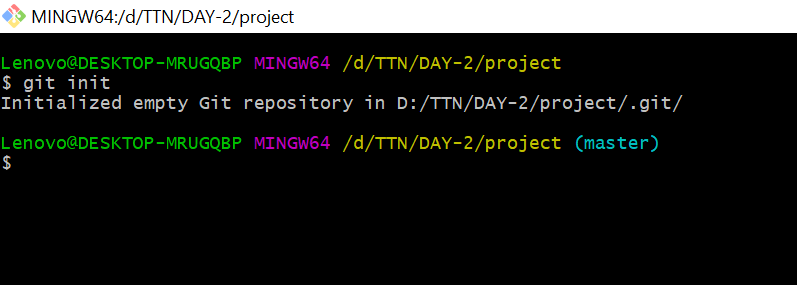
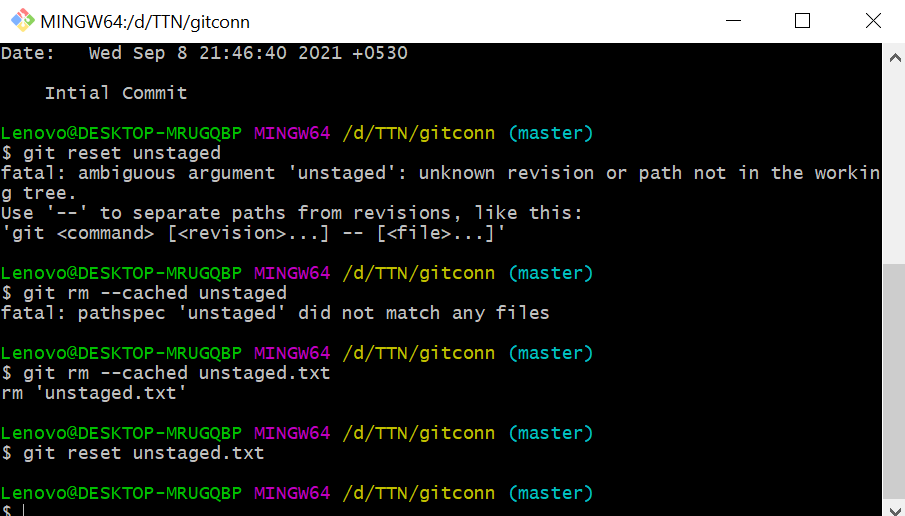
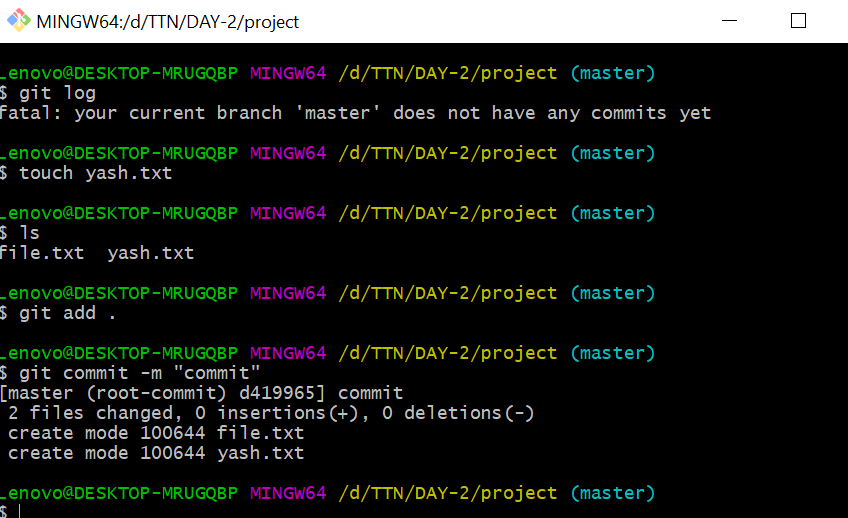
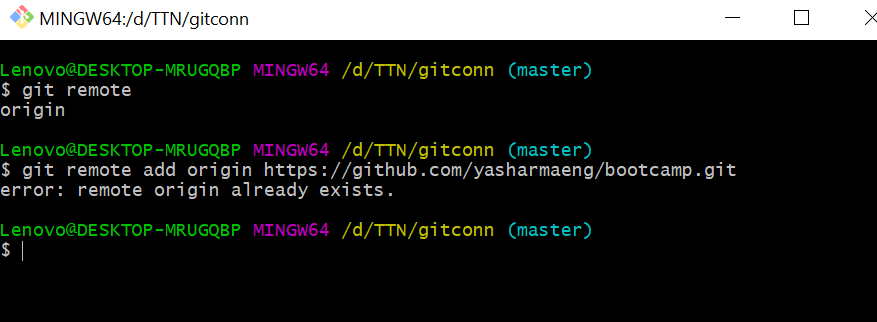
1. Initialize a Git Repository



1. Add files to the repository



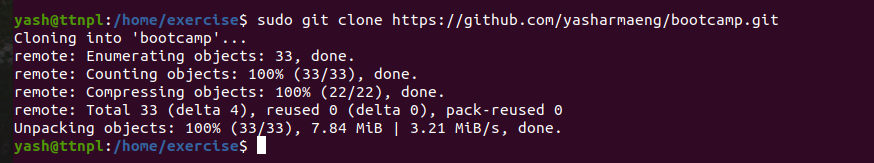
1. Unstage 1 file
2. Commit the file
3. Add a remote



1. Undo changes to a particular file
2. Push changes to Github



1. Clone the repository



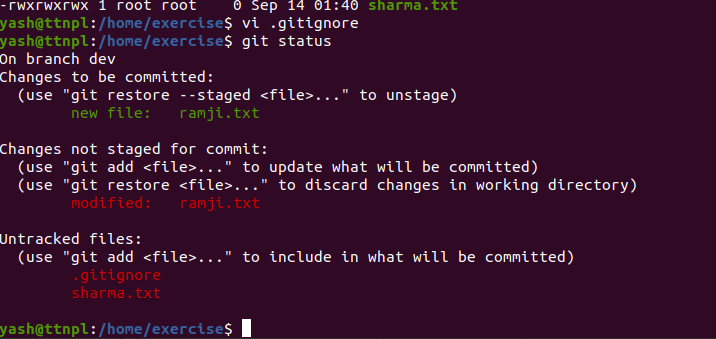
1. Add changes to one of the copies and pull the changes in the other.
2. Check differences between a file and its staged version



1. Ignore a few files to be checked in

Create a .gitignore file and add filenames which you want to ignore.

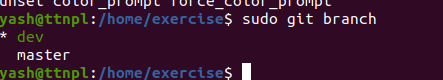
And then check git status.



1. Create a new branch.

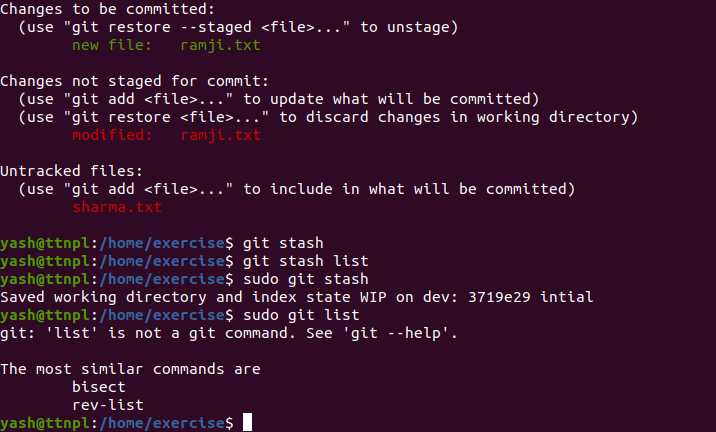


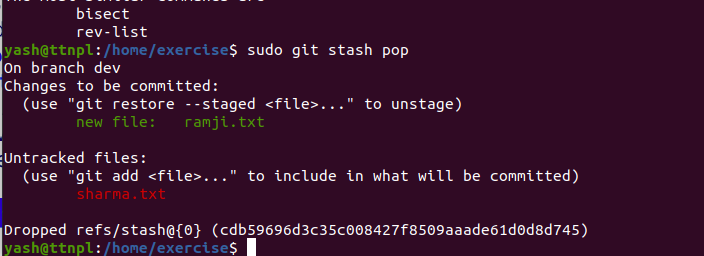
1. Diverge them with commits



1. Edit the same file at the same line on both branches and commit
2. Try merging and resolve merge conflicts
3. Stash the changes and pop them

STASH: If you are working in a branch and you have to work ion another branch just for few minutes and then continue again in the same branch so you use git stash. It is a place where all your work is stored in a secured and protected manner temporarily.





1. 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

