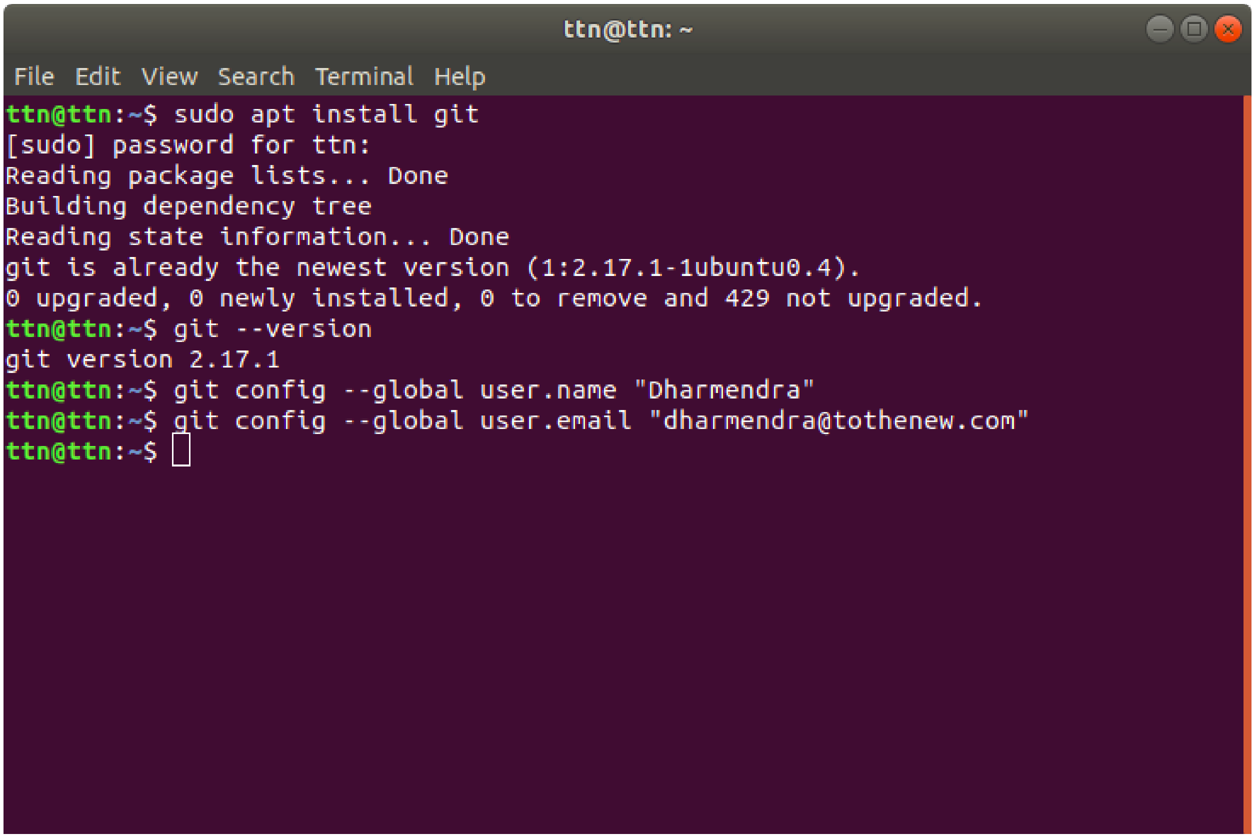
Q. 1 Git Setup https://confluence.atlassian.com/bitbucket/set-up-git-744723531.html

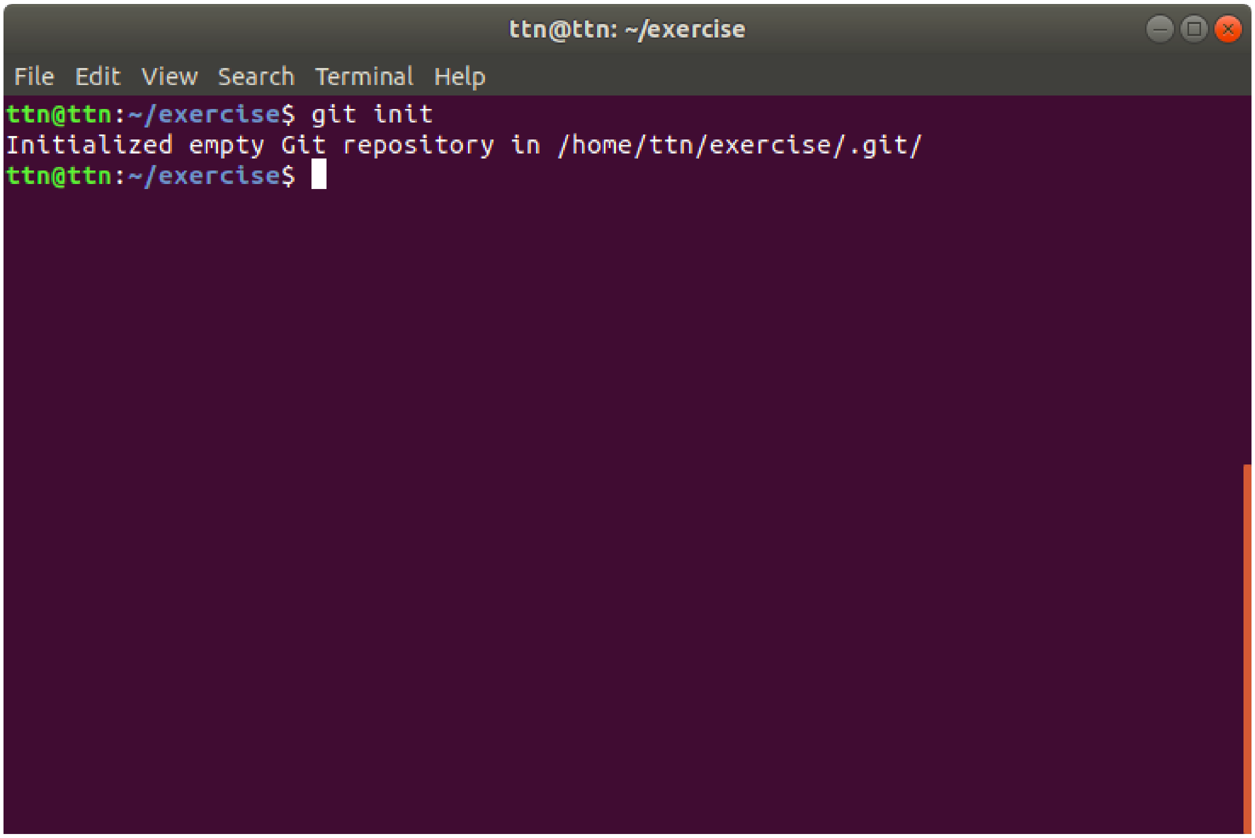
sudo apt-get install git
git --version
git config --global user.name “Dharmendra”
git config --global user.email “dharmendra@tothenew.com”



```
ttn@ttn:~$ sudo apt install git
[sudo] password for ttn:
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.17.1-1ubuntu0.4).
0 upgraded, 0 newly installed, 0 to remove and 429 not upgraded.
ttn@ttn:~$ git --version
git version 2.17.1
ttn@ttn:~$ git config --global user.name "Dharmendra"
ttn@ttn:~$ git config --global user.email "dharmendra@tothenew.com"
ttn@ttn:~$
```

Q. 2 Initialize a Git Repository

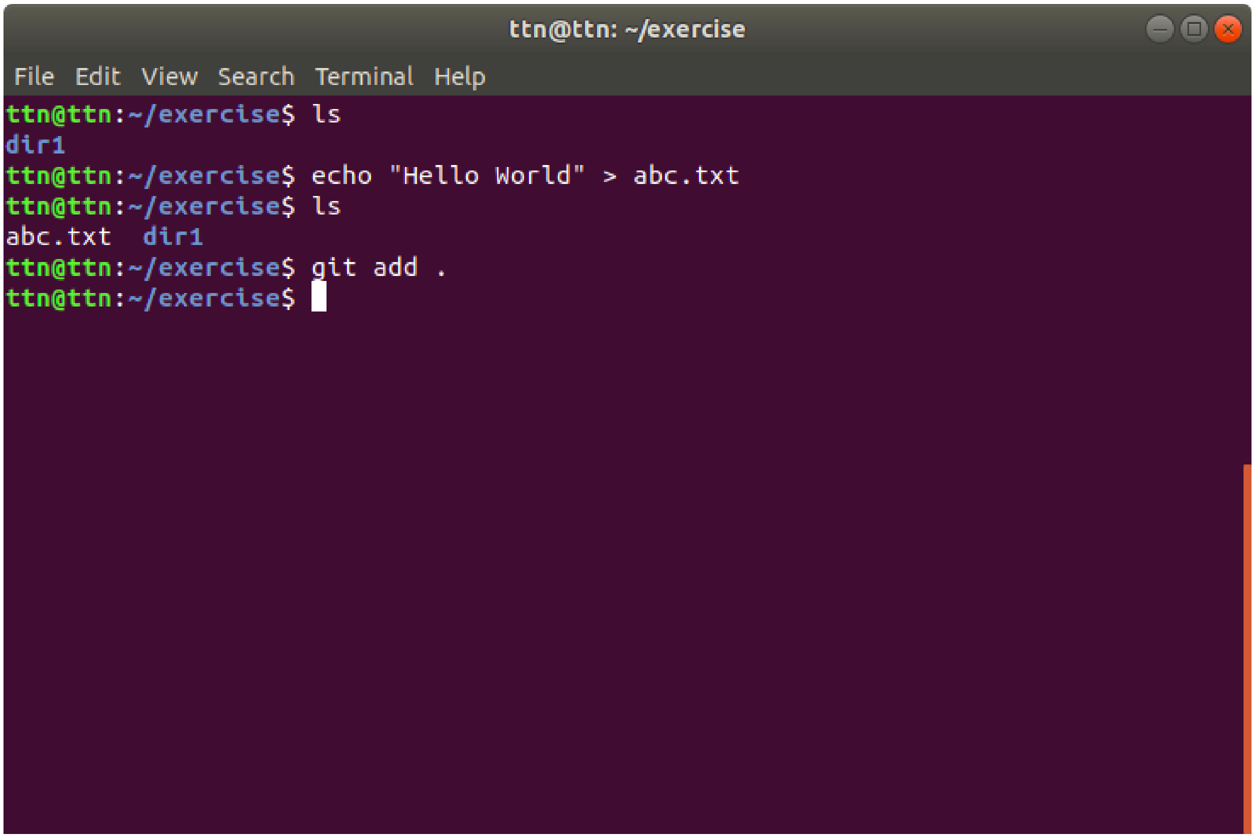
git init



```
ttn@ttn:~/exercise$ git init
Initialized empty Git repository in /home/ttn/exercise/.git/
ttn@ttn:~/exercise$ 
```

Q.3 Add files to the repository

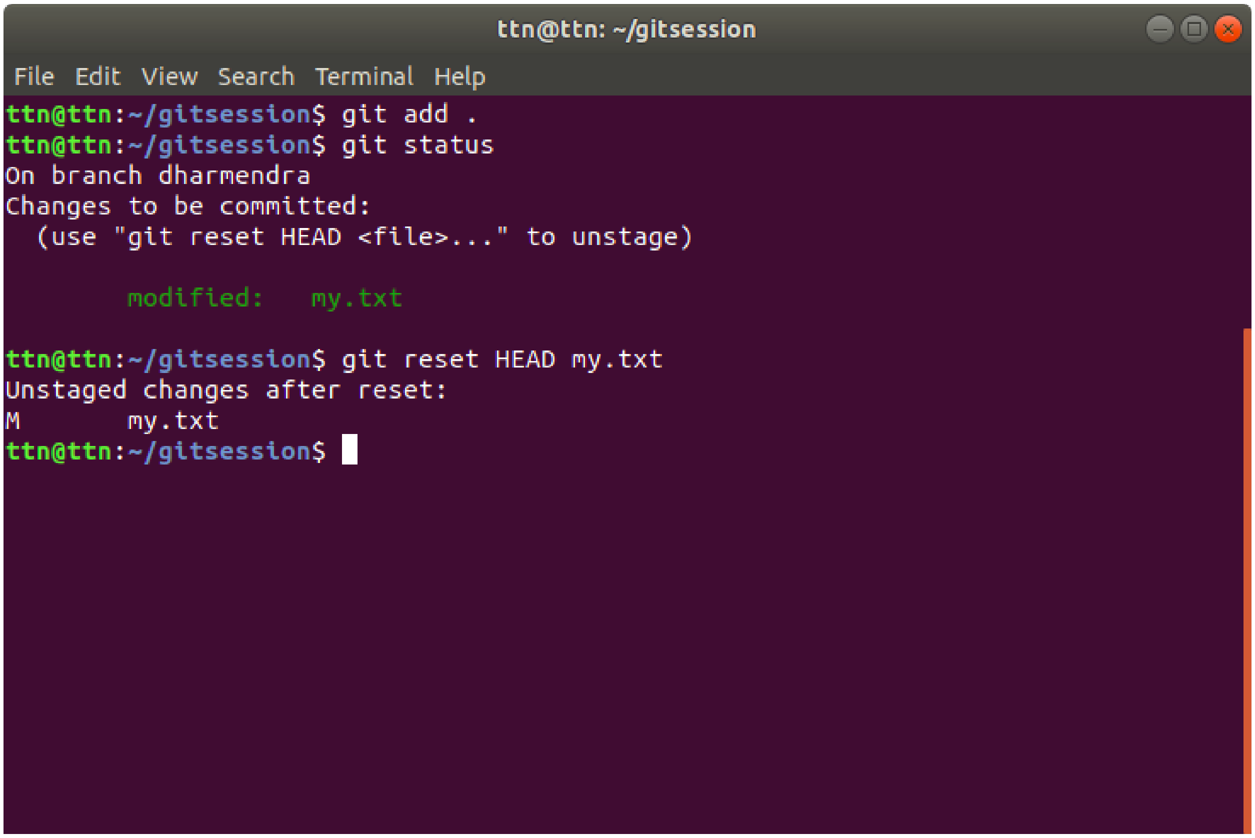
git add .



```
ttn@ttn:~/exercise$ ls
dir1
ttn@ttn:~/exercise$ echo "Hello World" > abc.txt
ttn@ttn:~/exercise$ ls
abc.txt  dir1
ttn@ttn:~/exercise$ git add .
ttn@ttn:~/exercise$ 
```

Q.4 Unstage 1 file

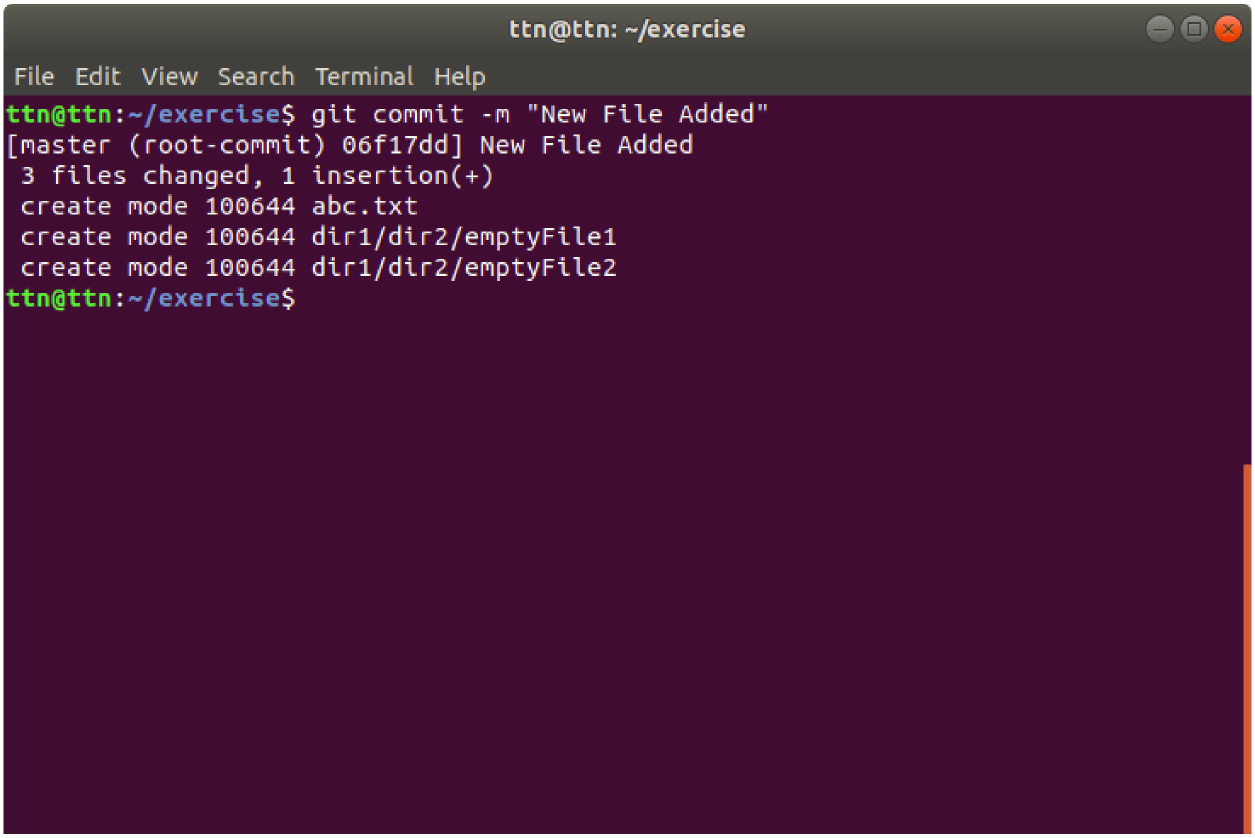
git reset HEAD my.txt



```
ttn@ttn:~/gitsession$ git add .
ttn@ttn:~/gitsession$ git status
On branch dharmendra
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   my.txt

ttn@ttn:~/gitsession$ git reset HEAD my.txt
Unstaged changes after reset:
M       my.txt
ttn@ttn:~/gitsession$ 
```

Q. 5 Commit the file

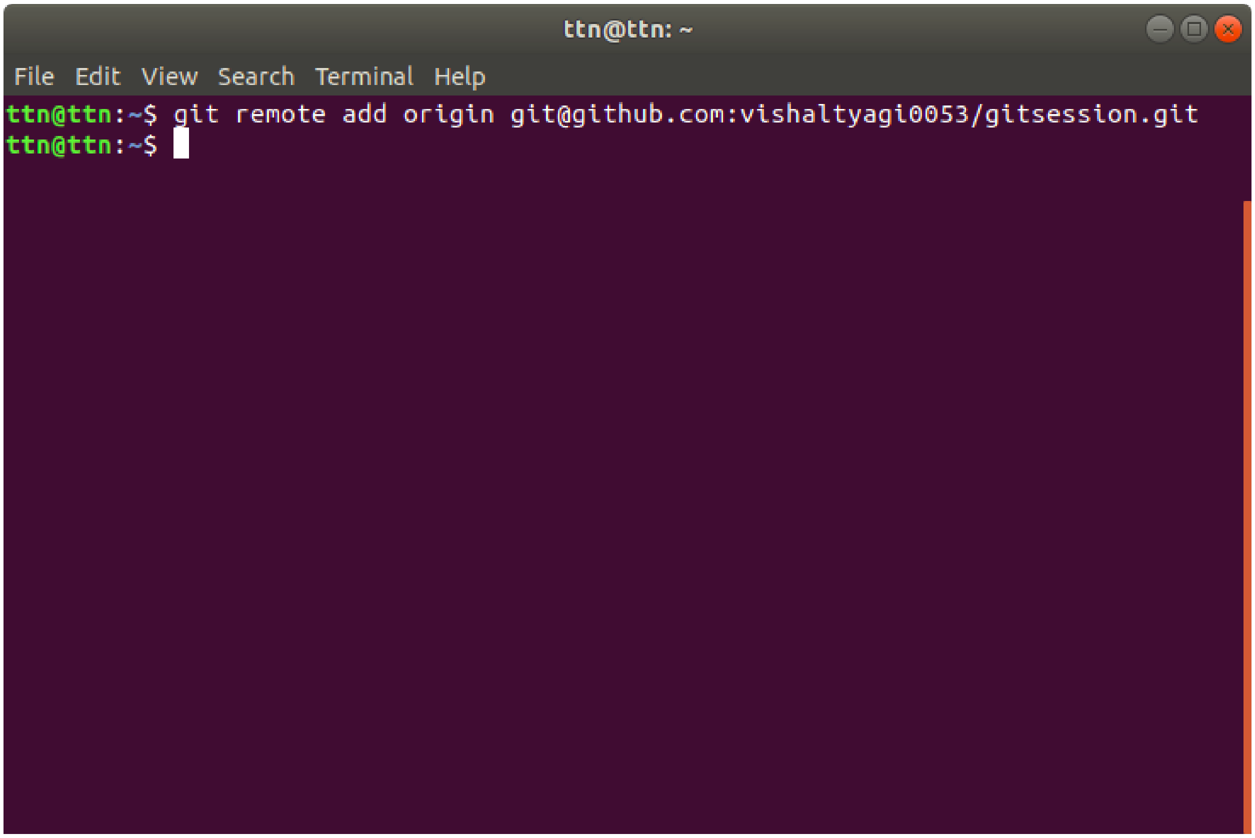
git commit -m “New File Added”



```
ttn@ttn:~/exercise$ git commit -m "New File Added"
[master (root-commit) 06f17dd] New File Added
 3 files changed, 1 insertion(+)
 create mode 100644 abc.txt
 create mode 100644 dir1/dir2/emptyFile1
 create mode 100644 dir1/dir2/emptyFile2
ttn@ttn:~/exercise$
```

Q. 6 Add a remote

git remote add origin repo_address



```
ttn@ttn:~$ git remote add origin git@github.com:vishaltyagi0053/gitsession.git
ttn@ttn:~$ 
```

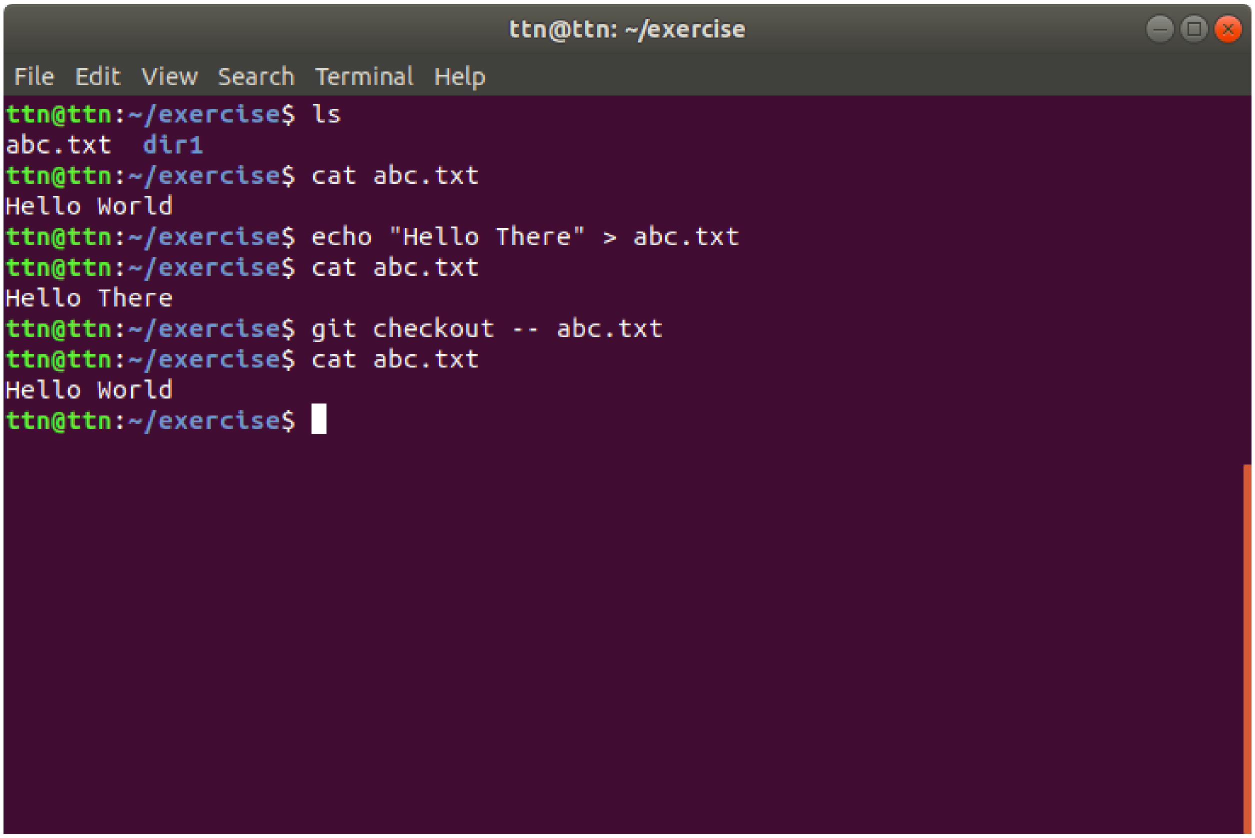
Q. 7 Undo changes to a particular file

Discard all local changes, but save them for possible re-use later

> git stash

Discarding local changes (permanently) to a file

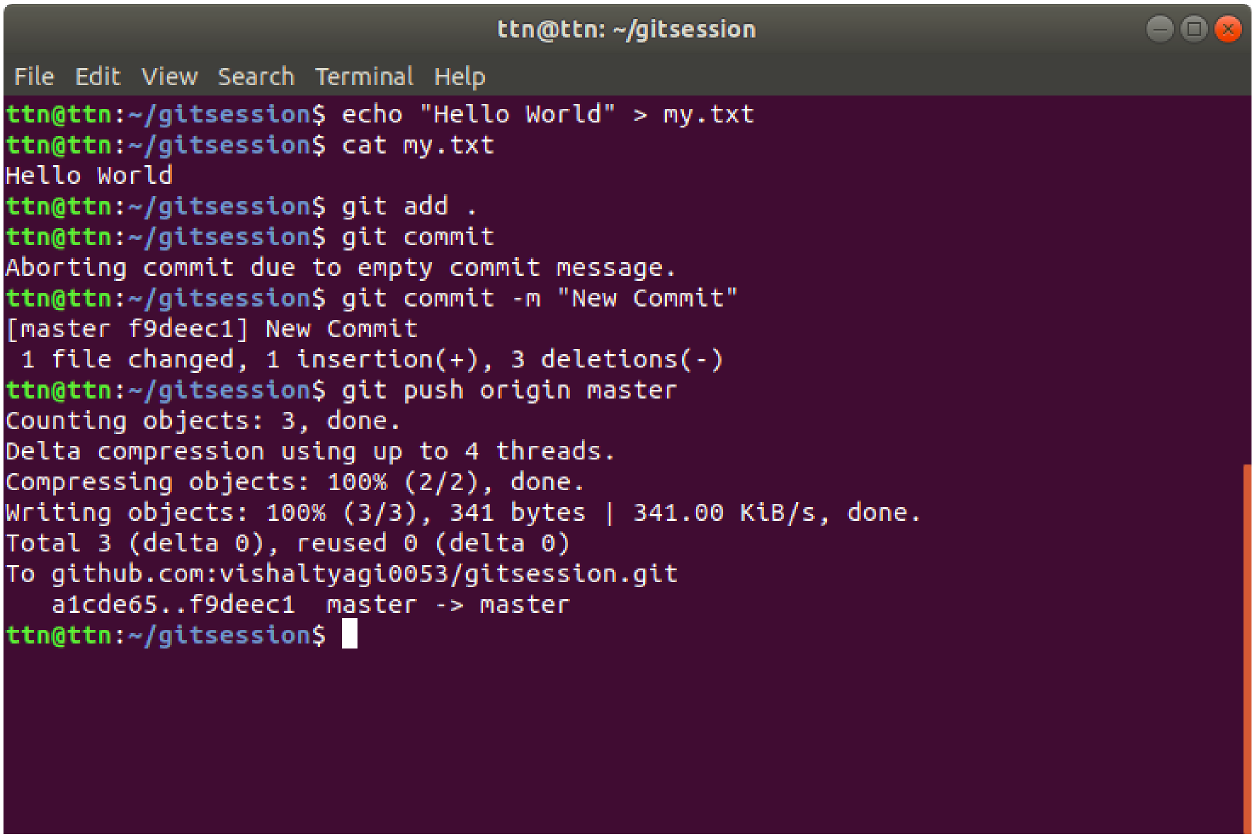
> **git checkout -- file_name**



```
ttn@ttn:~/exercise$ ls
abc.txt  dir1
ttn@ttn:~/exercise$ cat abc.txt
Hello World
ttn@ttn:~/exercise$ echo "Hello There" > abc.txt
ttn@ttn:~/exercise$ cat abc.txt
Hello There
ttn@ttn:~/exercise$ git checkout -- abc.txt
ttn@ttn:~/exercise$ cat abc.txt
Hello World
ttn@ttn:~/exercise$ 
```

Q.8 Push changes to Github

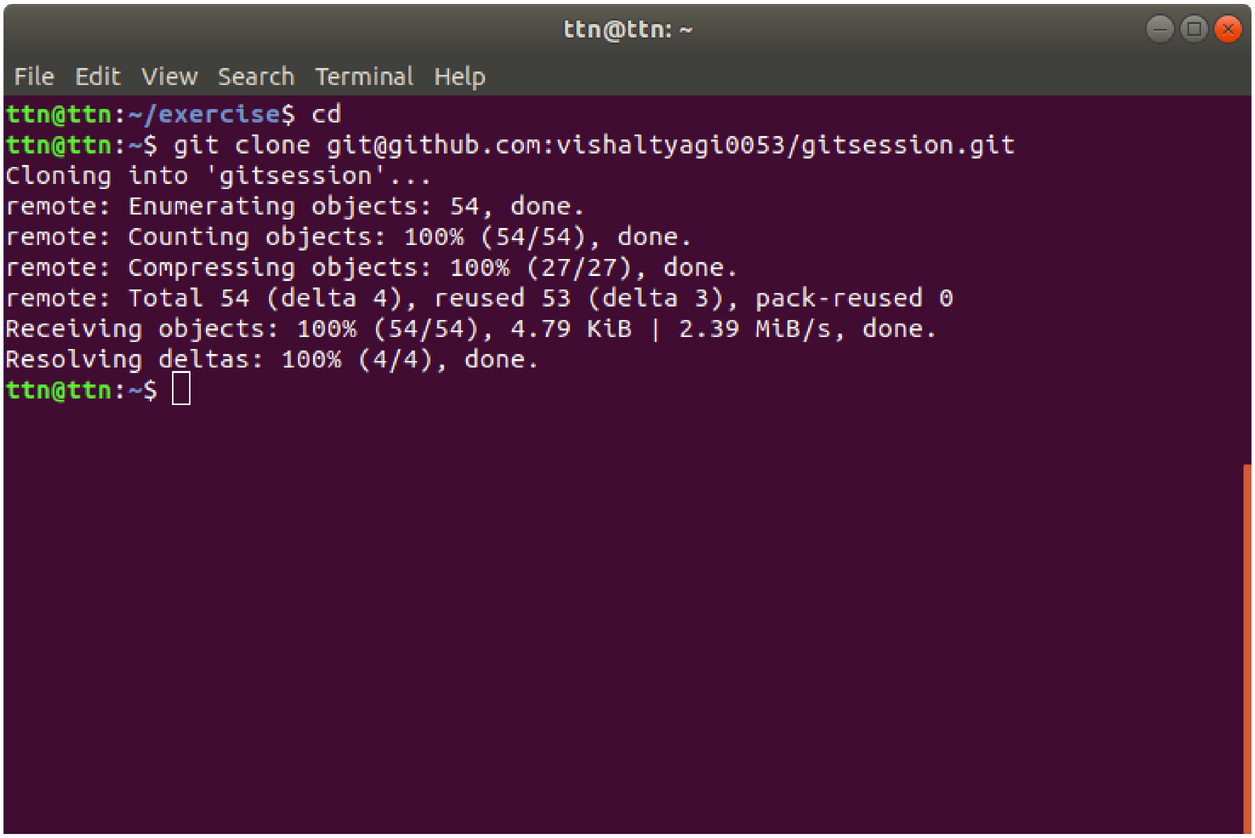
git add .
git commit -m “Commit Message”
git push origin master



```
ttn@ttn:~/gitsession$ echo "Hello World" > my.txt
ttn@ttn:~/gitsession$ cat my.txt
Hello World
ttn@ttn:~/gitsession$ git add .
ttn@ttn:~/gitsession$ git commit
Aborting commit due to empty commit message.
ttn@ttn:~/gitsession$ git commit -m "New Commit"
[master f9deec1] New Commit
 1 file changed, 1 insertion(+), 3 deletions(-)
ttn@ttn:~/gitsession$ git push origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 341 bytes | 341.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:vishaltyagi0053/gitsession.git
   a1cde65..f9deec1  master -> master
ttn@ttn:~/gitsession$ 
```

Q.9 Clone the repository

git clone repository_address



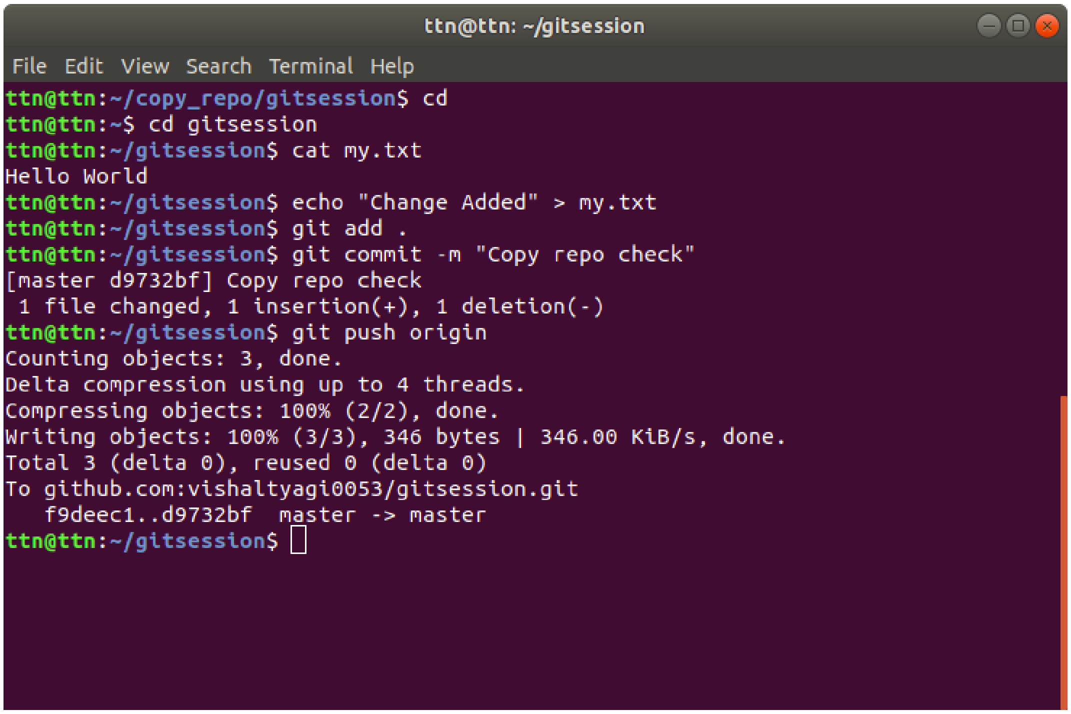
```
ttn@ttn:~/exercise$ cd
ttn@ttn:~$ git clone git@github.com:vishaltyagi0053/gitsession.git
Cloning into 'gitsession'...
remote: Enumerating objects: 54, done.
remote: Counting objects: 100% (54/54), done.
remote: Compressing objects: 100% (27/27), done.
remote: Total 54 (delta 4), reused 53 (delta 3), pack-reused 0
Receiving objects: 100% (54/54), 4.79 KiB | 2.39 MiB/s, done.
Resolving deltas: 100% (4/4), done.
ttn@ttn:~$ 
```

Q. 10 Add changes to one of the copies and pull the changes in the other.

-> git pull

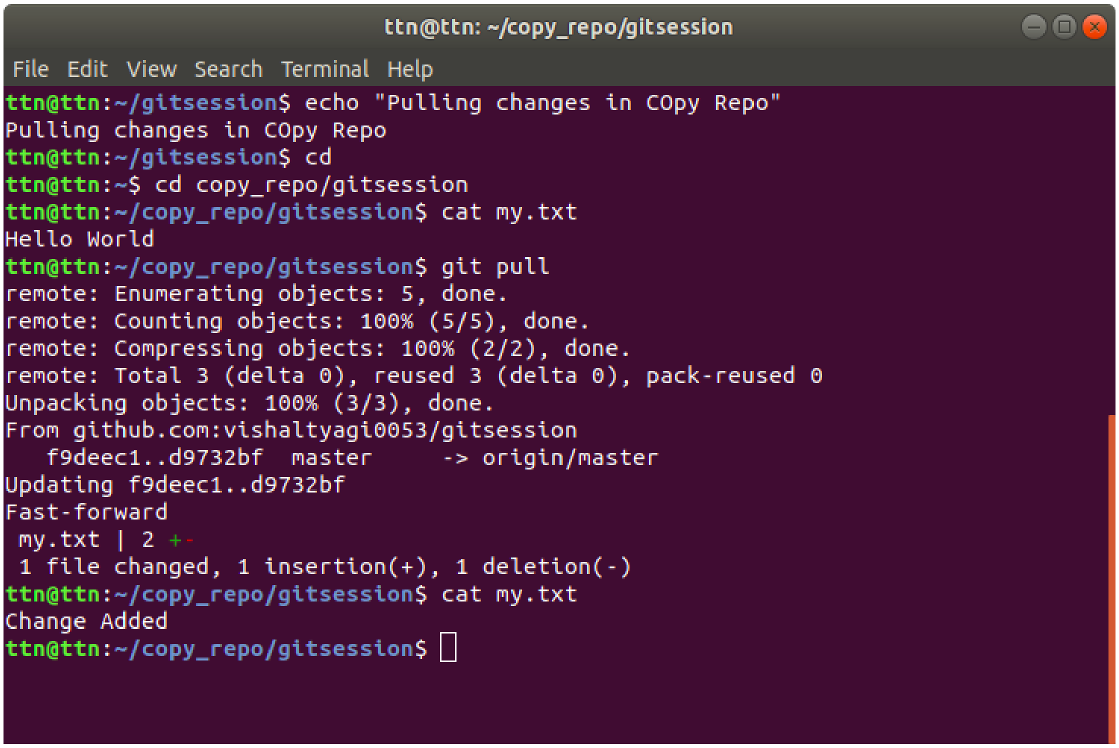
Here same repository is cloned in two directories for showcasing this method.

Adding changes to file



```
ttn@ttn:~/copy_repo/gitsession$ cd
ttn@ttn:~$ cd gitsession
ttn@ttn:~/gitsession$ cat my.txt
Hello World
ttn@ttn:~/gitsession$ echo "Change Added" > my.txt
ttn@ttn:~/gitsession$ git add .
ttn@ttn:~/gitsession$ git commit -m "Copy repo check"
[master d9732bf] Copy repo check
 1 file changed, 1 insertion(+), 1 deletion(-)
ttn@ttn:~/gitsession$ git push origin
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 346 bytes | 346.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:vishaltyagi0053/gitsession.git
   f9deec1..d9732bf  master -> master
ttn@ttn:~/gitsession$ 
```

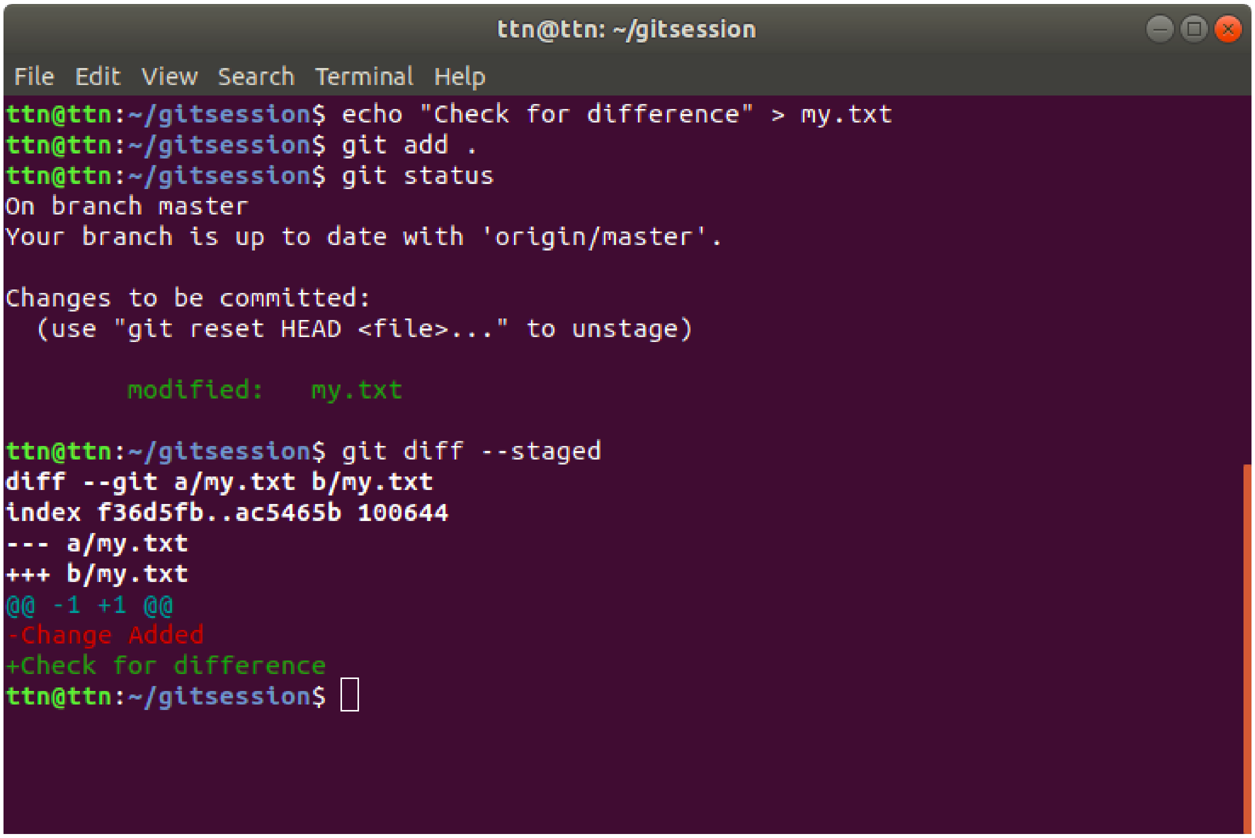
Pulling Changes in Copy Repo


ttn@ttn: ~/copy_repo/gitsession
File Edit View Search Terminal Help

```
ttn@ttn:~/gitsession$ echo "Pulling changes in COpy Repo"
Pulling changes in COpy Repo
ttn@ttn:~/gitsession$ cd
ttn@ttn:~$ cd copy_repo/gitsession
ttn@ttn:~/copy_repo/gitsession$ cat my.txt
Hello World
ttn@ttn:~/copy_repo/gitsession$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:vishaltyagi0053/gitsession
   f9deec1..d9732bf  master     -> origin/master
Updating f9deec1..d9732bf
Fast-forward
 my.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
ttn@ttn:~/copy_repo/gitsession$ cat my.txt
Change Added
ttn@ttn:~/copy_repo/gitsession$ 
```

Q.11 Check differences between a file and its staged version

git diff --staged

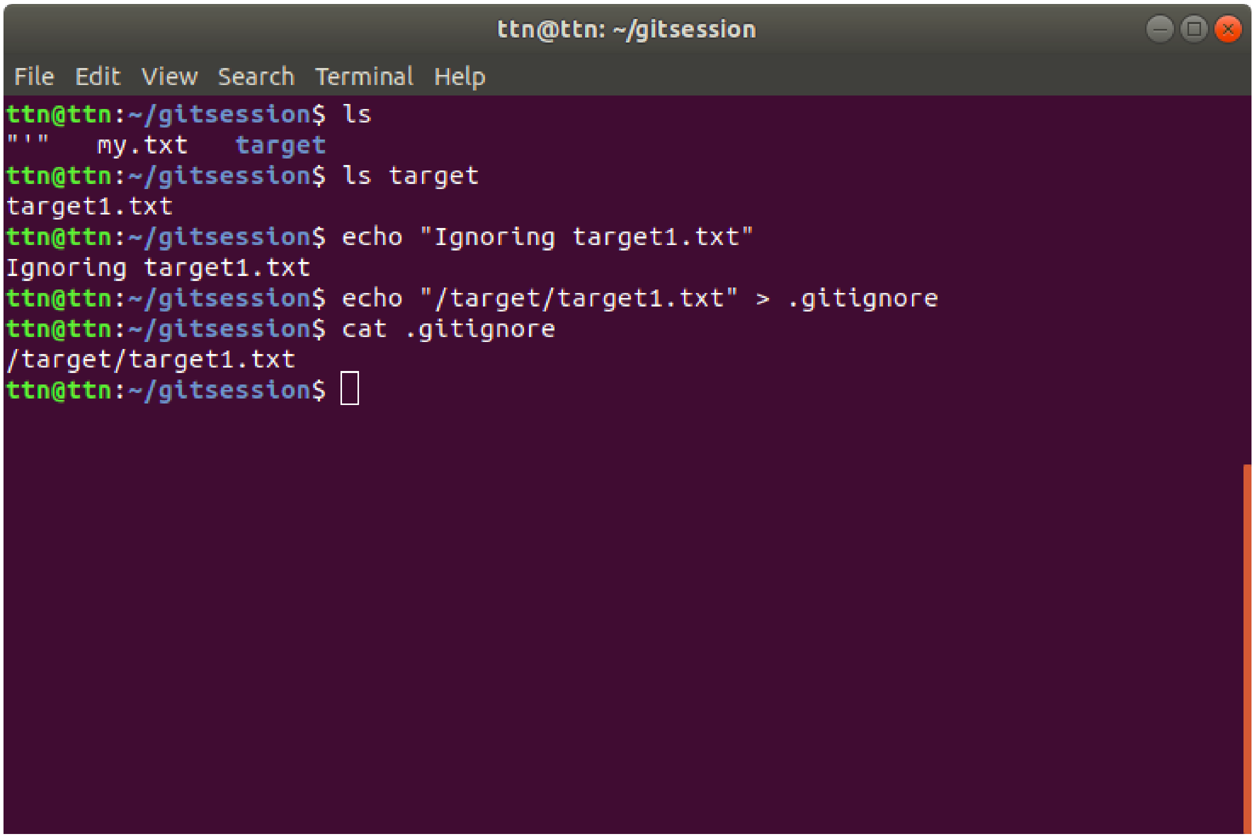


```
ttn@ttn:~/gitsession$ echo "Check for difference" > my.txt
ttn@ttn:~/gitsession$ git add .
ttn@ttn:~/gitsession$ git status
On branch master
Your branch is up to date with 'origin/master'.

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

        modified:   my.txt

ttn@ttn:~/gitsession$ git diff --staged
diff --git a/my.txt b/my.txt
index f36d5fb..ac5465b 100644
--- a/my.txt
+++ b/my.txt
@@ -1 +1 @@
-Change Added
+Check for difference
ttn@ttn:~/gitsession$
```

Q. 12 Ignore a few files to be checked in

-> We can add the file paths to the .gitignore file. Now these files will be ignored during checking.

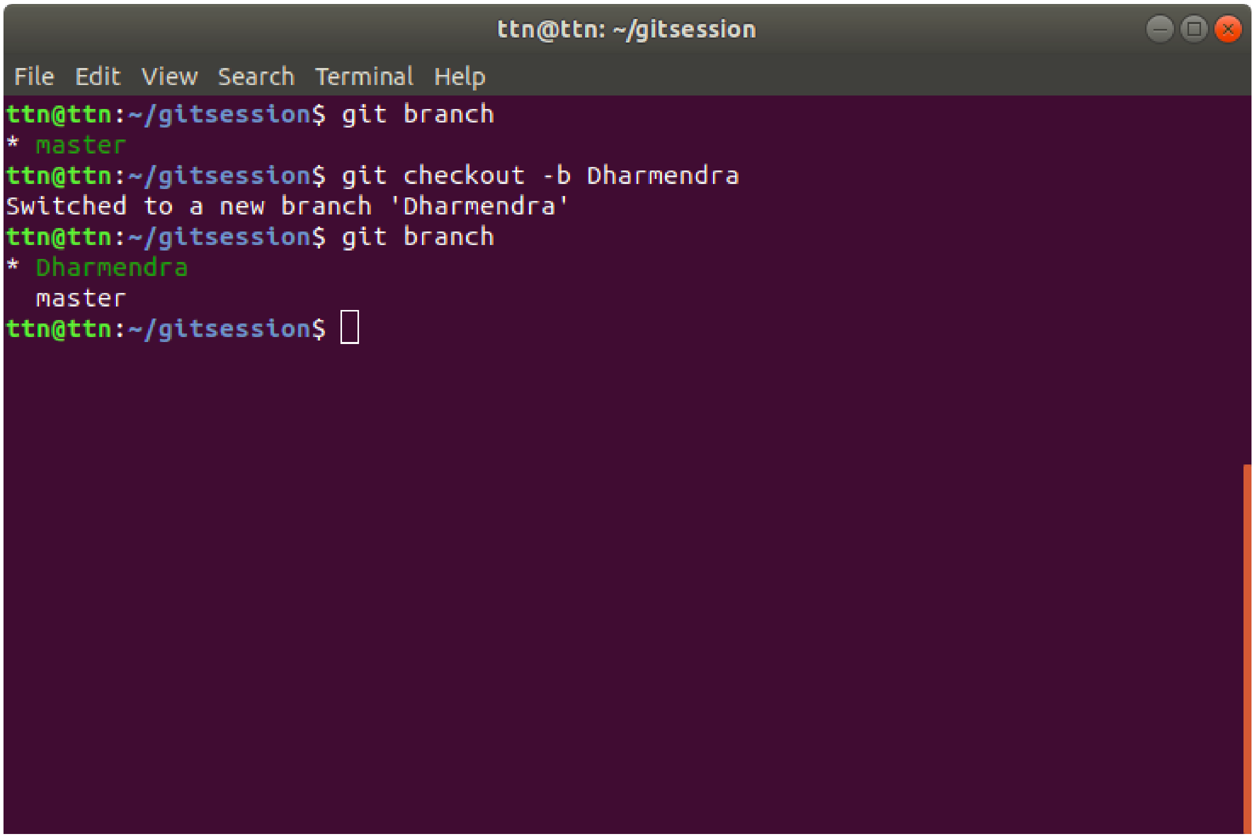
Here we are ignoring target1.txt located in target directory.



```
ttn@ttn:~/gitsession$ ls
"'"   my.txt   target
ttn@ttn:~/gitsession$ ls target
target1.txt
ttn@ttn:~/gitsession$ echo "Ignoring target1.txt"
Ignoring target1.txt
ttn@ttn:~/gitsession$ echo "/target/target1.txt" > .gitignore
ttn@ttn:~/gitsession$ cat .gitignore
/target/target1.txt
ttn@ttn:~/gitsession$ 
```

Q. 13 Create a new branch.

git branch -b branch_name

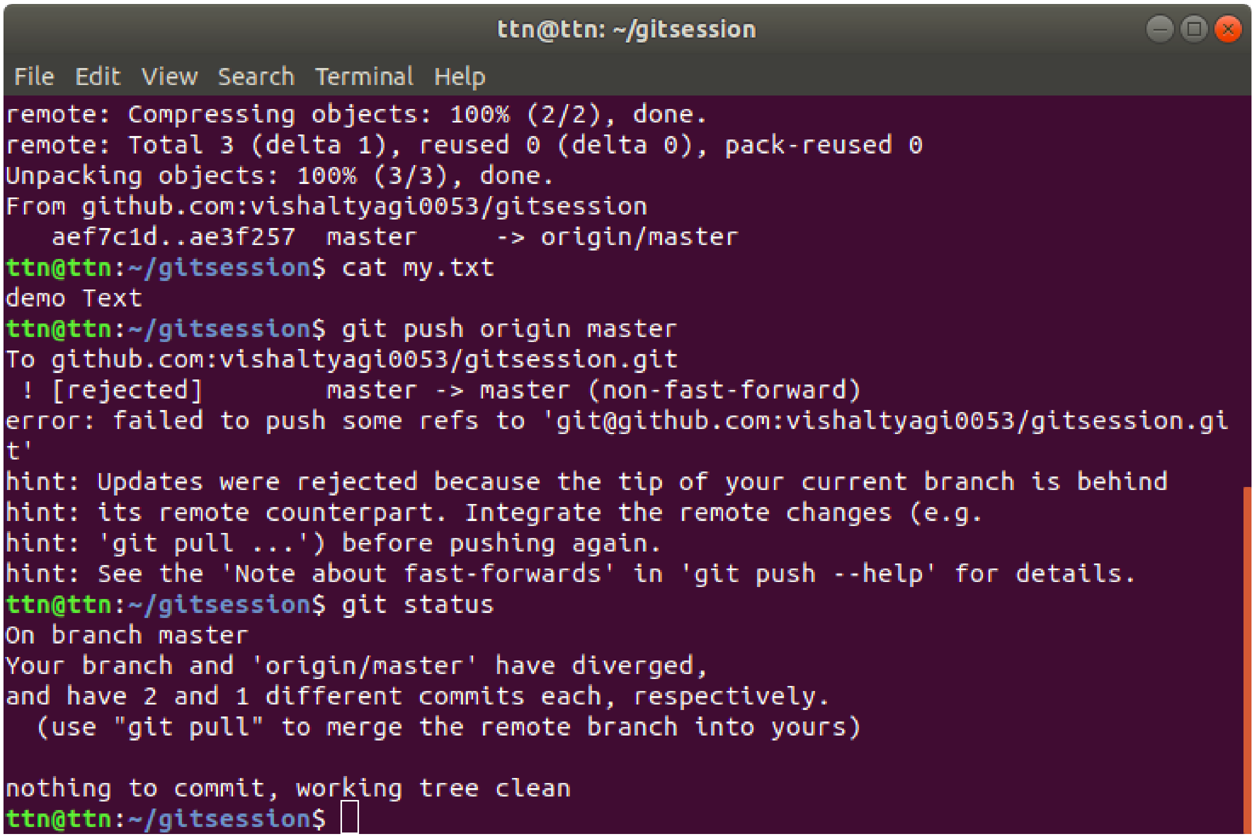


```
ttn@ttn:~/gitsession$ git branch
* master
ttn@ttn:~/gitsession$ git checkout -b Dharmendra
Switched to a new branch 'Dharmendra'
ttn@ttn:~/gitsession$ git branch
* Dharmendra
  master
ttn@ttn:~/gitsession$ 
```

Q. 14 Diverge them with commits

-> Edit and commit the same file in remote repo first then in local repo.

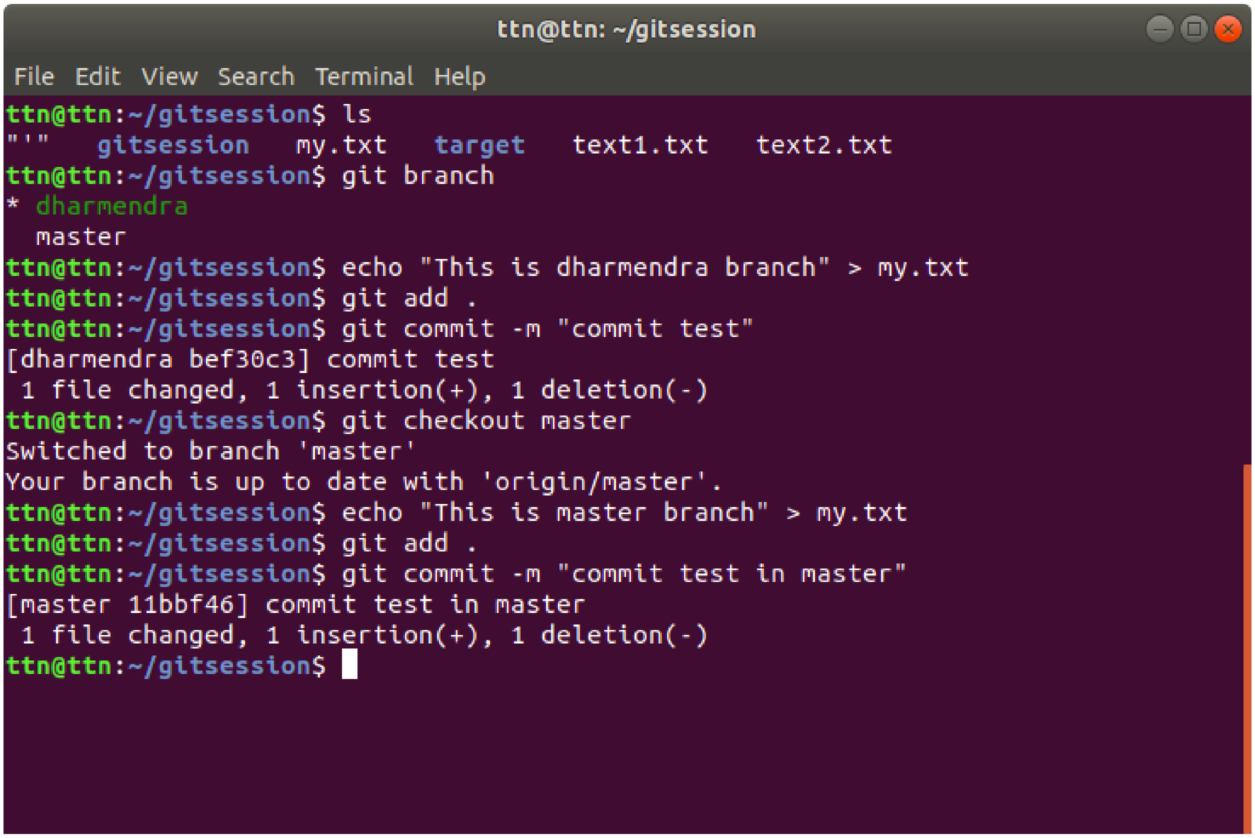
When we push the file in local repo we get a message that our branch is diverged.



```
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From github.com:vishaltyagi0053/gitsession
   aef7c1d..ae3f257  master     -> origin/master
ttn@ttn:~/gitsession$ cat my.txt
demo Text
ttn@ttn:~/gitsession$ git push origin master
To github.com:vishaltyagi0053/gitsession.git
 ! [rejected]        master -> master (non-fast-forward)
error: failed to push some refs to 'git@github.com:vishaltyagi0053/gitsession.gi
t'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
ttn@ttn:~/gitsession$ git status
On branch master
Your branch and 'origin/master' have diverged,
and have 2 and 1 different commits each, respectively.
  (use "git pull" to merge the remote branch into yours)

nothing to commit, working tree clean
ttn@ttn:~/gitsession$ 
```

Q. 15 Edit the same file at the same line on both branches and commit

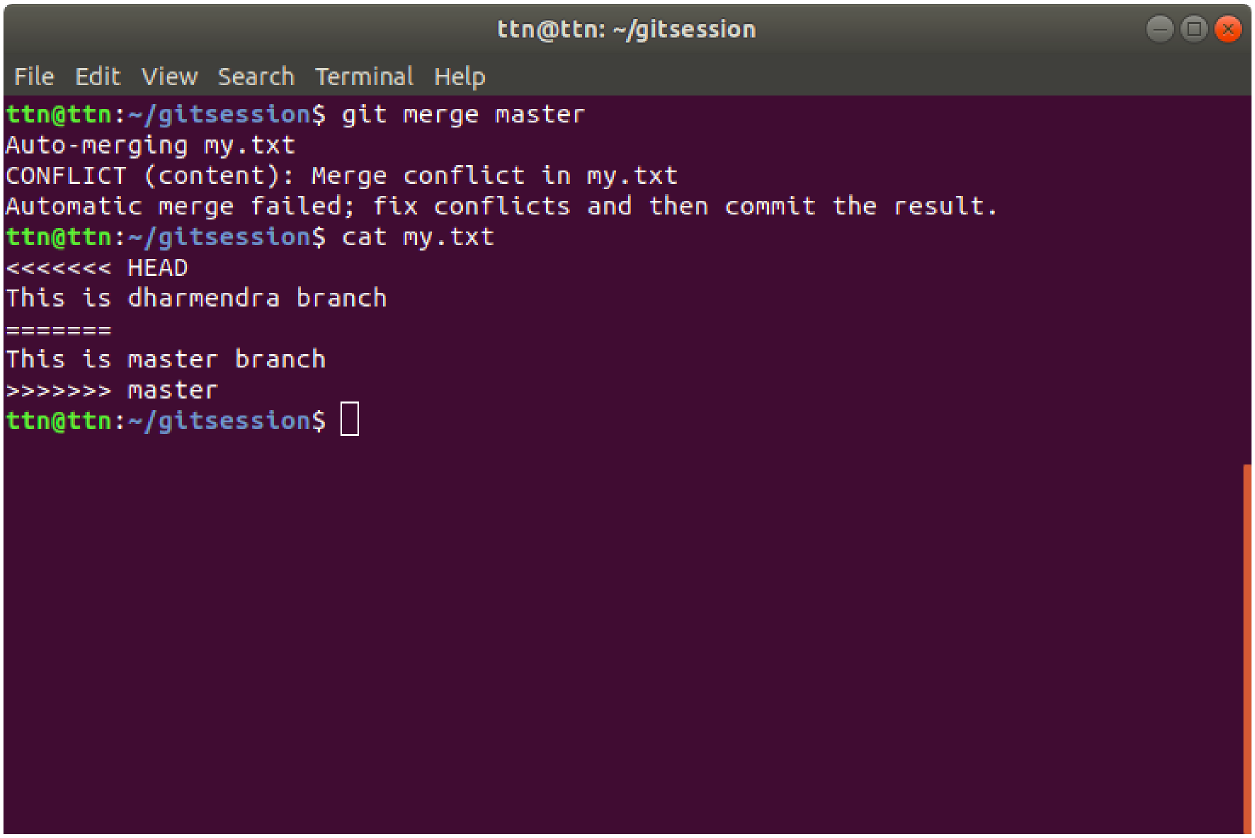
-> Here editing my.txt file line 1 in both branches and then performing commit on both one by one.



```
ttn@ttn:~/gitsession$ ls
"'"   gitsession   my.txt   target   text1.txt   text2.txt
ttn@ttn:~/gitsession$ git branch
* dharmendra
  master
ttn@ttn:~/gitsession$ echo "This is dharmendra branch" > my.txt
ttn@ttn:~/gitsession$ git add .
ttn@ttn:~/gitsession$ git commit -m "commit test"
[dharmendra bef30c3] commit test
 1 file changed, 1 insertion(+), 1 deletion(-)
ttn@ttn:~/gitsession$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
ttn@ttn:~/gitsession$ echo "This is master branch" > my.txt
ttn@ttn:~/gitsession$ git add .
ttn@ttn:~/gitsession$ git commit -m "commit test in master"
[master 11bbf46] commit test in master
 1 file changed, 1 insertion(+), 1 deletion(-)
ttn@ttn:~/gitsession$ 
```

Q. 16 Try merging and resolve merge conflicts

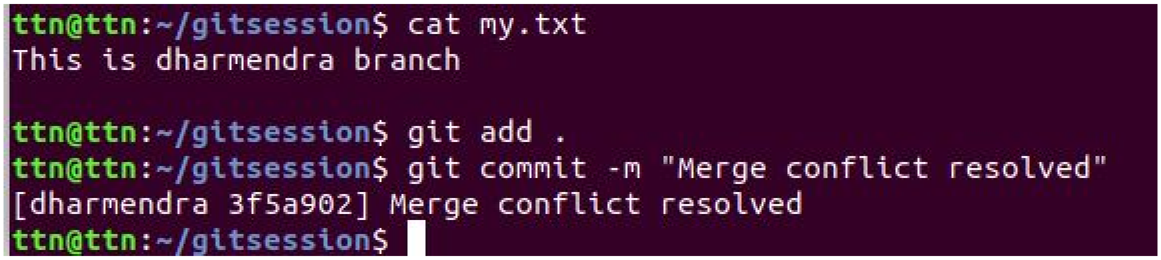
-> **Taking the previous question branches and try to merge them.**



```
ttn@ttn:~/gitsession$ git merge master
Auto-merging my.txt
CONFLICT (content): Merge conflict in my.txt
Automatic merge failed; fix conflicts and then commit the result.
ttn@ttn:~/gitsession$ cat my.txt
<<<<<<< HEAD
This is dharmendra branch
=======
This is master branch
>>>>>>> master
ttn@ttn:~/gitsession$ 
```

Resolving conflict by editing the file using vim editor.
-> vim my.txt

After edit,



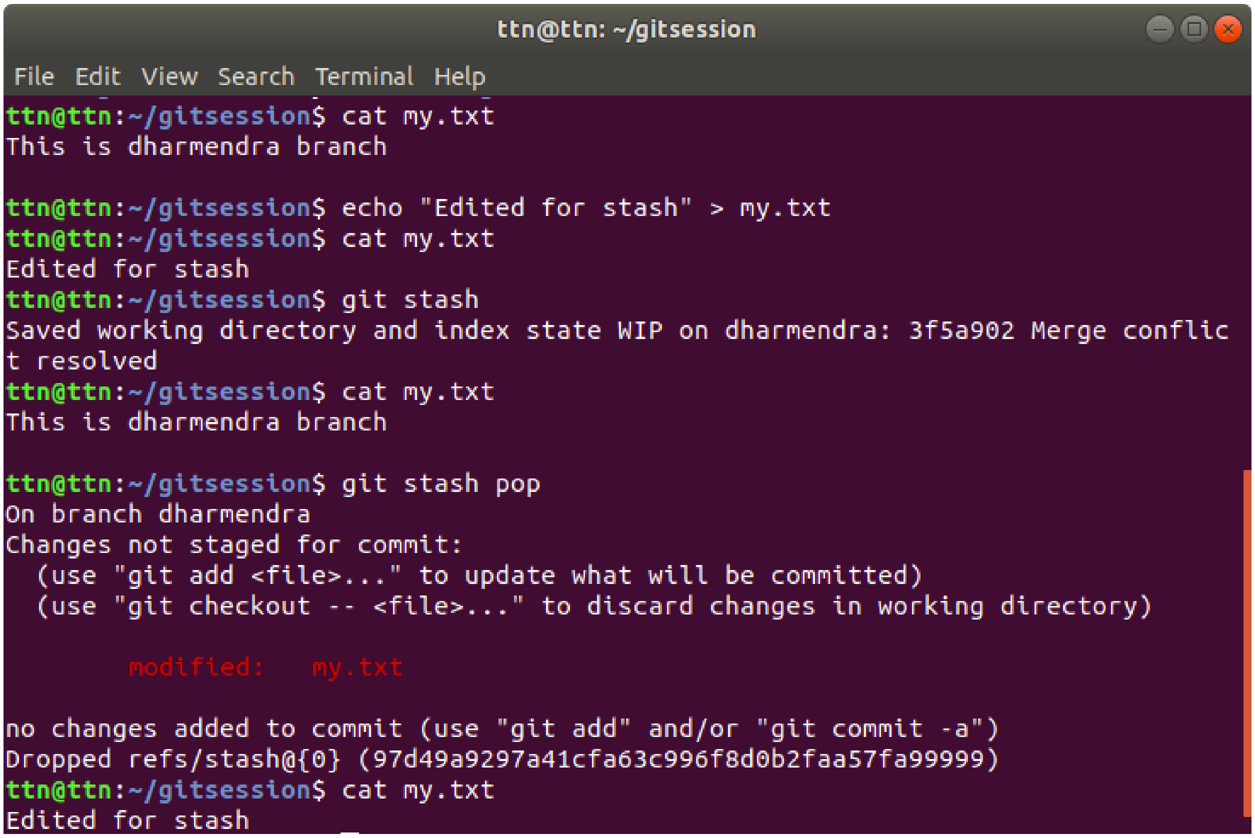
```
ttn@ttn:~/gitsession$ cat my.txt
This is dharmendra branch

ttn@ttn:~/gitsession$ git add .
ttn@ttn:~/gitsession$ git commit -m "Merge conflict resolved"
[dharmendra 3f5a902] Merge conflict resolved
ttn@ttn:~/gitsession$ 
```

Merge conflict resolved.

Q. 17 Stash the changes and pop them

-> **git stash**

-> git stash pop



```
ttn@ttn:~/gitsession$ cat my.txt
This is dharmendra branch

ttn@ttn:~/gitsession$ echo "Edited for stash" > my.txt
ttn@ttn:~/gitsession$ cat my.txt
Edited for stash
ttn@ttn:~/gitsession$ git stash
Saved working directory and index state WIP on dharmendra: 3f5a902 Merge conflic
t resolved
ttn@ttn:~/gitsession$ cat my.txt
This is dharmendra branch

ttn@ttn:~/gitsession$ git stash pop
On branch dharmendra
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:   my.txt

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (97d49a9297a41cfa63c996f8d0b2faa57fa99999)
ttn@ttn:~/gitsession$ cat my.txt
Edited for stash
```

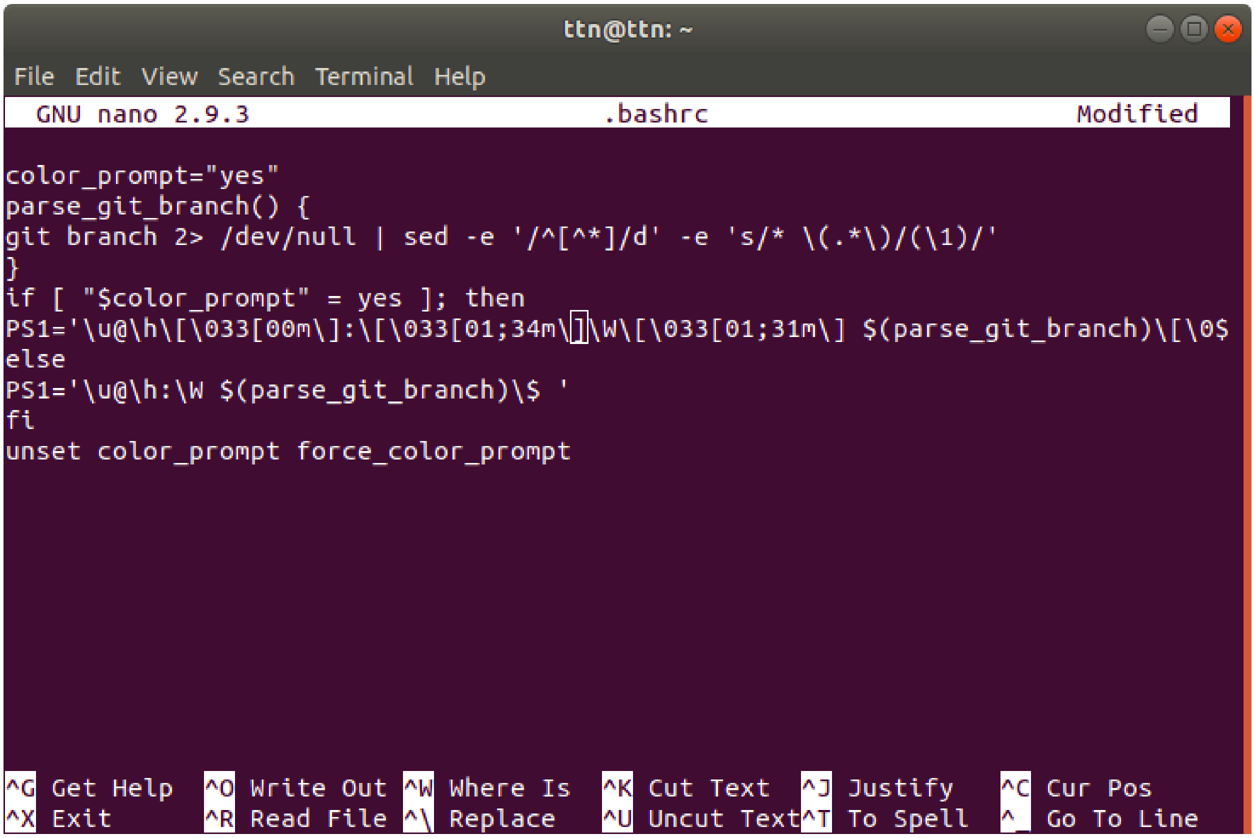
Q. 18 Add the following code to your .bashrc file : color_prompt="yes"

```
        parse_git_branch() {
        git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
        }
        if [ "$color_prompt" = yes ]; then
        PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\]
$(parse_git_branch)\[\033[00m\]\$ '
        else
        PS1='\u@\h:\W $(parse_git_branch)\$ '
        fi
        unset color_prompt force_color_prompt
```

->

Added the above code into .bashrc file using nano editor.

nano .bashrc



```
  GNU nano 2.9.3                      .bashrc                         Modified

color_prompt="yes"
parse_git_branch() {
git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
}
if [ "$color_prompt" = yes ]; then
PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\] $(parse_git_branch)\[\0$
else
PS1='\u@\h:\W $(parse_git_branch)\$ '
fi
unset color_prompt force_color_prompt











^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text^T To Spell    ^_ Go To Line
```