**DevOps Lab**

**Exercise 4**

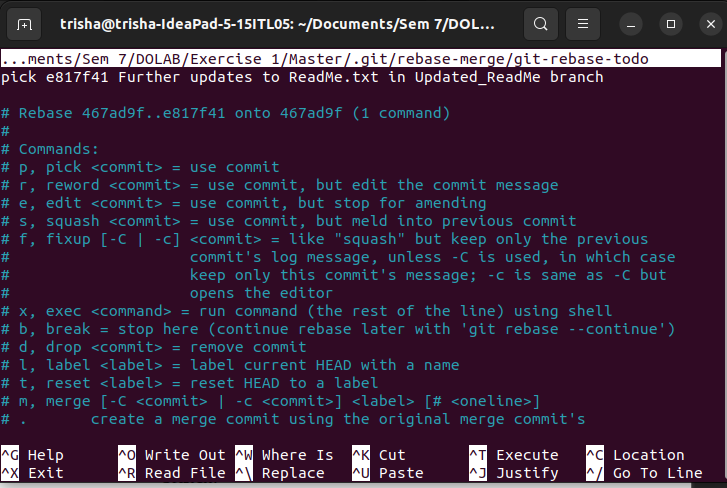
Trisha Balakrishnan

21011102105

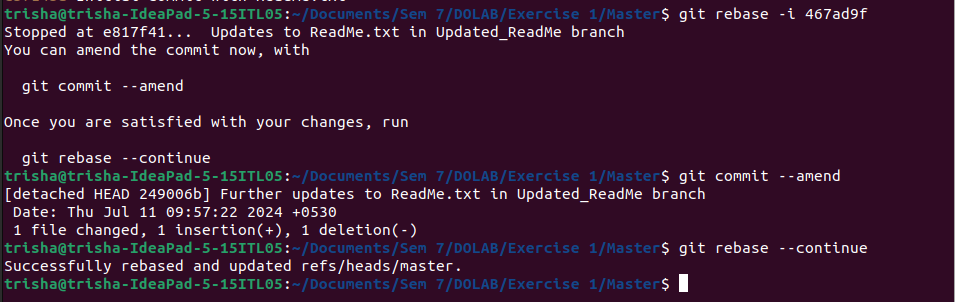
IoT-B

1. **Exploring Git Commands through Collaborative Coding – Advanced Git commands.**
2. **Use interactive rebase** to combine to multiple commits into one

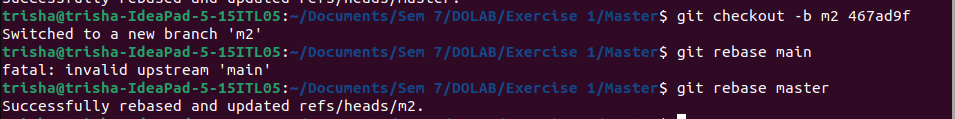
* Use git rebase -i to modify the commit history.
* Interactive Rebase to Edit Multiple Commits
* In the interactive rebase interface, change pick to edit or squash as needed to edit messages or combine commits.



* Rebase onto Another Branch.

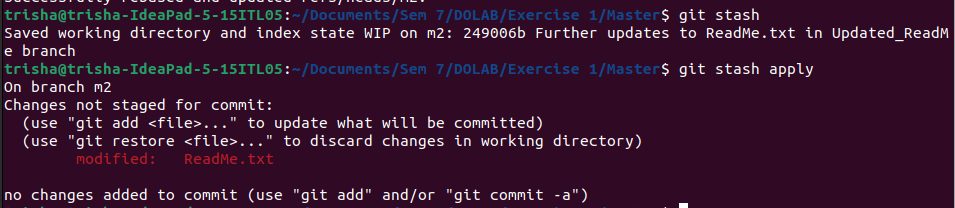


* Create a new branch from an earlier point in your commit history.
* Use rebase to apply the commits from main onto this new branch.

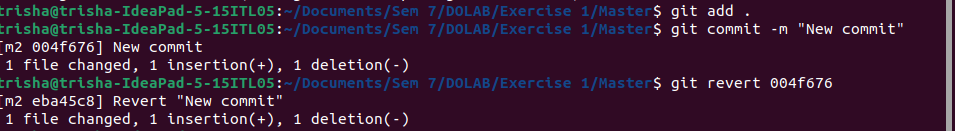


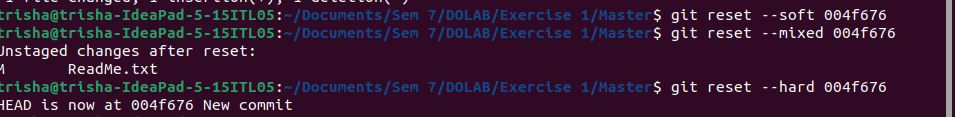
1. **Stash Changes**

* Make some changes in your working directory.
* Stash those changes, and then apply the stash.

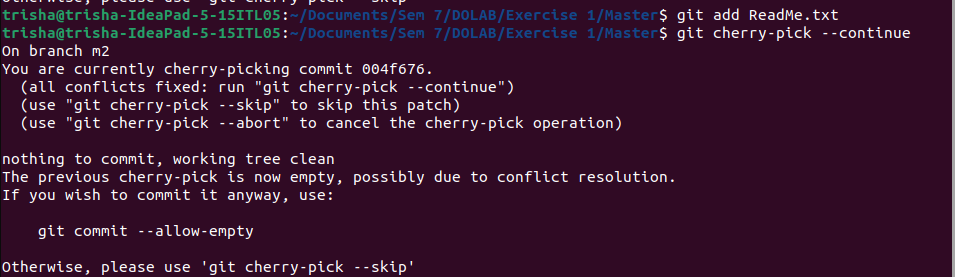
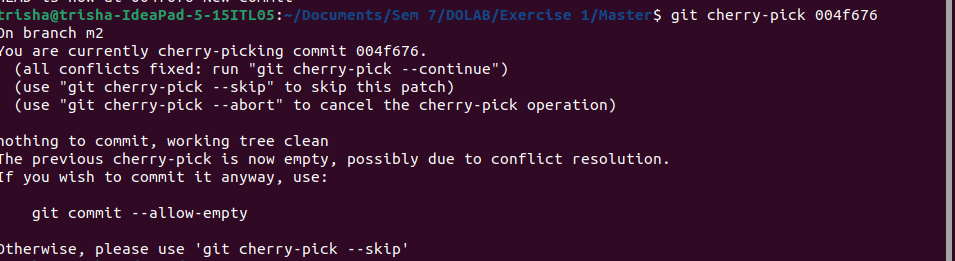


1. **Revert and Reset**

* Create a new commit, and then use git revert to undo it.
* Experiment with git reset to understand the difference between --soft, --mixed, and –hard.

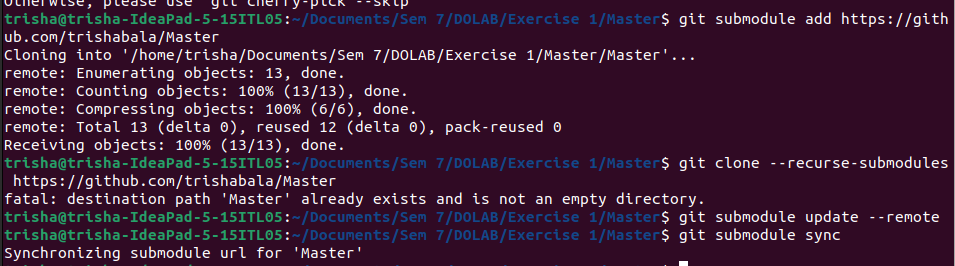


1. **Cherry-Pick a Commit**

* Resolve Conflicts During Cherry-Picking
* Resolve Conflicts During Cherry-Picking

1. **Working with Submodules**

* In your repository, add another repository as a submodule.
* Clone a Repository with Submodules
* Update and Synchronize Submodules



1. **Git Hooks**

* Create a Pre-Commit Hook that prevents commits with “TODO” comments
* Use the below script

#!/bin/sh

if grep -r "TODO" .; then

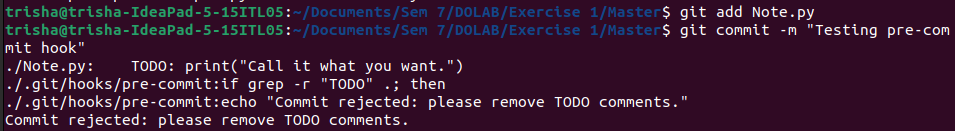
echo "Commit rejected: please remove TODO comments."

exit 1

fi



* Make the script executable and try committing a file with a TODO comment.



* Explore other types of hooks like post-merge, pre-push, or post-checkout, and create simple scripts for them.