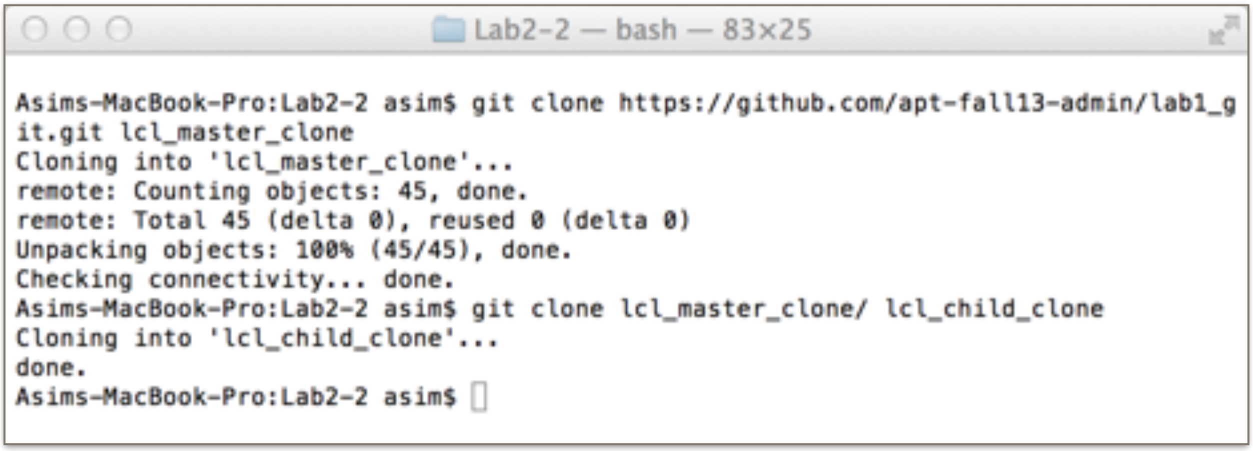
Version Control With Git

1.1 Copy HelloWorld.java over to a new file called HelloWorldEn.java in the same HelloWorld project.
Edit HelloWorld.java to have it print “Hallo Welt” (German for “Hello World”). You will see both the files in the “Unstaged Changes” section of “GIT Staging”
Investigate the differences between the original version (in the repository) and your German variant.
Commit and push the files in git.
Copy the German variant to a file named HelloWorldDe.java, revert HelloWorld.java and commit all files in.

Solution:

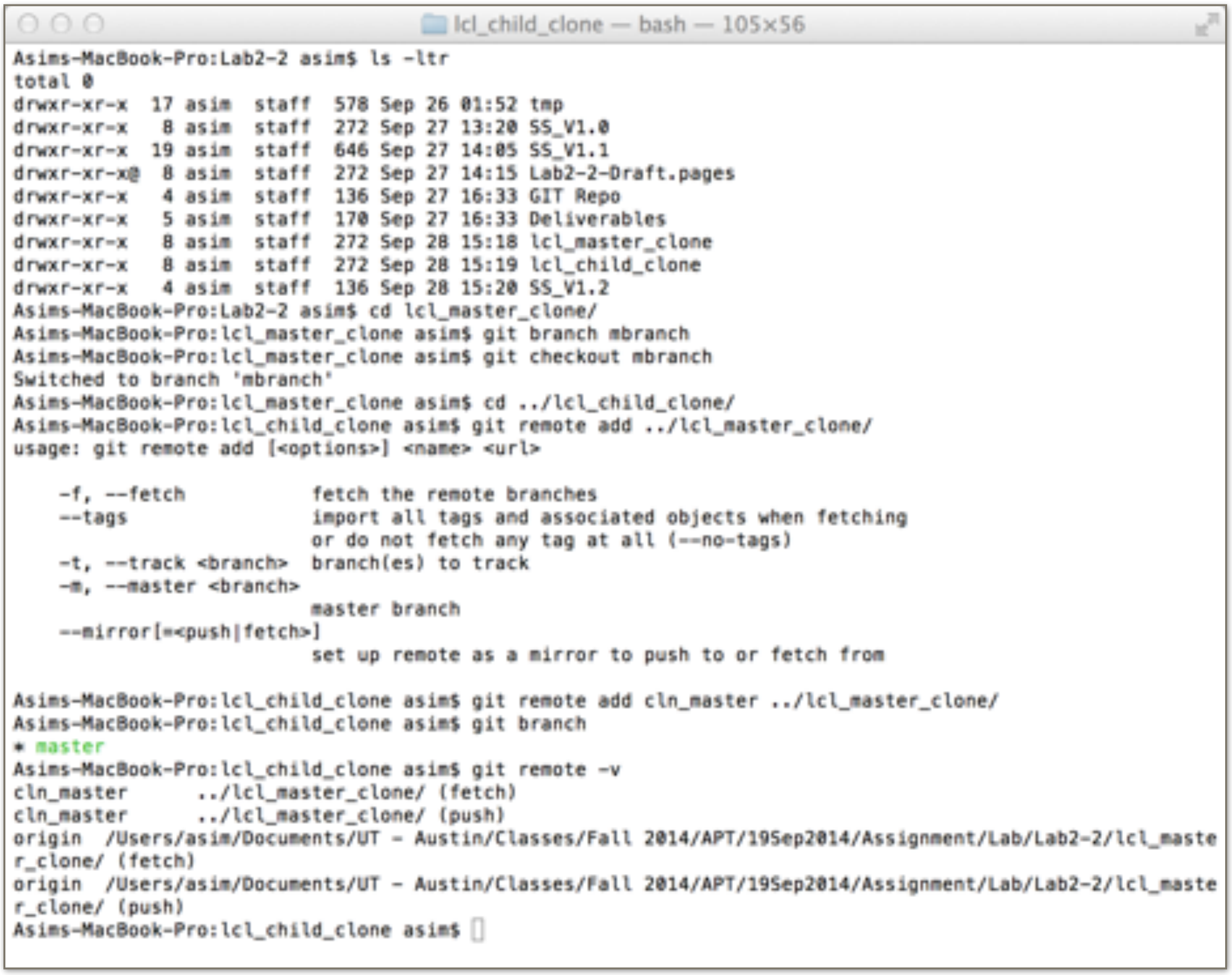
The initial figures contain the steps I performed in order to setup the environment to carry out the requirements for 1.1. I first cloned the repository into a local folder using the git clone command wherein it moved all the code into this folder. However this local clone had the origin as the main repository. In order to avoid pushing anything to that, we create another clone from the first one. The first clone will be referred to as Parent clone and the second one as Child clone from now on, going forward in this document



```
Asims-MacBook-Pro:Lab2-2 asim$ git clone https://github.com/apt-fall13-admin/lab1_g
it.git lcl_master_clone
Cloning into 'lcl_master_clone'...
remote: Counting objects: 45, done.
remote: Total 45 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (45/45), done.
Checking connectivity... done.
Asims-MacBook-Pro:Lab2-2 asim$ git clone lcl_master_clone/ lcl_child_clone
Cloning into 'lcl_child_clone'...
done.
Asims-MacBook-Pro:Lab2-2 asim$ 
```

Then we have to setup the Parent clone directory as the origin for the second clone, Child clone. For the purpose of doing this, we first create a branch in the Parent clone and checkout that branch. We then go back to the Child clone and do a git operation for remote add of the Parent Clone as the origin.

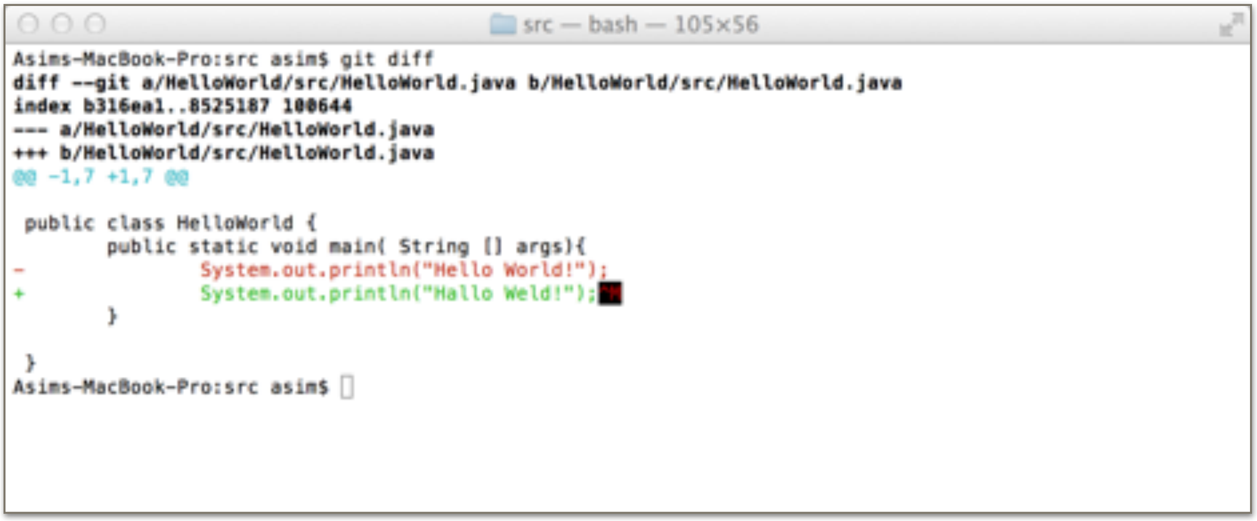
We can run the corresponding “git remote -v” command to ensure that this mapping has gone through.



```
Asims-MacBook-Pro:Lab2-2 asim$ ls -ltr
total 0
drwxr-xr-x  17 asim  staff  578 Sep 26 01:52 tmp
drwxr-xr-x   8 asim  staff  272 Sep 27 13:20 SS_V1.0
drwxr-xr-x  19 asim  staff  646 Sep 27 14:05 SS_V1.1
drwxr-xr-x@  8 asim  staff  272 Sep 27 14:15 Lab2-2-Draft.pages
drwxr-xr-x   4 asim  staff  136 Sep 27 16:33 GIT Repo
drwxr-xr-x   5 asim  staff  170 Sep 27 16:33 Deliverables
drwxr-xr-x   8 asim  staff  272 Sep 28 15:18 lcl_master_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 15:19 lcl_child_clone
drwxr-xr-x   4 asim  staff  136 Sep 28 15:20 SS_V1.2
Asims-MacBook-Pro:Lab2-2 asim$ cd lcl_master_clone/
Asims-MacBook-Pro:lcl_master_clone asim$ git branch mbranch
Asims-MacBook-Pro:lcl_master_clone asim$ git checkout mbranch
Switched to branch 'mbranch'
Asims-MacBook-Pro:lcl_master_clone asim$ cd ../lcl_child_clone/
Asims-MacBook-Pro:lcl_child_clone asim$ git remote add ../lcl_master_clone/
usage: git remote add [<options>] <name> <url>

    -f, --fetch           fetch the remote branches
    --tags                import all tags and associated objects when fetching
                          or do not fetch any tag at all (--no-tags)
    -t, --track <branch>  branch(es) to track
    -m, --master <branch>
                          master branch
    --mirror[=<push|fetch>]
                          set up remote as a mirror to push to or fetch from

Asims-MacBook-Pro:lcl_child_clone asim$ git remote add cln_master ../lcl_master_clone/
Asims-MacBook-Pro:lcl_child_clone asim$ git branch
* master
Asims-MacBook-Pro:lcl_child_clone asim$ git remote -v
cln_master      ../lcl_master_clone/ (fetch)
cln_master      ../lcl_master_clone/ (push)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:lcl_child_clone asim$ 
```

After the initial setups are completed, the file HelloWorld.java is copied over into HelloWorldEn.java. It is then edited to print “Hallo Welt” instead of “Hell World”.

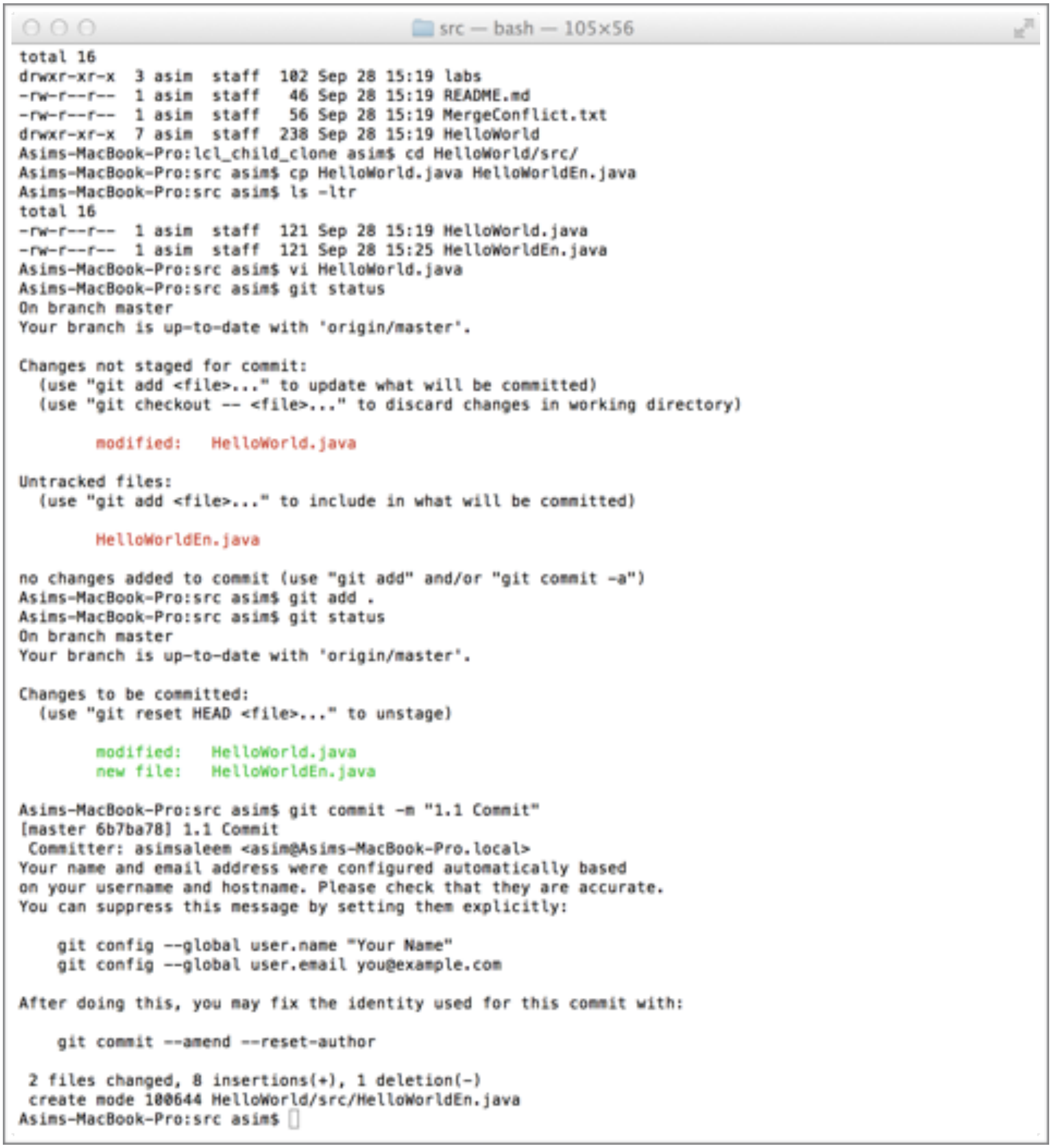
Once the changes are made and the file is saved, a quick diff gives us an indication of the changes we have made



```
Asims-MacBook-Pro:src asim$ git diff
diff --git a/HelloWorld/src/HelloWorld.java b/HelloWorld/src/HelloWorld.java
index b316ea1..8525187 100644
--- a/HelloWorld/src/HelloWorld.java
+++ b/HelloWorld/src/HelloWorld.java
@@ -1,7 +1,7 @@

 public class HelloWorld {
        public static void main( String [] args){
-               System.out.println("Hello World!");
+               System.out.println("Hallo Weld!");^M
        }

 }
Asims-MacBook-Pro:src asim$ 
```

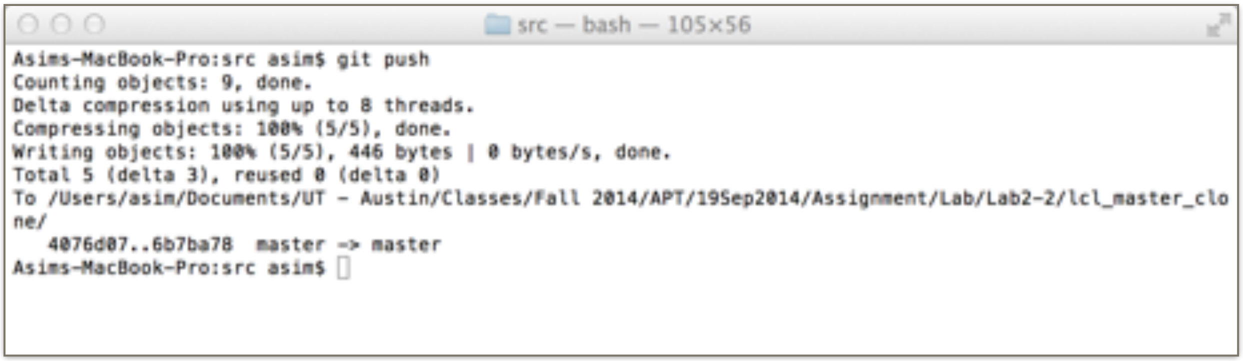
It can also be noticed that the modified filed HelloWorld.java and the newly created file “HelloWorldEn.java” appear in the Unstaged changes section of “Git Staging”

The files are added for the purpose of Committing and committed as shown below:



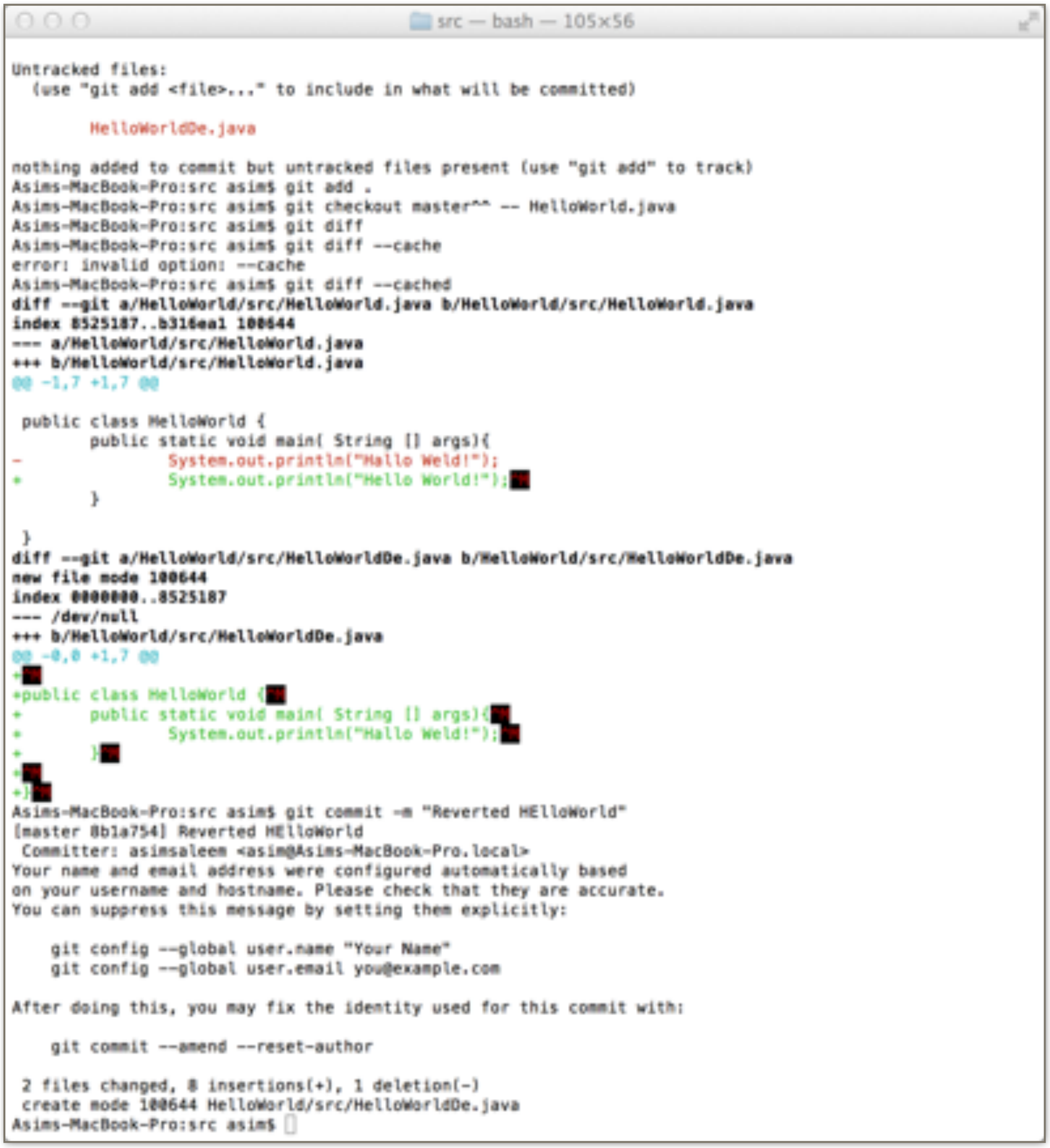
```
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 15:19 labs
-rw-r--r--  1 asim  staff   46 Sep 28 15:19 README.md
-rw-r--r--  1 asim  staff   56 Sep 28 15:19 MergeConflict.txt
drwxr-xr-x  7 asim  staff  238 Sep 28 15:19 HelloWorld
Asims-MacBook-Pro:lcl_child_clone asim$ cd HelloWorld/src/
Asims-MacBook-Pro:src asim$ cp HelloWorld.java HelloWorldEn.java
Asims-MacBook-Pro:src asim$ ls -ltr
total 16
-rw-r--r--  1 asim  staff  121 Sep 28 15:19 HelloWorld.java
-rw-r--r--  1 asim  staff  121 Sep 28 15:25 HelloWorldEn.java
Asims-MacBook-Pro:src asim$ vi HelloWorld.java
Asims-MacBook-Pro:src asim$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   HelloWorld.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        HelloWorldEn.java

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:src asim$ git add .
Asims-MacBook-Pro:src asim$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   HelloWorld.java
        new file:   HelloWorldEn.java

Asims-MacBook-Pro:src asim$ git commit -m "1.1 Commit"
[master 6b7ba78] 1.1 Commit
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 2 files changed, 8 insertions(+), 1 deletion(-)
 create mode 100644 HelloWorld/src/HelloWorldEn.java
Asims-MacBook-Pro:src asim$
```

After a Commit is done, the file is then Pushed to the Parent clone



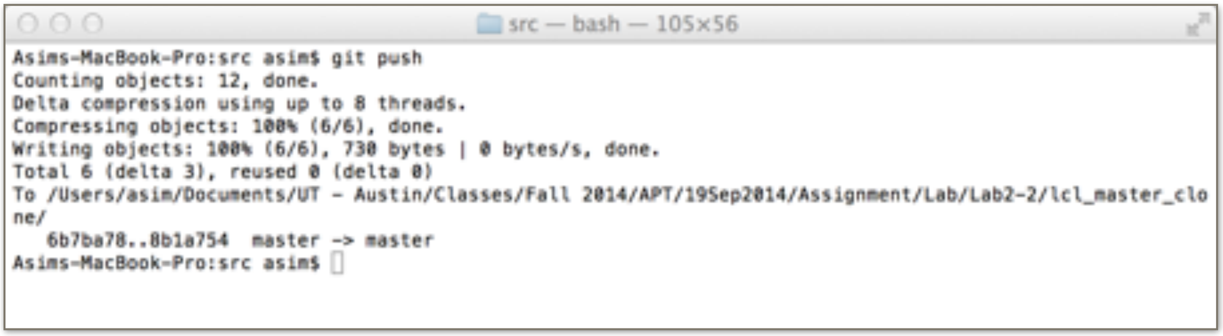
```
Asims-MacBook-Pro:src asim$ git push
Counting objects: 9, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 446 bytes | 0 bytes/s, done.
Total 5 (delta 3), reused 0 (delta 0)
To /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_master_clo
ne/
   4076d07..6b7ba78  master -> master
Asims-MacBook-Pro:src asim$ 
```

Once the file is pushed, we create a new German variant of the file called “HelloWorldDe.java” by making a copy of the current version of “HelloWorld.java”. The modified “HelloWorld.java is reverted back to its original version displaying “Hello Wordl!” jn English. After this the files are all committed in as show below.



```
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        HelloWorldDe.java

nothing added to commit but untracked files present (use "git add" to track)
Asims-MacBook-Pro:src asim$ git add .
Asims-MacBook-Pro:src asim$ git checkout master^^ -- HelloWorld.java
Asims-MacBook-Pro:src asim$ git diff
Asims-MacBook-Pro:src asim$ git diff --cache
error: invalid option: --cache
Asims-MacBook-Pro:src asim$ git diff --cached
diff --git a/HelloWorld/src/HelloWorld.java b/HelloWorld/src/HelloWorld.java
index 8525187..b316ea1 100644
--- a/HelloWorld/src/HelloWorld.java
+++ b/HelloWorld/src/HelloWorld.java
@@ -1,7 +1,7 @@

 public class HelloWorld {
        public static void main( String [] args){
-               System.out.println("Hallo Weld!");
+               System.out.println("Hello World!");^M
        }

 }
diff --git a/HelloWorld/src/HelloWorldDe.java b/HelloWorld/src/HelloWorldDe.java
new file mode 100644
index 0000000..8525187
--- /dev/null
+++ b/HelloWorld/src/HelloWorldDe.java
@@ -0,0 +1,7 @@
+^M
+public class HelloWorld {^M
+       public static void main( String [] args){^M
+               System.out.println("Hallo Weld!");^M
+       }^M
+^M
+}^M
Asims-MacBook-Pro:src asim$ git commit -m "Reverted HElloWorld"
[master 8b1a754] Reverted HElloWorld
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 2 files changed, 8 insertions(+), 1 deletion(-)
 create mode 100644 HelloWorld/src/HelloWorldDe.java
Asims-MacBook-Pro:src asim$ 
```

The files that are committed in are then pushed to the repository using the git push command



src — bash — 105×56

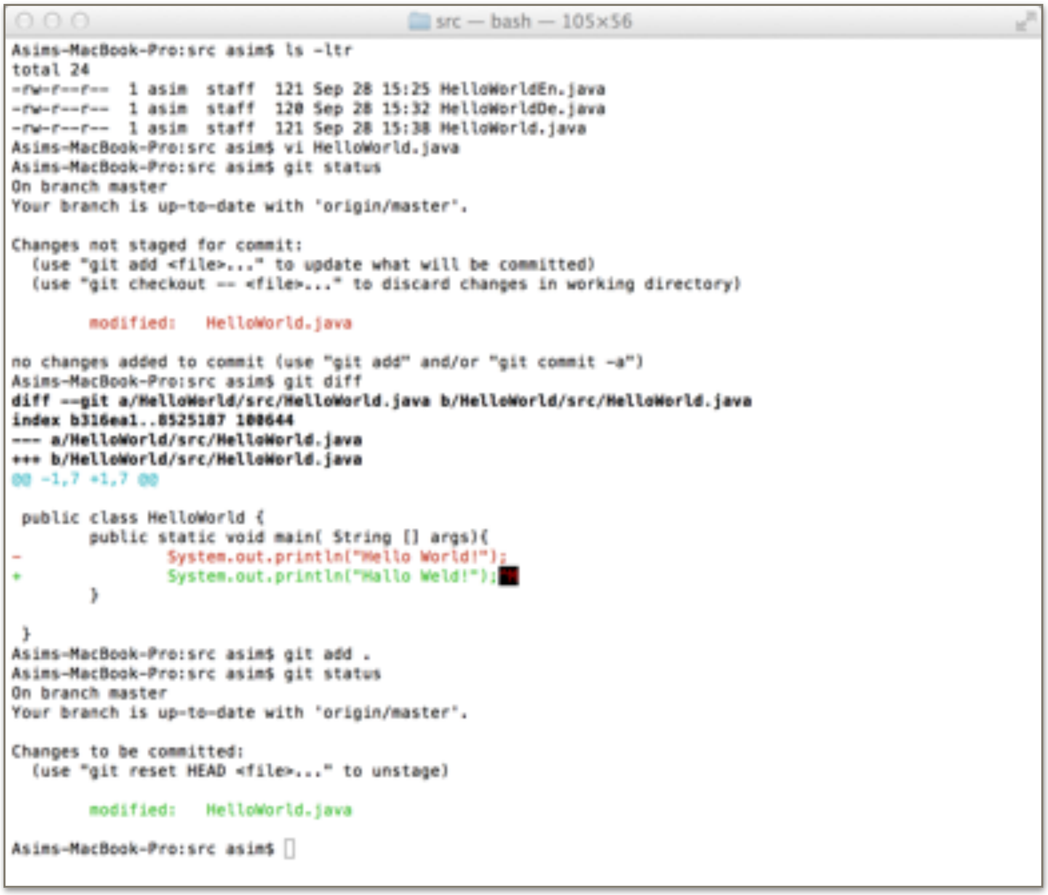
```
Asims-MacBook-Pro:src asim$ git push
Counting objects: 12, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 730 bytes | 0 bytes/s, done.
Total 6 (delta 3), reused 0 (delta 0)
To /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_master_clo
ne/
   6b7ba78..8b1a754  master -> master
Asims-MacBook-Pro:src asim$ 
```

With this step all the requirements expected are met.

1.2 Create a patch for the HelloWorld.java to convert it to the German variant. There are many ways you can do this. Name the patch HelloWorldDe.patch. Apply this patch to HelloWorld.java and see if it works.
Create a corresponding patch HelloWorldFr.patch which translates the Hello World message into French ("Bonjour, monde").What happens if you try to use both patches one after the other? Explain.
Check in all the variants and the patches. *(8 marks)*

Solution:

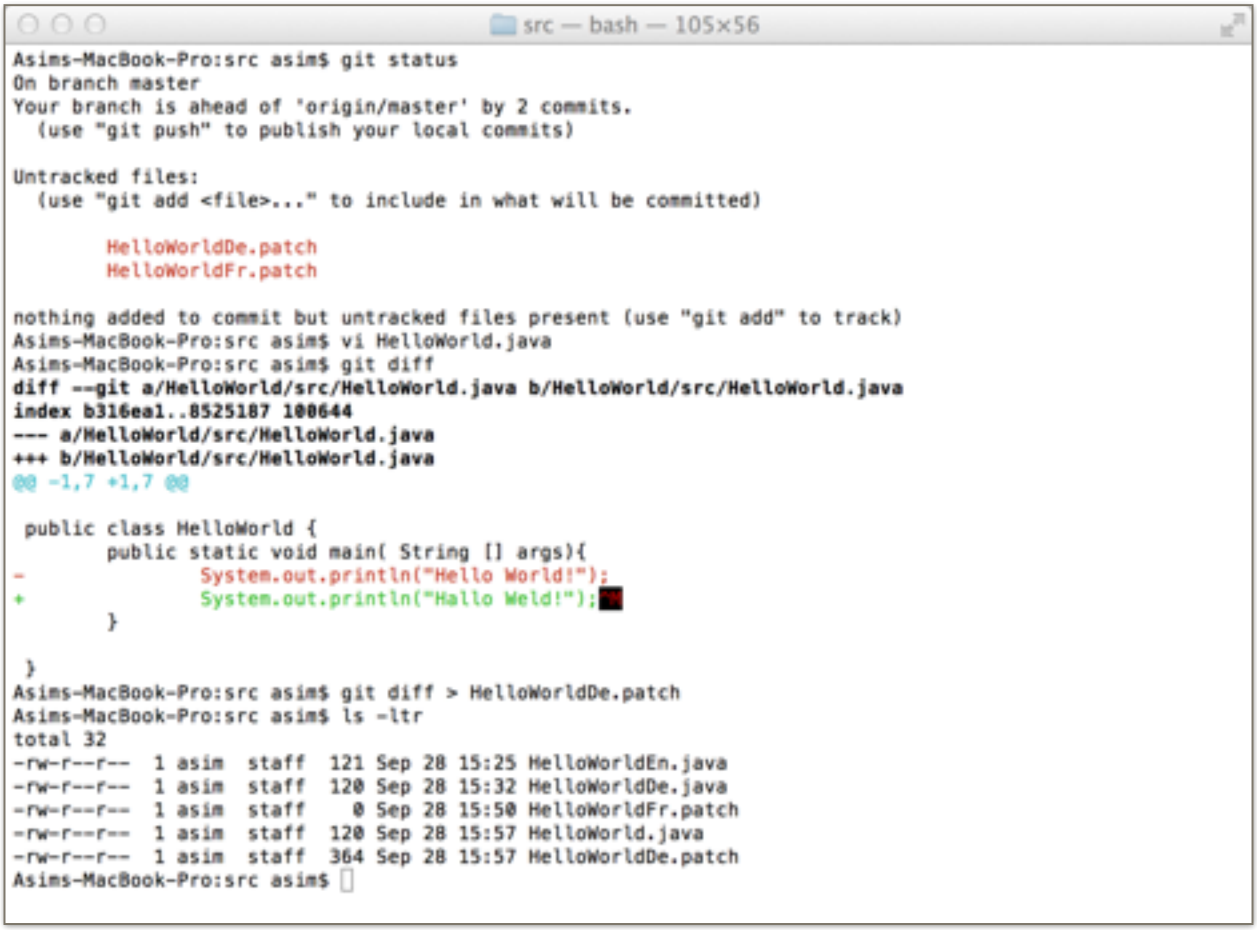
In order to create a patch for HelloWorld.java, I had to carry out the following steps. To start with, we modify the original file to constitute the German version.



src — bash — 105×56

```
Asims-MacBook-Pro:src asim$ ls -ltr
total 24
-rw-r--r--  1 asim  staff  121 Sep 28 15:25 HelloWorldEn.java
-rw-r--r--  1 asim  staff  120 Sep 28 15:32 HelloWorldDe.java
-rw-r--r--  1 asim  staff  121 Sep 28 15:38 HelloWorld.java
Asims-MacBook-Pro:src asim$ vi HelloWorld.java
Asims-MacBook-Pro:src asim$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   HelloWorld.java

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:src asim$ git diff
diff --git a/HelloWorld/src/HelloWorld.java b/HelloWorld/src/HelloWorld.java
index b316ea1..8525187 100644
--- a/HelloWorld/src/HelloWorld.java
+++ b/HelloWorld/src/HelloWorld.java
@@ -1,7 +1,7 @@

 public class HelloWorld {
        public static void main( String [] args){
-               System.out.println("Hello World!");
+               System.out.println("Hallo Weld!");^M
        }

 }
Asims-MacBook-Pro:src asim$ git add .
Asims-MacBook-Pro:src asim$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   HelloWorld.java

Asims-MacBook-Pro:src asim$ 
```

A git diff shown in the screenshot illustrates that the changes made are available in the modified file. This file is then added to the Staging area by executing the “git add .” command and then committed.

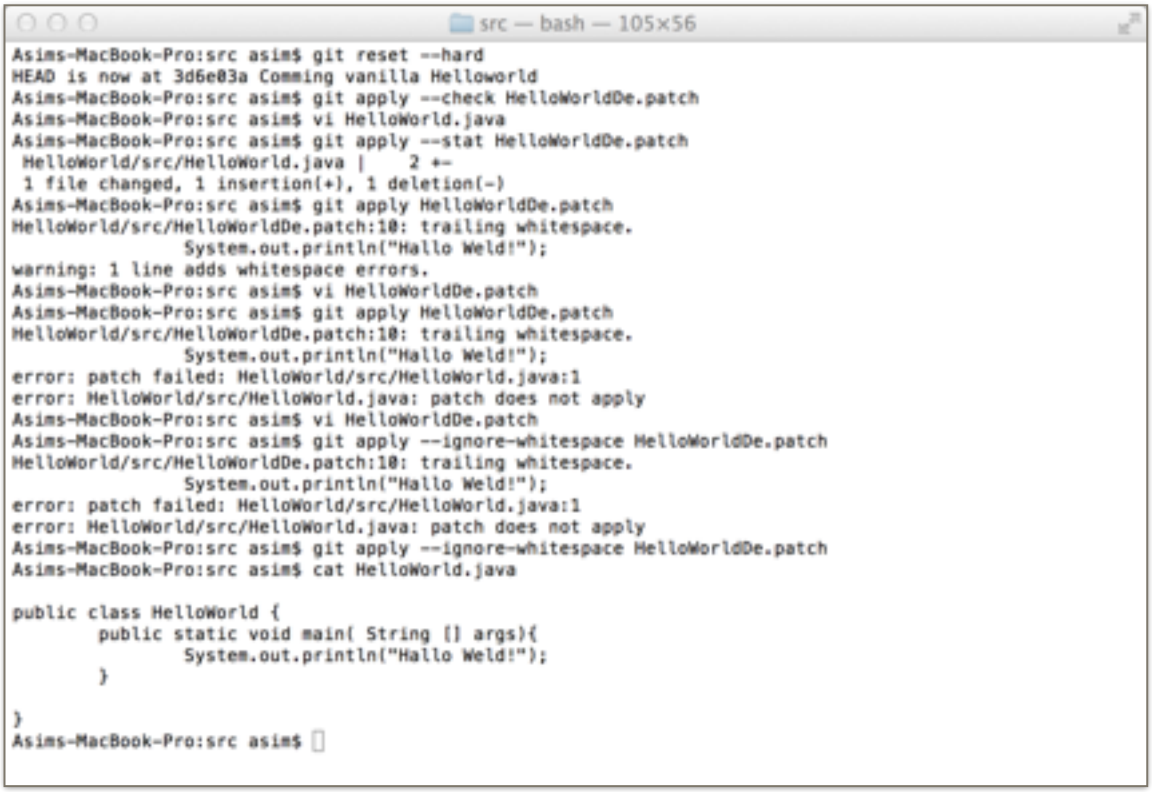


```
Asims-MacBook-Pro:src asim$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        HelloWorldDe.patch
        HelloWorldFr.patch

nothing added to commit but untracked files present (use "git add" to track)
Asims-MacBook-Pro:src asim$ vi HelloWorld.java
Asims-MacBook-Pro:src asim$ git diff
diff --git a/HelloWorld/src/HelloWorld.java b/HelloWorld/src/HelloWorld.java
index b316ea1..8525187 100644
--- a/HelloWorld/src/HelloWorld.java
+++ b/HelloWorld/src/HelloWorld.java
@@ -1,7 +1,7 @@

 public class HelloWorld {
        public static void main( String [] args){
-               System.out.println("Hello World!");
+               System.out.println("Hallo Weld!");^M
        }

 }
Asims-MacBook-Pro:src asim$ git diff > HelloWorldDe.patch
Asims-MacBook-Pro:src asim$ ls -ltr
total 32
-rw-r--r--  1 asim  staff  121 Sep 28 15:25 HelloWorldEn.java
-rw-r--r--  1 asim  staff  120 Sep 28 15:32 HelloWorldDe.java
-rw-r--r--  1 asim  staff    0 Sep 28 15:50 HelloWorldFr.patch
-rw-r--r--  1 asim  staff  120 Sep 28 15:57 HelloWorld.java
-rw-r--r--  1 asim  staff  364 Sep 28 15:57 HelloWorldDe.patch
Asims-MacBook-Pro:src asim$ 
```

A new patch file called “HelloWorldDe.patch” is then generated by executing the command “git diff > HelloWorldDe.patch”. Once a patch is generated, the HelloWorld.java code is reset back to the original version. This is done in order to apply the German patch.

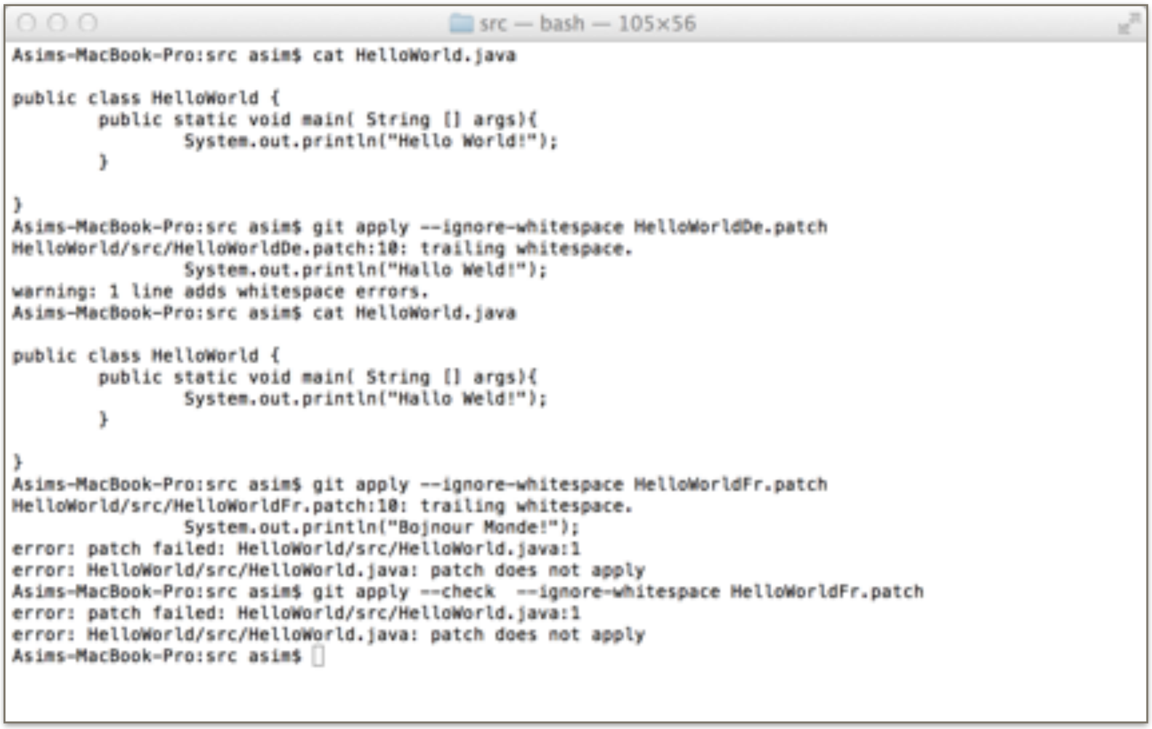
The outcome of applying the patch is first checked by executing the git apply command with the —check flag and then the files that will be impacted by applying the patch are checked by running the same git apply command with the —stat flag. Once this is done, the patch is finally applied. However it is noticed that it fails the first time around because of a trailing white space problem. This problem seems to occur depending on the environment used to generate the patch.

The patch is then re-executed with the —ignore-whitespace flag and even though an error is thrown, it is observed that the patch is applied correctly. The HelloWorld.java code now contains the german wordings.



```
Asims-MacBook-Pro:src asim$ git reset --hard
HEAD is now at 3d6e03a Comming vanilla Helloworld
Asims-MacBook-Pro:src asim$ git apply --check HelloWorldDe.patch
Asims-MacBook-Pro:src asim$ vi HelloWorld.java
Asims-MacBook-Pro:src asim$ git apply --stat HelloWorldDe.patch
 HelloWorld/src/HelloWorld.java |    2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
Asims-MacBook-Pro:src asim$ git apply HelloWorldDe.patch
HelloWorld/src/HelloWorldDe.patch:10: trailing whitespace.
                System.out.println("Hallo Weld!");
warning: 1 line adds whitespace errors.
Asims-MacBook-Pro:src asim$ vi HelloWorldDe.patch
Asims-MacBook-Pro:src asim$ git apply HelloWorldDe.patch
HelloWorld/src/HelloWorldDe.patch:10: trailing whitespace.
                System.out.println("Hallo Weld!");
error: patch failed: HelloWorld/src/HelloWorld.java:1
error: HelloWorld/src/HelloWorld.java: patch does not apply
Asims-MacBook-Pro:src asim$ vi HelloWorldDe.patch
Asims-MacBook-Pro:src asim$ git apply --ignore-whitespace HelloWorldDe.patch
HelloWorld/src/HelloWorldDe.patch:10: trailing whitespace.
                System.out.println("Hallo Weld!");
error: patch failed: HelloWorld/src/HelloWorld.java:1
error: HelloWorld/src/HelloWorld.java: patch does not apply
Asims-MacBook-Pro:src asim$ git apply --ignore-whitespace HelloWorldDe.patch
Asims-MacBook-Pro:src asim$ cat HelloWorld.java

public class HelloWorld {
        public static void main( String [] args){
                System.out.println("Hallo Weld!");
        }

}
Asims-MacBook-Pro:src asim$ 
```

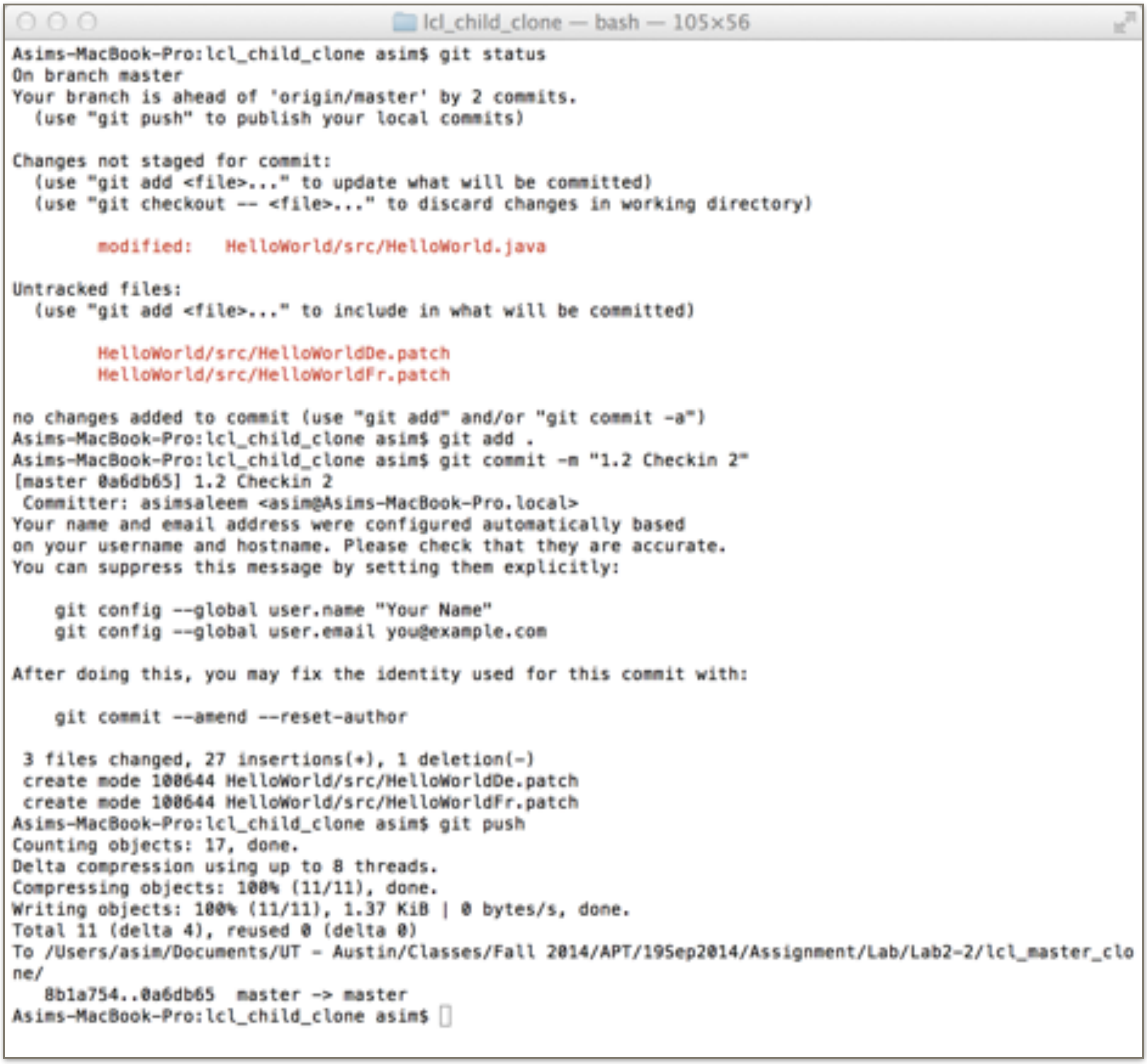
Now we reset the code for HelloWorld.java and then using that as the basis we create a French version of the patch called “HelloWorldFr.patch”.



```
Asims-MacBook-Pro:src asim$ cat HelloWorld.java

public class HelloWorld {
        public static void main( String [] args){
                System.out.println("Hello World!");
        }

}
Asims-MacBook-Pro:src asim$ git apply --ignore-whitespace HelloWorldDe.patch
HelloWorld/src/HelloWorldDe.patch:10: trailing whitespace.
                System.out.println("Hallo Weld!");
warning: 1 line adds whitespace errors.
Asims-MacBook-Pro:src asim$ cat HelloWorld.java

public class HelloWorld {
        public static void main( String [] args){
                System.out.println("Hallo Weld!");
        }

}
Asims-MacBook-Pro:src asim$ git apply --ignore-whitespace HelloWorldFr.patch
HelloWorld/src/HelloWorldFr.patch:10: trailing whitespace.
                System.out.println("Bojnour Monde!");
error: patch failed: HelloWorld/src/HelloWorld.java:1
error: HelloWorld/src/HelloWorld.java: patch does not apply
Asims-MacBook-Pro:src asim$ git apply --check  --ignore-whitespace HelloWorldFr.patch
error: patch failed: HelloWorld/src/HelloWorld.java:1
error: HelloWorld/src/HelloWorld.java: patch does not apply
Asims-MacBook-Pro:src asim$ 
```

When we try to apply the French patch after applying the German patch it fails. The reason this happens is because when the French patch is generated, the HelloWorld.java commit it is referring to has only the English version. Now when the German patch is applied and the HelloWorld.java code is updated, the French patch has no reference to this change. As a result when it is applied, it fails.

This is the reason why applying the French patch after the German patch does not work. After doing all this, the files are all checked in and pushed

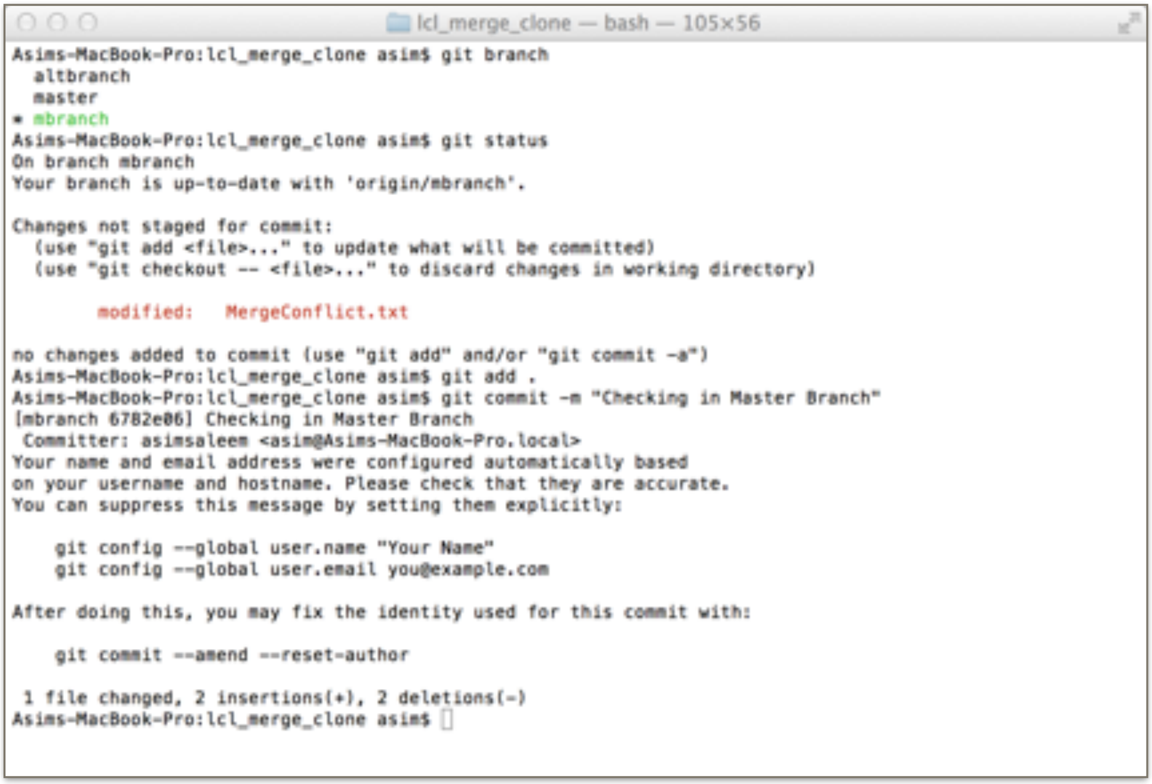


```
Asims-MacBook-Pro:lcl_child_clone asim$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   HelloWorld/src/HelloWorld.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        HelloWorld/src/HelloWorldDe.patch
        HelloWorld/src/HelloWorldFr.patch

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:lcl_child_clone asim$ git add .
Asims-MacBook-Pro:lcl_child_clone asim$ git commit -m "1.2 Checkin 2"
[master 0a6db65] 1.2 Checkin 2
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 3 files changed, 27 insertions(+), 1 deletion(-)
 create mode 100644 HelloWorld/src/HelloWorldDe.patch
 create mode 100644 HelloWorld/src/HelloWorldFr.patch
Asims-MacBook-Pro:lcl_child_clone asim$ git push
Counting objects: 17, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 1.37 KiB | 0 bytes/s, done.
Total 11 (delta 4), reused 0 (delta 0)
To /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_master_clo
ne/
   8b1a754..0a6db65  master -> master
Asims-MacBook-Pro:lcl_child_clone asim$ 
```

1.3 Edit the file MergeConflict.txt and replace the first line
Jack and Jill
by
Jane and Joe
Create a branch from the HEAD version. In the branch, edit MergeConflict.txt and replace the last line
Red Hat, Green Hat
by
Blue Hat, Purple Hat.
Commit the first set of changes to MergeConflict.txt (on master), then the changes on the branch. Finally merge the changes from the branch. Now "Cherry Pick" the original MergeConflict.txt commit into the HEAD of your master. Explain what happens, why it happens, and how you resolved the situation. Push in all changes to your remote repository. *(10 marks)*

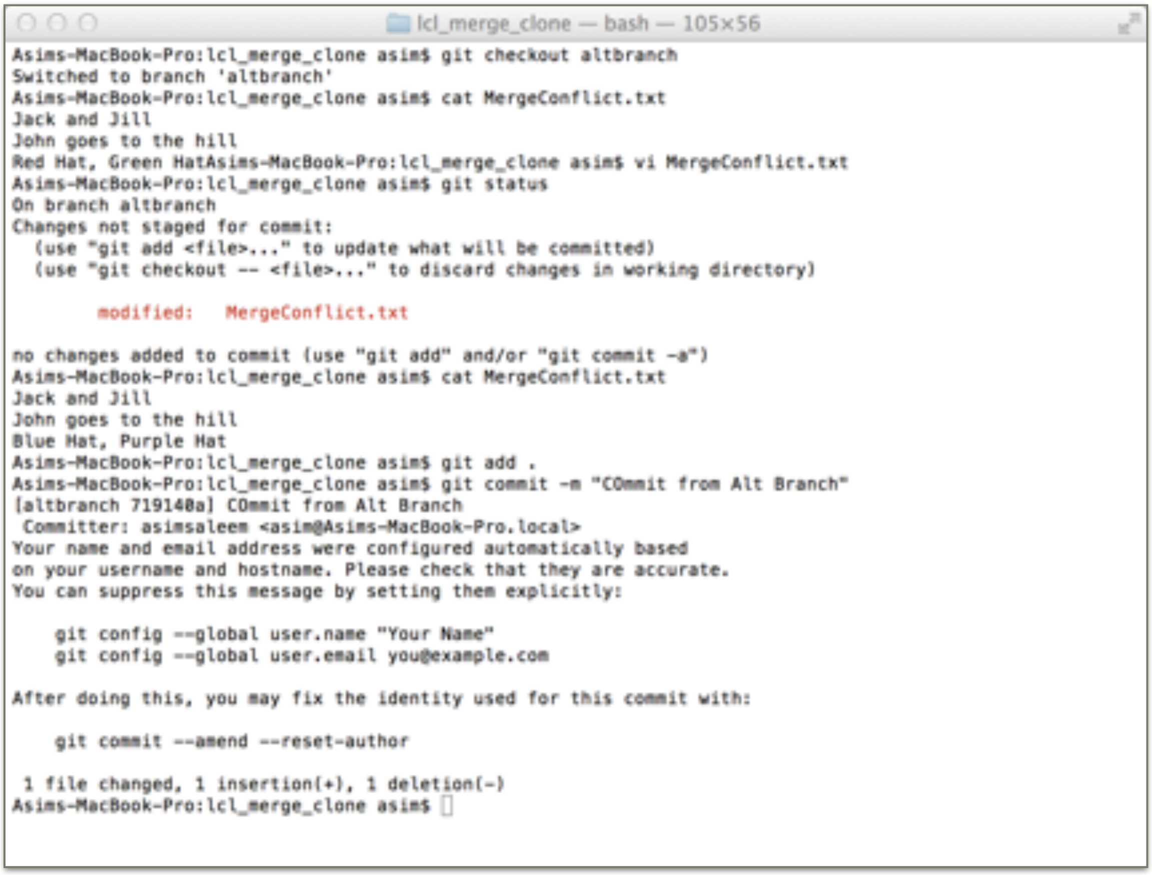
Solution:

For the purpose of this solution, we first modify the MergeConflict.txt by replacing "Jack and Jill" with "Jane and Joe" in the HEAD and then commit the file in



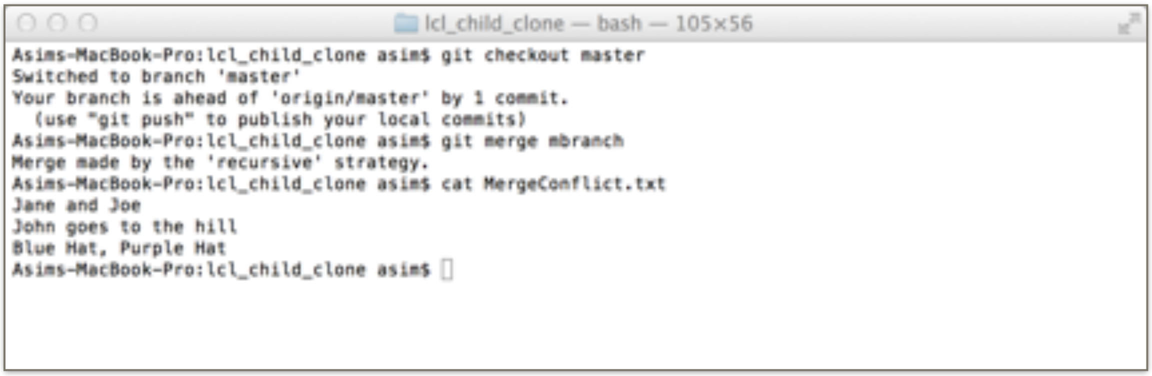
```
Asims-MacBook-Pro:lcl_merge_clone asim$ git branch
  altbranch
  master
* mbranch
Asims-MacBook-Pro:lcl_merge_clone asim$ git status
On branch mbranch
Your branch is up-to-date with 'origin/mbranch'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MergeConflict.txt

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:lcl_merge_clone asim$ git add .
Asims-MacBook-Pro:lcl_merge_clone asim$ git commit -m "Checking in Master Branch"
[mbranch 6782e06] Checking in Master Branch
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 2 insertions(+), 2 deletions(-)
Asims-MacBook-Pro:lcl_merge_clone asim$ 
```

Once the commit is performed, we then create a branch from here and checkout that branch. In this checked out branch, we make the other expected change, which is to replace "Red Hat, Green Hat" with "Blue Hat, Purple Hat"



```
Asims-MacBook-Pro:lcl_merge_clone asim$ git checkout altbranch
Switched to branch 'altbranch'
Asims-MacBook-Pro:lcl_merge_clone asim$ cat MergeConflict.txt
Jack and Jill
John goes to the hill
Red Hat, Green HatAsims-MacBook-Pro:lcl_merge_clone asim$ vi MergeConflict.txt
Asims-MacBook-Pro:lcl_merge_clone asim$ git status
On branch altbranch
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MergeConflict.txt

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:lcl_merge_clone asim$ cat MergeConflict.txt
Jack and Jill
John goes to the hill
Blue Hat, Purple Hat
Asims-MacBook-Pro:lcl_merge_clone asim$ git add .
Asims-MacBook-Pro:lcl_merge_clone asim$ git commit -m "COmmit from Alt Branch"
[altbranch 719140a] COmmit from Alt Branch
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+), 1 deletion(-)
Asims-MacBook-Pro:lcl_merge_clone asim$ 
```

Now the changes from the checkout branch are then committed. After this we go back to the initial branch from where this branch was created in the first place. We then check out that branch to "cherry pick" and noticed that it is taken care of by Git by using the recursive strategy and then merge this branch in

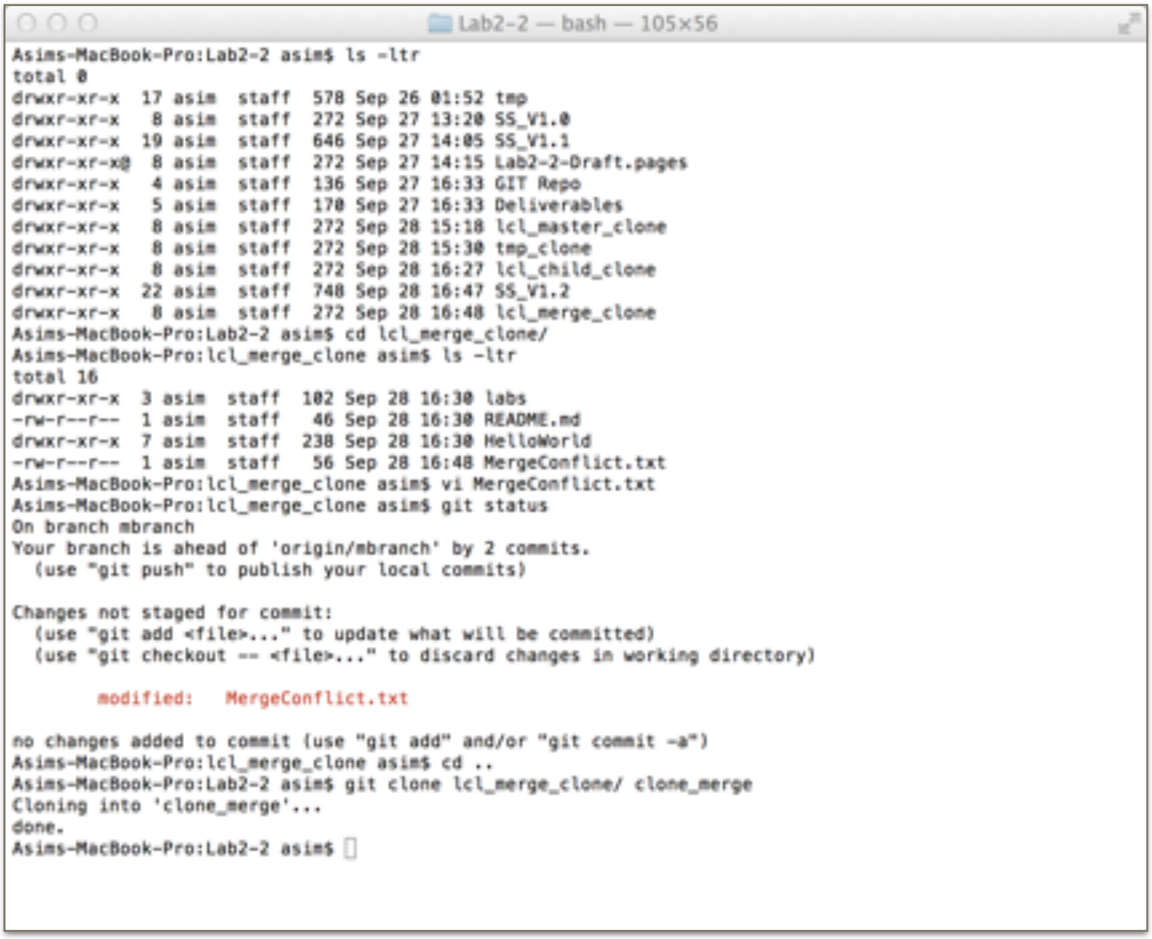


```
Asims-MacBook-Pro:lcl_child_clone asim$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
Asims-MacBook-Pro:lcl_child_clone asim$ git merge mbranch
Merge made by the 'recursive' strategy.
Asims-MacBook-Pro:lcl_child_clone asim$ cat MergeConflict.txt
Jane and Joe
John goes to the hill
Blue Hat, Purple Hat
Asims-MacBook-Pro:lcl_child_clone asim$ 
```

1.4 Start with the original MergeConflict.txt. Now delete the last line,
Red Hat, Green Hat
Clone your lab1_git directory in a separate place on your computer.
Commit and push the first set of changes to MergeConflict.txt.
How will you undo those changes from the second directory? Is it possible to force your original file through without pulling in the new changes? Explain. (6 marks)

Solution:

As a first step to this process, we edit the original MergeConflict.txt and delete the last line. After the line is deleted, a new clone is created from the Child clone and this is where the next set of changes expected would be carried out.

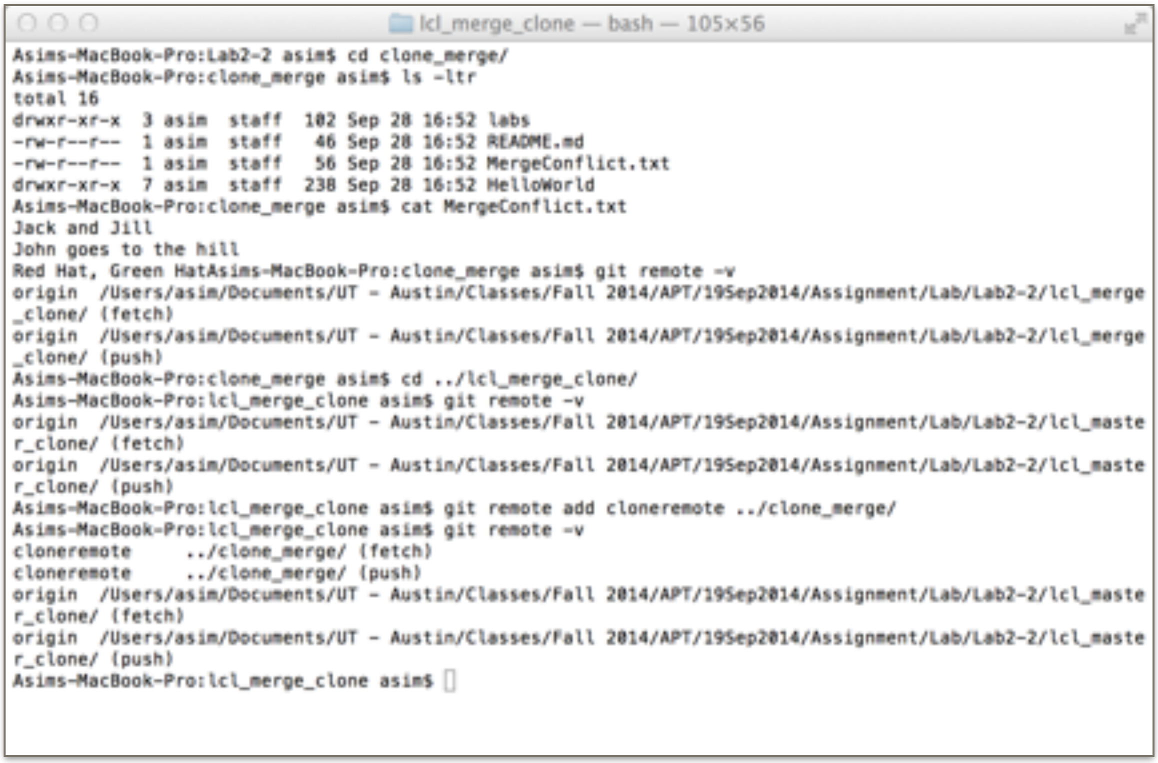


Lab2-2 — bash — 105×56

```
Asims-MacBook-Pro:Lab2-2 asim$ ls -ltr
total 0
drwxr-xr-x  17 asim  staff  578 Sep 26 01:52 tmp
drwxr-xr-x   8 asim  staff  272 Sep 27 13:20 SS_V1.0
drwxr-xr-x  19 asim  staff  646 Sep 27 14:05 SS_V1.1
drwxr-xr-x@  8 asim  staff  272 Sep 27 14:15 Lab2-2-Draft.pages
drwxr-xr-x   4 asim  staff  136 Sep 27 16:33 GIT Repo
drwxr-xr-x   5 asim  staff  170 Sep 27 16:33 Deliverables
drwxr-xr-x   8 asim  staff  272 Sep 28 15:18 lcl_master_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 15:30 tmp_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 16:27 lcl_child_clone
drwxr-xr-x  22 asim  staff  748 Sep 28 16:47 SS_V1.2
drwxr-xr-x   8 asim  staff  272 Sep 28 16:48 lcl_merge_clone
Asims-MacBook-Pro:Lab2-2 asim$ cd lcl_merge_clone/
Asims-MacBook-Pro:lcl_merge_clone asim$ ls -ltr
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 16:30 labs
-rw-r--r--  1 asim  staff   46 Sep 28 16:30 README.md
drwxr-xr-x  7 asim  staff  238 Sep 28 16:30 HelloWorld
-rw-r--r--  1 asim  staff   56 Sep 28 16:48 MergeConflict.txt
Asims-MacBook-Pro:lcl_merge_clone asim$ vi MergeConflict.txt
Asims-MacBook-Pro:lcl_merge_clone asim$ git status
On branch mbranch
Your branch is ahead of 'origin/mbranch' by 2 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MergeConflict.txt

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:lcl_merge_clone asim$ cd ..
Asims-MacBook-Pro:Lab2-2 asim$ git clone lcl_merge_clone/ clone_merge
Cloning into 'clone_merge'...
done.
Asims-MacBook-Pro:Lab2-2 asim$ 
```

The new clone is created and we set up the remote origin for this clone as the parent from which it was cloned from. The status of the remote is checked by using the “git remote -v” command and once verified, we are ready to carry out the next set of operations.

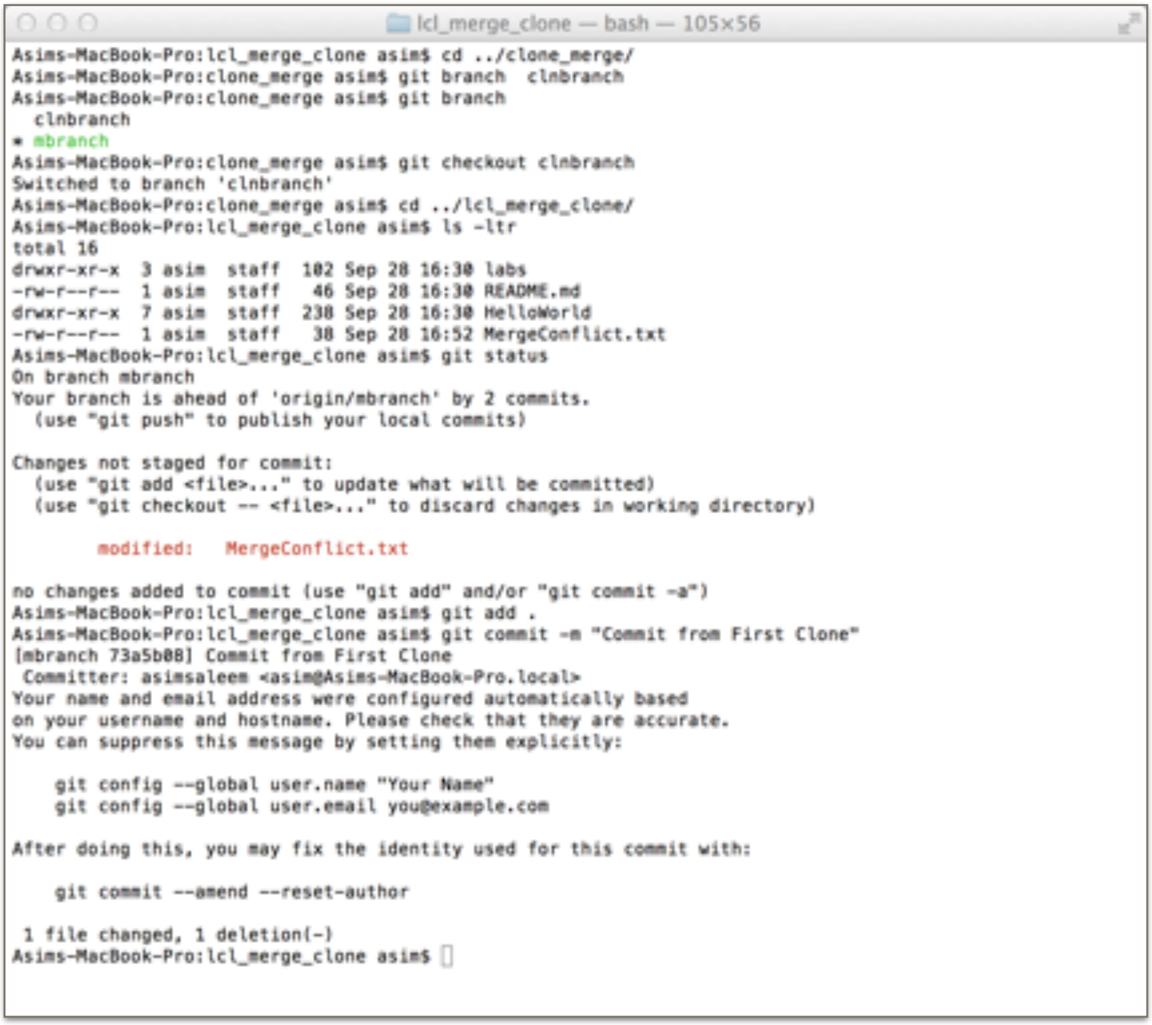
The screenshot of this setup is provided below:



```
Asims-MacBook-Pro:Lab2-2 asim$ cd clone_merge/
Asims-MacBook-Pro:clone_merge asim$ ls -ltr
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 16:52 labs
-rw-r--r--  1 asim  staff   46 Sep 28 16:52 README.md
-rw-r--r--  1 asim  staff   56 Sep 28 16:52 MergeConflict.txt
drwxr-xr-x  7 asim  staff  238 Sep 28 16:52 HelloWorld
Asims-MacBook-Pro:clone_merge asim$ cat MergeConflict.txt
Jack and Jill
John goes to the hill
Red Hat, Green HatAsims-MacBook-Pro:clone_merge asim$ git remote -v
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_merge
_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_merge
_clone/ (push)
Asims-MacBook-Pro:clone_merge asim$ cd ../lcl_merge_clone/
Asims-MacBook-Pro:lcl_merge_clone asim$ git remote -v
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:lcl_merge_clone asim$ git remote add cloneremote ../clone_merge/
Asims-MacBook-Pro:lcl_merge_clone asim$ git remote -v
cloneremote     ../clone_merge/ (fetch)
cloneremote     ../clone_merge/ (push)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:lcl_merge_clone asim$ 
```

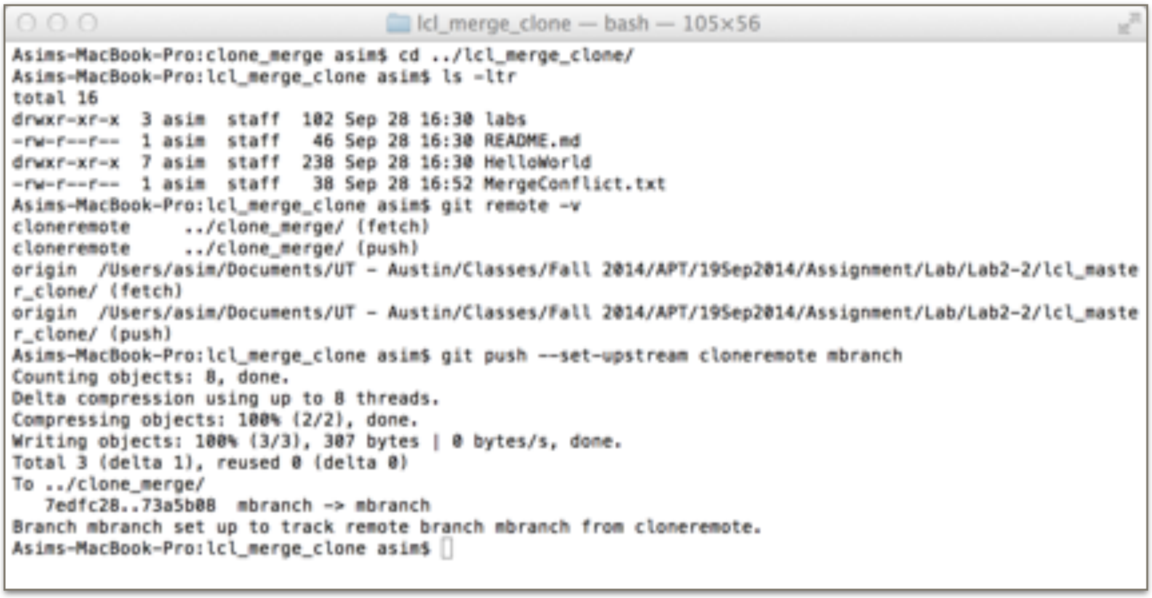
It has to be kept in mind that when the first set of changes were made to the MergeConflict.txt in the parent, it was not committed. In order to do this, we go back to the parent branch of and then commit the version of the MergeConflict.txt with the last line deleted.

Since the file was not committed, it has to be noticed that this change is not visible in the cloned child as yet.



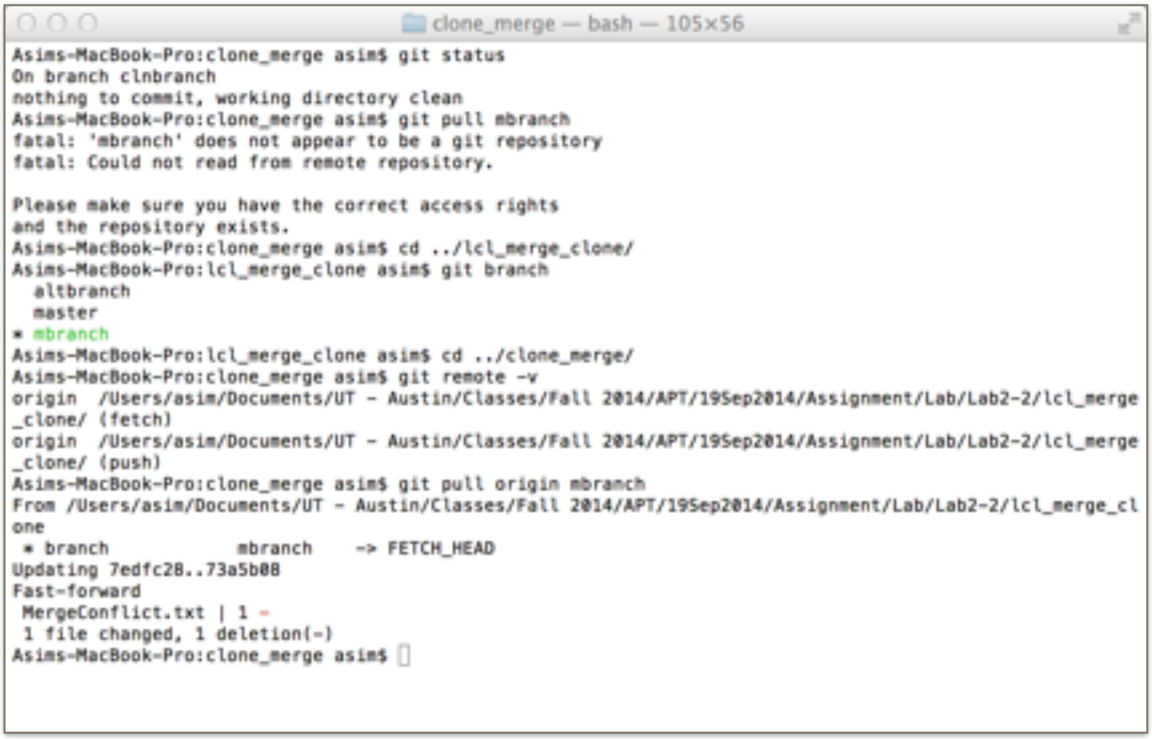
```
Asims-MacBook-Pro:lcl_merge_clone asim$ cd ../clone_merge/
Asims-MacBook-Pro:clone_merge asim$ git branch  clnbranch
Asims-MacBook-Pro:clone_merge asim$ git branch
  clnbranch
* mbranch
Asims-MacBook-Pro:clone_merge asim$ git checkout clnbranch
Switched to branch 'clnbranch'
Asims-MacBook-Pro:clone_merge asim$ cd ../lcl_merge_clone/
Asims-MacBook-Pro:lcl_merge_clone asim$ ls -ltr
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 16:30 labs
-rw-r--r--  1 asim  staff   46 Sep 28 16:30 README.md
drwxr-xr-x  7 asim  staff  238 Sep 28 16:30 HelloWorld
-rw-r--r--  1 asim  staff   38 Sep 28 16:52 MergeConflict.txt
Asims-MacBook-Pro:lcl_merge_clone asim$ git status
On branch mbranch
Your branch is ahead of 'origin/mbranch' by 2 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MergeConflict.txt

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:lcl_merge_clone asim$ git add .
Asims-MacBook-Pro:lcl_merge_clone asim$ git commit -m "Commit from First Clone"
[mbranch 73a5b08] Commit from First Clone
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 deletion(-)
Asims-MacBook-Pro:lcl_merge_clone asim$ 
```

In order to be able to push the code, we make the necessary setup

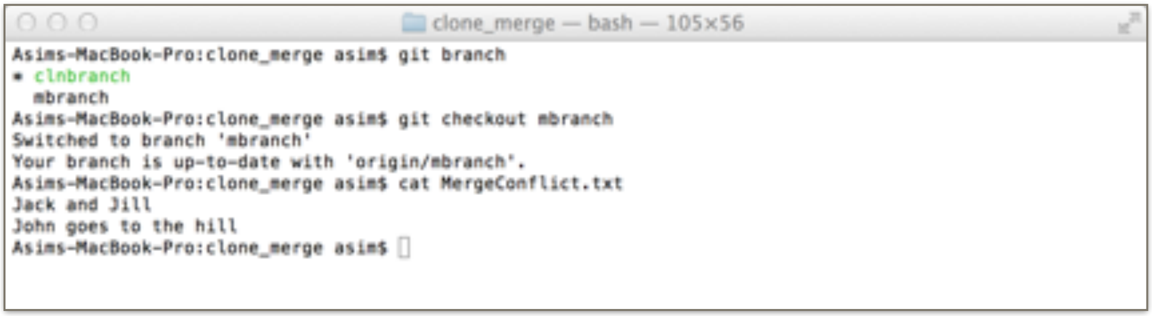


```
Asims-MacBook-Pro:clone_merge asim$ cd ../lcl_merge_clone/
Asims-MacBook-Pro:lcl_merge_clone asim$ ls -ltr
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 16:30 labs
-rw-r--r--  1 asim  staff   46 Sep 28 16:30 README.md
drwxr-xr-x  7 asim  staff  238 Sep 28 16:30 HelloWorld
-rw-r--r--  1 asim  staff   38 Sep 28 16:52 MergeConflict.txt
Asims-MacBook-Pro:lcl_merge_clone asim$ git remote -v
cloneremote     ../clone_merge/ (fetch)
cloneremote     ../clone_merge/ (push)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:lcl_merge_clone asim$ git push --set-upstream cloneremote mbranch
Counting objects: 8, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 307 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To ../clone_merge/
   7edfc28..73a5b08  mbranch -> mbranch
Branch mbranch set up to track remote branch mbranch from cloneremote.
Asims-MacBook-Pro:lcl_merge_clone asim$ 
```

Since the MergeConflict.txt in the parent was modified to remove the last line which did not make it to the newly created clone, we do a "git pull" from the master branch and then we observe that the changes are reflected in the clone.



```
Asims-MacBook-Pro:clone_merge asim$ git status
On branch clnbranch
nothing to commit, working directory clean
Asims-MacBook-Pro:clone_merge asim$ git pull mbranch
fatal: 'mbranch' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
Asims-MacBook-Pro:clone_merge asim$ cd ../lcl_merge_clone/
Asims-MacBook-Pro:lcl_merge_clone asim$ git branch
  altbranch
  master
* mbranch
Asims-MacBook-Pro:lcl_merge_clone asim$ cd ../clone_merge/
Asims-MacBook-Pro:clone_merge asim$ git remote -v
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_merge
_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_merge
_clone/ (push)
Asims-MacBook-Pro:clone_merge asim$ git pull origin mbranch
From /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_merge_cl
one
 * branch            mbranch    -> FETCH_HEAD
Updating 7edfc28..73a5b08
Fast-forward
 MergeConflict.txt | 1 -
 1 file changed, 1 deletion(-)
Asims-MacBook-Pro:clone_merge asim$ 
```



```
Asims-MacBook-Pro:clone_merge asim$ git branch
* clnbranch
  mbranch
Asims-MacBook-Pro:clone_merge asim$ git checkout mbranch
Switched to branch 'mbranch'
Your branch is up-to-date with 'origin/mbranch'.
Asims-MacBook-Pro:clone_merge asim$ cat MergeConflict.txt
Jack and Jill
John goes to the hill
Asims-MacBook-Pro:clone_merge asim$ 
```

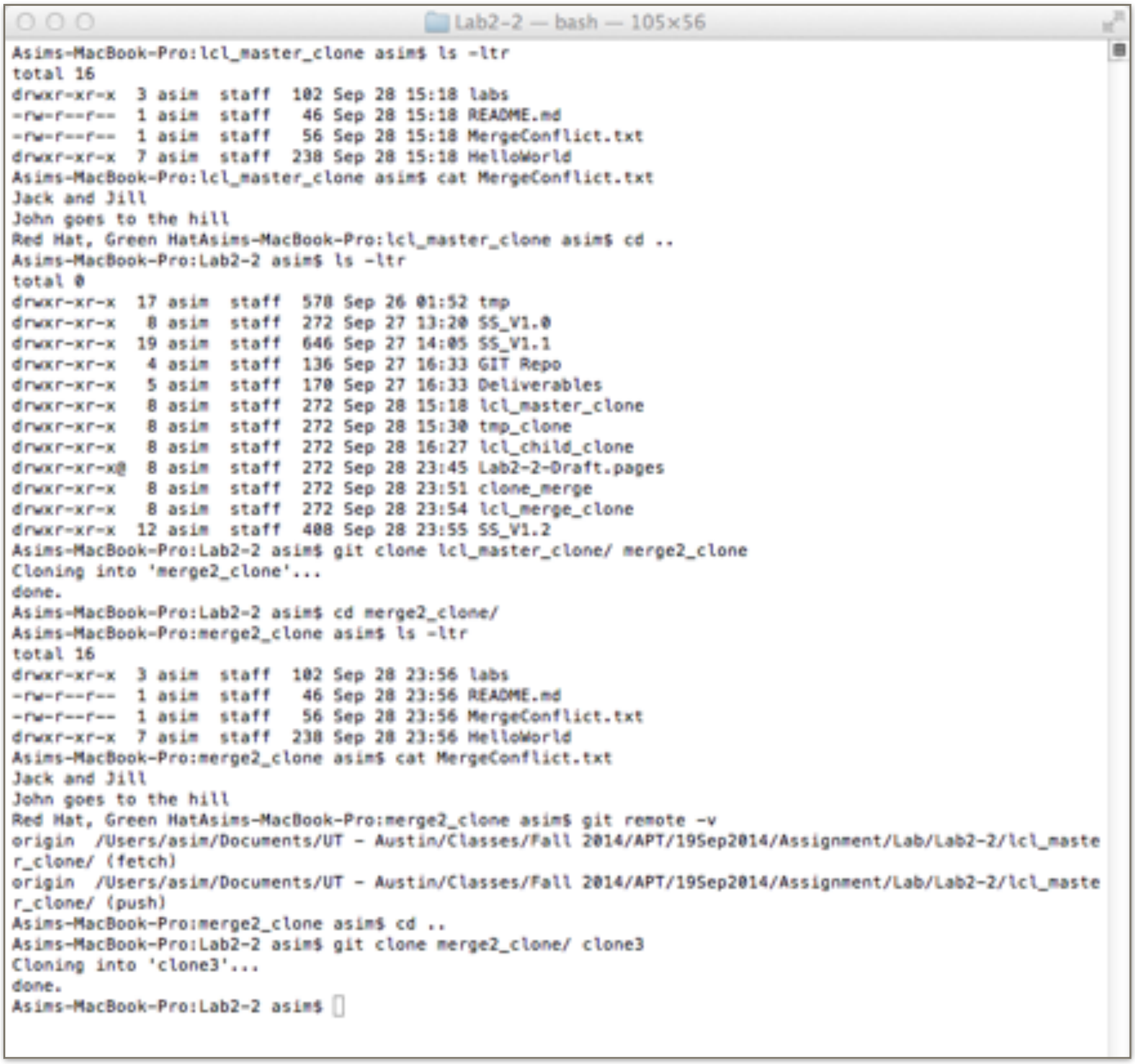
Since the "git pull" command was carried out to pull in the changes, if we need to "undo" these changes, then we would be able to carry that out by executing "git revert" in the clone branch. This will throw away any changes that were pulled in.

On the other hand, to answer the question about whether it is possible to force the original file through without pulling in new changes, yes it is. We can definitely do that by using a git push command with the force flag. This will take care of pushing it, but this is not an option which should be used a lot because it could be possibly dangerous

1.5 Start with the original file MergeConflict.txt. Edit it and replace all the three lines by
Jane and Joe
John goes up the hill
Blue Hat, Purple Ha
From your cloned lab1_git directory , edit the file MergeConflict.txt and replace the line
John goes to the hill
by
John goes down the hill
Commit and push the first set of changes to MergeConflict.txt, then the changes from the second repository. Explain what happens, why it happens, and how you resolved things (7 marks)

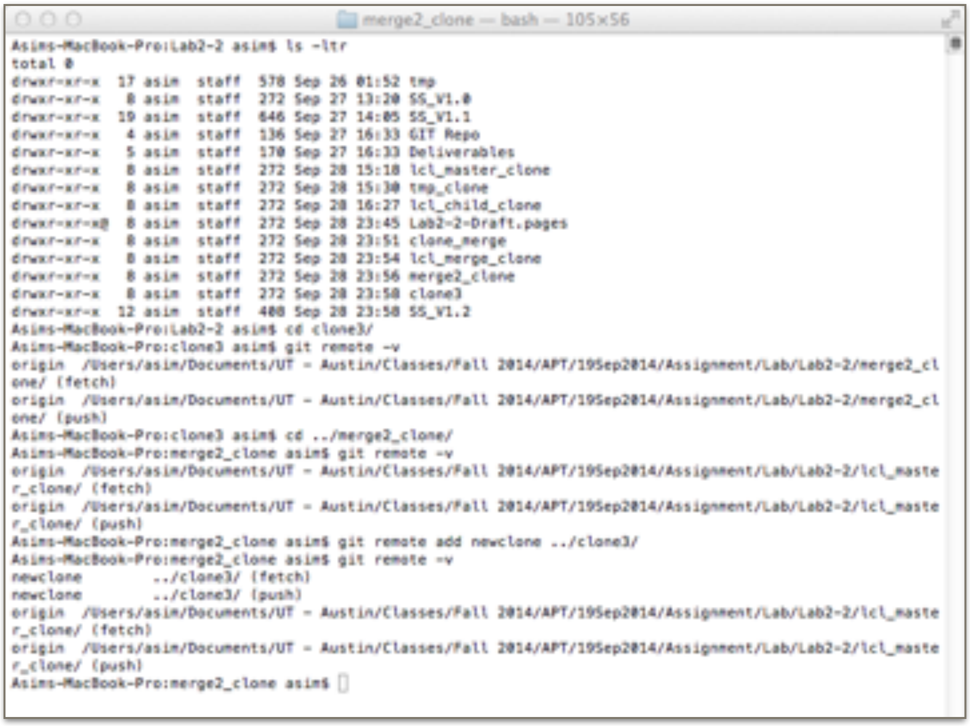
Solution:

In order to carry out this task, we create a new clone which is clean in order to be able to make changes as desired



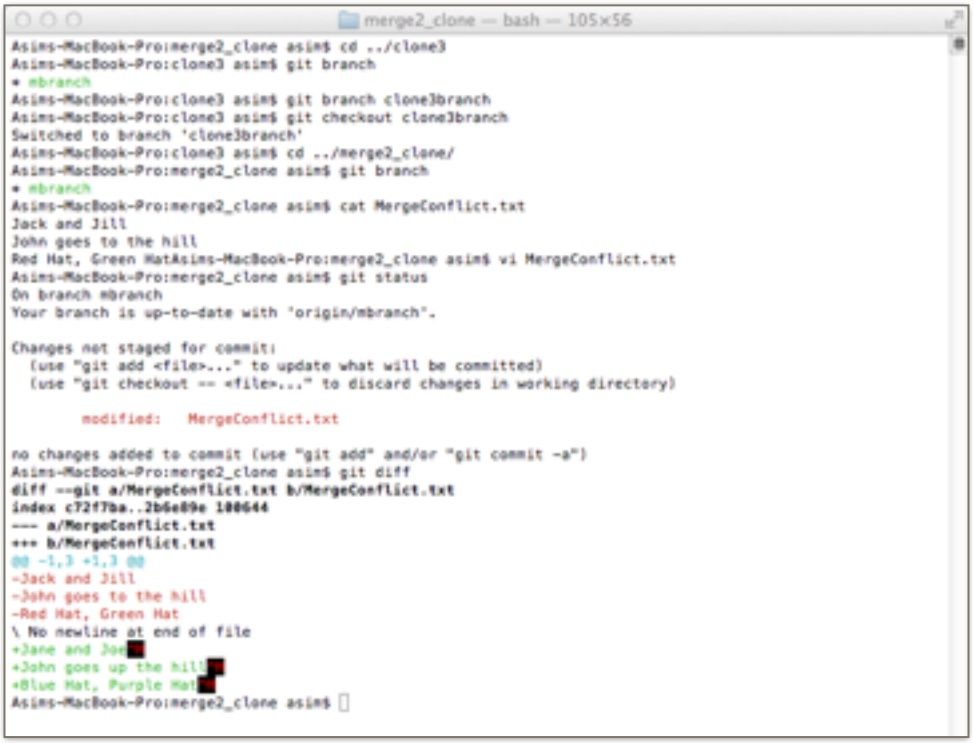
```
Asims-MacBook-Pro:lcl_master_clone asim$ ls -ltr
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 15:18 labs
-rw-r--r--  1 asim  staff   46 Sep 28 15:18 README.md
-rw-r--r--  1 asim  staff   56 Sep 28 15:18 MergeConflict.txt
drwxr-xr-x  7 asim  staff  238 Sep 28 15:18 HelloWorld
Asims-MacBook-Pro:lcl_master_clone asim$ cat MergeConflict.txt
Jack and Jill
John goes to the hill
Red Hat, Green HatAsims-MacBook-Pro:lcl_master_clone asim$ cd ..
Asims-MacBook-Pro:Lab2-2 asim$ ls -ltr
total 0
drwxr-xr-x  17 asim  staff  578 Sep 26 01:52 tmp
drwxr-xr-x   8 asim  staff  272 Sep 27 13:20 SS_V1.0
drwxr-xr-x  19 asim  staff  646 Sep 27 14:05 SS_V1.1
drwxr-xr-x   4 asim  staff  136 Sep 27 16:33 GIT Repo
drwxr-xr-x   5 asim  staff  170 Sep 27 16:33 Deliverables
drwxr-xr-x   8 asim  staff  272 Sep 28 15:18 lcl_master_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 15:30 tmp_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 16:27 lcl_child_clone
drwxr-xr-x@  8 asim  staff  272 Sep 28 23:45 Lab2-2-Draft.pages
drwxr-xr-x   8 asim  staff  272 Sep 28 23:51 clone_merge
drwxr-xr-x   8 asim  staff  272 Sep 28 23:54 lcl_merge_clone
drwxr-xr-x  12 asim  staff  408 Sep 28 23:55 SS_V1.2
Asims-MacBook-Pro:Lab2-2 asim$ git clone lcl_master_clone/ merge2_clone
Cloning into 'merge2_clone'...
done.
Asims-MacBook-Pro:Lab2-2 asim$ cd merge2_clone/
Asims-MacBook-Pro:merge2_clone asim$ ls -ltr
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 23:56 labs
-rw-r--r--  1 asim  staff   46 Sep 28 23:56 README.md
-rw-r--r--  1 asim  staff   56 Sep 28 23:56 MergeConflict.txt
drwxr-xr-x  7 asim  staff  238 Sep 28 23:56 HelloWorld
Asims-MacBook-Pro:merge2_clone asim$ cat MergeConflict.txt
Jack and Jill
John goes to the hill
Red Hat, Green HatAsims-MacBook-Pro:merge2_clone asim$ git remote -v
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:merge2_clone asim$ cd ..
Asims-MacBook-Pro:Lab2-2 asim$ git clone merge2_clone/ clone3
Cloning into 'clone3'...
done.
Asims-MacBook-Pro:Lab2-2 asim$ 
```

We then setup the remote as required



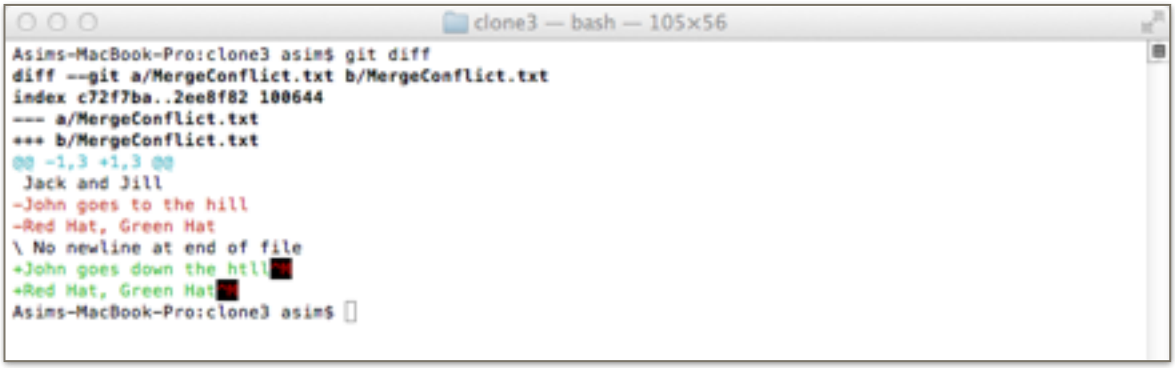
```
Asims-MacBook-Pro:Lab2-2 asim$ ls -ltr
total 0
drwxr-xr-x  17 asim  staff  578 Sep 26 01:52 tmp
drwxr-xr-x   8 asim  staff  272 Sep 27 13:20 SS_V1.0
drwxr-xr-x  19 asim  staff  646 Sep 27 14:05 SS_V1.1
drwxr-xr-x   4 asim  staff  136 Sep 27 16:33 GIT Repo
drwxr-xr-x   5 asim  staff  170 Sep 27 16:33 Deliverables
drwxr-xr-x   8 asim  staff  272 Sep 28 15:18 lcl_master_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 15:30 tmp_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 16:27 lcl_child_clone
drwxr-xr-x@  8 asim  staff  272 Sep 28 23:45 Lab2-2-Draft.pages
drwxr-xr-x   8 asim  staff  272 Sep 28 23:51 clone_merge
drwxr-xr-x   8 asim  staff  272 Sep 28 23:54 lcl_merge_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 23:56 merge2_clone
drwxr-xr-x   8 asim  staff  272 Sep 28 23:58 clone3
drwxr-xr-x  12 asim  staff  408 Sep 28 23:58 SS_V1.2
Asims-MacBook-Pro:Lab2-2 asim$ cd clone3/
Asims-MacBook-Pro:clone3 asim$ git remote -v
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/merge2_cl
one/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/merge2_cl
one/ (push)
Asims-MacBook-Pro:clone3 asim$ cd ../merge2_clone/
Asims-MacBook-Pro:merge2_clone asim$ git remote -v
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:merge2_clone asim$ git remote add newclone ../clone3/
Asims-MacBook-Pro:merge2_clone asim$ git remote -v
newclone        ../clone3/ (fetch)
newclone        ../clone3/ (push)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:merge2_clone asim$ 
```

In the original repository, we make the change required. That is, we modify the MergeConflict.txt file so that the entire content is replaced with new data



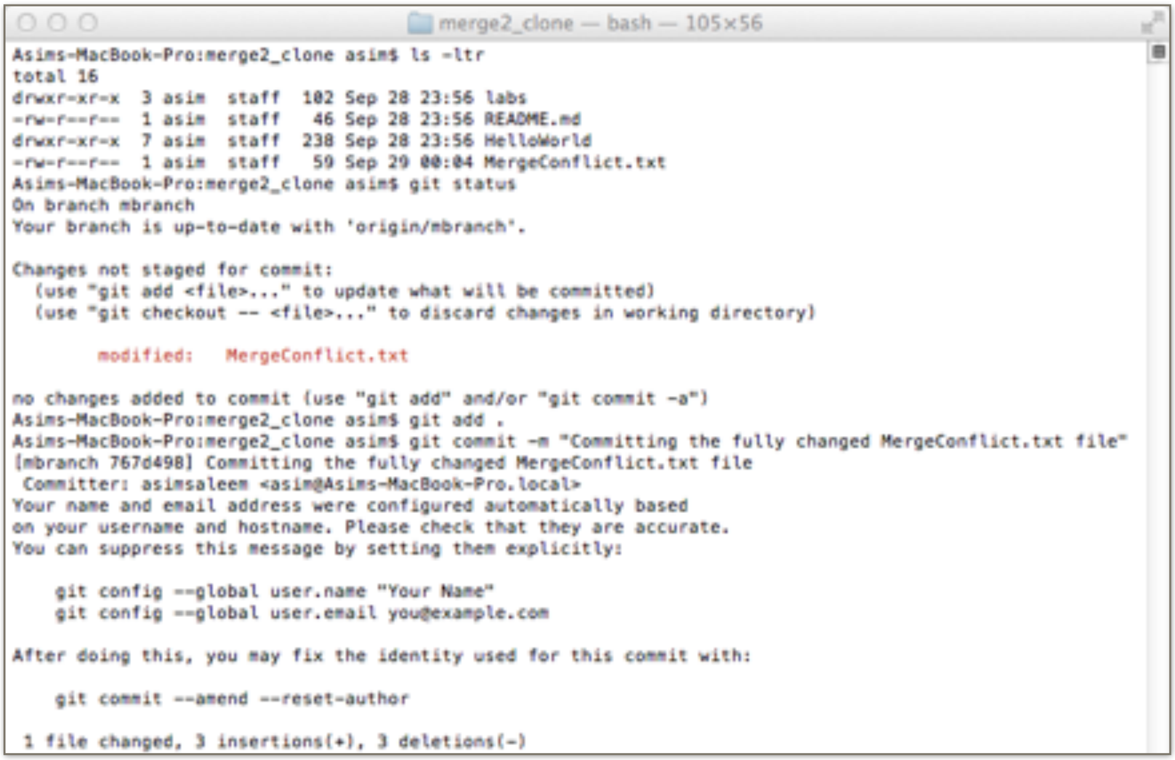
```
Asims-MacBook-Pro:merge2_clone asim$ cd ../clone3
Asims-MacBook-Pro:clone3 asim$ git branch
* mbranch
Asims-MacBook-Pro:clone3 asim$ git branch clone3branch
Asims-MacBook-Pro:clone3 asim$ git checkout clone3branch
Switched to branch 'clone3branch'
Asims-MacBook-Pro:clone3 asim$ cd ../merge2_clone/
Asims-MacBook-Pro:merge2_clone asim$ git branch
* mbranch
Asims-MacBook-Pro:merge2_clone asim$ cat MergeConflict.txt
Jack and Jill
John goes to the hill
Red Hat, Green HatAsims-MacBook-Pro:merge2_clone asim$ vi MergeConflict.txt
Asims-MacBook-Pro:merge2_clone asim$ git status
On branch mbranch
Your branch is up-to-date with 'origin/mbranch'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MergeConflict.txt

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:merge2_clone asim$ git diff
diff --git a/MergeConflict.txt b/MergeConflict.txt
index c72f7ba..2b6e89e 100644
--- a/MergeConflict.txt
+++ b/MergeConflict.txt
@@ -1,3 +1,3 @@
-Jack and Jill
-John goes to the hill
-Red Hat, Green Hat
\ No newline at end of file
+Jane and Joe^M
+John goes up the hill^M
+Blue Hat, Purple Hat^M
Asims-MacBook-Pro:merge2_clone asim$ 
```

We then make the required changes in the cloned directory, wherein only one line of the file is changed. It has to be observed though, that this line of change is in the middle of the file. The other thing that needs to be kept in mind is that even the file in the original directory has not been committed or pushed yet.



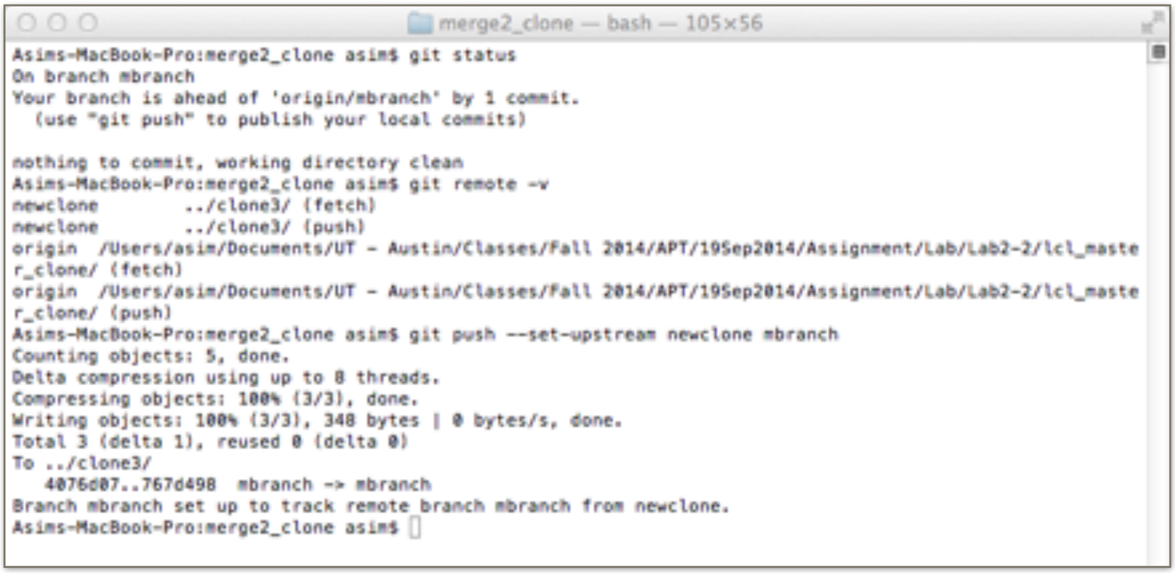
```
Asims-MacBook-Pro:clone3 asim$ git diff
diff --git a/MergeConflict.txt b/MergeConflict.txt
index c72f7ba..2ee8f82 100644
--- a/MergeConflict.txt
+++ b/MergeConflict.txt
@@ -1,3 +1,3 @@
 Jack and Jill
-John goes to the hill
-Red Hat, Green Hat
\ No newline at end of file
+John goes down the hill^M
+Red Hat, Green Hat^M
Asims-MacBook-Pro:clone3 asim$ 
```

So in order to do that, we go back to the original directory and commit the modified MergeConflict.txt file in.



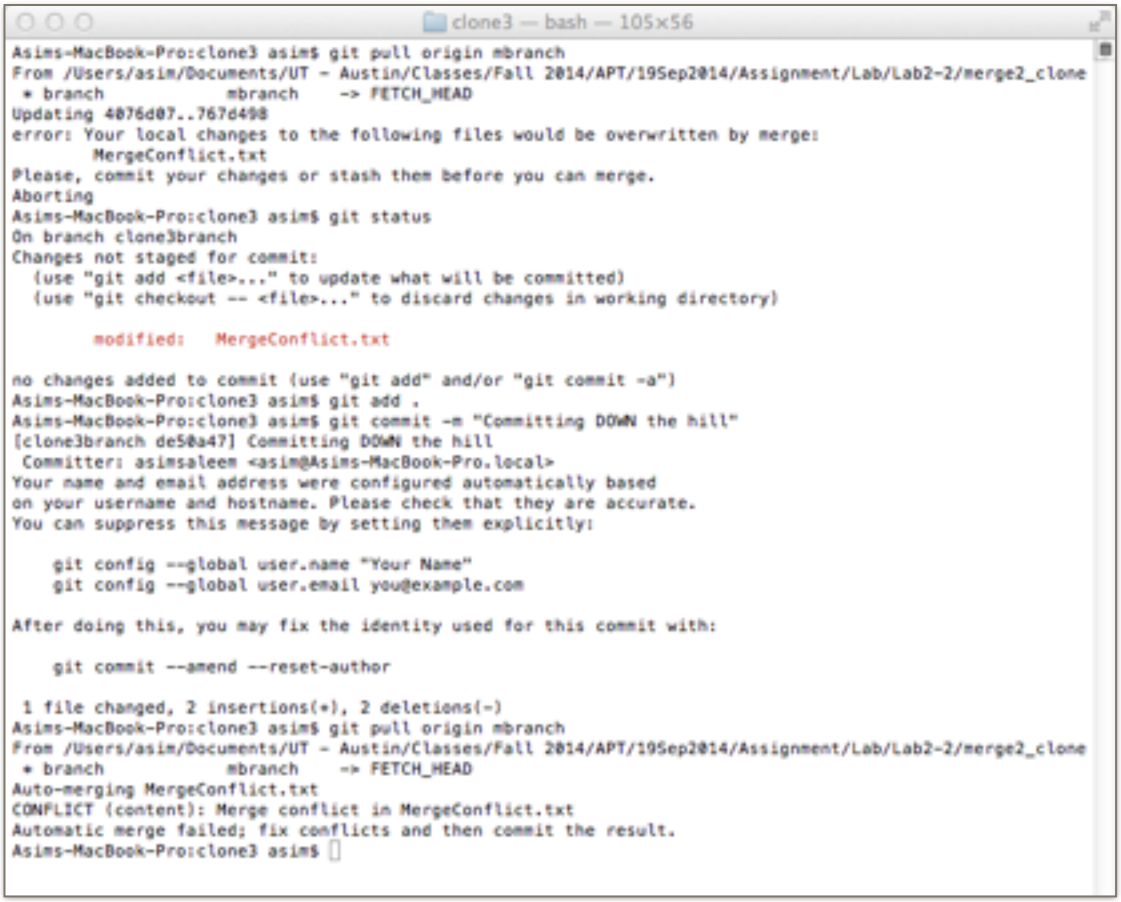
```
Asims-MacBook-Pro:merge2_clone asim$ ls -ltr
total 16
drwxr-xr-x  3 asim  staff  102 Sep 28 23:56 labs
-rw-r--r--  1 asim  staff   46 Sep 28 23:56 README.md
drwxr-xr-x  7 asim  staff  238 Sep 28 23:56 HelloWorld
-rw-r--r--  1 asim  staff   59 Sep 29 00:04 MergeConflict.txt
Asims-MacBook-Pro:merge2_clone asim$ git status
On branch mbranch
Your branch is up-to-date with 'origin/mbranch'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MergeConflict.txt

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:merge2_clone asim$ git add .
Asims-MacBook-Pro:merge2_clone asim$ git commit -m "Committing the fully changed MergeConflict.txt file"
[mbranch 767d498] Committing the fully changed MergeConflict.txt file
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 3 insertions(+), 3 deletions(-)
```

After this the newly committed file is pushed to the repo.



```
Asims-MacBook-Pro:merge2_clone asim$ git status
On branch mbranch
Your branch is ahead of 'origin/mbranch' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working directory clean
Asims-MacBook-Pro:merge2_clone asim$ git remote -v
newclone        ../clone3/ (fetch)
newclone        ../clone3/ (push)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (fetch)
origin  /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/lcl_maste
r_clone/ (push)
Asims-MacBook-Pro:merge2_clone asim$ git push --set-upstream newclone mbranch
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 348 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To ../clone3/
   4076d07..767d498  mbranch -> mbranch
Branch mbranch set up to track remote branch mbranch from newclone.
Asims-MacBook-Pro:merge2_clone asim$ 
```

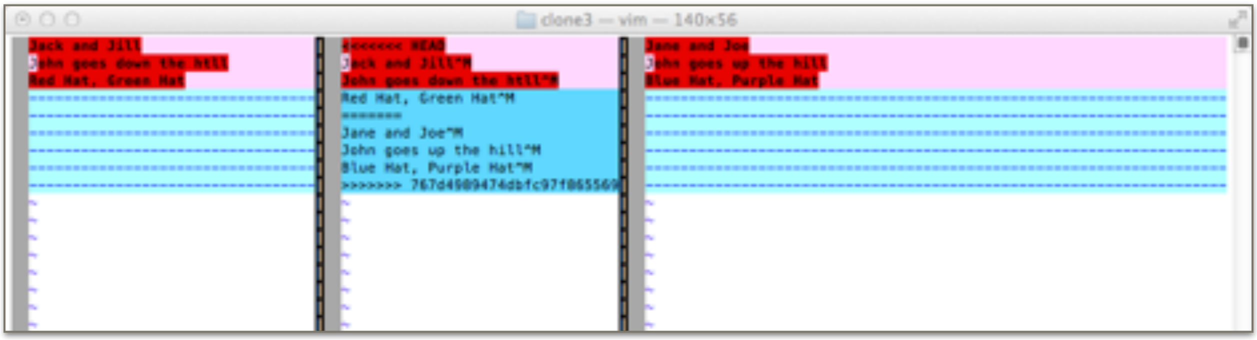
So in order to pull the commit made in the original directory in the cloned directory, we do a git pull from the clone. Before that, we first commit the modified MergeConflict.txt file in the clone directory also.



```
Asims-MacBook-Pro:clone3 asim$ git pull origin mbranch
From /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/merge2_clone
 * branch            mbranch    -> FETCH_HEAD
Updating 4076d07..767d498
error: Your local changes to the following files would be overwritten by merge:
        MergeConflict.txt
Please, commit your changes or stash them before you can merge.
Aborting
Asims-MacBook-Pro:clone3 asim$ git status
On branch clone3branch
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   MergeConflict.txt

no changes added to commit (use "git add" and/or "git commit -a")
Asims-MacBook-Pro:clone3 asim$ git add .
Asims-MacBook-Pro:clone3 asim$ git commit -m "Committing DOWN the hill"
[clone3branch de50a47] Committing DOWN the hill
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 2 insertions(+), 2 deletions(-)
Asims-MacBook-Pro:clone3 asim$ git pull origin mbranch
From /Users/asim/Documents/UT - Austin/Classes/Fall 2014/APT/19Sep2014/Assignment/Lab/Lab2-2/merge2_clone
 * branch            mbranch    -> FETCH_HEAD
Auto-merging MergeConflict.txt
CONFLICT (content): Merge conflict in MergeConflict.txt
Automatic merge failed; fix conflicts and then commit the result.
Asims-MacBook-Pro:clone3 asim$ 
```

It is observed that when we do a git pull after committing in the cloned directory, we notice a problem with that operation. The error message clearly states that the “Automatic Merge failed: fix conflicts and then commit the result”. From the error message it is very clear that this happens due to a conflict in the MergeConflict.txt file checked in both at the original directory and the current clone directory. Git is basically unable to do a automatic merge because of the overlapping of changes in the same locations. It is easier to allow the user to correct such merge conflicts, so that valid code is not thrown away by the automatic merge operation.

In order to solve this, we have to use a merge tool and manually review the conflicts that Git encountered. We run the git mergetool command with vimdiff2 in order to compare the conflicts. The screen appears like this

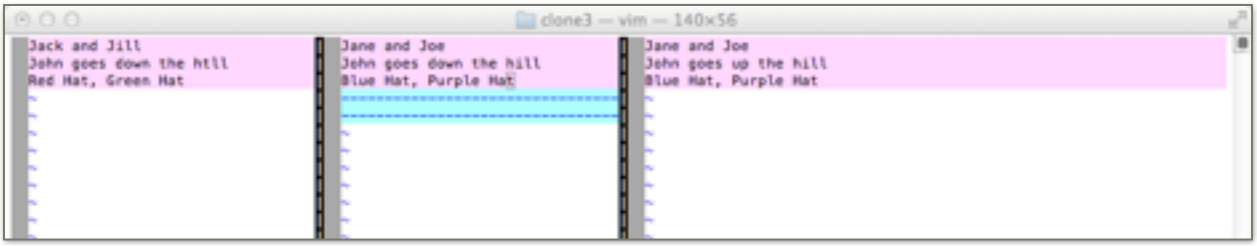


```
Jack and Jill
John goes down the hill
Red Hat, Green Hat
```

```
<<<<<<< HEAD
Jack and Jill^M
John goes down the hill^M
Red Hat, Green Hat^M
=======
Jane and Joe^M
John goes up the hill^M
Blue Hat, Purple Hat^M
>>>>>>> 767d4989474dbfc97f865569
```

```
Jane and Joe
John goes up the hill
Blue Hat, Purple Hat
```

We then make the necessary changes to the center portion of this conflict display

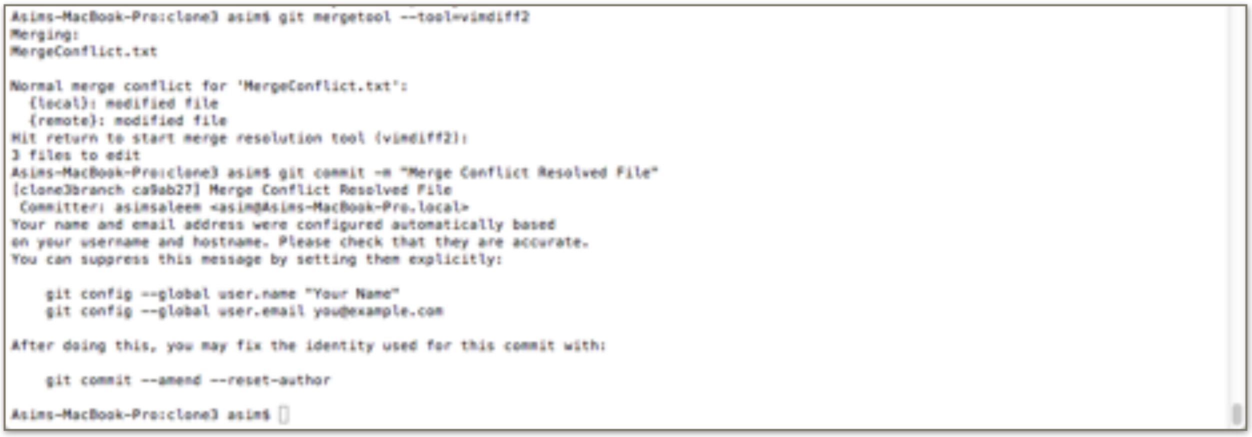


```
Jack and Jill
John goes down the hill
Red Hat, Green Hat
```

```
Jane and Joe
John goes down the hill
Blue Hat, Purple Hat
```

```
Jane and Joe
John goes up the hill
Blue Hat, Purple Hat
```

Once done, we are ready to save this file and commit it as shown below:



```
Asims-MacBook-Pro:clone3 asim$ git mergetool --tool=vimdiff2
Merging:
MergeConflict.txt

Normal merge conflict for 'MergeConflict.txt':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (vimdiff2):
3 files to edit
Asims-MacBook-Pro:clone3 asim$ git commit -m "Merge Conflict Resolved File"
[clone3branch ca9ab27] Merge Conflict Resolved File
 Committer: asimsaleem <asim@Asims-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

Asims-MacBook-Pro:clone3 asim$
```

So basically to resolve this issue, we use a mergetool, fix the conflict and then save the file which is now ready to be committed and pushed.

1.6 Write a hook script that aborts an attempt to commit if any line in any file is more than 80 chars long. Show your file(s), and a screenshot showing your check works.   (15 marks)

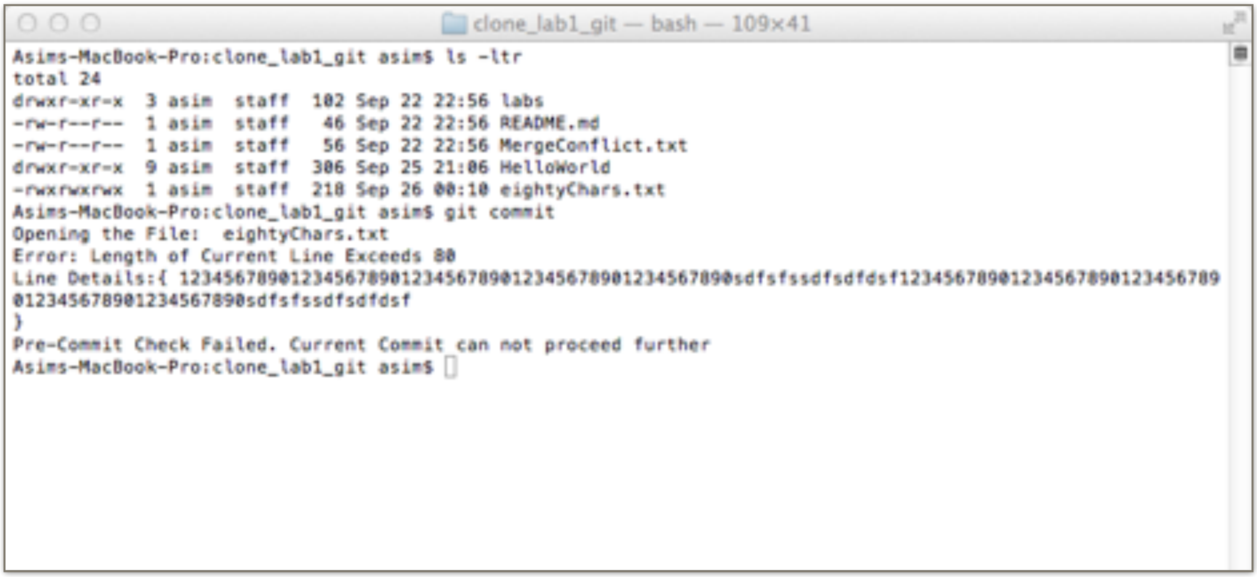
Solution:

In order to meet this requirement, we need to write a pre-commit hook. A pre-commit hook basically executes before an actual commit can even be done. So to achieve the required result, we add a new file called "pre-commit" into the hook folder of git. The code encapsulated within this file will be executed whenever there is a commit operation to be carried out.

The way it's structured, the code first walks through the current working directory. We have to make sure that we only apply the required condition of not having a line more than 80 characters only to valid files rather than hidden or system files. To do this, we ignore all files that begin with a '.' (dot). Once those files are filtered out, we can then iterate through all the files and open each of those individually.

The files are then opened and we iterate through each of the lines and verify if their length is greater than or equal to 80. If greater than 80, then the code exits with a value of 1 which indicates that the check failed and therefore should not be allowed to proceed with the commit.

A screenshot of the error message shown to the user on executing commit with a file that has one of the lines with more than 80 characters is shown below:



```
Asims-MacBook-Pro:clone_lab1_git asim$ ls -ltr
total 24
drwxr-xr-x  3 asim  staff  102 Sep 22 22:56 labs
-rw-r--r--  1 asim  staff   46 Sep 22 22:56 README.md
-rw-r--r--  1 asim  staff   56 Sep 22 22:56 MergeConflict.txt
drwxr-xr-x  9 asim  staff  306 Sep 25 21:06 HelloWorld
-rwxrwxrwx  1 asim  staff  218 Sep 26 00:10 eightyChars.txt
Asims-MacBook-Pro:clone_lab1_git asim$ git commit
Opening the File:  eightyChars.txt
Error: Length of Current Line Exceeds 80
Line Details:{ 12345678901234567890123456789012345678901234567890sdfsfssdfsdfdsf1234567890123456789012345678
90123456789012345678 90sdfsfssdfsdfdsf
}
Pre-Commit Check Failed. Current Commit can not proceed further
Asims-MacBook-Pro:clone_lab1_git asim$ 
```

The code written inside the "pre-commit" file to handle such a scenario is given below:

```
#!/usr/bin/python

import os
import sys

#Get the current directory
current_working_directory = os.path.join(os.path.dirname(__file__), '../..')
#Walkthrough the Entire Directory to get the List of Files
for roots, dirs, files in os.walk(current_working_directory):
    #Open the Individual Files to Read for the size
    for individual_file in files:
        #Ignore all Hidden files and focus only on the valid ones
        if not individual_file.startswith('.'):
            #Opening the File in Read only mode is FINE, since we are not writing anything
            print "Opening the File: ", individual_file
            os.chdir(current_working_directory)
            f = open(individual_file, 'r')

            #Read the individual lines of the Files to check their length
            for line in f:
                '''
                Check the Length of the Line to determine if Commit should be stopped
                if any line is more than 80 chars long
                '''
                if len(line) > 80:
                    print "Error: Length of Current Line Exceeds 80"
                    print "Line Details:{", line, "}"
                    print "Pre-Commit Check Failed. Current Commit can not proceed further"
                    sys.exit(1)

print "Pre-Commit Check Passed. Proceed with Commit"
```