iNeuron < Back Curriculum | Analytics | Certificate

GIT Rewrite

Score: 10/11

✓ **1. What does GIT Rewrite History allow you to do?**

- Change the commit messages only
- [✓ Edit, remove, or reorder commits]
- Merge branches in the repository
- Push commits to remote branch directly

Explanation

GIT Rewrite History allows you to modify commit history by editing, removing, or reordering commits.

✓ **2. What does 'git rebase' do?**

- Create a new branch
- Merge branches together
- [✓ Apply commits from one branch to another]
- Delete a branch

Explanation

'git rebase' is used to apply commits from one branch to another. It is often used to maintain a linear project history.

✓ **3. What is interactive rebasing in GIT?**

Automatically rebasing all commits

Rebasing without any user interaction

[✓ Reordering, editing, or deleting previous commits]

Creating a new commit from scratch

Explanation

Interactive rebasing allows you to combine, edit, or delete previous commits before finalizing the changes. It offers a more granular control over the commit history.

✓ **4. How does 'git cherry-pick' work?**

Merge all commits from one branch to another

[✓ Apply a specific commit from one branch to another]

Create a new branch from an existing commit

Delete all commits in a branch

Explanation

'git cherry-pick' is used to apply a specific commit from one branch to another. It allows you to pick individual commits and apply them to a new branch or location in the repository.

✓ **5. Which command is used to start an interactive rebase in GIT?**

git rebase -start

git rebase -interactive

[✓ git rebase -i]

git rebase -modify

Explanation

The command 'git rebase -i' is used to start an interactive rebase in GIT.

✓ **6. What does 'git commit --amend' do?**

Create a new commit

[✓ Modify the last commit]

Delete the last commit

Undo all changes in the last commit

Explanation

'git commit --amend' is used to modify the last commit. It allows you to change the commit message or add changes to the previous commit.

✓ **7. Why is it important to be cautious when rewriting history in a shared repository?**

It has no impact on other developers

[✓ It can cause inconsistencies and conflicts for other developers]

It automatically updates other developers' branches

Explanation

Rewriting history can lead to inconsistencies and conflicts if other developers have already based their work on the existing history. It is important to communicate any history changes with the team and choose rewriting options carefully.

It speeds up the repository performance

✓ **8. In which order are the commits applied during a rebase operation?**

- Newest to oldest
- Random order
- [✓ Oldest to newest]
- Based on commit message length

Explanation

During a rebase operation, commits are applied from the oldest to the newest, allowing for a linear integration of commits from one branch to another.

✓ **9. What can be a potential consequence of force-pushing rewritten history to a remote repository?**

- It automatically merges the rewritten history
- It preserves other developers' commit history
- [✓ It can cause confusion and potential loss of work for other developers]
- It improves the repository's commit history

Explanation

Force-pushing rewritten history to a remote repository can lead to rewriting the commit history for other developers, causing confusion and potential loss of their work. It is important to use force-push with caution and communicate with the team.

✗ **10. What is the difference between 'git rebase' and 'git merge'?**

- Both commands rewrite the commit history

Explanation

[✓ 'git rebase' creates new commits, 'git merge' integrates commits without altering the original history]

'git merge' creates new commits, 'git rebase' integrates commits without altering the original history

There is no difference between the two commands

The main difference is that 'git rebase' rewrites the commit history by creating new commits, while 'git merge' integrates the commits from one branch into another without altering the original commit history.

✓

11. When using interactive rebasing, what keyword can be used to squash commits into one?

Condense

Combine

[✓ Squash]

Merge

Explanation

The keyword 'squash' can be used during interactive rebasing to combine multiple commits into a single commit.