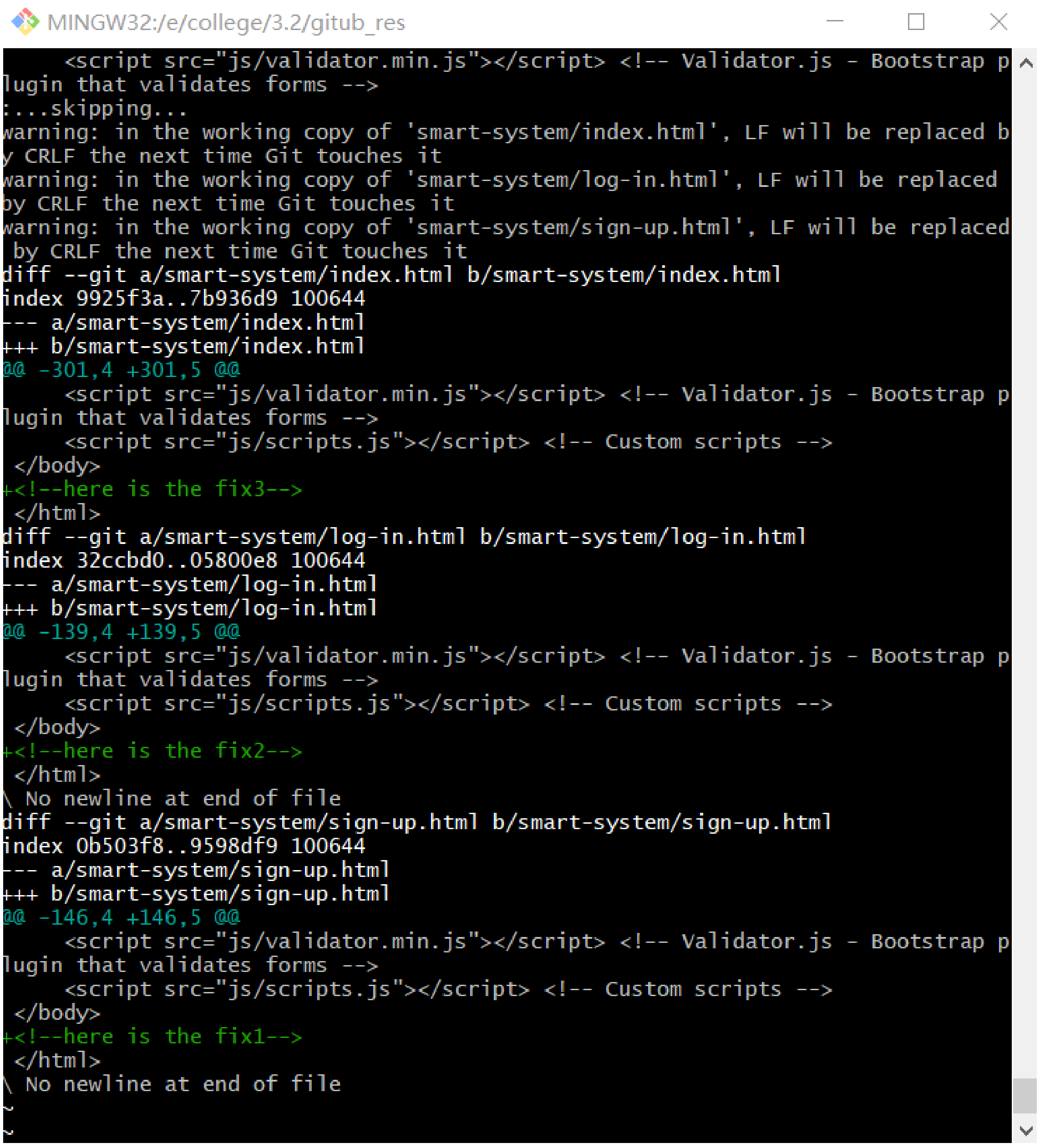
- 手工对团队作业中的3个文件进行修改；
- 查看上次提交之后都有哪些文件修改、具体修改内容是什么（查看修改后的文件和暂存区域中相应文件的差别）；

对文件进行修改后，在`git bash`中执行`git status`查看修改的状态：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   smart-system/index.html
        modified:   smart-system/log-in.html
        modified:   smart-system/sign-up.html

no changes added to commit (use "git add" and/or "git commit -a")
```

根据要求，我们在这三个文件中添加三处注释表示修改，通过执行`git diff`命令进行比对：


MINGW32:/e/college/3.2/gitub_res

```
     <script src="js/validator.min.js"></script> <!-- Validator.js - Bootstrap p
lugin that validates forms -->
:...skipping...
warning: in the working copy of 'smart-system/index.html', LF will be replaced b
y CRLF the next time Git touches it
warning: in the working copy of 'smart-system/log-in.html', LF will be replaced
by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/sign-up.html', LF will be replaced
 by CRLF the next time Git touches it
diff --git a/smart-system/index.html b/smart-system/index.html
index 9925f3a..7b936d9 100644
--- a/smart-system/index.html
+++ b/smart-system/index.html
@@ -301,4 +301,5 @@
     <script src="js/validator.min.js"></script> <!-- Validator.js - Bootstrap p
lugin that validates forms -->
     <script src="js/scripts.js"></script> <!-- Custom scripts -->
 </body>
+<!--here is the fix3-->
 </html>
diff --git a/smart-system/log-in.html b/smart-system/log-in.html
index 32ccbd0..05800e8 100644
--- a/smart-system/log-in.html
+++ b/smart-system/log-in.html
@@ -139,4 +139,5 @@
     <script src="js/validator.min.js"></script> <!-- Validator.js - Bootstrap p
lugin that validates forms -->
     <script src="js/scripts.js"></script> <!-- Custom scripts -->
 </body>
+<!--here is the fix2-->
 </html>
\ No newline at end of file
diff --git a/smart-system/sign-up.html b/smart-system/sign-up.html
index 0b503f8..9598df9 100644
--- a/smart-system/sign-up.html
+++ b/smart-system/sign-up.html
@@ -146,4 +146,5 @@
     <script src="js/validator.min.js"></script> <!-- Validator.js - Bootstrap p
lugin that validates forms -->
     <script src="js/scripts.js"></script> <!-- Custom scripts -->
 </body>
+<!--here is the fix1-->
 </html>
\ No newline at end of file
~
~
```

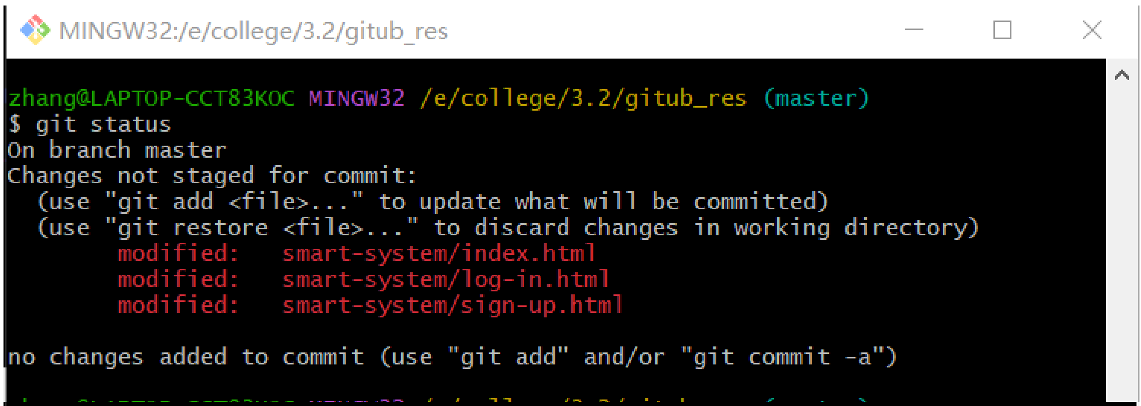
R4：重新提交

使用`git commit -a`打开日志编辑器进行日志填写并提交

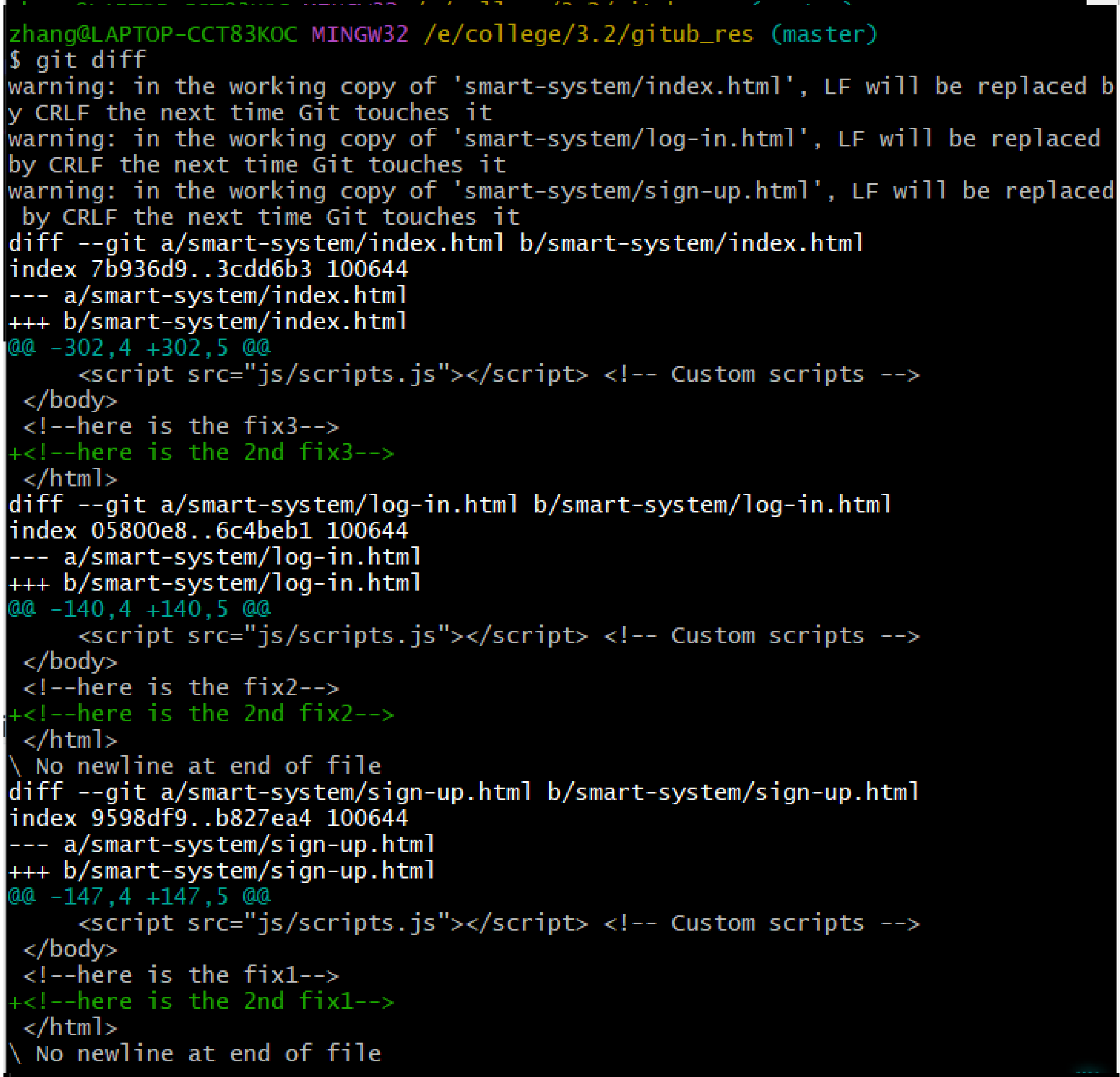
R5：重新提交2nd

- 再次对这3个文件进行修改；

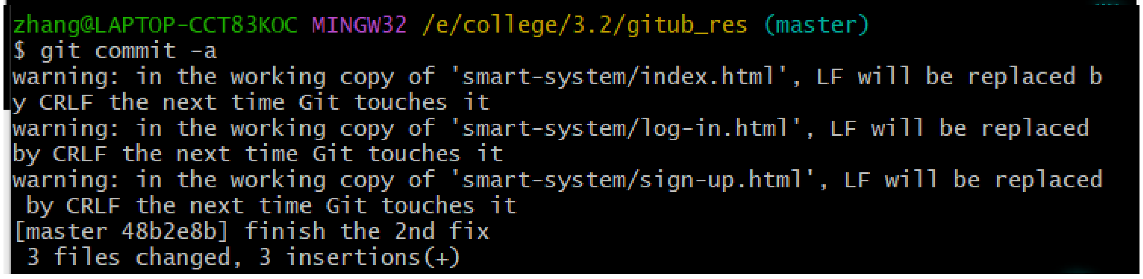
同理使用 `git status` 和 `git diff` 输出差异并使用 `git commit -a` 提交：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   smart-system/index.html
        modified:   smart-system/log-in.html
        modified:   smart-system/sign-up.html

no changes added to commit (use "git add" and/or "git commit -a")
```



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git diff
warning: in the working copy of 'smart-system/index.html', LF will be replaced b
y CRLF the next time Git touches it
warning: in the working copy of 'smart-system/log-in.html', LF will be replaced
by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/sign-up.html', LF will be replaced
 by CRLF the next time Git touches it
diff --git a/smart-system/index.html b/smart-system/index.html
index 7b936d9..3cdd6b3 100644
--- a/smart-system/index.html
+++ b/smart-system/index.html
@@ -302,4 +302,5 @@
     <script src="js/scripts.js"></script> <!-- Custom scripts -->
 </body>
 <!--here is the fix3-->
+<!--here is the 2nd fix3-->
 </html>
diff --git a/smart-system/log-in.html b/smart-system/log-in.html
index 05800e8..6c4beb1 100644
--- a/smart-system/log-in.html
+++ b/smart-system/log-in.html
@@ -140,4 +140,5 @@
     <script src="js/scripts.js"></script> <!-- Custom scripts -->
 </body>
 <!--here is the fix2-->
+<!--here is the 2nd fix2-->
 </html>
\ No newline at end of file
diff --git a/smart-system/sign-up.html b/smart-system/sign-up.html
index 9598df9..b827ea4 100644
--- a/smart-system/sign-up.html
+++ b/smart-system/sign-up.html
@@ -147,4 +147,5 @@
     <script src="js/scripts.js"></script> <!-- Custom scripts -->
 </body>
 <!--here is the fix1-->
+<!--here is the 2nd fix1-->
 </html>
\ No newline at end of file
```

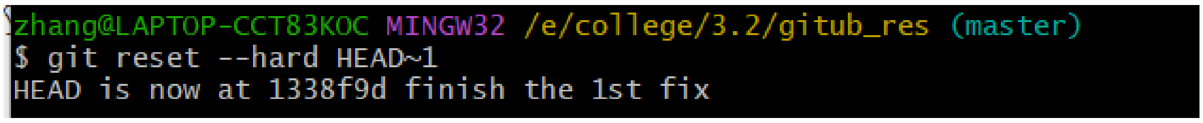


```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git commit -a
warning: in the working copy of 'smart-system/index.html', LF will be replaced b
y CRLF the next time Git touches it
warning: in the working copy of 'smart-system/log-in.html', LF will be replaced
by CRLF the next time Git touches it
warning: in the working copy of 'smart-system/sign-up.html', LF will be replaced
 by CRLF the next time Git touches it
[master 48b2e8b] finish the 2nd fix
 3 files changed, 3 insertions(+)
```

R6：把最后一次提交撤销

这里我们直接使用破坏性的撤销修改方法，即撤回提交记录同时撤回文件数据的修改：

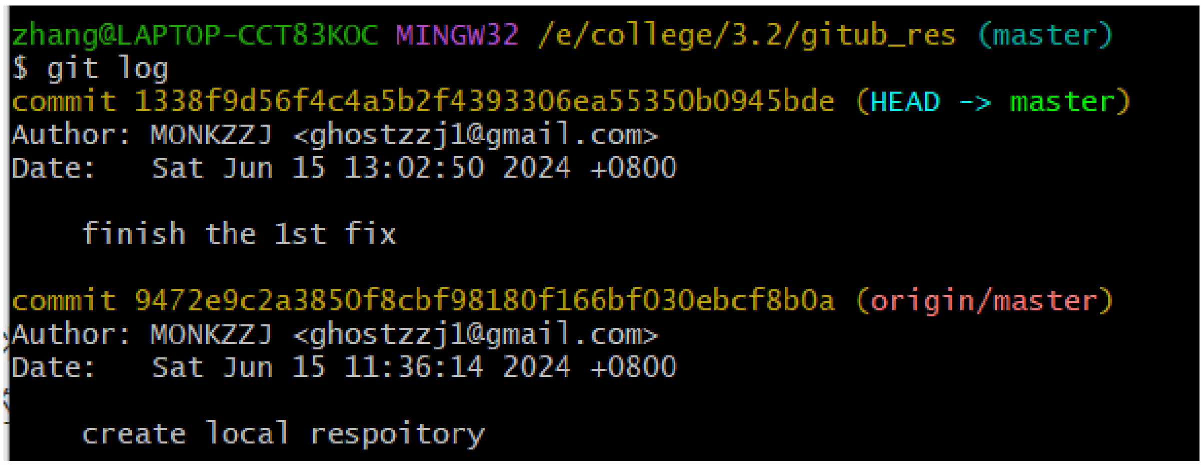
运行`git reset --hard HEAD~1`指令删除提交：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git reset --hard HEAD~1
HEAD is now at 1338f9d finish the 1st fix
```

R7：查询提交记录

使用`git log`可以查看日志提交记录：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/gitub_res (master)
$ git log
commit 1338f9d56f4c4a5b2f4393306ea55350b0945bde (HEAD -> master)
Author: MONKZZJ <ghostzzj1@gmail.com>
Date:   Sat Jun 15 13:02:50 2024 +0800

    finish the 1st fix

commit 9472e9c2a3850f8cbf98180f166bf030ebcf8b0a (origin/master)
Author: MONKZZJ <ghostzzj1@gmail.com>
Date:   Sat Jun 15 11:36:14 2024 +0800

    create local respoitory
```

可以看到第二次修改的日志已经被删除。

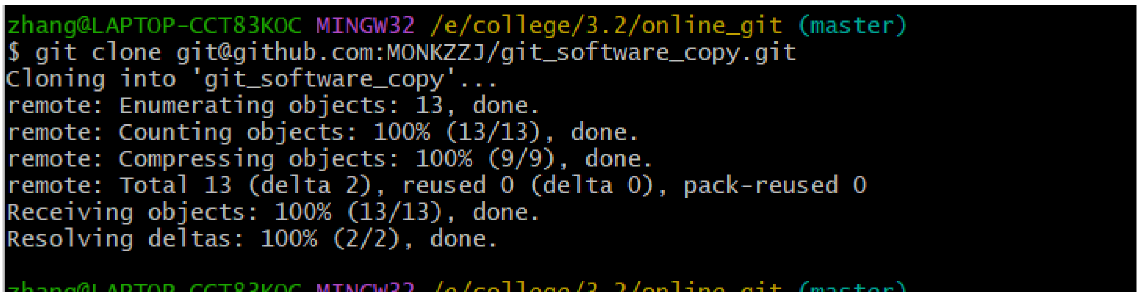
分支管理

- 在你的Github网站上，通过web界面建立一个project，将不少于10个文件（程序代码、文档等）加入进去，形成初始分支B1；在B1基础上建立两个并行的分支B2、B3，手工对B2和B3上的某些文件进行不同程度的修改并提交；

R8：Git Clone

- 从Github上(URL)克隆一个已有的git仓库到本地；

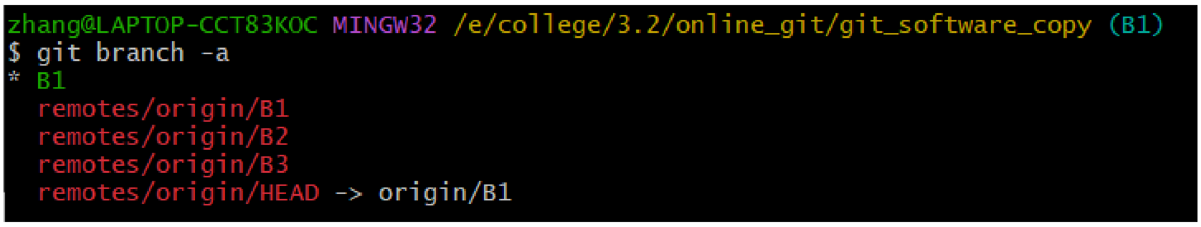
  指令：`git clone git@github.com:MONKZZJ/git_software_copy.git`



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git (master)
$ git clone git@github.com:MONKZZJ/git_software_copy.git
Cloning into 'git_software_copy'...
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 13 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (13/13), done.
Resolving deltas: 100% (2/2), done.

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git (master)
```

R9：获得该仓库的全部分支

使用 `git branch -a` 查看所有分支：



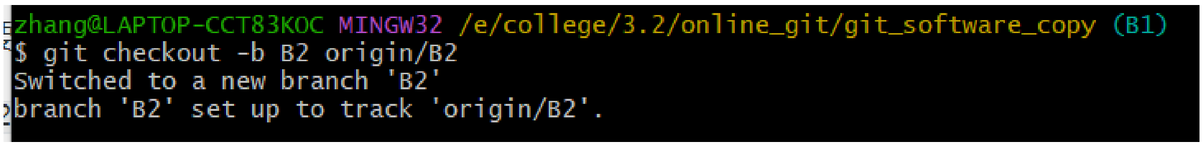
```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B1)
$ git branch -a
* B1
  remotes/origin/B1
  remotes/origin/B2
  remotes/origin/B3
  remotes/origin/HEAD -> origin/B1
```

其中remotes指远程分支，现在我们在根目录B1下。

R10：在B2分支基础上创建一个新分支C4

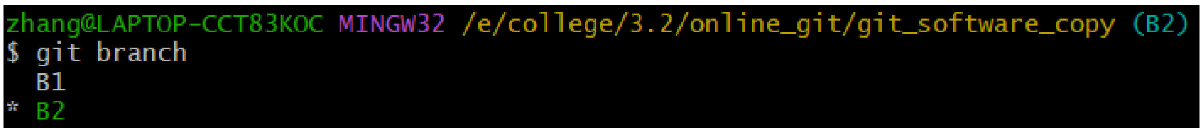
目前在B1分支，如果切换到B2分支需要根据远程分支追踪到本地分支

输入`git checkout -b B2 origin/B2`跟踪到B2分支并在本地复制：



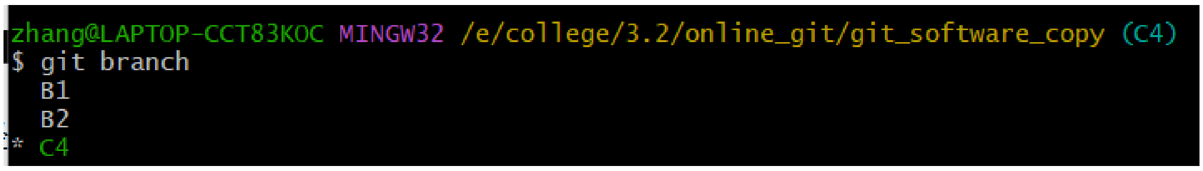
```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B1)
$ git checkout -b B2 origin/B2
Switched to a new branch 'B2'
branch 'B2' set up to track 'origin/B2'.
```

现在执行`git branch`可以看到已经存在B2分支：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B2)
$ git branch
  B1
* B2
```

在该分支下使用`git checkout -b C4`创建C4分支：使用`git branch`查看分支情况：

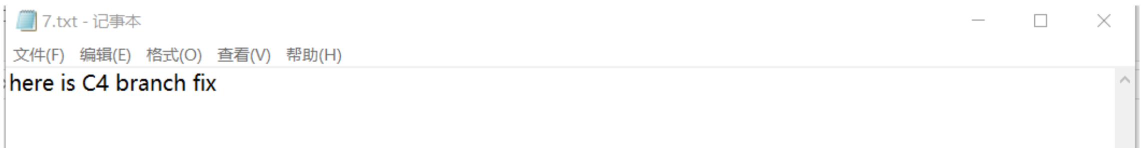


```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (C4)
$ git branch
  B1
  B2
* C4
```

R11：分支文件修改1

- 在C4上，对4个文件进行修改并提交；

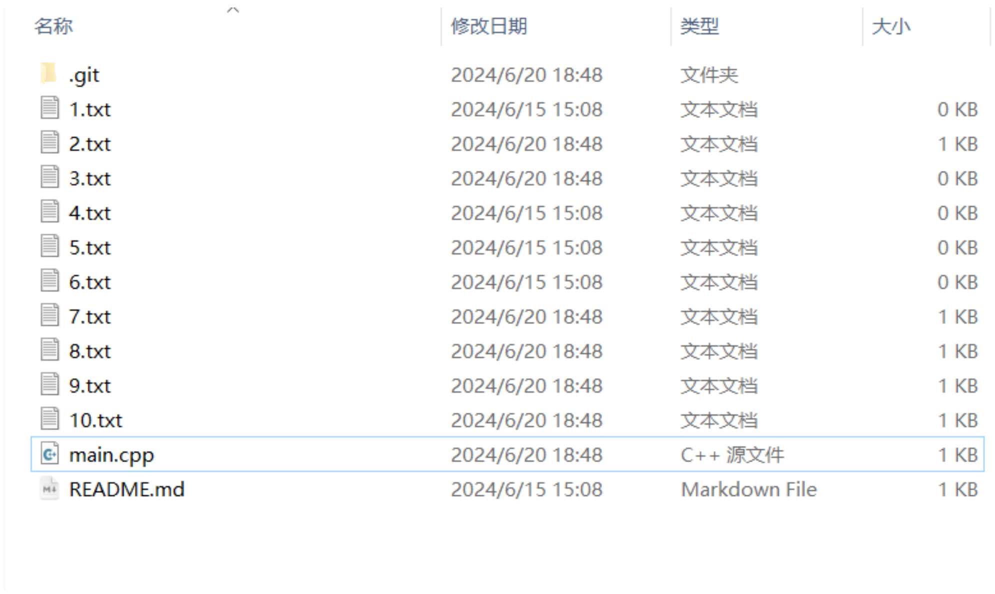
  这里我们修改7~10.txt，添加信息（7.txt为例）：



```
here is C4 branch fix
```

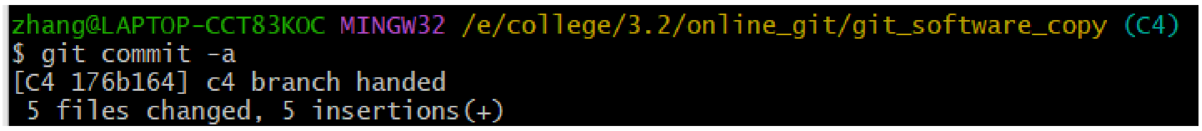
  同时修改main.cpp文件添加注释：

  分支修改文件后列表如下：



名称	修改日期	类型	大小
.git	2024/6/20 18:48	文件夹	
1.txt	2024/6/15 15:08	文本文档	0 KB
2.txt	2024/6/20 18:48	文本文档	1 KB
3.txt	2024/6/20 18:48	文本文档	0 KB
4.txt	2024/6/15 15:08	文本文档	0 KB
5.txt	2024/6/15 15:08	文本文档	0 KB
6.txt	2024/6/15 15:08	文本文档	0 KB
7.txt	2024/6/20 18:48	文本文档	1 KB
8.txt	2024/6/20 18:48	文本文档	1 KB
9.txt	2024/6/20 18:48	文本文档	1 KB
10.txt	2024/6/20 18:48	文本文档	1 KB
main.cpp	2024/6/20 18:48	C++ 源文件	1 KB
README.md	2024/6/15 15:08	Markdown File	1 KB

  使用 `git commit -a`编辑日志进行提交：

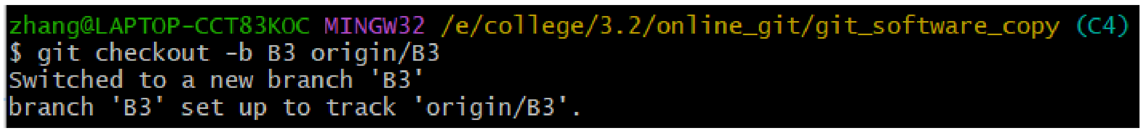


```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (C4)
$ git commit -a
[C4 176b164] c4 branch handed
 5 files changed, 5 insertions(+)
```

R12：分支文件修改2

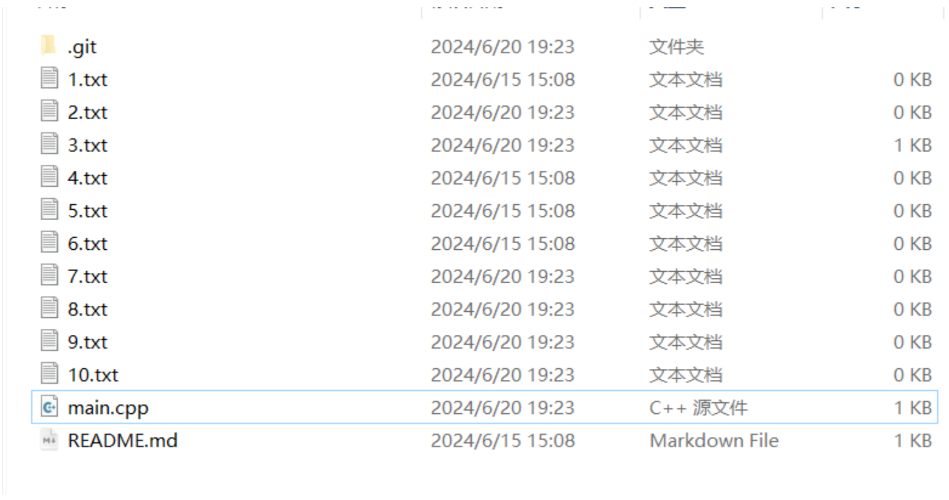
- 在B3分支上对同样的4个文件做不同修改并提交；

  首先我们通过`git checkout -b B3 origin/B3`将远程仓库内的B3分支克隆到本地，并切换到B3分支：



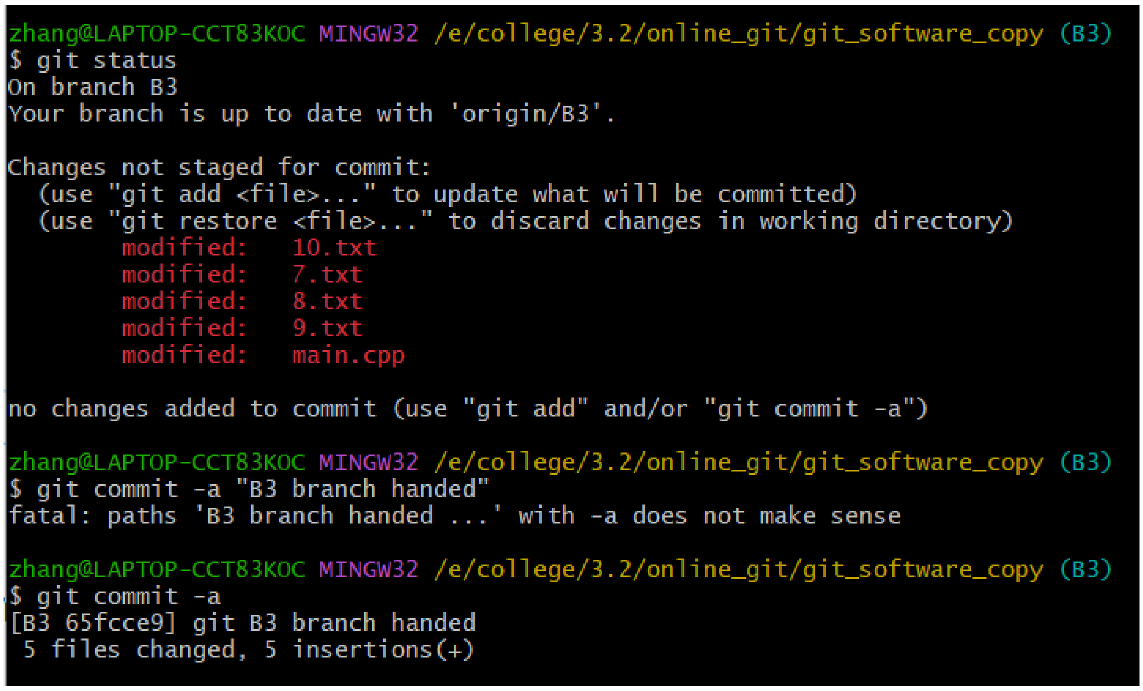
```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (C4)
$ git checkout -b B3 origin/B3
Switched to a new branch 'B3'
branch 'B3' set up to track 'origin/B3'.
```

切换后结果如下：



.git	2024/6/20 19:23	文件夹	
1.txt	2024/6/15 15:08	文本文档	0 KB
2.txt	2024/6/20 19:23	文本文档	0 KB
3.txt	2024/6/20 19:23	文本文档	1 KB
4.txt	2024/6/15 15:08	文本文档	0 KB
5.txt	2024/6/15 15:08	文本文档	0 KB
6.txt	2024/6/15 15:08	文本文档	0 KB
7.txt	2024/6/20 19:23	文本文档	0 KB
8.txt	2024/6/20 19:23	文本文档	0 KB
9.txt	2024/6/20 19:23	文本文档	0 KB
10.txt	2024/6/20 19:23	文本文档	0 KB
main.cpp	2024/6/20 19:23	C++ 源文件	1 KB
README.md	2024/6/15 15:08	Markdown File	1 KB

可以看到做出的C4分支修改在B3不生效。我们修改对应文件，注释为B3修改：

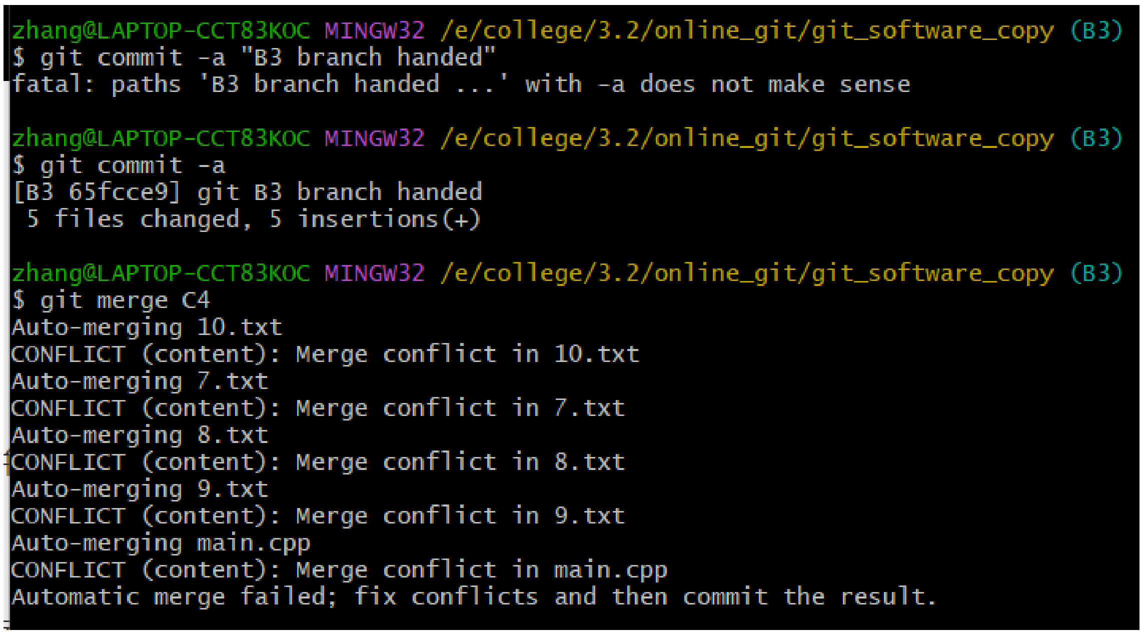


```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git status
On branch B3
Your branch is up to date with 'origin/B3'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   10.txt
        modified:   7.txt
        modified:   8.txt
        modified:   9.txt
        modified:   main.cpp

no changes added to commit (use "git add" and/or "git commit -a")

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git commit -a "B3 branch handed"
fatal: paths 'B3 branch handed ...' with -a does not make sense

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git commit -a
[B3 65fcce9] git B3 branch handed
 5 files changed, 5 insertions(+)
```

R13：分支合并

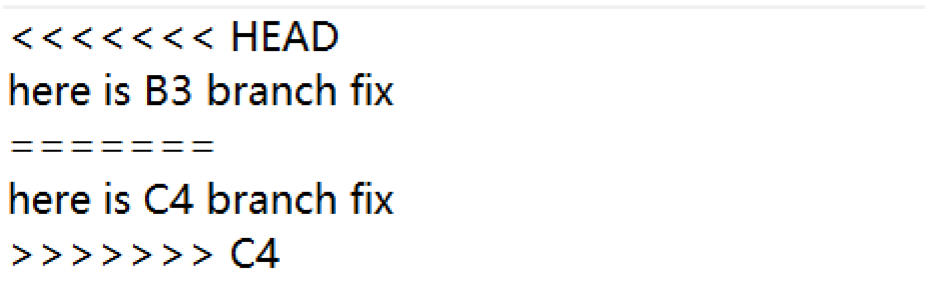
- 将C4和B3分支合并，若有冲突，手工消解；

我们现在在B3分支下，所以合并两分支需要执行 `git merge C4` 将C4的内容合并到B3：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git commit -a "B3 branch handed"
fatal: paths 'B3 branch handed ...' with -a does not make sense

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git commit -a
[B3 65fcce9] git B3 branch handed
 5 files changed, 5 insertions(+)

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git merge C4
Auto-merging 10.txt
CONFLICT (content): Merge conflict in 10.txt
Auto-merging 7.txt
CONFLICT (content): Merge conflict in 7.txt
Auto-merging 8.txt
CONFLICT (content): Merge conflict in 8.txt
Auto-merging 9.txt
CONFLICT (content): Merge conflict in 9.txt
Auto-merging main.cpp
CONFLICT (content): Merge conflict in main.cpp
Automatic merge failed; fix conflicts and then commit the result.
```

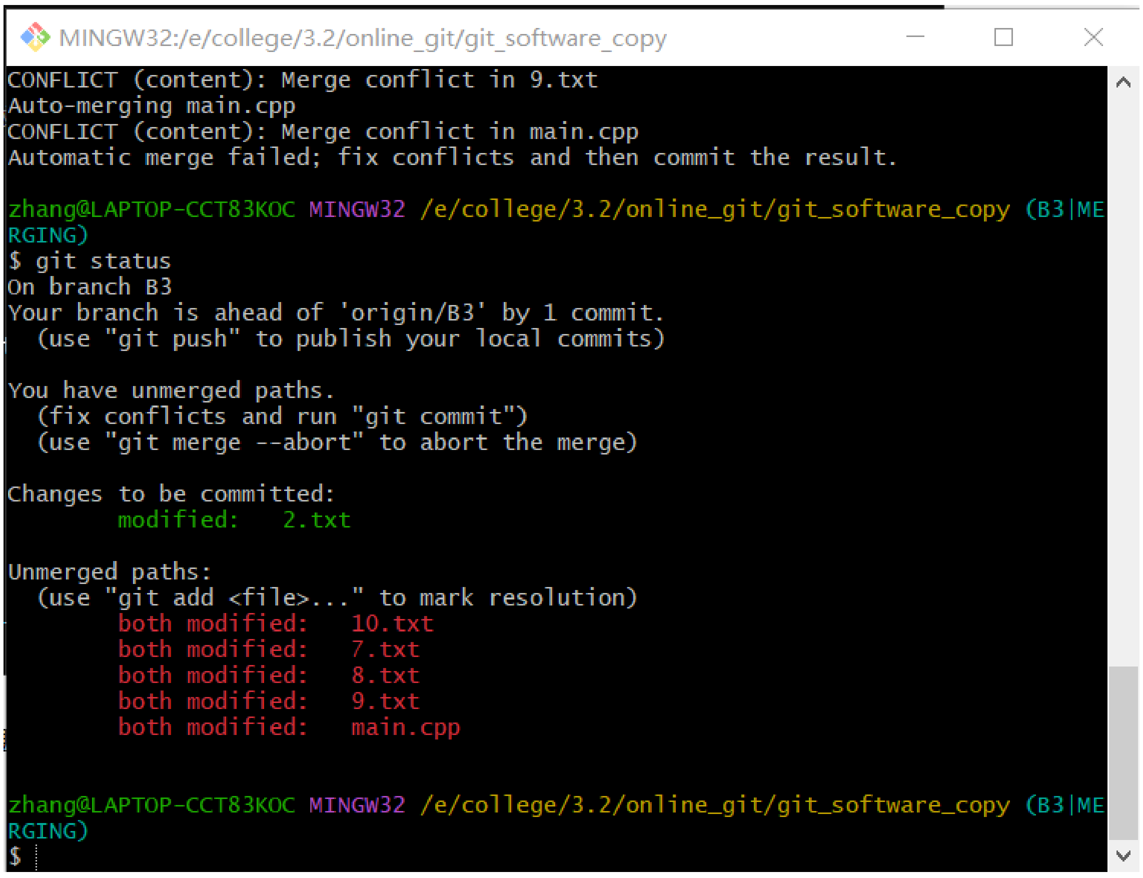
出现冲突，打开对应文件可以看到冲突信息：



```
<<<<<<< HEAD
here is B3 branch fix
=======
here is C4 branch fix
>>>>>>> C4
```

对于冲突这里要求手工消解，保留B3分支内容：

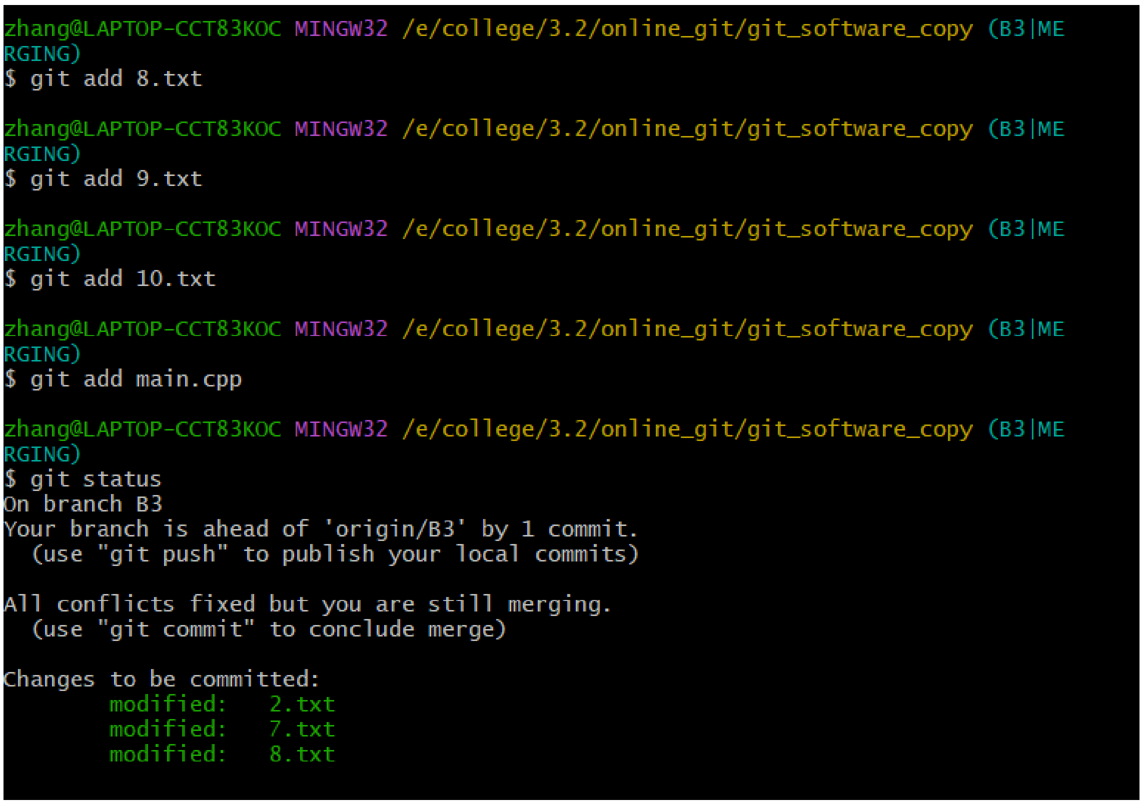
使用 `git status` 查看合并状态：



MINGW32:/e/college/3.2/online_git/git_software_copy

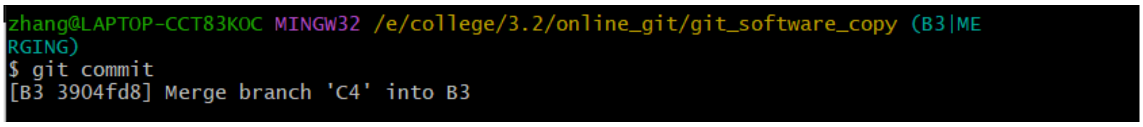
```
CONFLICT (content): Merge conflict in 9.txt
Auto-merging main.cpp
CONFLICT (content): Merge conflict in main.cpp
Automatic merge failed; fix conflicts and then commit the result.

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$ git status
On branch B3
Your branch is ahead of 'origin/B3' by 1 commit.
  (use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Changes to be committed:
        modified:   2.txt

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   10.txt
        both modified:   7.txt
        both modified:   8.txt
        both modified:   9.txt
        both modified:   main.cpp


zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$
```

2.txt为C4（B2中）与B3不发生冲突的文件，7~9.txt文件是C4与B3发生冲突的文件，冲突文件需要手动上传：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$ git add 8.txt

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$ git add 9.txt

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$ git add 10.txt

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$ git add main.cpp

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$ git status
On branch B3
Your branch is ahead of 'origin/B3' by 1 commit.
  (use "git push" to publish your local commits)

All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
        modified:   2.txt
        modified:   7.txt
        modified:   8.txt
```

进行修改后进行提交则可以看到C4已经合并到B3分支：

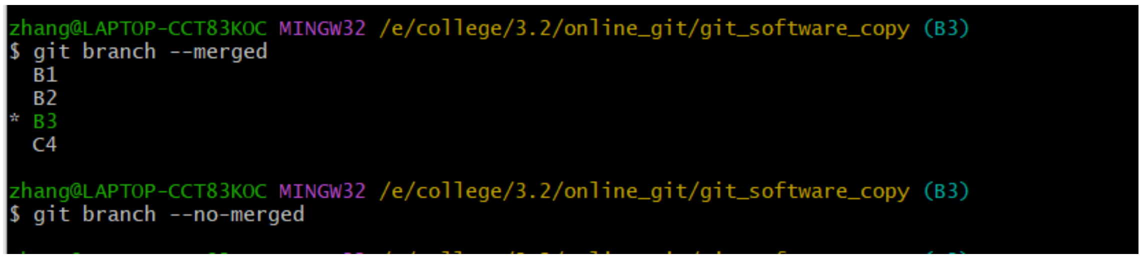


```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3|ME
RGING)
$ git commit
[B3 3904fd8] Merge branch 'C4' into B3
```

R14：查看分支

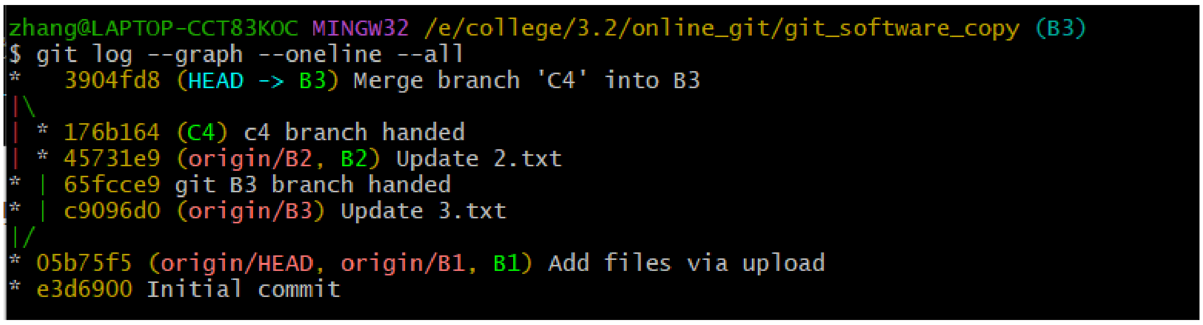
- 查看目前哪些分支已经合并、哪些分支尚未合并；

  使用`git branch --merged`指令查看已合并分支，使用`git branch --no-merged`指令查看未合并分支：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git branch --merged
  B1
  B2
* B3
  C4

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git branch --no-merged
```

发现合并问题，B1不应该在合并分支中，选择使用`git log --graph --oneline --all`查看分支集成情况：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B3)
$ git log --graph --oneline --all
*   3904fd8 (HEAD -> B3) Merge branch 'C4' into B3
|\
| * 176b164 (C4) c4 branch handed
| * 45731e9 (origin/B2, B2) Update 2.txt
* | 65fcce9 git B3 branch handed
* | c9096d0 (origin/B3) Update 3.txt
|/
* 05b75f5 (origin/HEAD, origin/B1, B1) Add files via upload
* e3d6900 Initial commit
```

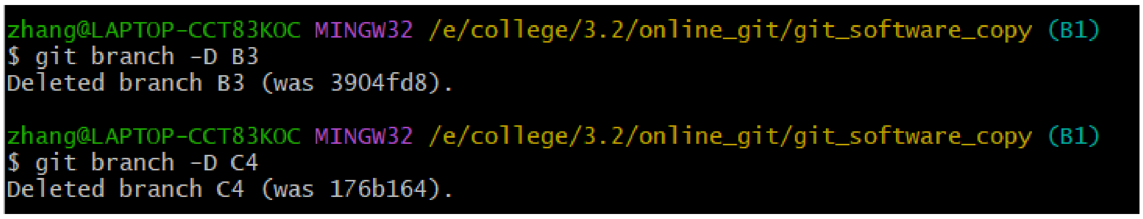
可以看到根据逻辑，B3和C4应当已经合并，B1与B2未参与合并。

R15：分支归属修改

- 将C4和B3合并后的分支删除，将尚未合并的分支合并到一个新分支上，分支名字为你的学号；

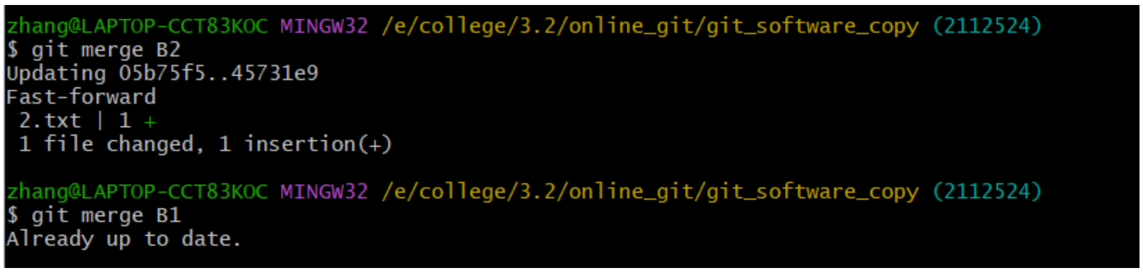
  我们要想删除分支，需要切换到其他分支，这里我们选择切换到B1分支（根节点）

  使用`git branch -D C4`和`git branch -D B3`删除分支：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B1)
$ git branch -D B3
Deleted branch B3 (was 3904fd8).

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (B1)
$ git branch -D C4
Deleted branch C4 (was 176b164).
```

  创建新分支2112524，将B1和B2分支内容合并到该分支：



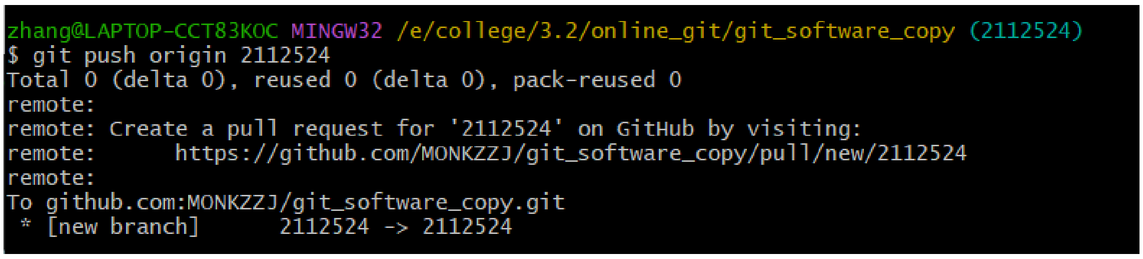
```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (2112524)
$ git merge B2
Updating 05b75f5..45731e9
Fast-forward
 2.txt | 1 +
 1 file changed, 1 insertion(+)

zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (2112524)
$ git merge B1
Already up to date.
```

远程分支管理

R16：本地上传

- 将本地以你的学号命名的分支推送到Github上；

  使用`git push origin 2112524`指令推送到github：



```
zhang@LAPTOP-CCT83KOC MINGW32 /e/college/3.2/online_git/git_software_copy (2112524)
$ git push origin 2112524
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for '2112524' on GitHub by visiting:
remote:      https://github.com/MONKZZJ/git_software_copy/pull/new/2112524
remote:
To github.com:MONKZZJ/git_software_copy.git
 * [new branch]      2112524 -> 2112524
```

R17：操作上传

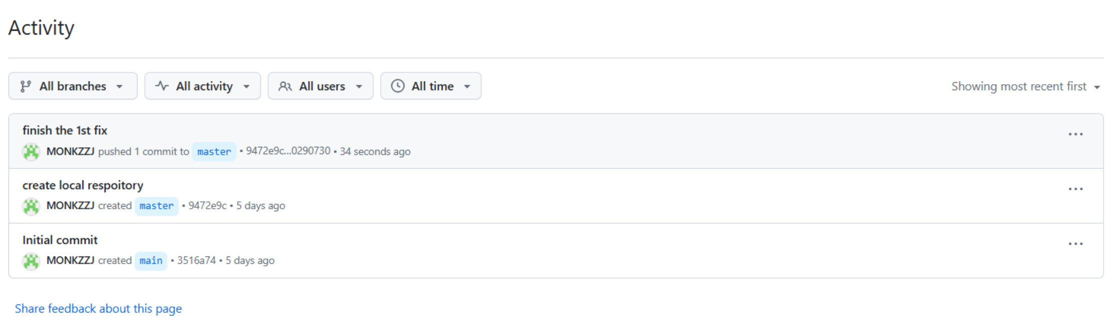
- 将R1到R7各步骤得到的结果推送到Github上；

  同理在R1与R7操作的仓库中添加`git push origin master`指令推送到原分支：

R18：查看状态

- 在Github网站以web页面的方式查看你的两个仓库的当前状态。

仓库（创建和提交）结果

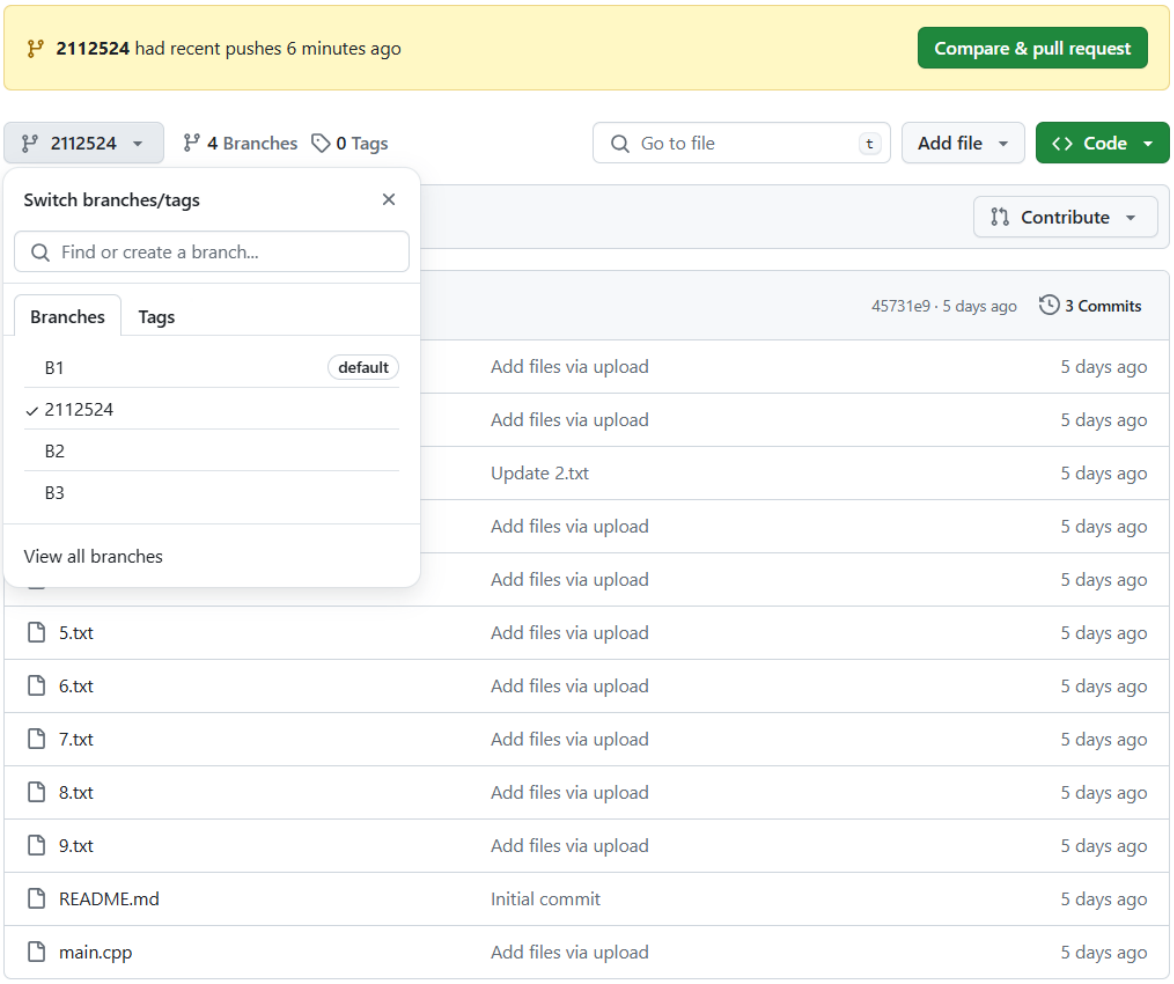


finish the 1st fix
MONKZZJ pushed 1 commit to master • 9472e9c...0290730 • 34 seconds ago

create local respoitory
MONKZZJ created master • 9472e9c • 5 days ago

Initial commit
MONKZZJ created main • 3516a74 • 5 days ago

仓库（分支管理）结果


2112524 had recent pushes 6 minutes ago

Compare & pull request

2112524 · 4 Branches · 0 Tags · Go to file · Add file · Code

Switch branches/tags

Find or create a branch...

Branches · Tags

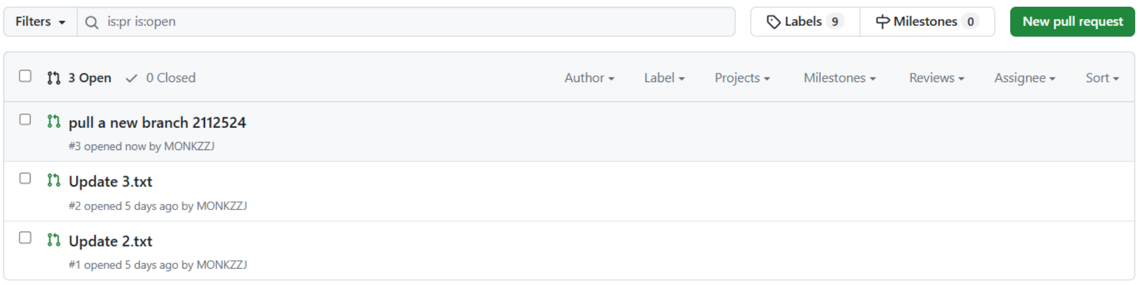
- B1 default
- ✓ 2112524
- B2
- B3

View all branches

Contribute

45731e9 · 5 days ago · 3 Commits

	Add files via upload	5 days ago
	Add files via upload	5 days ago
	Update 2.txt	5 days ago
	Add files via upload	5 days ago
	Add files via upload	5 days ago
5.txt	Add files via upload	5 days ago
6.txt	Add files via upload	5 days ago
7.txt	Add files via upload	5 days ago
8.txt	Add files via upload	5 days ago
9.txt	Add files via upload	5 days ago
README.md	Initial commit	5 days ago
main.cpp	Add files via upload	5 days ago


Filters · is:pr is:open · Labels 9 · Milestones 0 · New pull request

3 Open · 0 Closed · Author · Label · Projects · Milestones · Reviews · Assignee · Sort

pull a new branch 2112524
#3 opened now by MONKZZJ

Update 3.txt
#2 opened 5 days ago by MONKZZJ

Update 2.txt
#1 opened 5 days ago by MONKZZJ

由于我们只提交了新分支2112524，所以在远程仓库中并没有B3和C4的后续操作情况。

小结

实验反思

实验过程因为之前电脑本地存储有中国 Gitee 平台的 SSH 公钥，并用于 git 操作台，导致在 Github 上进行 git 操作需要临时建立会话来保证运行，而且因为在运行时多采用了强制更新命令，对个人开发也许影响不大但是在团队开发中一定会造成恶劣影响。

思考题

1. **比较之前的开发经验，使用git的优点？**

　　使用Git可以更方便的管理工程的进度，记录里程碑等信息，在对应里程碑处做好备份防止后期出现不可逆的损坏造成重新开始的惨 剧，这时只需要根据git恢复断点数据就可以了。

2. **在个人开发和团队开发中，git起到的作用有何主要差异？**

对于个人开发：

- Git允许开发者在本地轻松地保存和恢复代码的不同版本，帮助管理和跟踪代码变化。
- 开发者可以创建多个分支进行不同的实验，安全地测试新功能或修复bug而不影响主代码库。
- 通过将本地仓库推送到远程仓库（如GitHub、GitLab），开发者可以确保代码不会因硬件故障丢失。

对于团队开发：

- 团队成员可以在不同的分支上独立工作，进行代码的并行开发，减少冲突和依赖问题。
- Git的pull request（合并请求）功能允许团队成员相互审查代码，确保代码质量和一致性。

  > 这里需要注意的是我在进行实验的时候使用临时会话所以在Git看来我是不拥有该仓库的"其他成员"，尽管我使用的SSH公钥就是本人......所以在实验中需要进行pull request。

- Git与多种CI/CD工具和项目管理工具集成，提升团队协作效率。

3. **之前是否用过其他的版本控制软件？如果有，同git相比有哪些优缺点？如果没有查阅资料对比一下不同版本控制系统的差别。**

之前的Gitee本质上是一样的，控制台都是可以使用Git bash。

不同版本的控制系统还有 SVN，Mercurial和 Perforce等：

- **SVN：**
  - Git是分布式的，每个开发者有完整的仓库副本；SVN是集中式的，所有开发者从单一中央仓库检出代码。
  - Git的分支操作快速高效且空间占用小，合并能力强；SVN的分支操作较慢且占用更多空间。
  - Git支持多种工作流（如Git Flow），SVN的工作流较为固定。
  - Git在处理大项目和大量文件时性能优越，SVN在小项目中性能也不错。
- **Mercurial：**

  两者都是分布式版本控制系统，都支持高效的分支和合并操作。

  - Mercurial相对更易于学习和使用，其命令和行为对新手更友好；Git具有更丰富的功能和配置选项。
  - Git的社区和第三方工具支持更加广泛，应用更加普遍。
- **Perforce：**
  - Perforce是集中式版本控制系统，适合大型企业和对集中管理有需求的团队；Git是分布式的。
  - Perforce在处理大型代码库和二进制文件时表现出色，Git在分布式开发和灵活工作流上更有优势。

4. **在什么情况下适合使用git、什么情况下没必要使用git？**

   在分布式团队处理项目或是大型，复杂项目适合使用git进行管理；

   项目团队规模较小且集中，或者需要用户权限控制和版本控制时，需要更加专业的代码管理工具如perforce等。