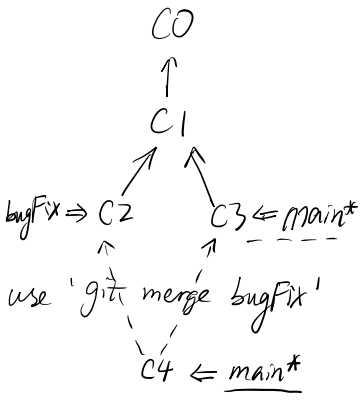
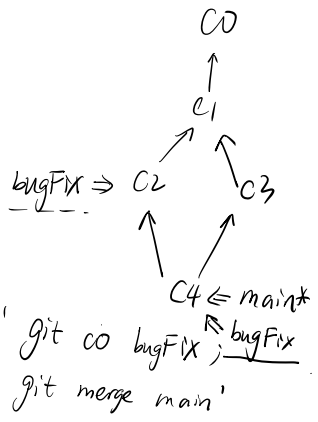
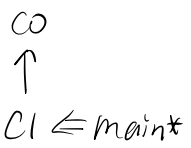
2. Merging

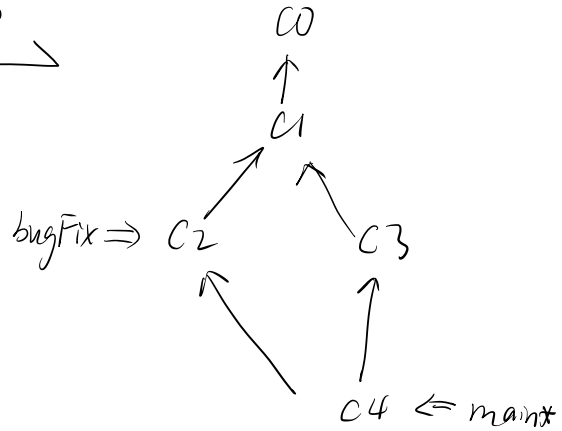




Practice:



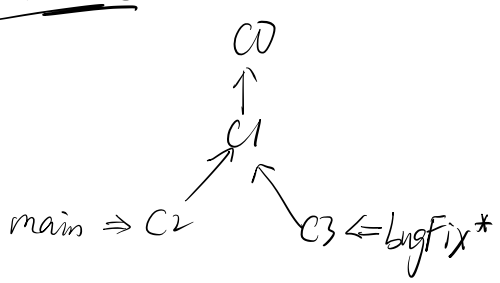
?



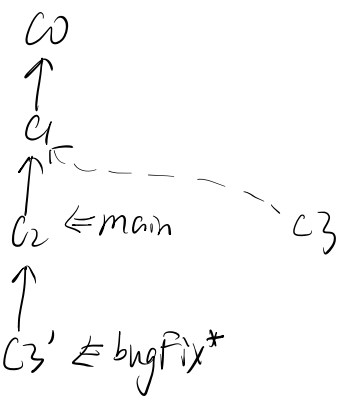
Sol:

```
'''
git checkout -b bugFix
git commit
git checkout main
git commit
git merge bugFix
'''
```

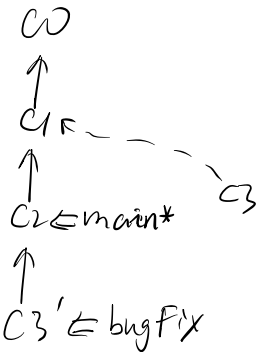
3. Rebase



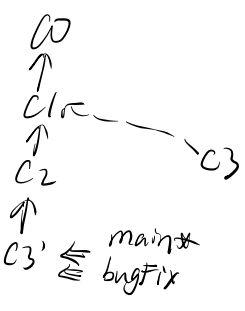
git rebase main



git co main

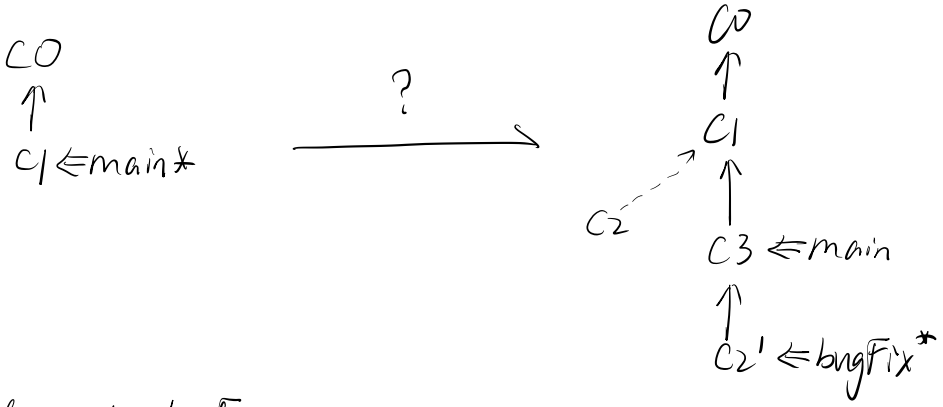


git rebase bugFix [1]



[1] Since main is the ancestor of bugFix, then git simply move main branch reference forward in history.

Practice

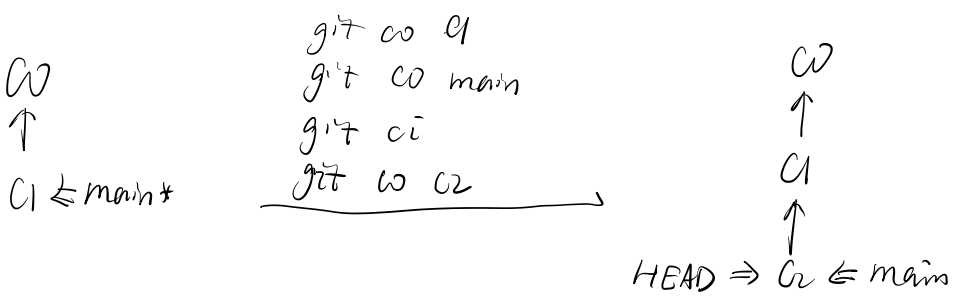


Solution:

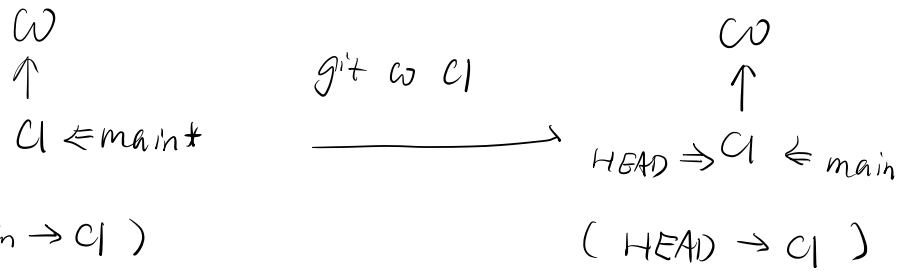
```
git checkout -b bugFix
git commit ...
git co main
git commit
git checkout bugFix
git rebase main
```

Ramping up

1. 'HEAD'



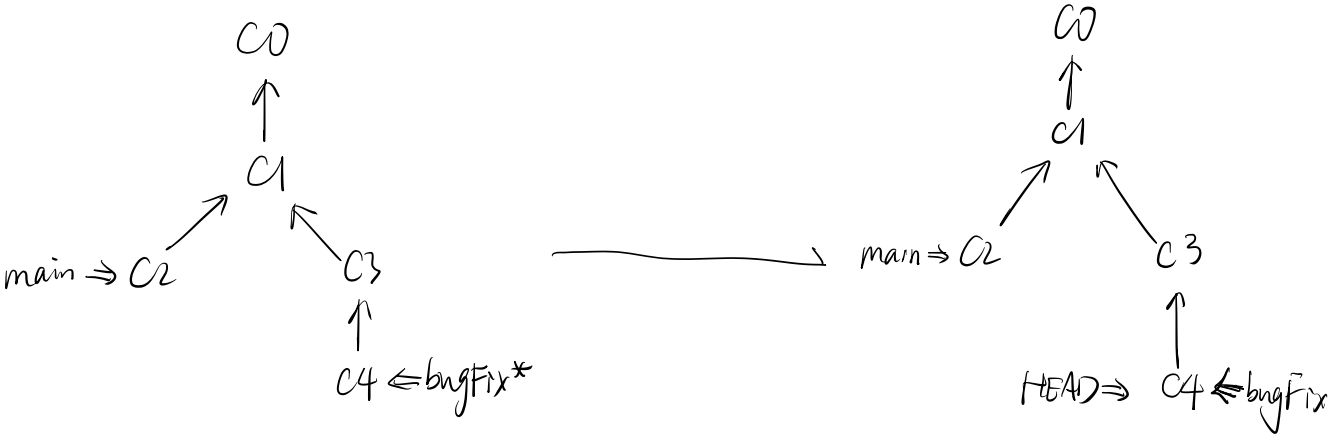
2. Detaching 'HEAD'



(HEAD → main → C1)

(HEAD → C1)

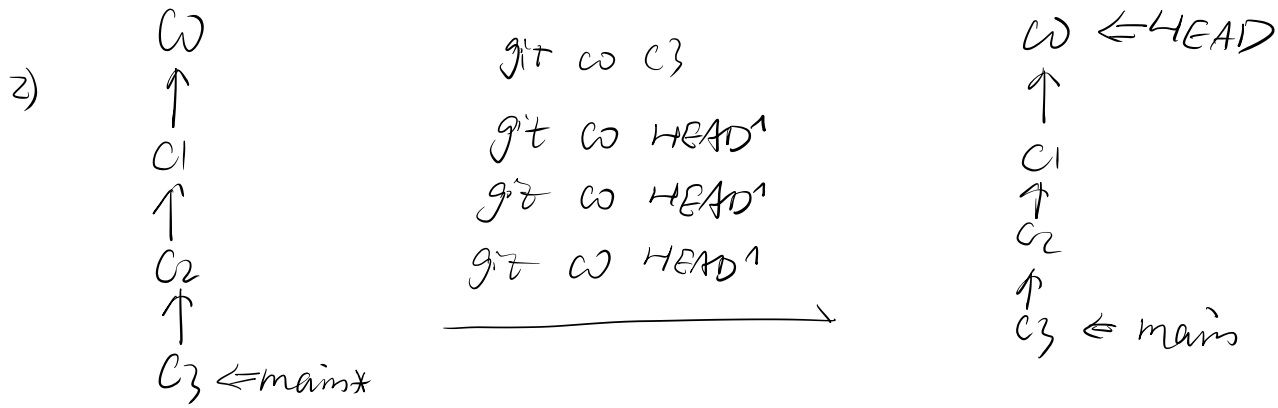
Practice:



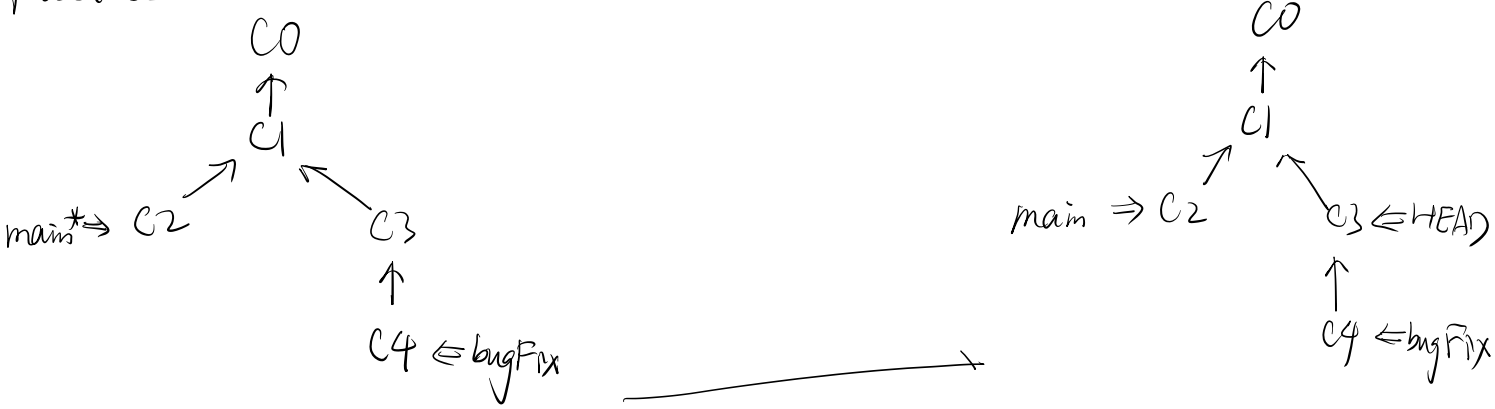
Solution

git checkout C4

3. Relative Refs



Practice:



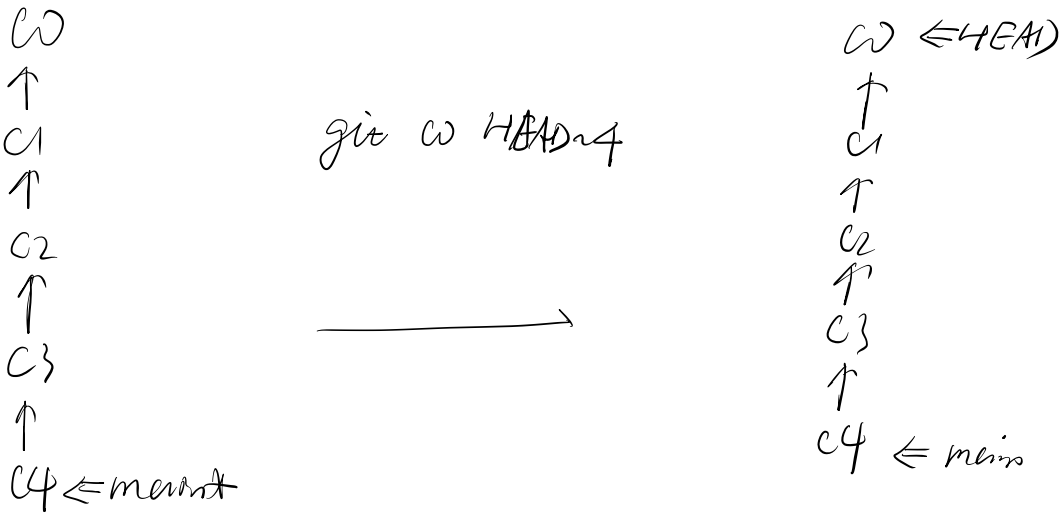
Solution:

git co bugFix^ (relative)

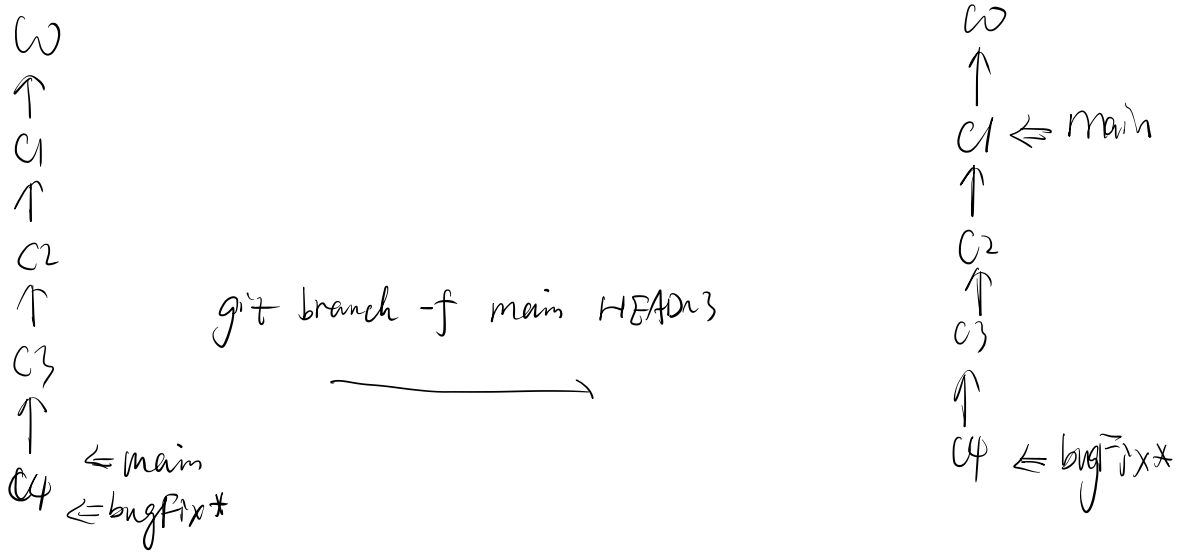
OR

git co c3 (hash)

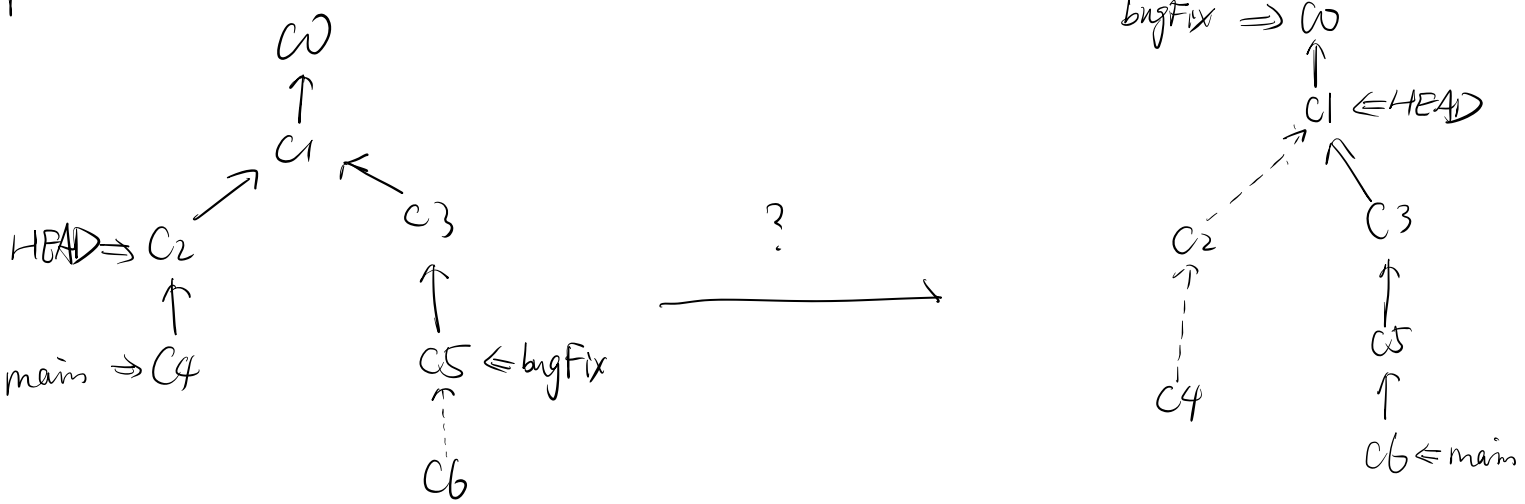
3)



4)



Practice:



Solution: `git co c6`

`git branch -f main HEAD`

`git branch -f bugFix HEAD~4`

~~HEAD HEAD~3~~

`git co c1`

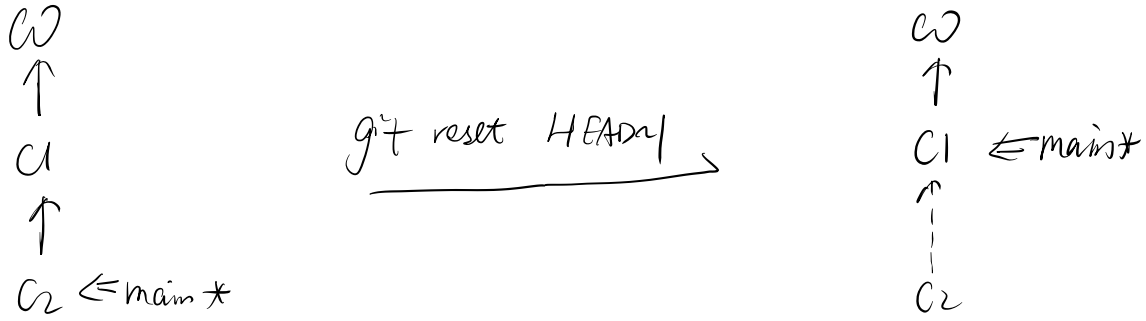
`git branch -f main c6`

`git checkout HEAD~1`

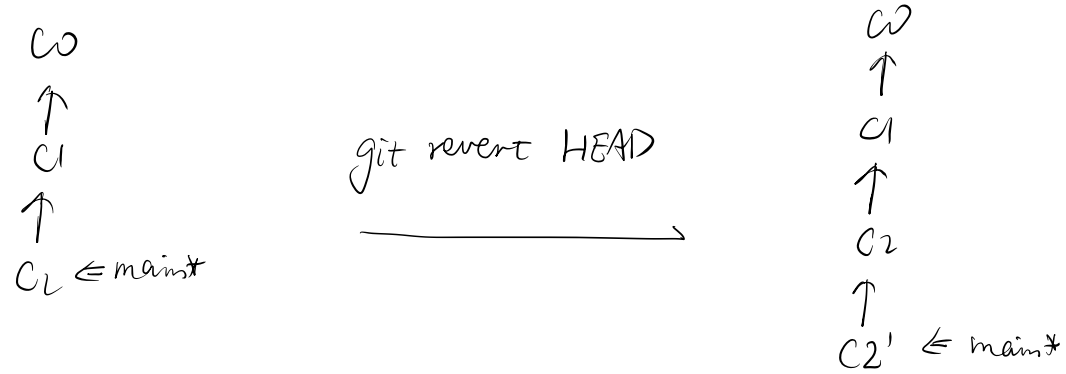
`git branch -f bugFix HEAD~1`

4. Reversing in Git

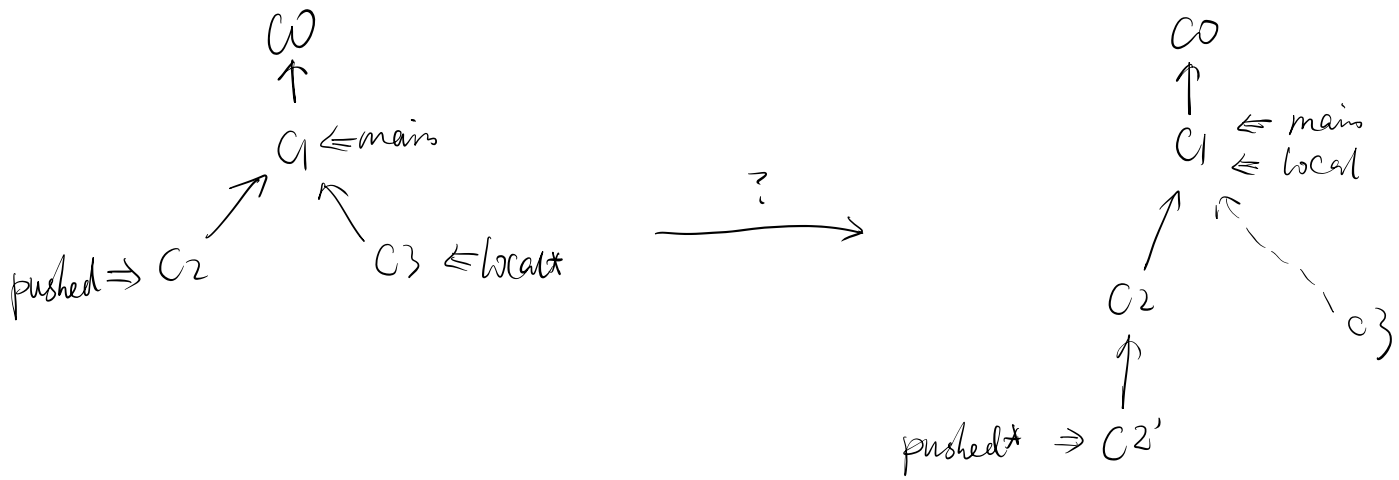
1)



2)



Practice:



Solution: git reset HEAD~1

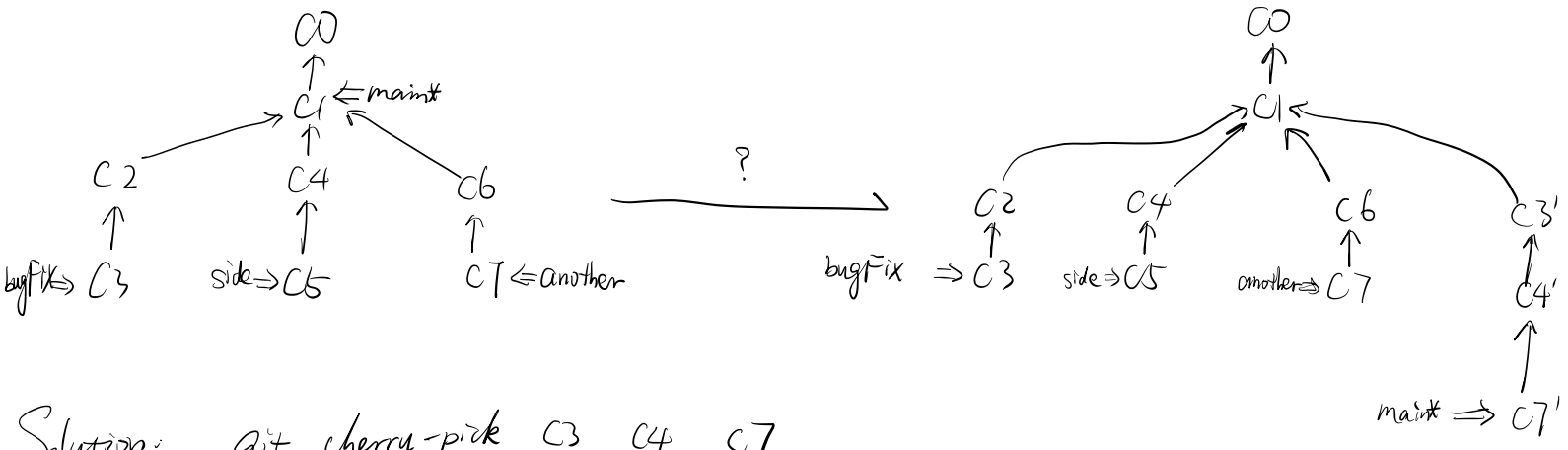
git co pushed

git revert pushed /HEAD

Moving Work Around

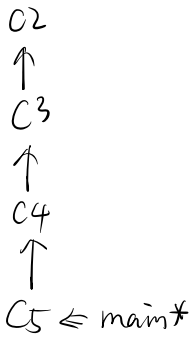
1. Cherry-pick (copy)

Practice:



Solution: git cherry-pick C3 C4 C7

2. Git interactive rebase 'git rebase -i '



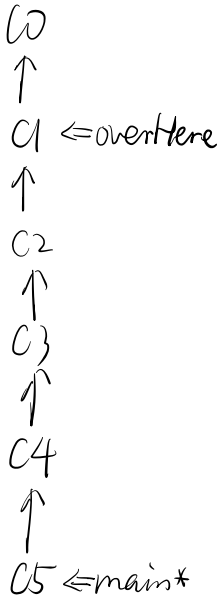
git rebase -i HEAD~4

→ (drag/pick/...)

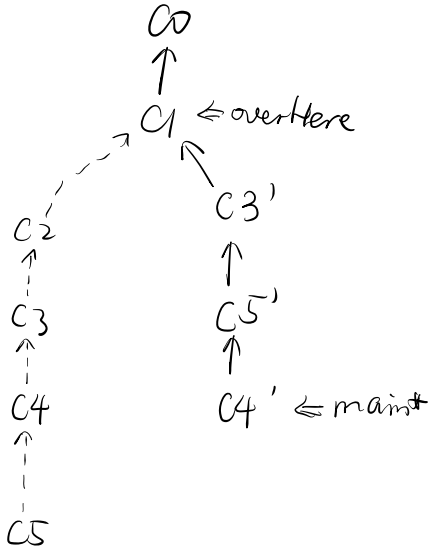
(not include (last: C1))

reorder/pick from { C2, C3, C4, C5

Practice:



?



Solution:

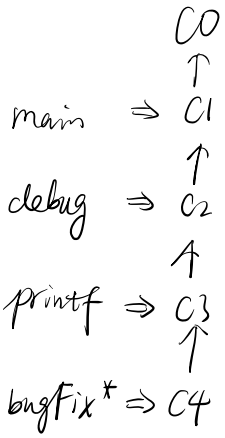
1) git branch -f main HEAD~4/overHere

git cherry-pick C3 C5 C4

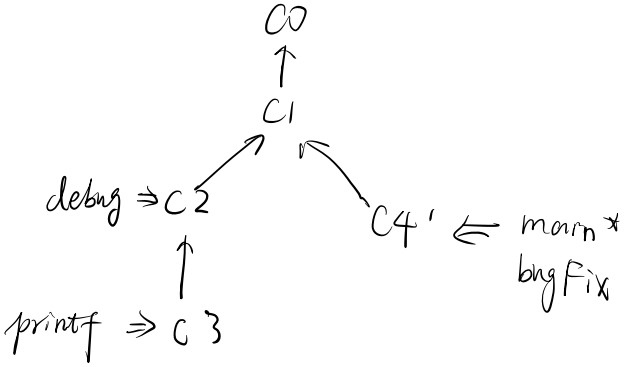
2) git rebase -i overHere

A Mixed bag

Practice



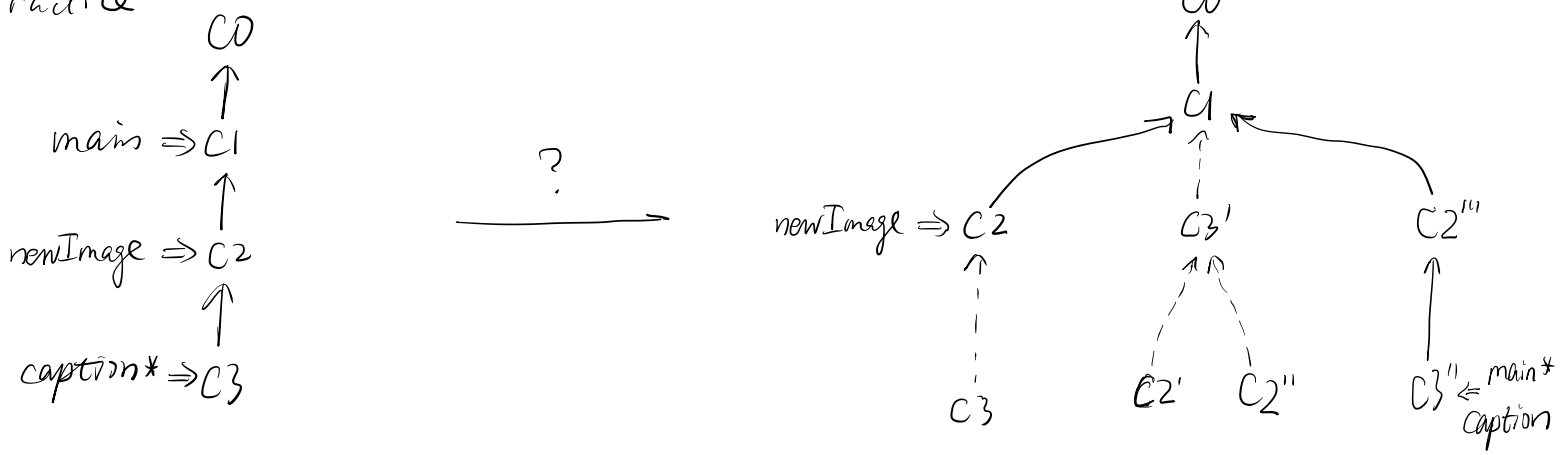
?



Solutions: 1) git checkout main
git cherry-pick C4

2) git rebase -i HEAD~3
(pick C4)
(git branch -f main HEAD)
git rebase bugFix main

Practice



Solutions:

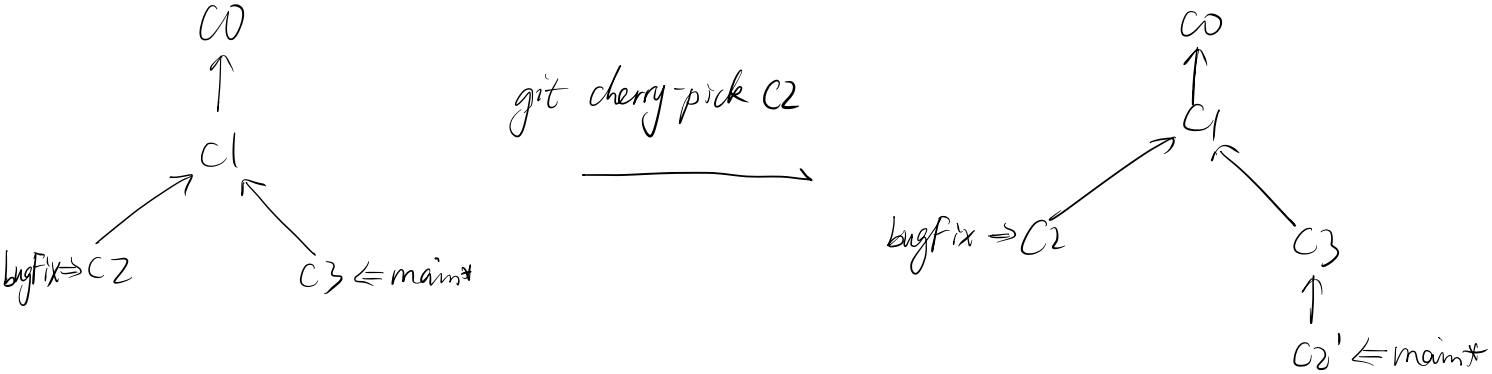
git rebase -i main +

git commit --amend

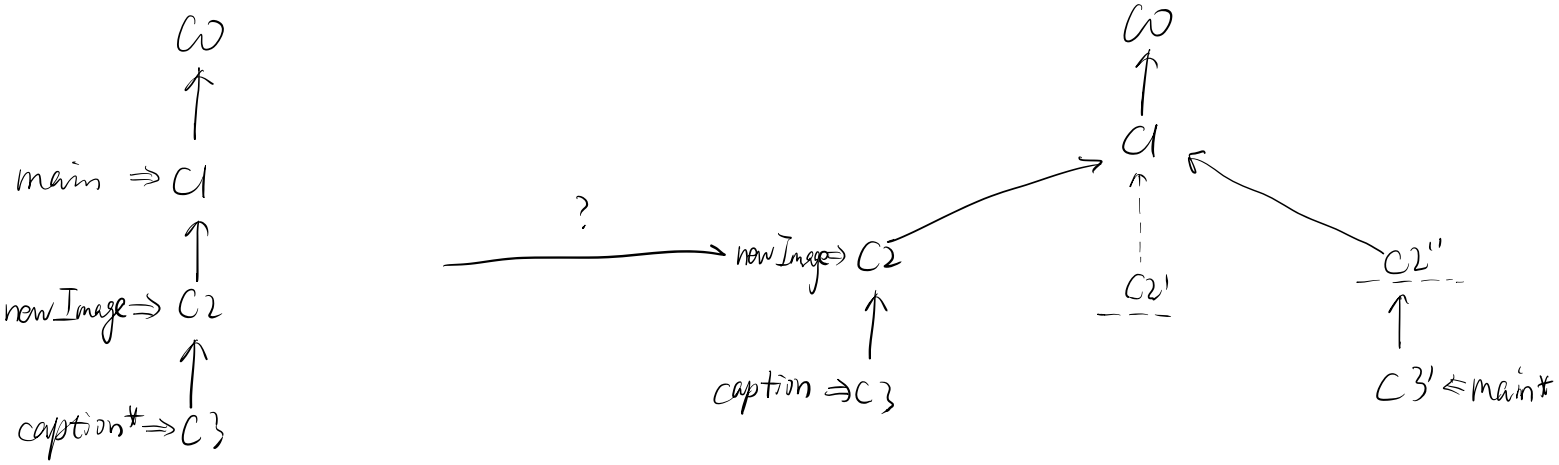
git rebase -i main +

git branch -f main caption / git rebase caption main (order!)

~ ~ ~



Practice: using cherry-pick

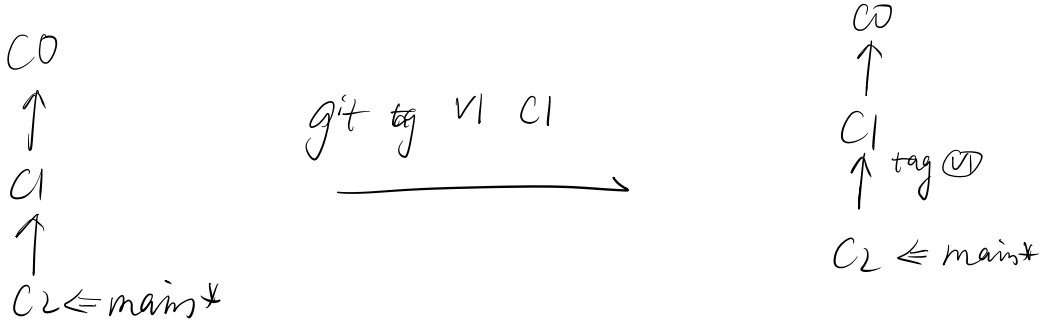


Solution:

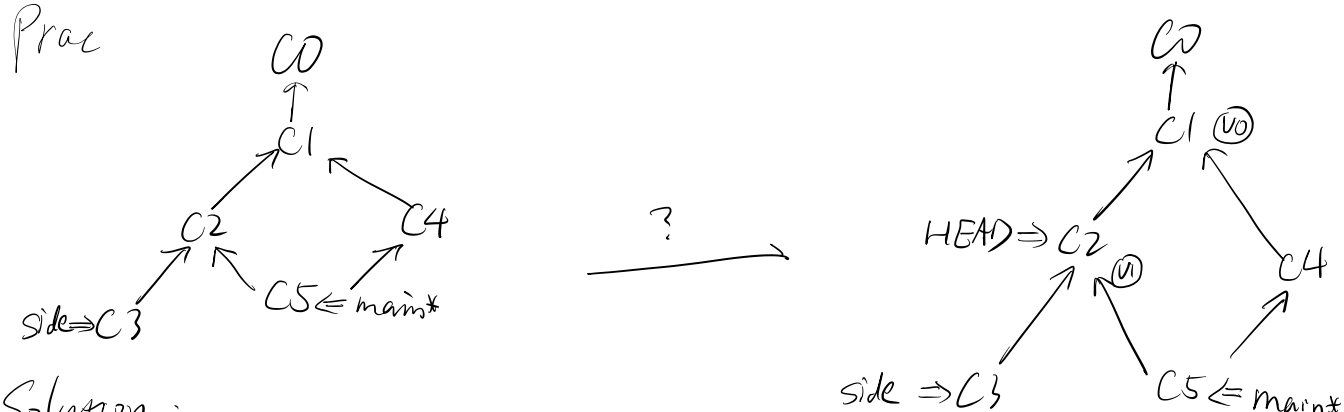
```
git checkout C1
git cherrypick C2
git checkout main
git cherry-pick C2' C3
```

```
git checkout main
git cherry-pick C2
git commit --amend (C2'→C2'')
git cherry-pick C3
```

Git Tags



Prac

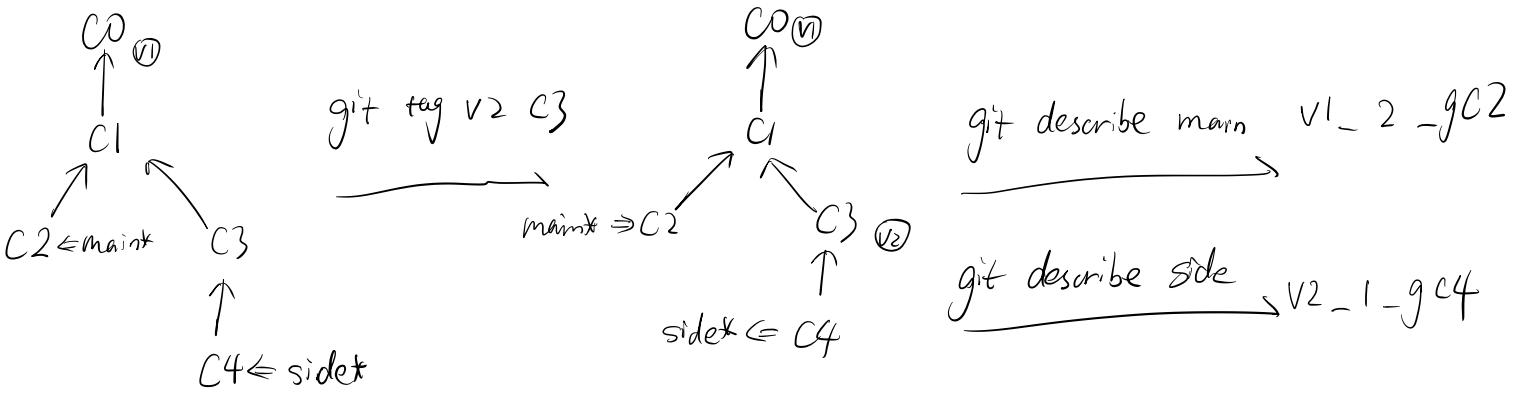


Solution:

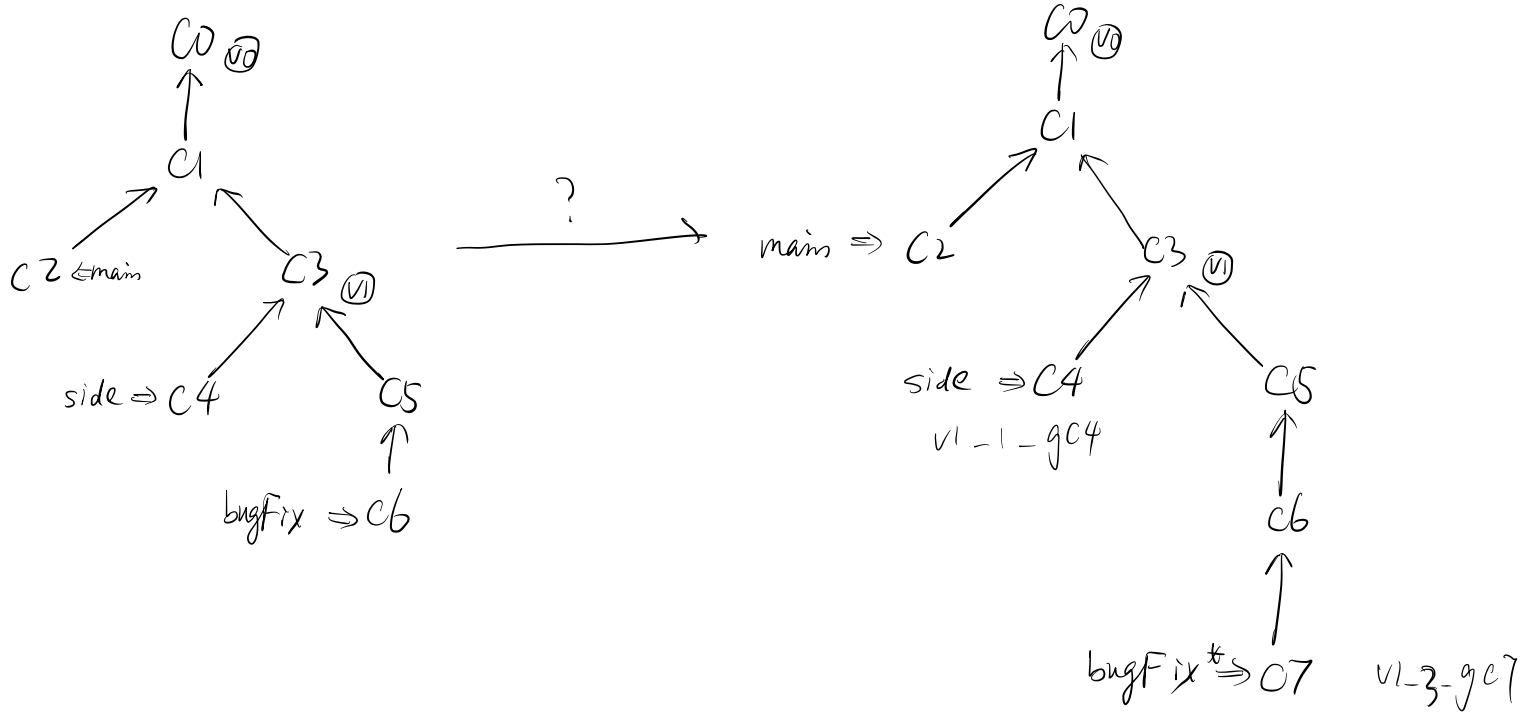
```
git tag V1 C2
git tag V0 C1
git checkout C2
```

```
git tag side C1
git tag main C2
git checkout V1
```

Git describe

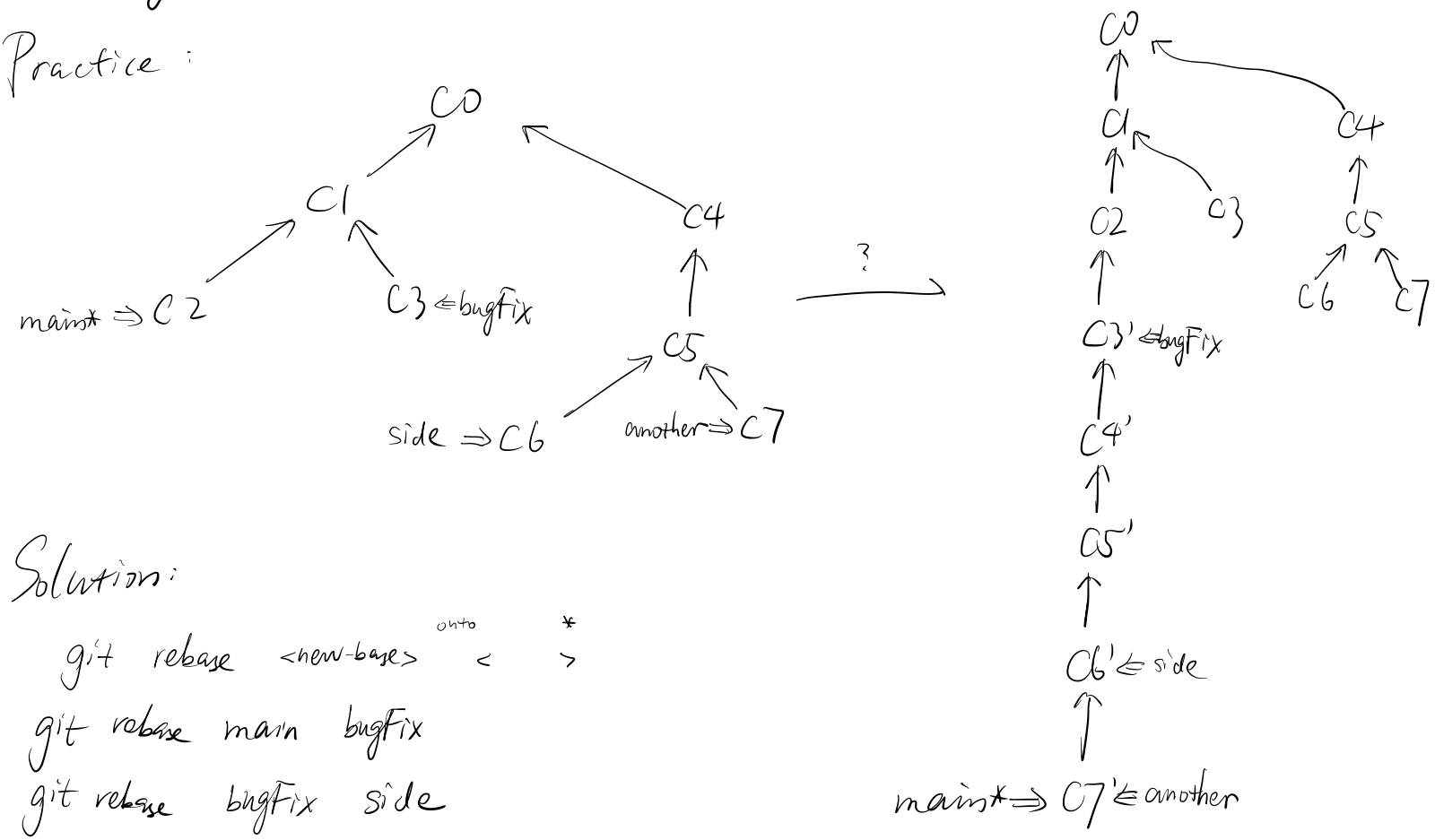


Practice



Rebasing Multiple Branches

Practice:



Solution:

git rebase <new-base> onto < * >

git rebase main bugFix

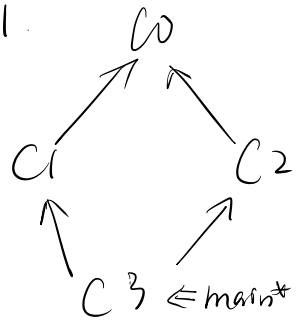
git rebase bugFix side

git rebase side another

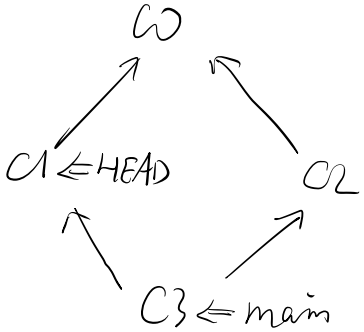
git rebase another main

Multiple parents

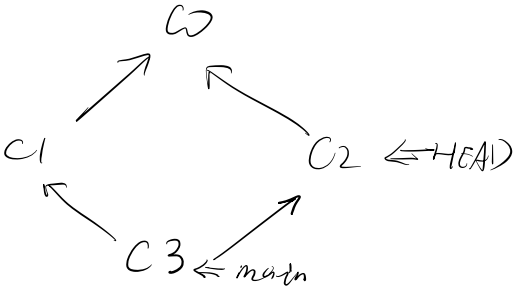
eg 1.



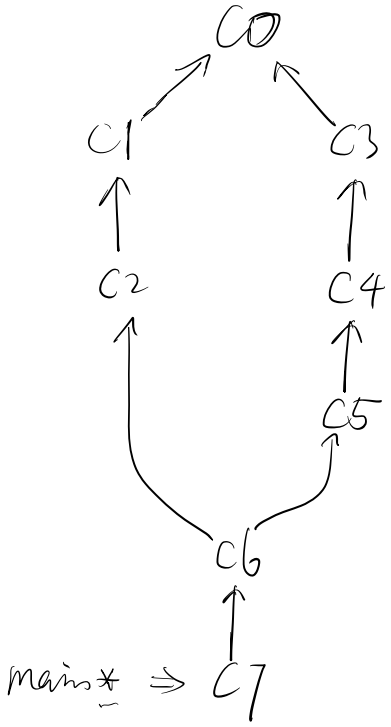
git checkout main^



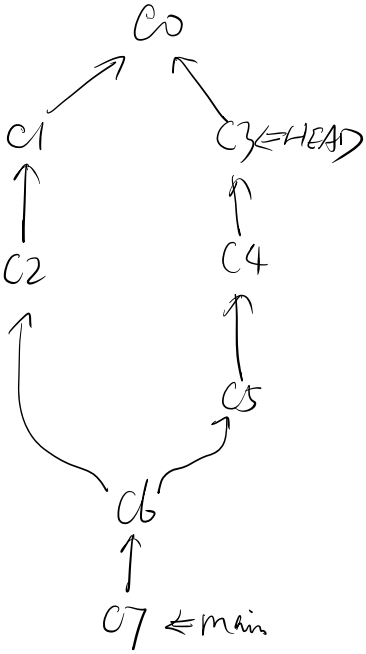
git checkout main^2



eg 2



1. git checkout HEAD^



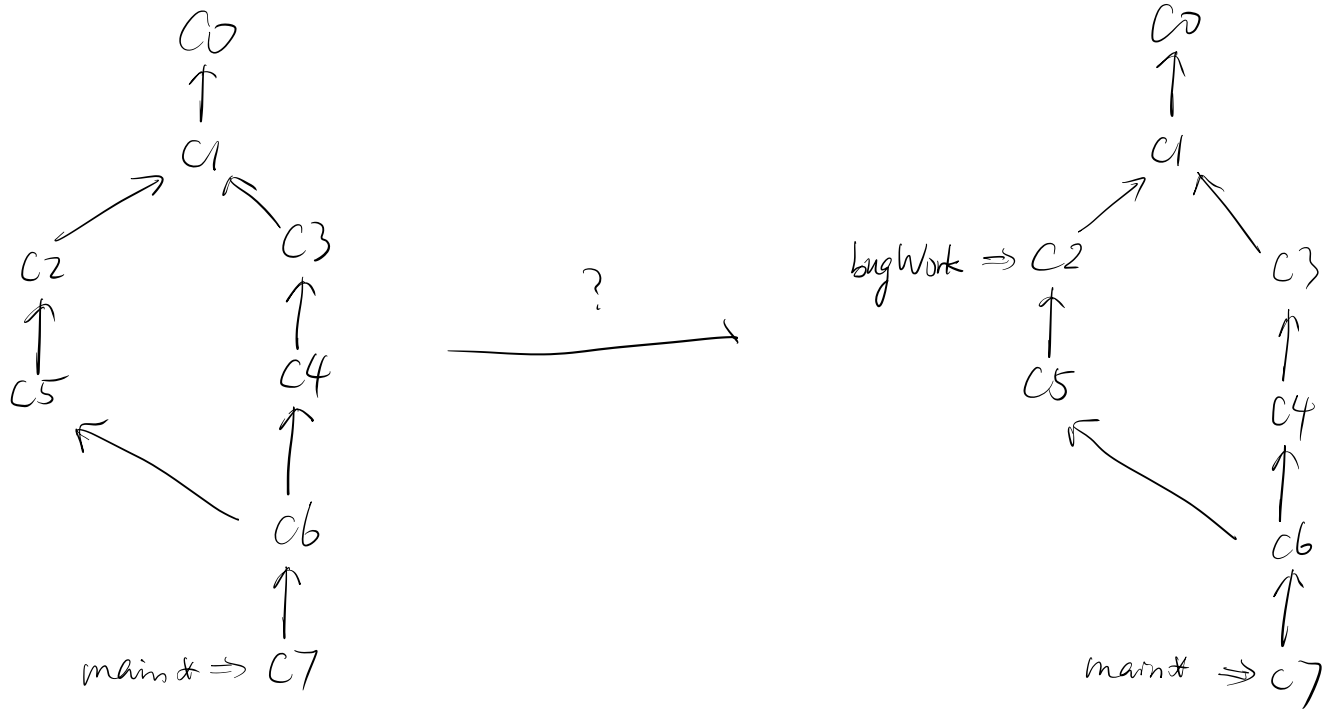
2. git checkout HEAD^2

3. git checkout HEAD^2

OR chaining :

git checkout HEAD~^2~2

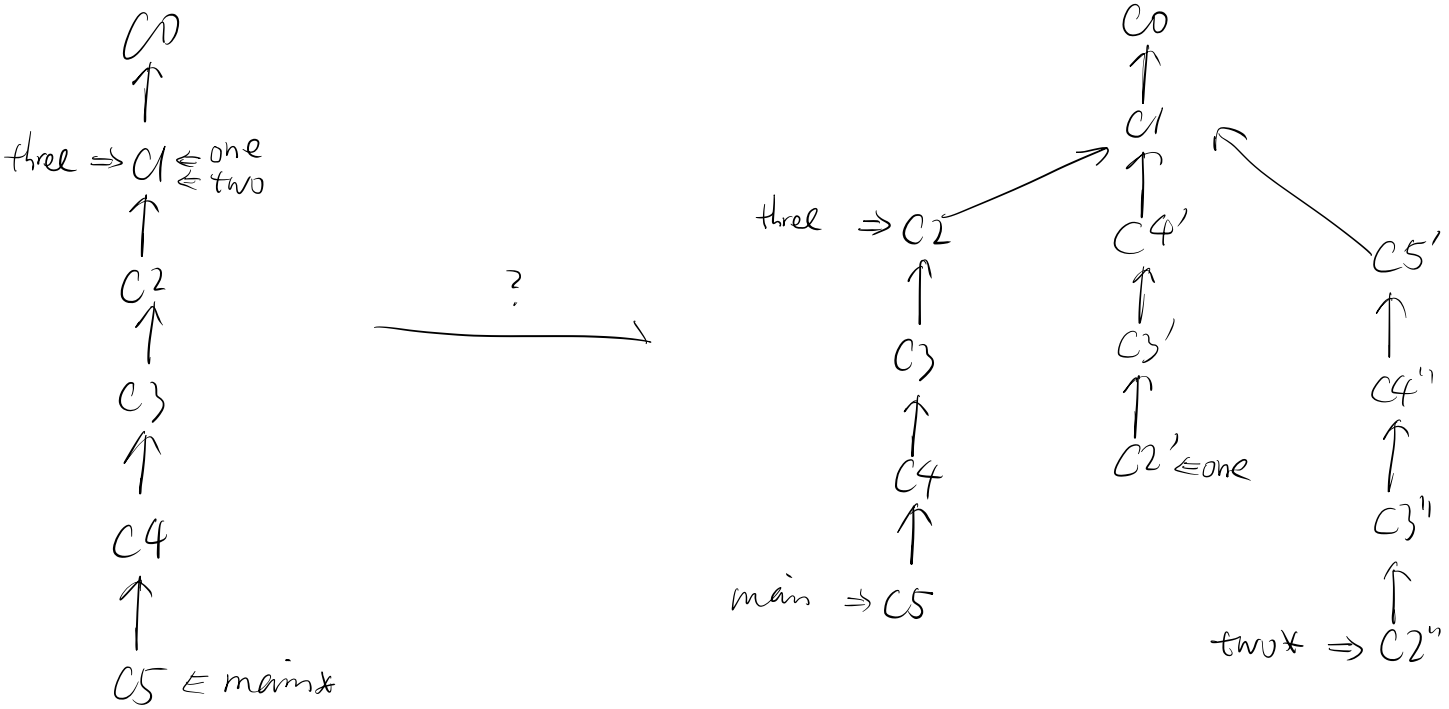
Practice:



Solution:

git branch bugWork main~1^2~

OR main~1^2^1



Solution: git checkout one

git cherry-pick C4 C3 C2

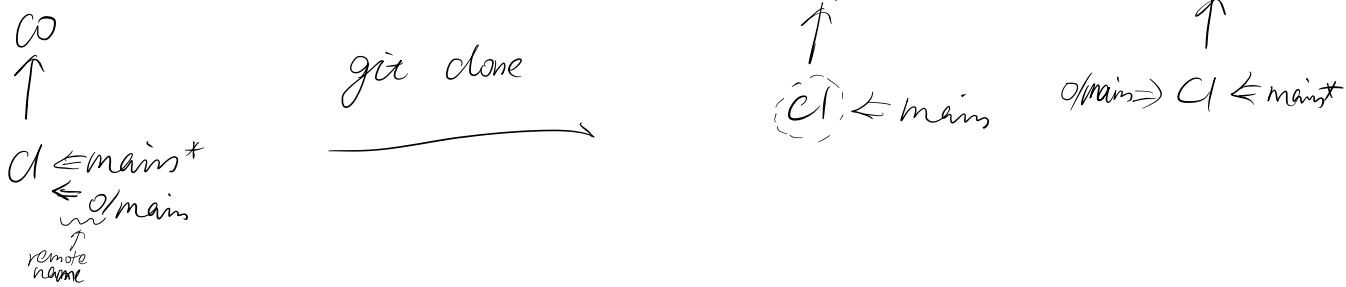
git checkout two

git checkout C5 C4' C3' C2'

git branch -f three C2

Remote

1. git clone

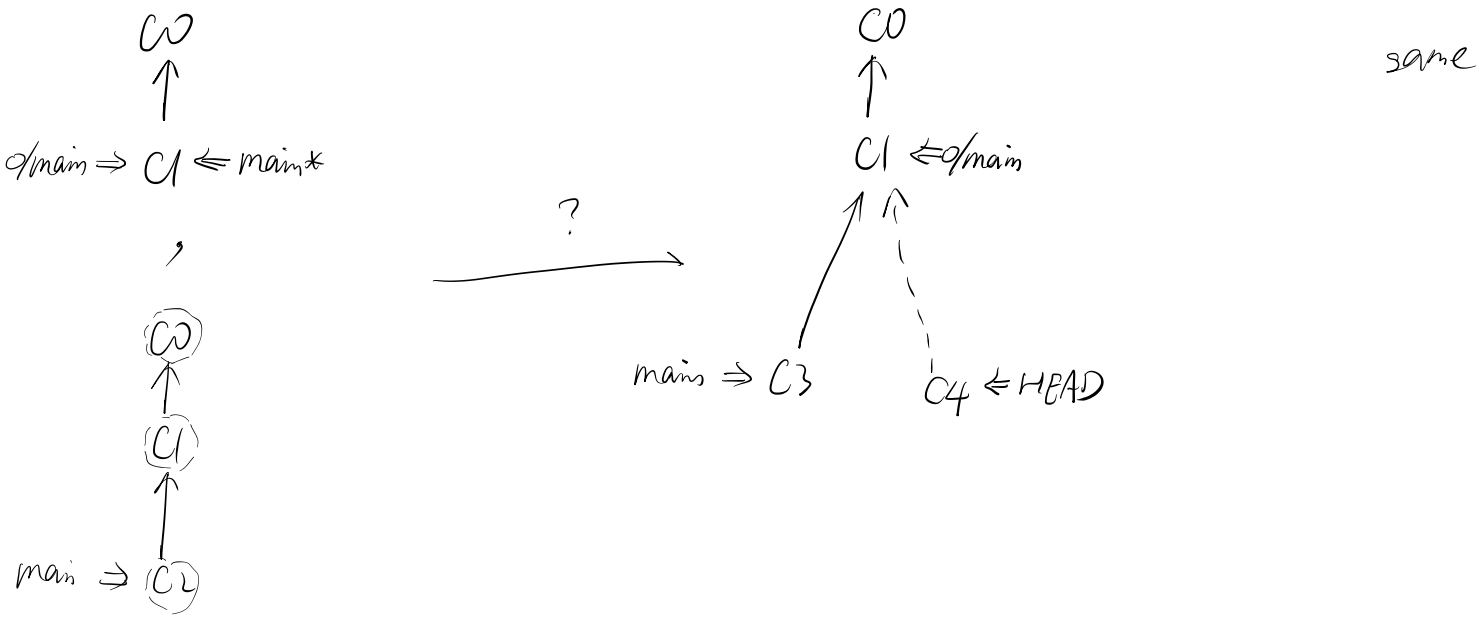


2. Remote branches

have the properties that when you check them out, you are put into detached "HEAD" mode.

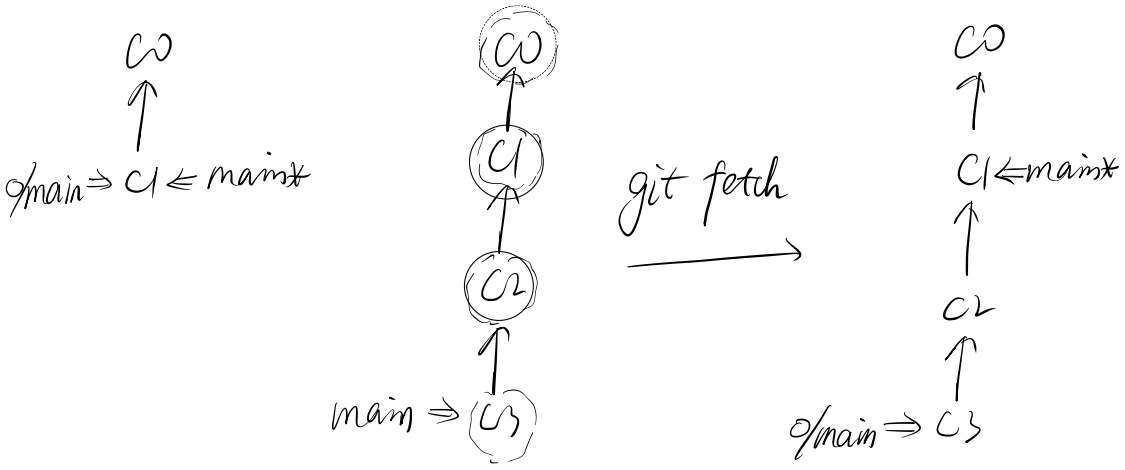
<remote name>/<branch name>

Practice:



Solution: git commit ...
git checkout o/main
git commit

3. Fetch



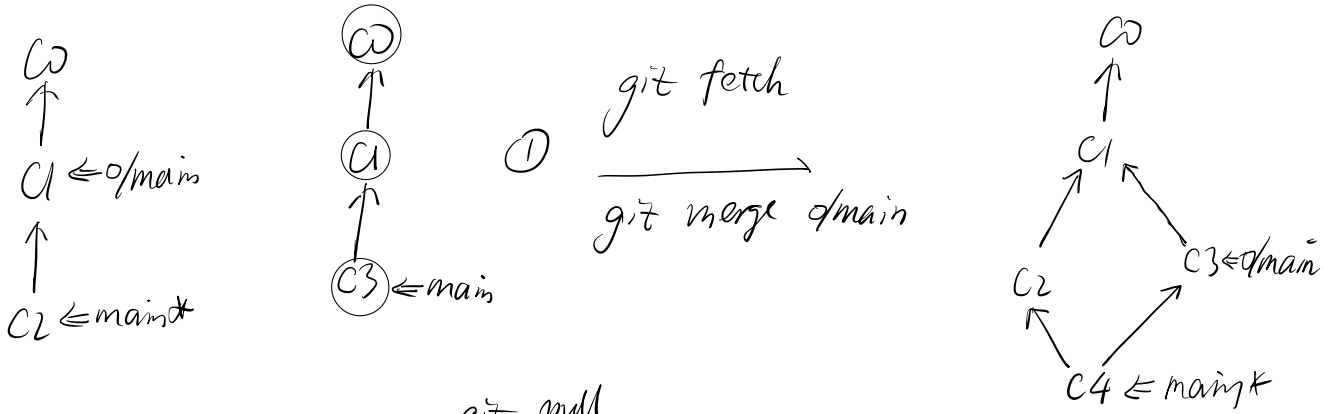
git fetch: 1. downloads commits local doesn't have (sync)
2. updates where our remote branch points (eg, o/main)

Doesn't do: 1. doesn't change anything about local state

4. Pull

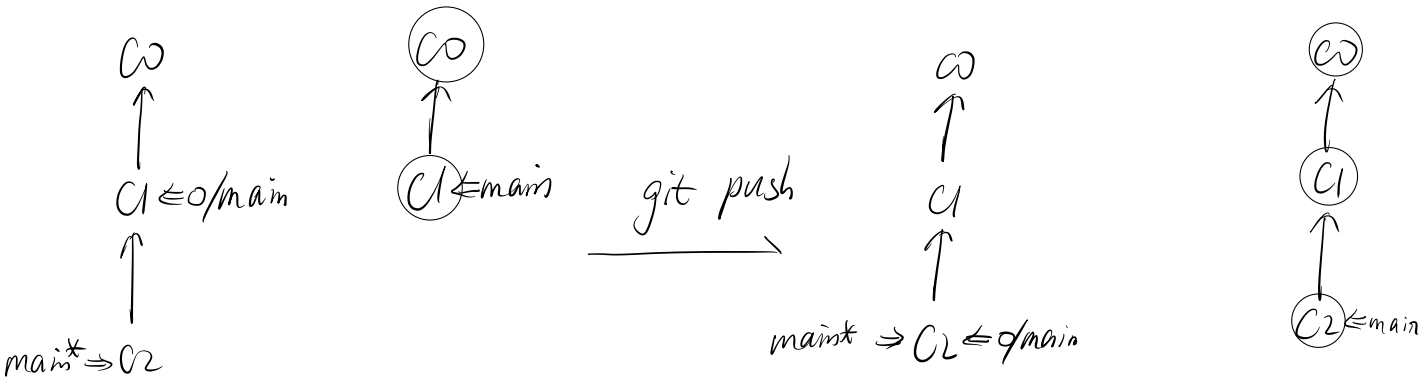
Once you have new commits locally, you can incorporate them as if they were just commits on other branches:

git cherry-pick o/main
git rebase o/main ≈ git pull
git merge o/main



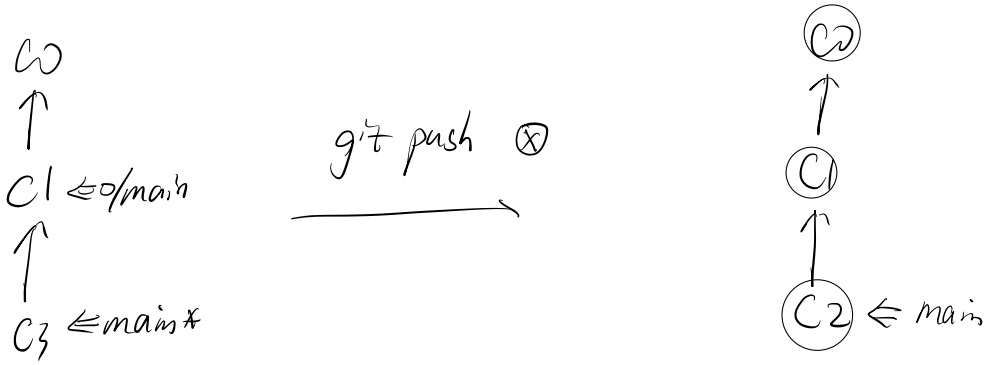
5. Push

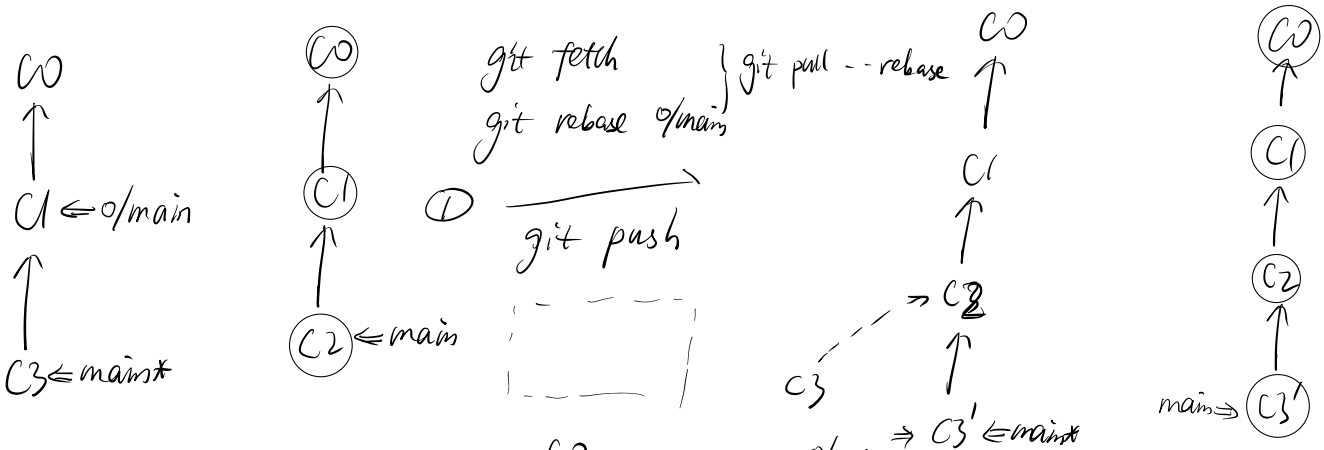
git push with no argument varies depending on 'push.default'

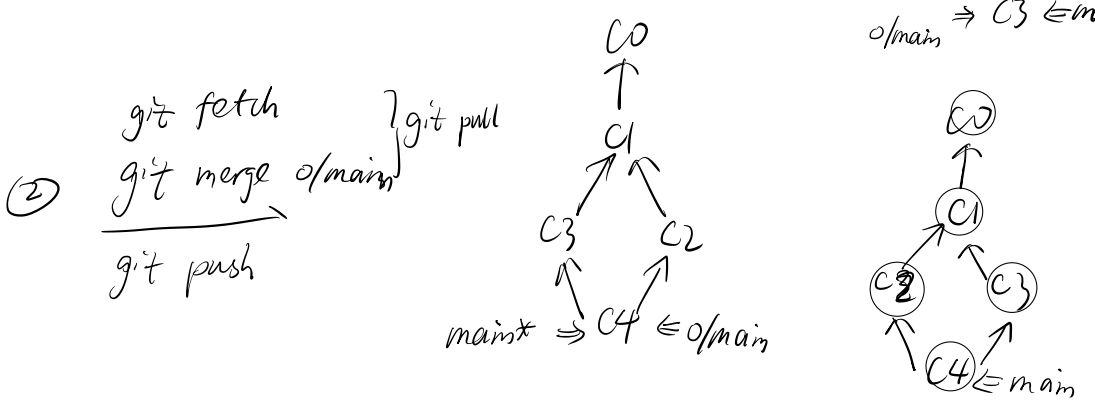


6. Diverged work

Fail when:



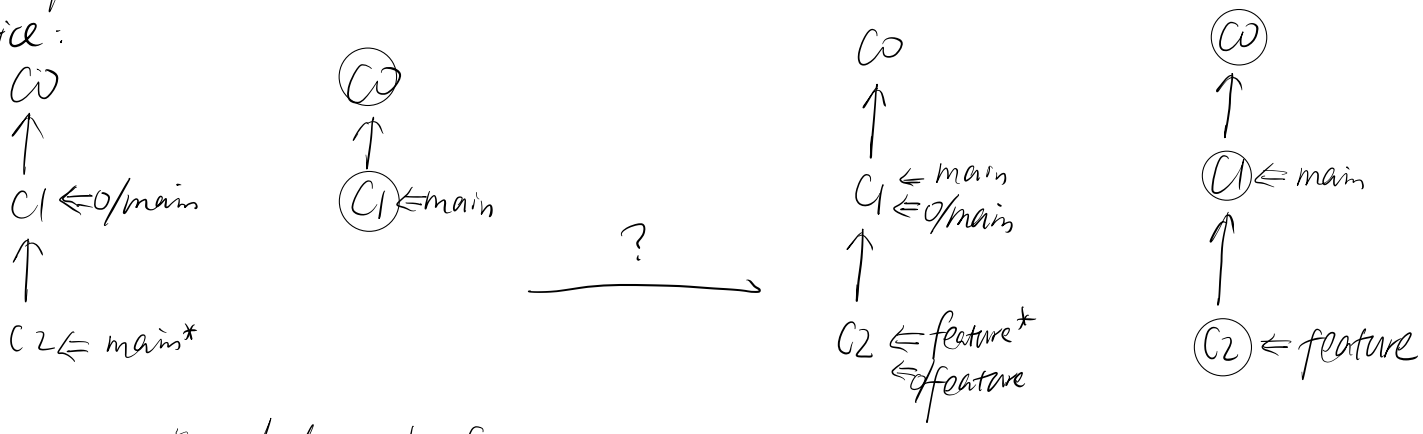




Remote Rejected! Locked Main

The remote rejected the push directly to main because of the policy on main requiring pull request to instead be used.

Practice:



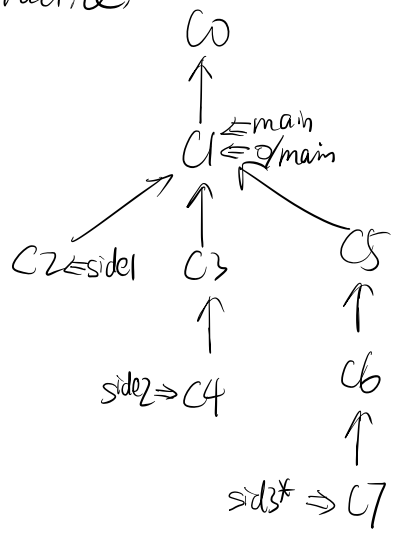
Solution: git checkout -b feature
git push orgin feature
- git reset --hard o/main

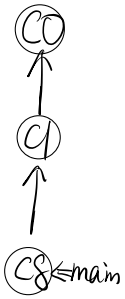
(reset main to point to C1 to sync with remote)

OR git branch -f main C1

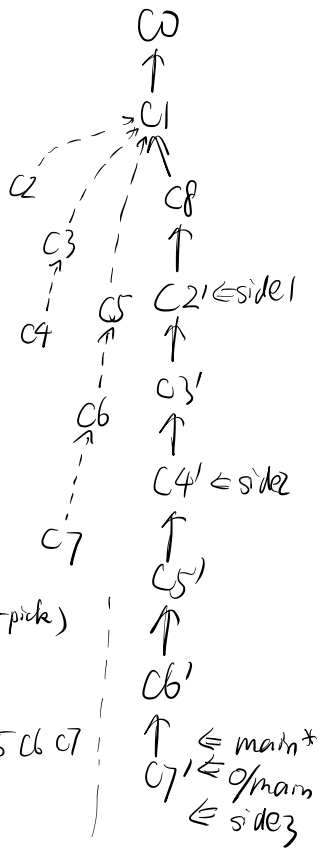
Merging Feature Branches

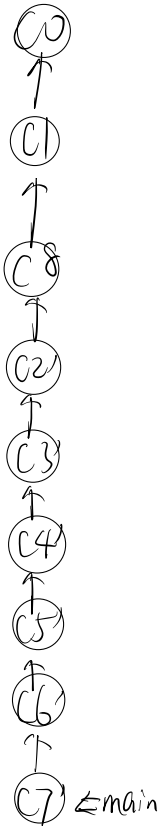
Practice:





?





Solution: 10 commands

1. git branch -f main side1
2. git checkout main
3. git pull --rebase
4. git checkout side2
5. git rebase main
6. git checkout side3
7. git rebase side2
8. git branch -f main side3
9. git checkout main
10. git push

6 commands (cherry-pick)

```
git checkout side1
git cherry-pick C3 C4 C5 C6 C7
git branch -f main side1
git checkout main
git pull --rebase
git push
```

6 commands (rebase)

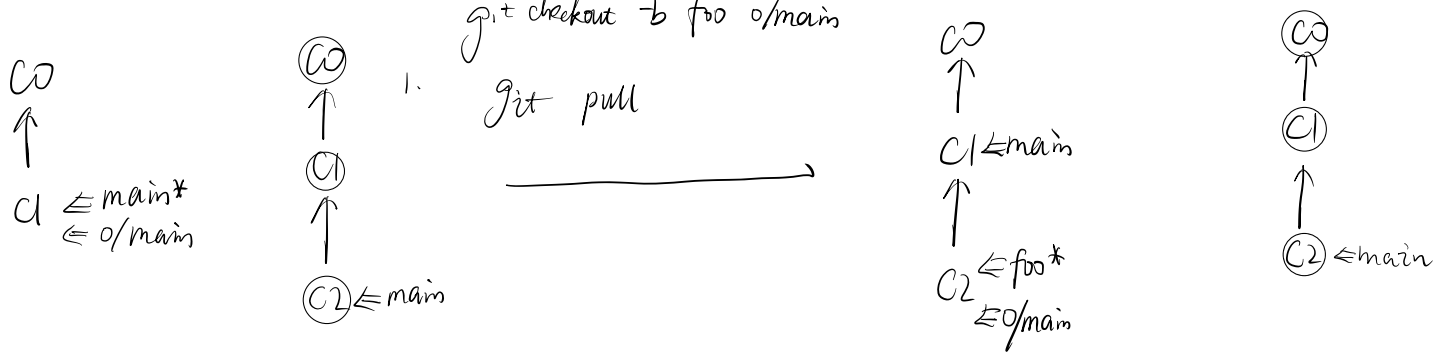
1. git fetch
2. git rebase o/main side1
3. git rebase side1 side2
4. git rebase side2 side3
5. git rebase side3 main
6. git push

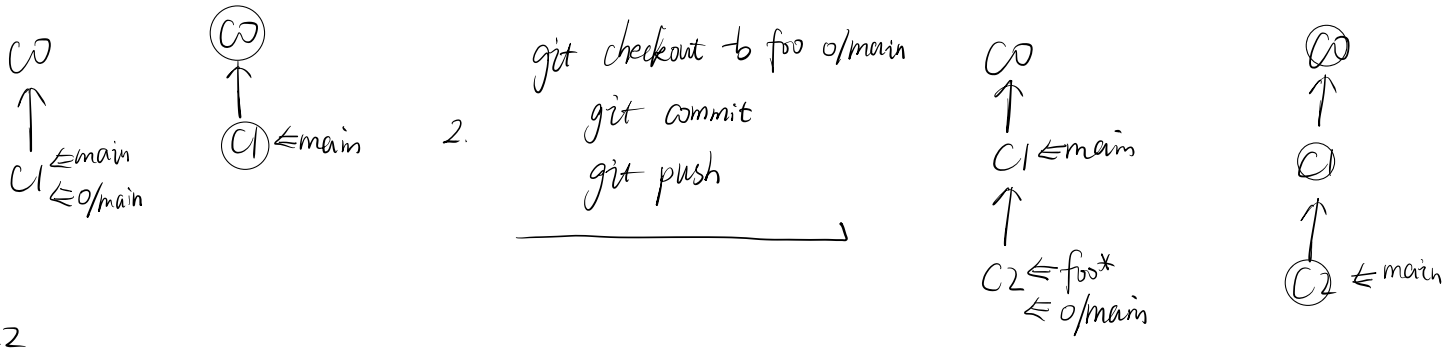
Why not merge?

```
git checkout main
git pull
git merge side1
git merge side2
git merge side3
git push.
```

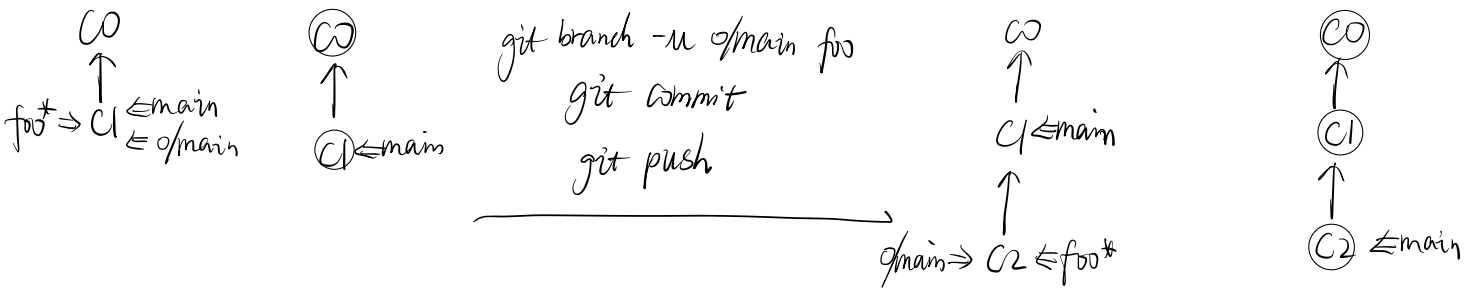
Remote tracking

#1

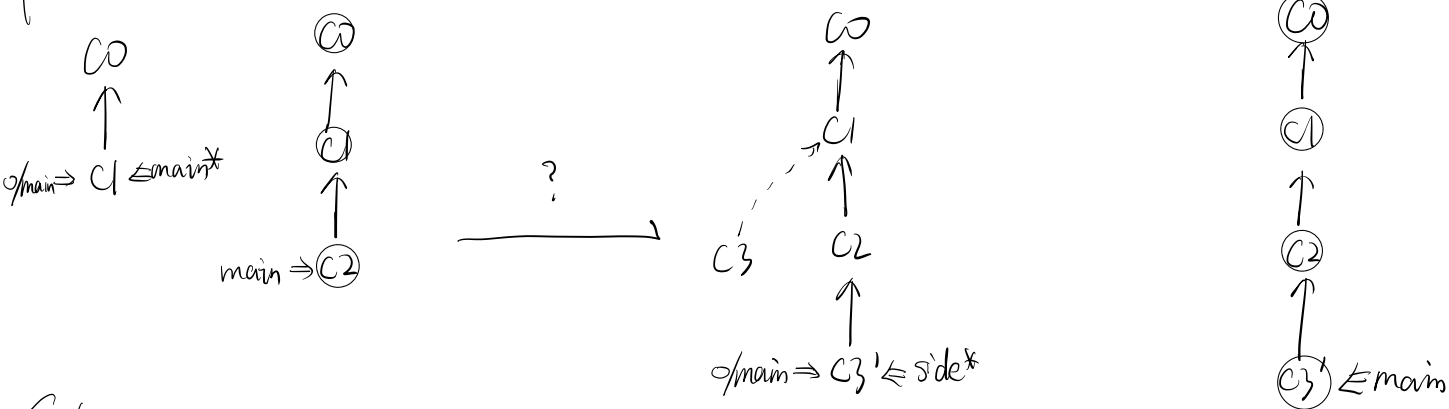




#2



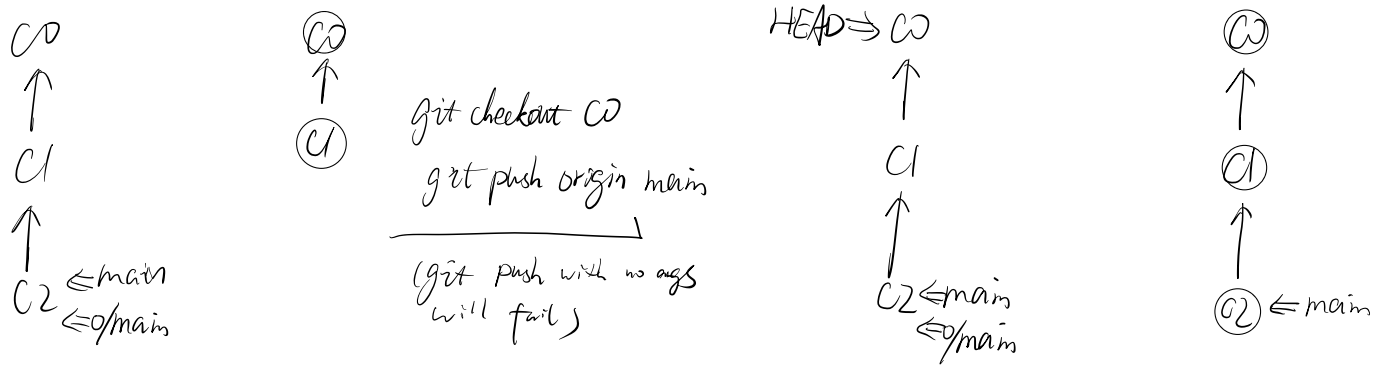
Practice



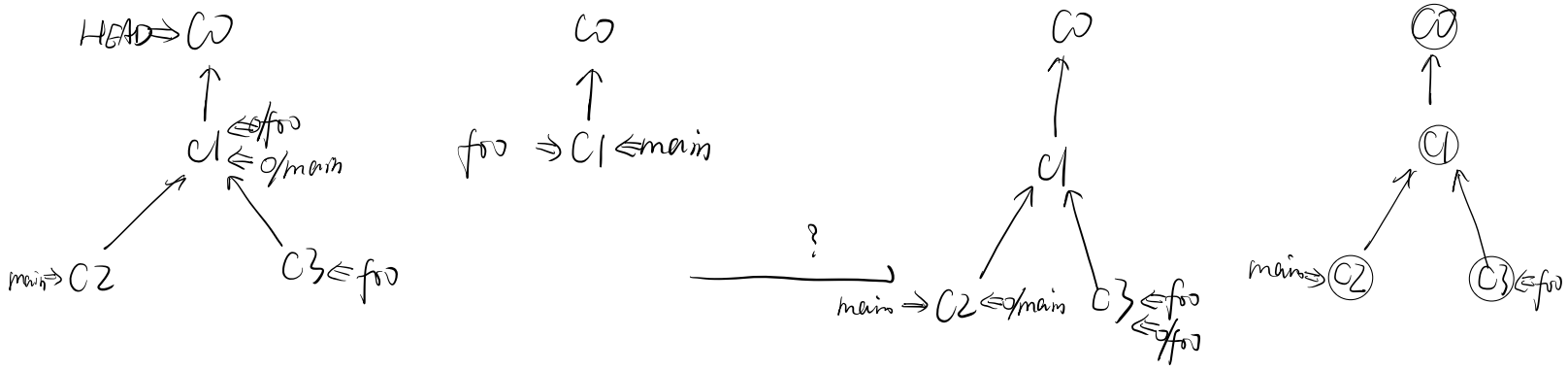
Solution:

```
git checkout -b side o/main
git commit
git pull --rebase
git push
```

Push arguments



Practice:

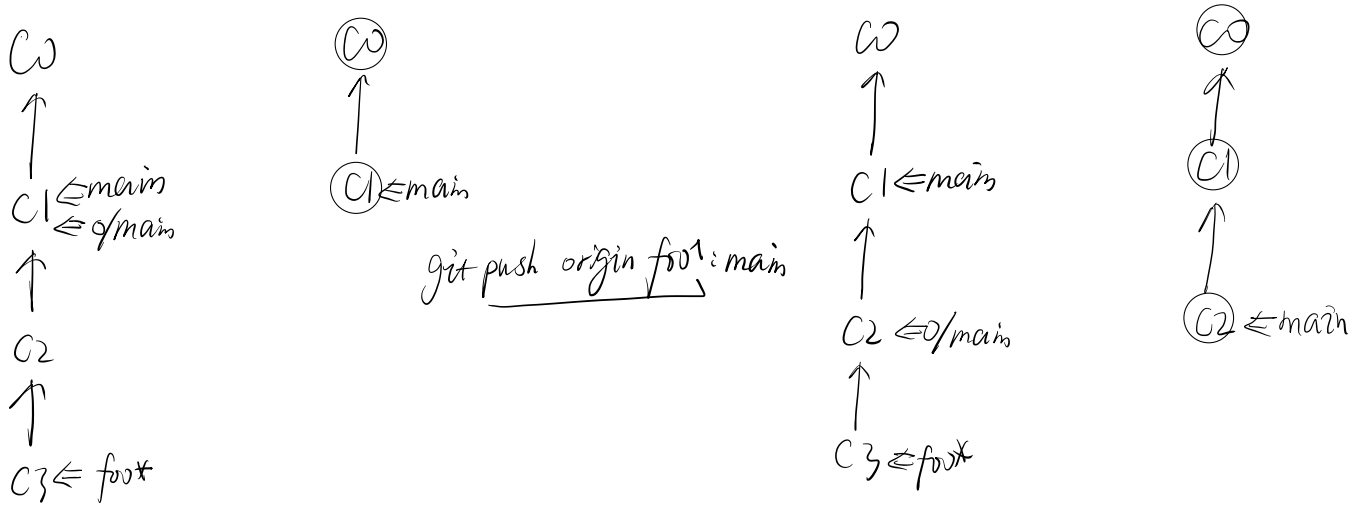


Solution:

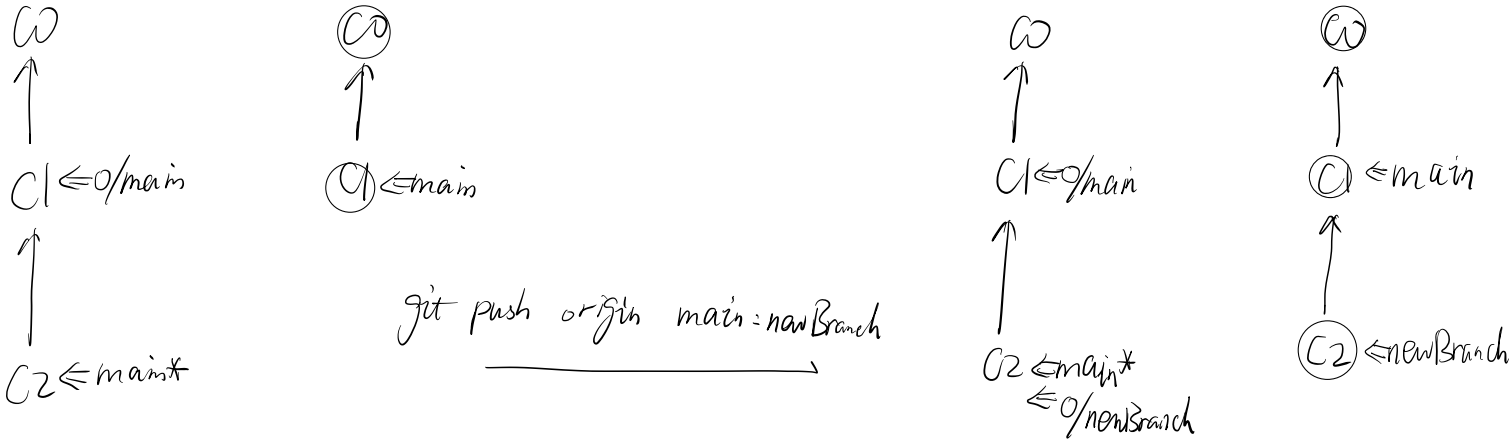
git push origin main

git push origin foo

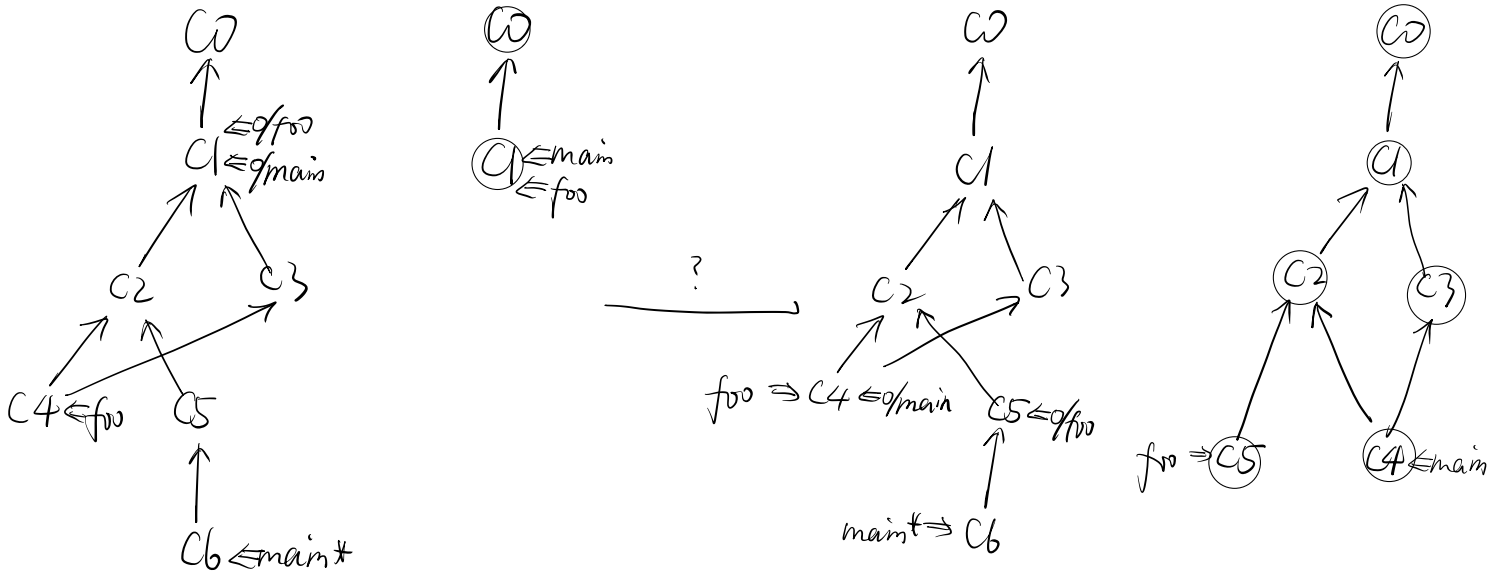
<place> argument



What if the destination you want to update doesn't exist?



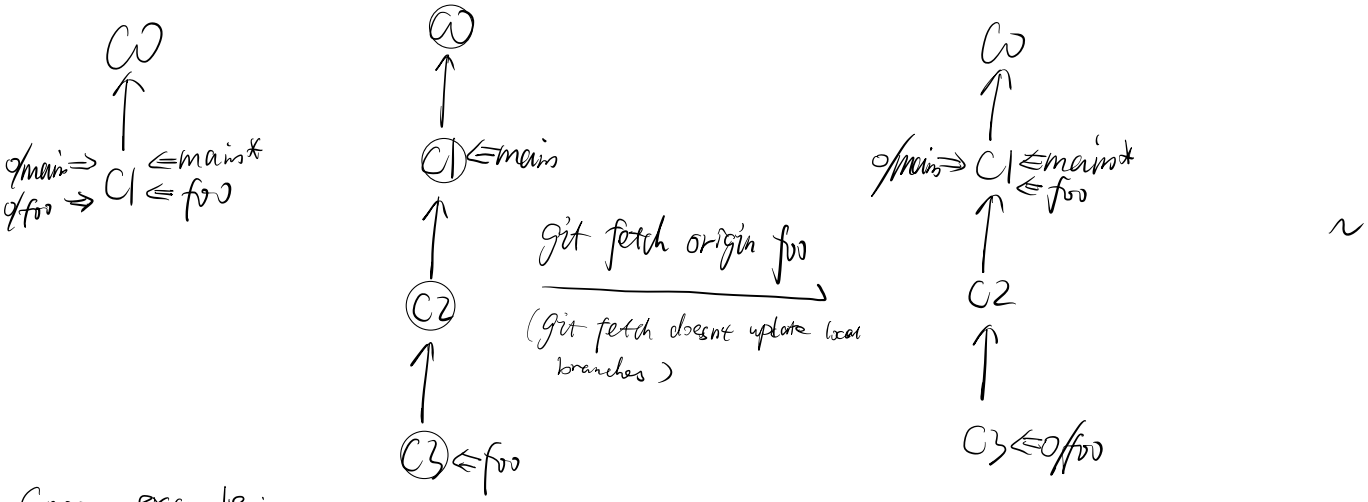
Practice



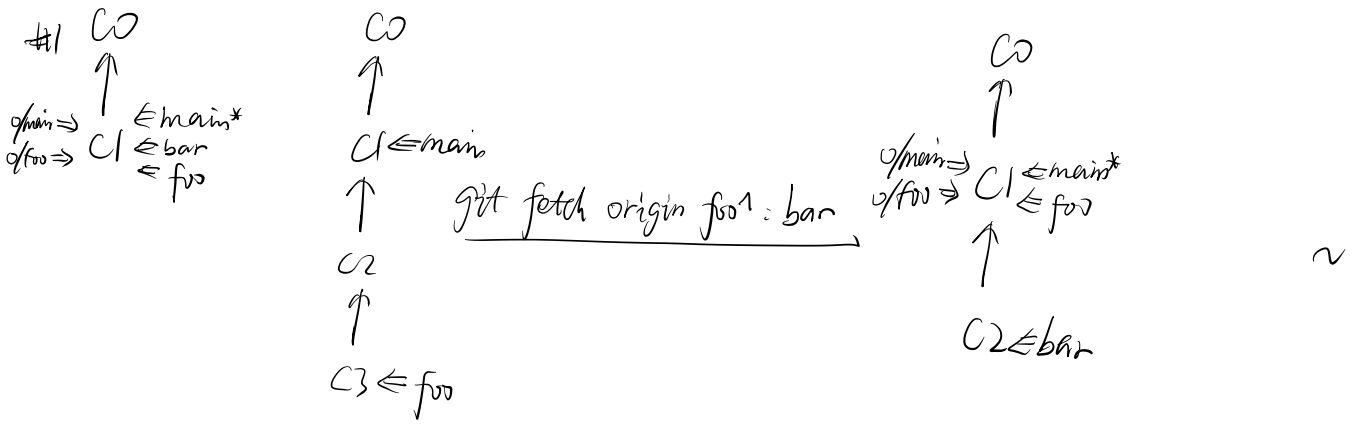
Solution: git push origin foo:main

git push origin main^:foo

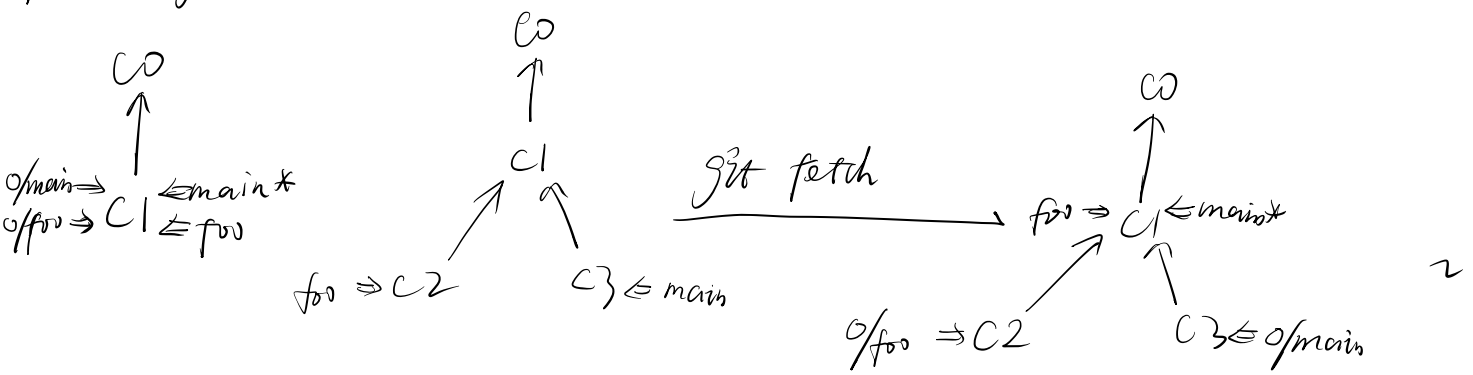
Fetch arguments



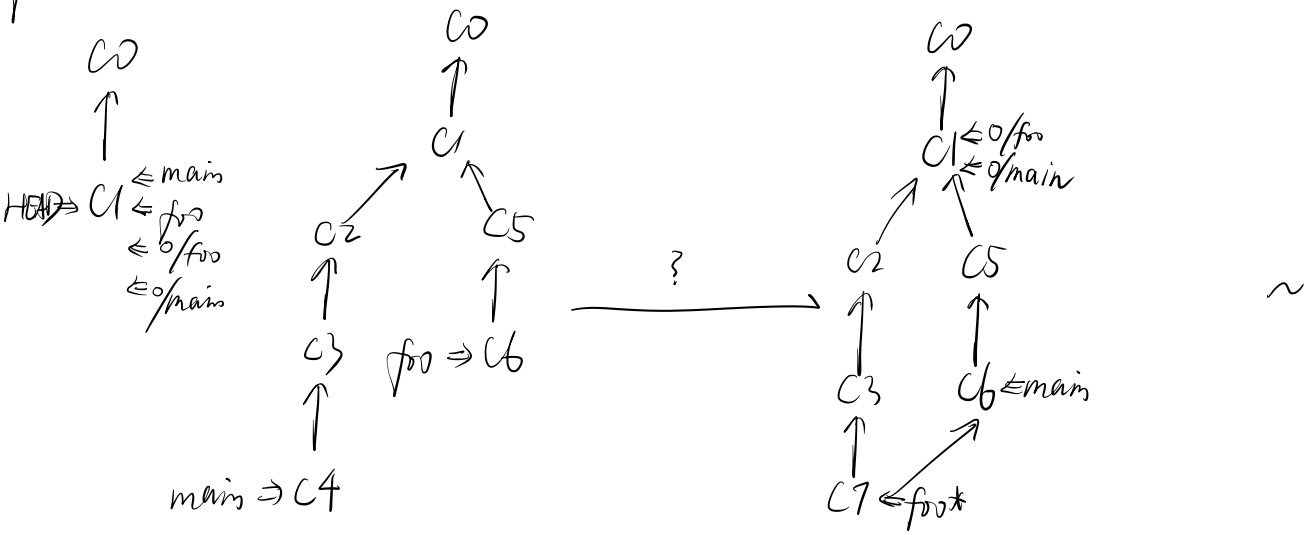
Crazy example:



No args:



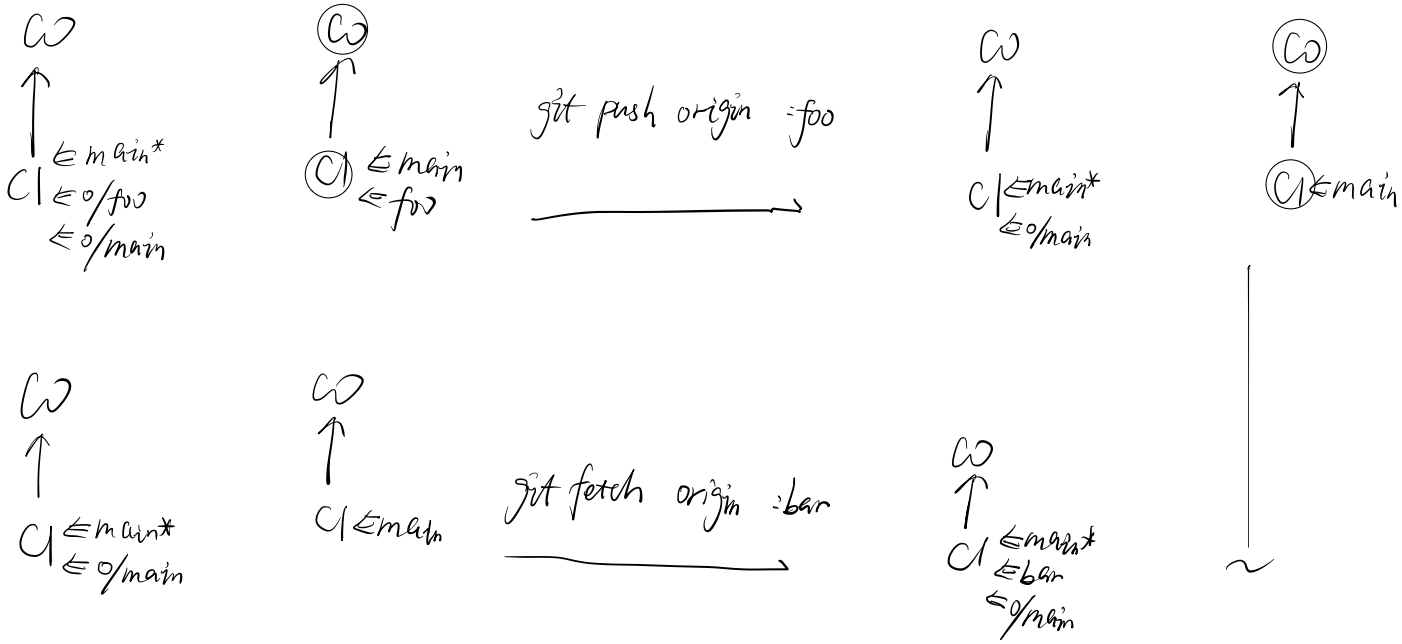
Practice:



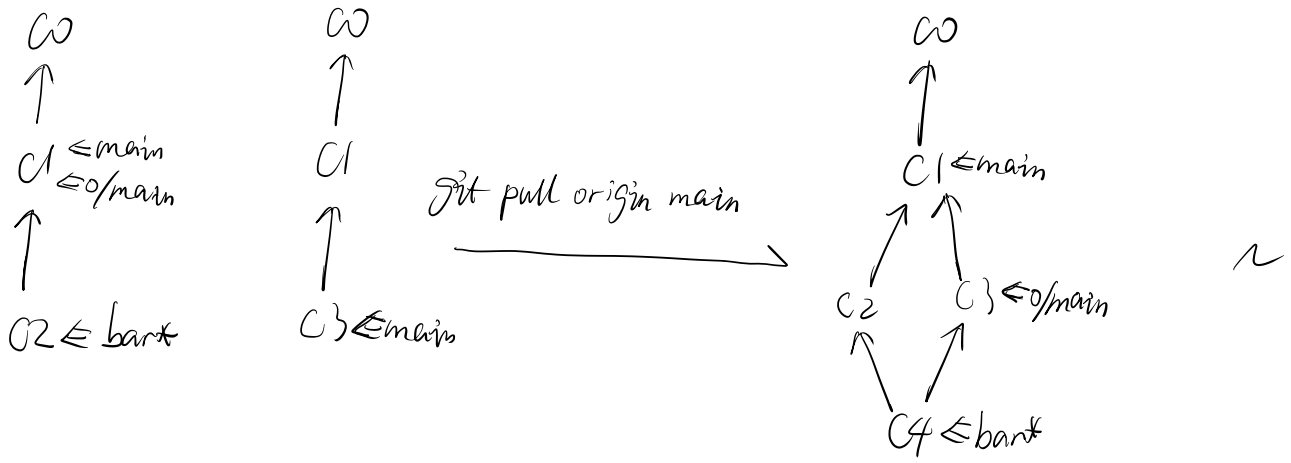
Solution:

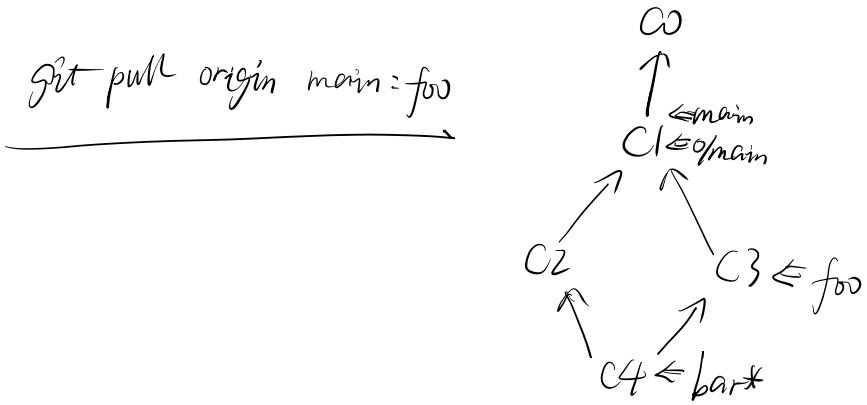
```
git fetch origin foo:main
git fetch origin main~1:foo
git checkout foo
git merge main
```

Source of nothing!

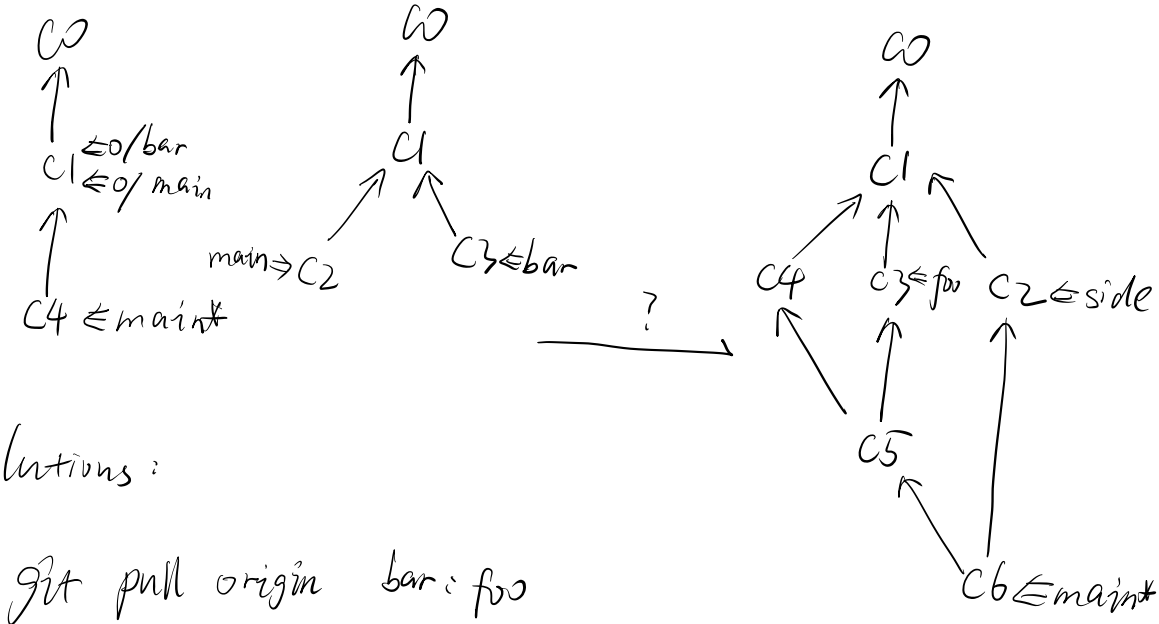


Git pull arguments





Practice



Solutions:

git pull origin bar:foo

git pull origin main:side