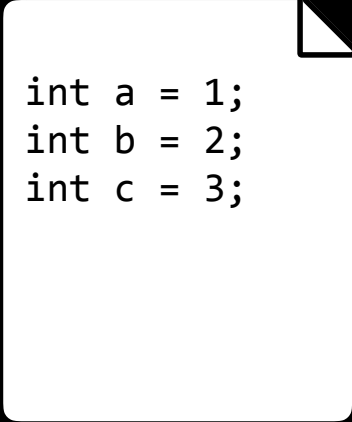


Git and GitHub

CS50 Seminar
Brian Yu

What is Git?

Keep track of changes to code.

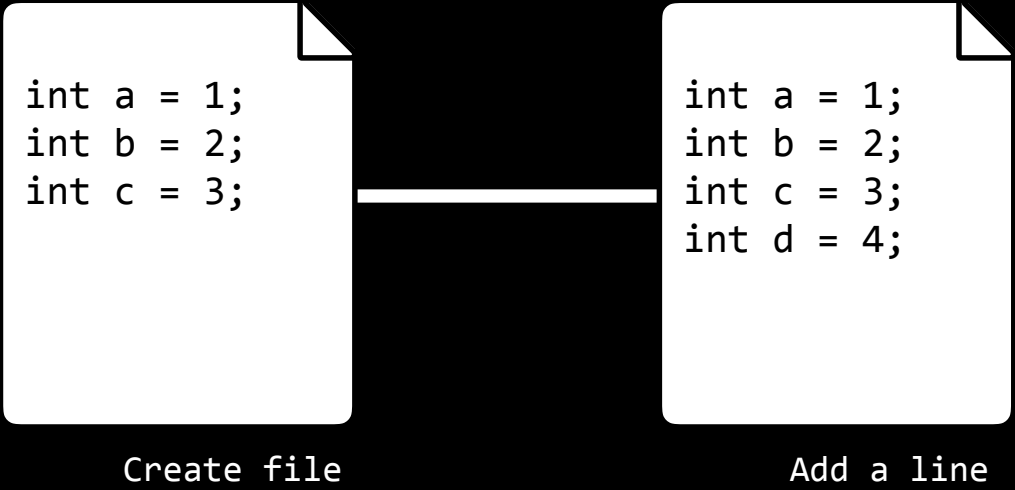


```
int a = 1;  
int b = 2;  
int c = 3;
```

Create file

Keep track of changes to code.

```
int a = 1;  
int b = 2;  
int c = 3;
```



Create file

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

Add a line

Keep track of changes to code.

```
int a = 1;  
int b = 2;  
int c = 3;
```

Create file

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

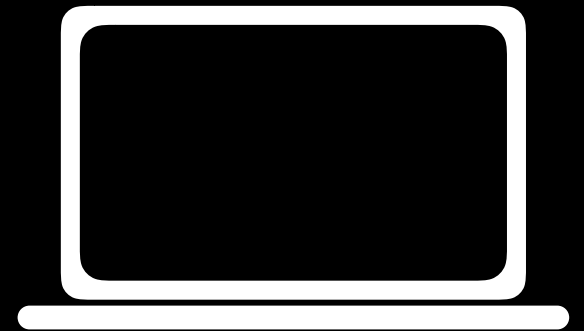
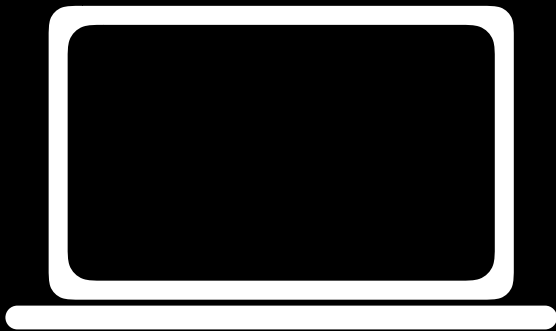
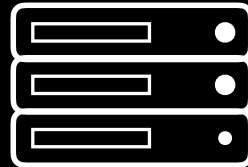
Add a line

```
int a = 1;  
int c = 3;  
int d = 4;
```

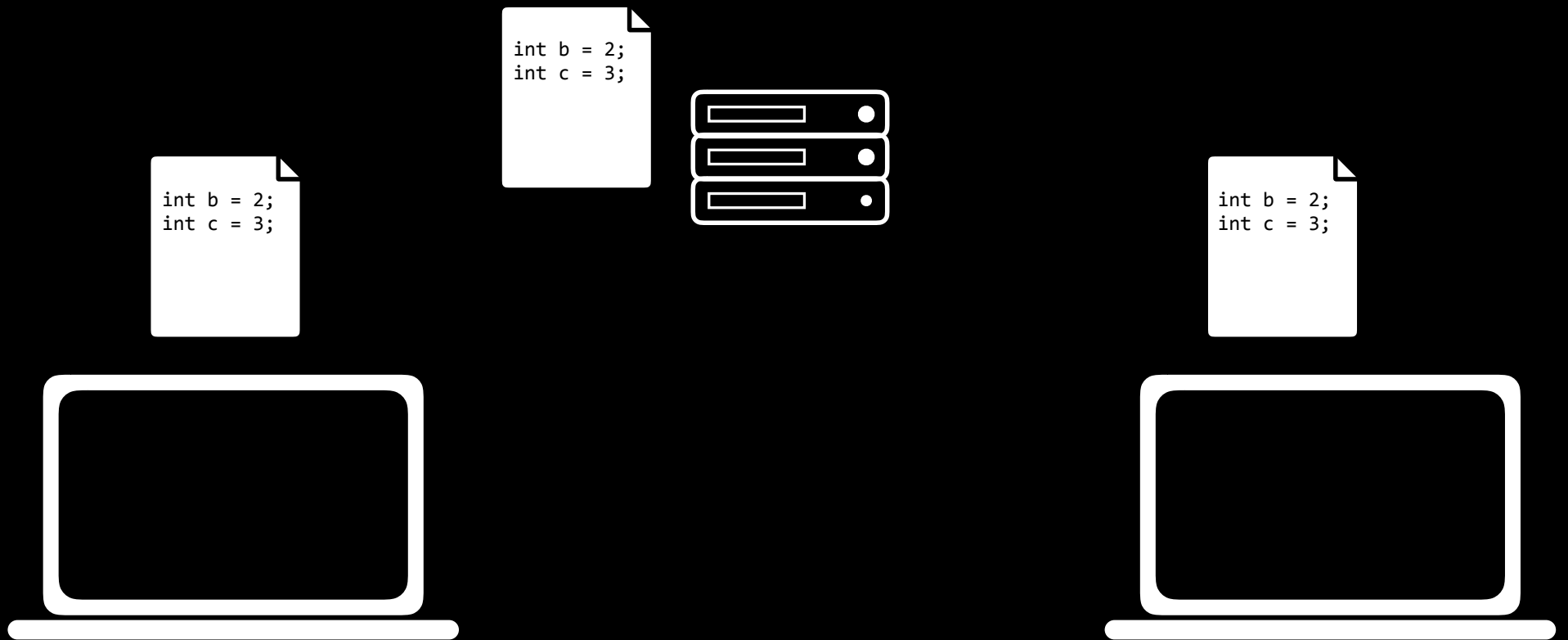
Remove a line

Synchronizes code between different people.

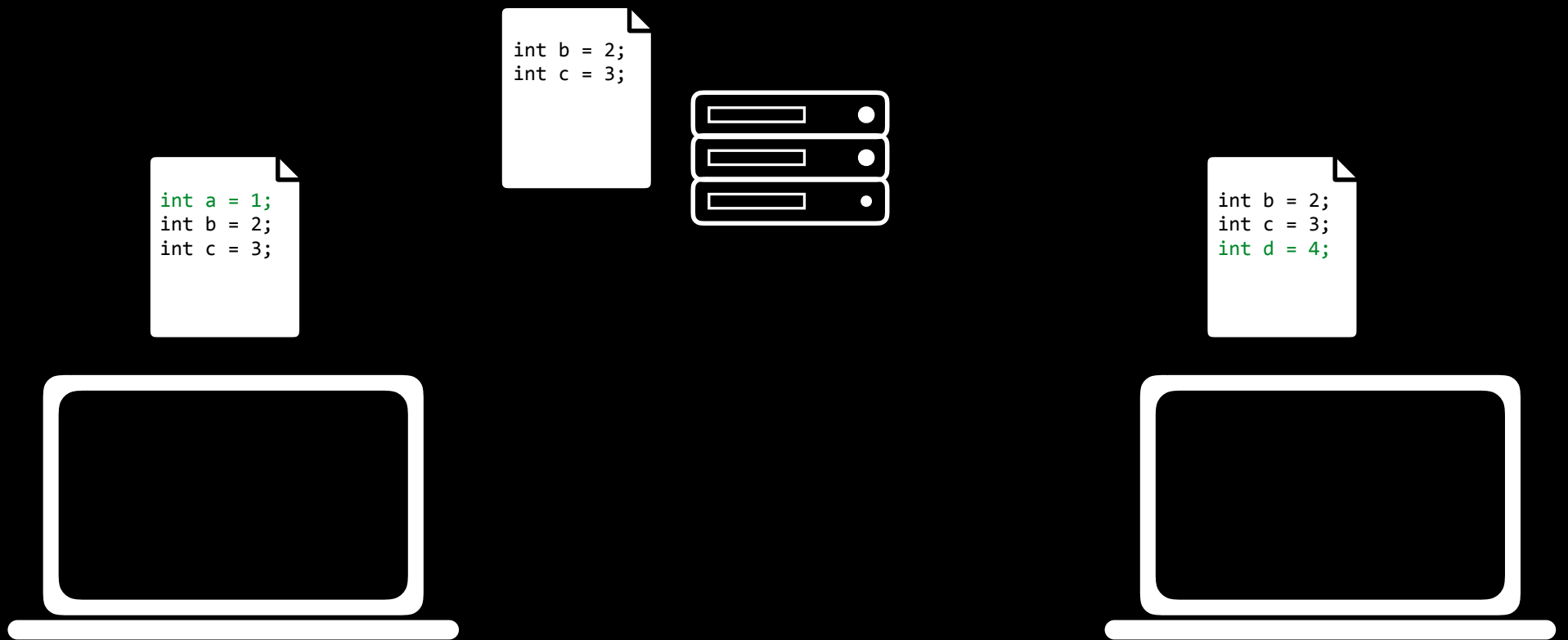
```
int b = 2;  
int c = 3;
```



Synchronizes code between different people.



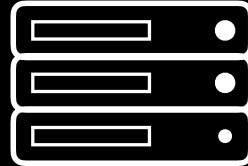
Synchronizes code between different people.



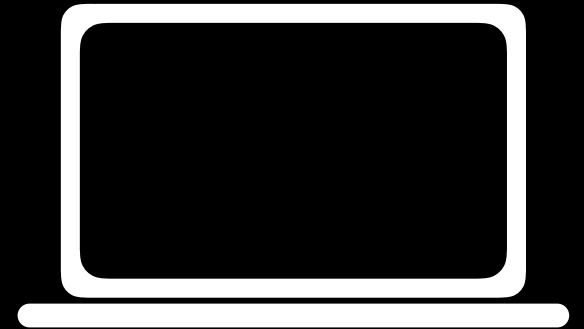
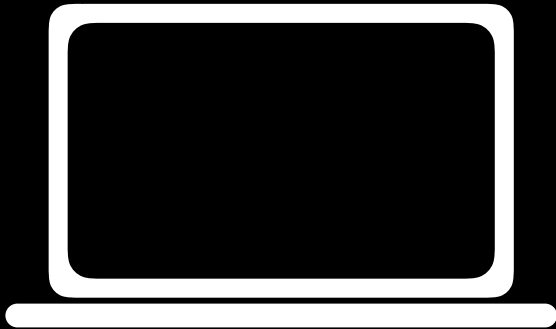
Synchronizes code between different people.

```
int a = 1;  
int b = 2;  
int c = 3;
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



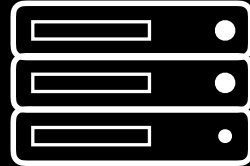
```
int b = 2;  
int c = 3;  
int d = 4;
```



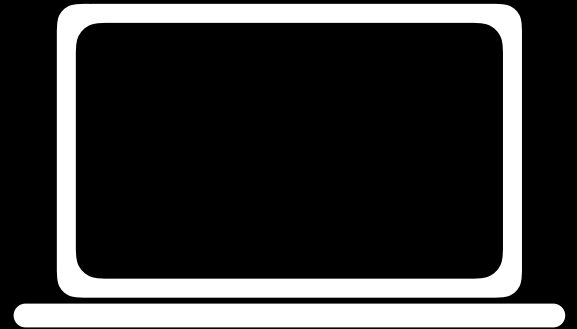
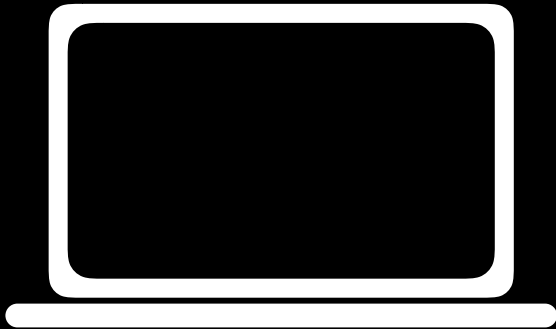
Synchronizes code between different people.

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

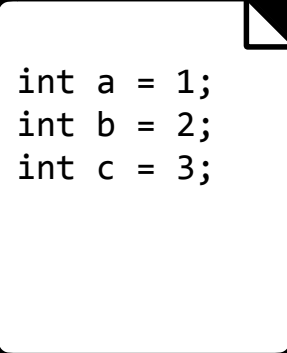
```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



Test changes to code without losing the original.



```
int a = 1;  
int b = 2;  
int c = 3;
```

Test changes to code without losing the original.

```
int a = 1;  
int b = 2;  
int c = 3;
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

Test changes to code without losing the original.

```
int a = 1;  
int b = 2;  
int c = 3;
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

Revert back to old versions of code.

```
int a = 1;  
int b = 2;  
int c = 3;
```

Create file

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

Add a line

```
int a = 1;  
int c = 3;  
int d = 4;
```

Remove a line

Revert back to old versions of code.

```
int a = 1;  
int b = 2;  
int c = 3;
```

Create file

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

Add a line

What is Git?

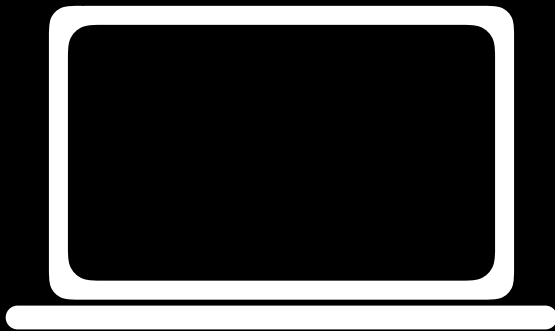
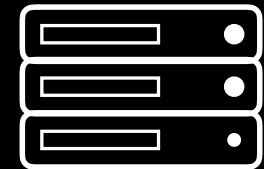
- Keeps track of changes to code.
- Synchronizes code between different people.
- Test changes to code without losing the original.
- Revert back to old versions of code.


```
git clone
```

git clone <url>

- makes a copy of a repository
- stores it on your computer
- a "fork" creates your own copy of someone else's repository

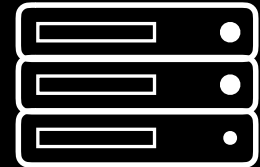
```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



git clone <url>

- makes a copy of a repository
- stores it on your computer
- a "fork" creates your own copy of someone else's repository

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

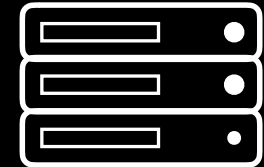


```
git clone <url>
```

git clone <url>

- makes a copy of a repository
- stores it on your computer
- a "fork" creates your own copy of someone else's repository

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
git clone <url>
```

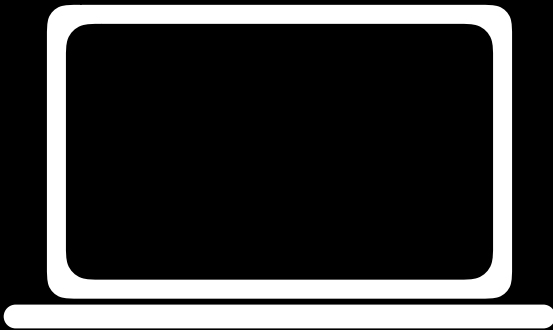
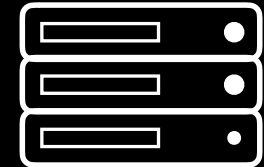
```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

```
git add
```

git add <filename>

- adds a file to "staging area"
- tells git to include the file in the next revision to the repository
- `git add *` adds all changed files

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

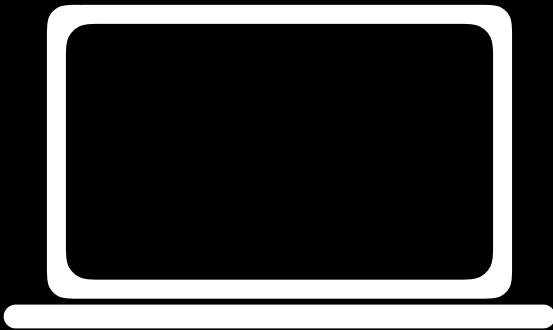
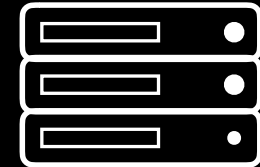


```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

git add <filename>

- adds a file to "staging area"
- tells git to include the file in the next revision to the repository
- `git add *` adds all changed files

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

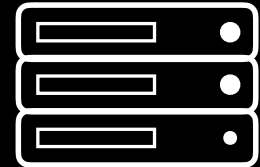


```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

git add <filename>

- adds a file to "staging area"
- tells git to include the file in the next revision to the repository
- `git add *` adds all changed files

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



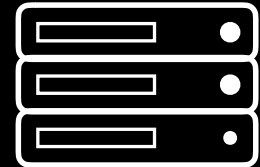
```
git add foo.c
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```


git add <filename>

- adds a file to "staging area"
- tells git to include the file in the next revision to the repository
- `git add *` adds all changed files

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
git add foo.c
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

Changes to be committed:

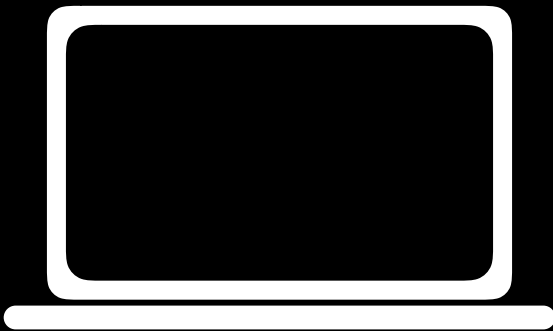
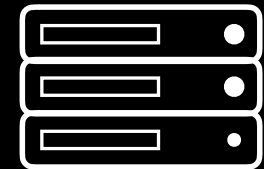
```
modified: foo.c
```

```
git commit
```

git commit -m "message"

- saves the changes to repository as a new revision (a "commit")
- records a message
- `git commit -am "message"` adds and commits in same step

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

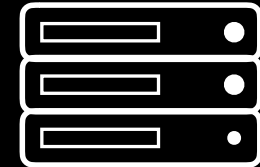


```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

git commit -m "message"

- saves the changes to repository as a new revision (a "commit")
- records a message
- `git commit -am "message"` adds and commits in same step

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
git commit -m  
"Add line"
```

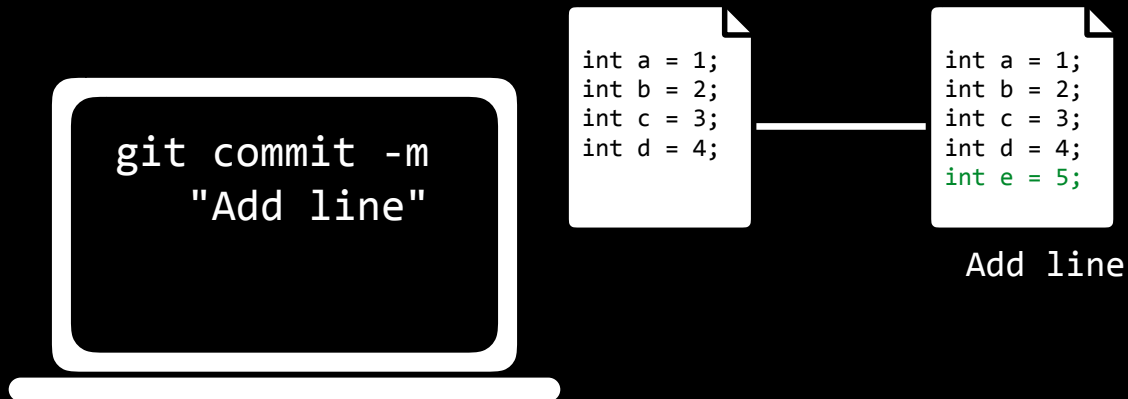
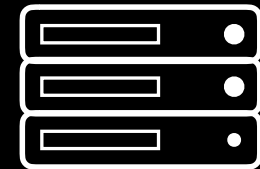
```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```



git commit -m "message"

- saves the changes to repository as a new revision (a "commit")
- records a message
- `git commit -am "message"` adds and commits in same step

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```

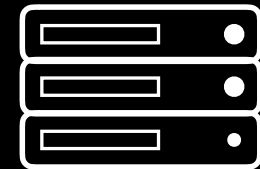


```
git status
```

git status

- shows current status of repository

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

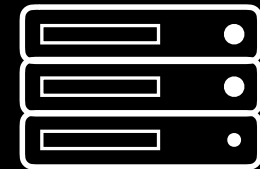
Add line



git status

- shows current status of repository

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```



```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;
```

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

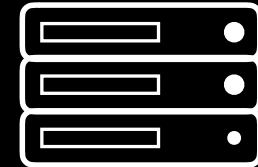
Add line



git status

- shows current status of repository

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```



```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e = 5;
```

Add line



git status

On branch master

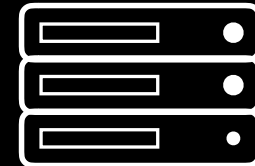
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

```
git push
```

git push

- sends committed changes to remote repository
- more explicitly, could write `git push origin master`

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```



```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```

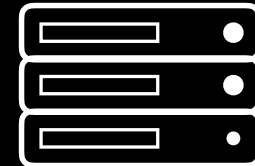
```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e = 5;
```

Add line

git push

- sends committed changes to remote repository
- more explicitly, could write `git push origin master`

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```



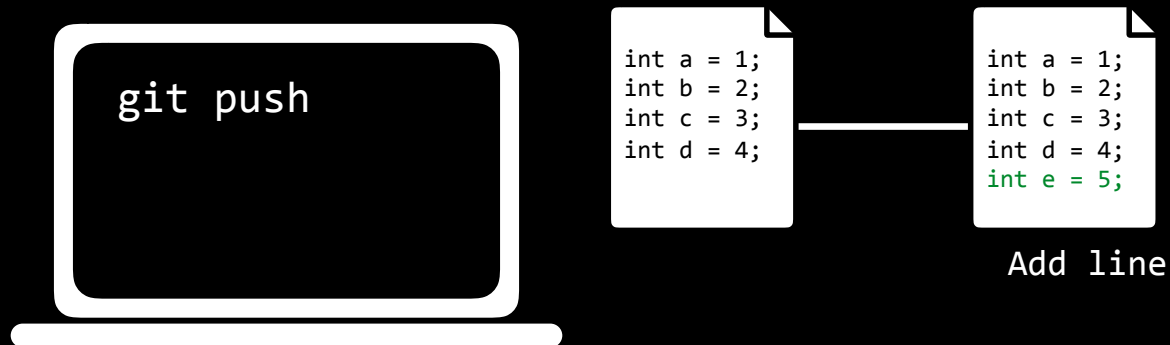
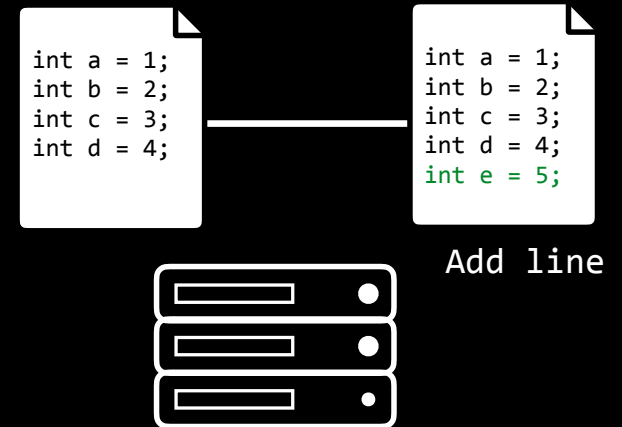
```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e = 5;
```

Add line

git push

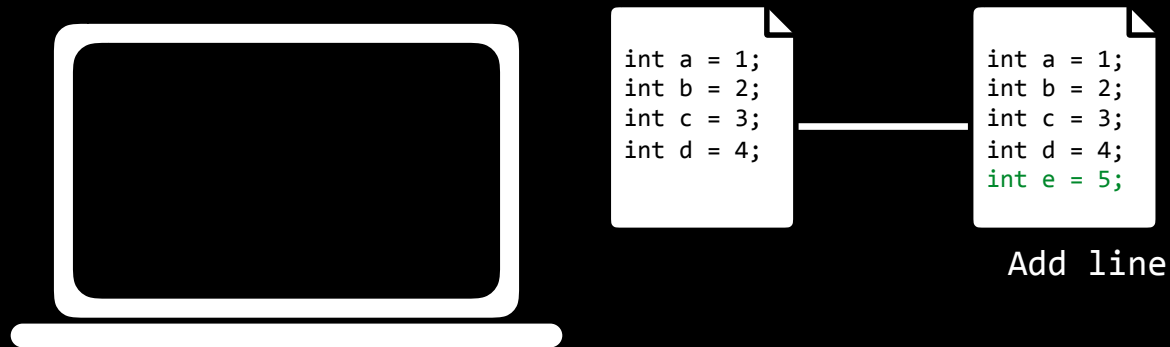
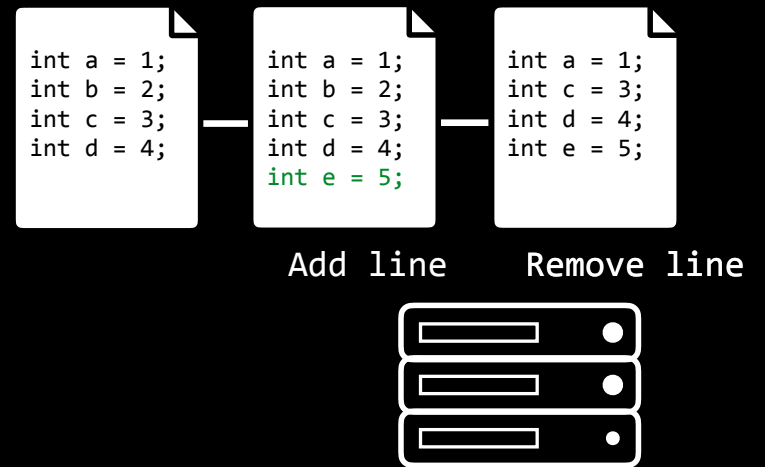
- sends committed changes to remote repository
- more explicitly, could write `git push origin master`



```
git pull
```

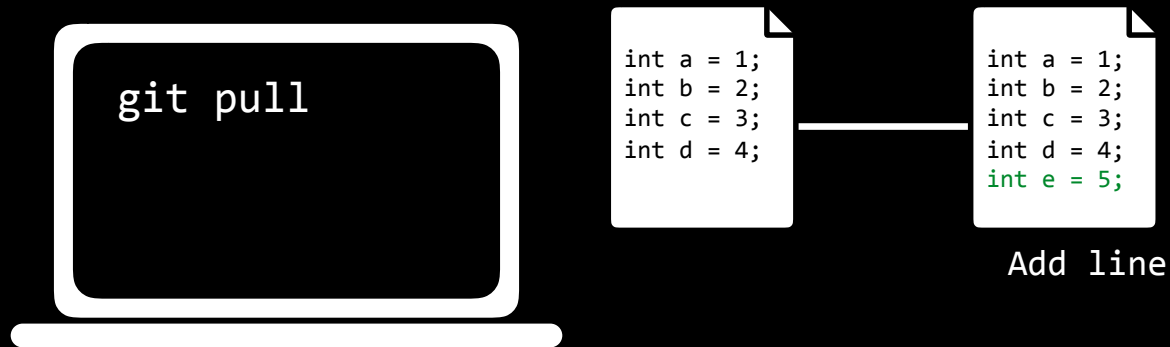
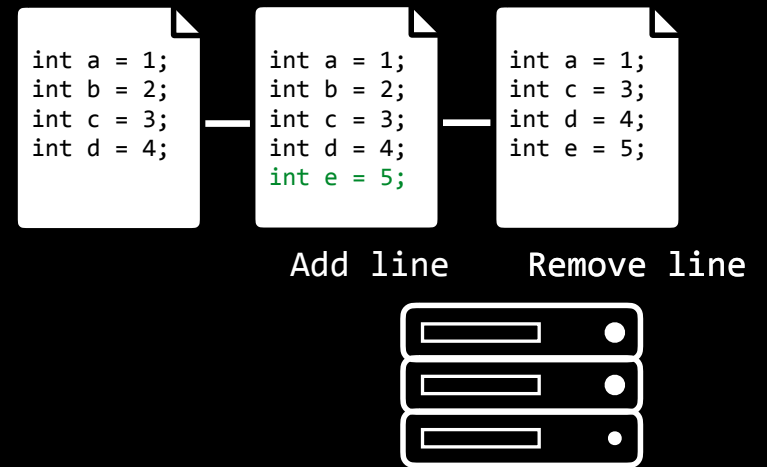
git pull

- retrieves changes from remote repository



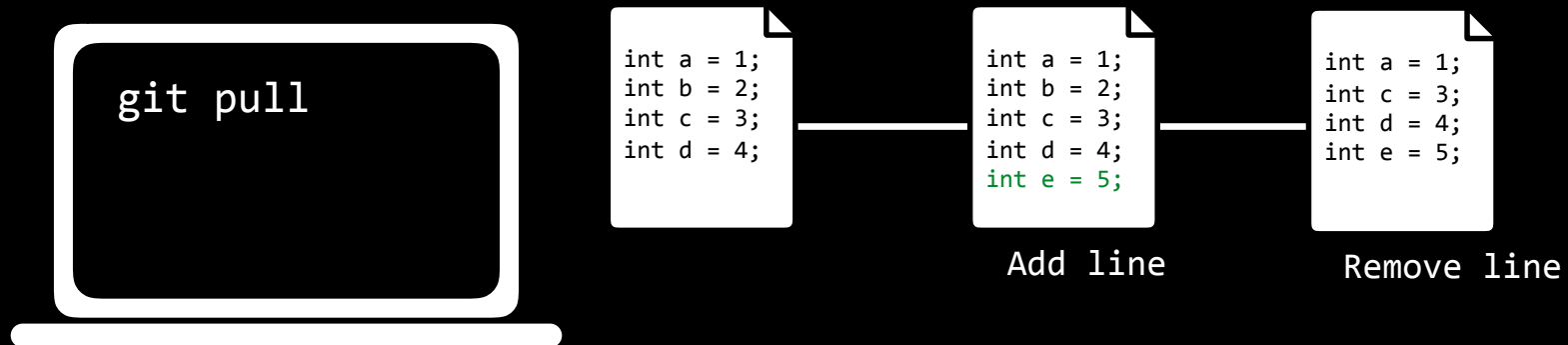
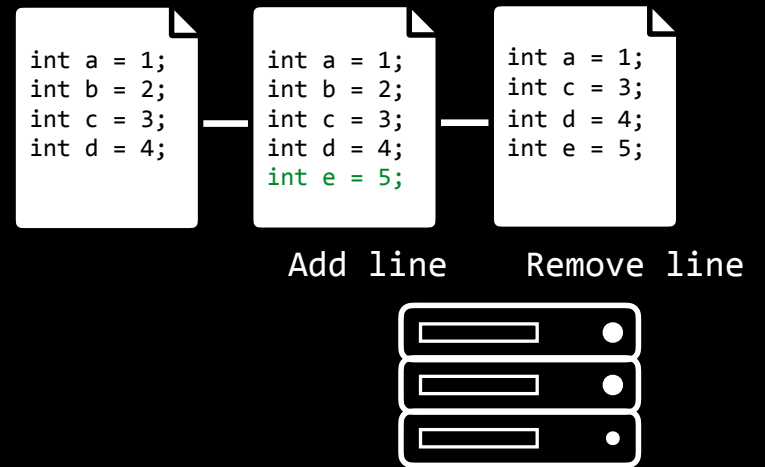
git pull

- retrieves changes from remote repository



git pull

- retrieves changes from remote repository



Merge Conflicts


Merge Conflicts

- when two different commits can't be automatically merged
- need to be resolved



Merge Conflicts


- when two different commits can't be automatically merged
- need to be resolved



```
git pull
```

Merge Conflicts

- when two different commits can't be automatically merged
- need to be resolved


A white-outlined icon of a terminal window with a rounded top and a horizontal bar at the bottom, representing a command prompt.

```
git pull
```

```
CONFLICT (content): Merge conflict in foo.c  
Automatic merge failed; fix conflicts and then  
commit the result.
```

Merge Conflicts

- when two different commits can't be automatically merged
- need to be resolved



```
git pull
```

```
int a = 1;
<<<<<<< HEAD
int b = 2;
=====
int b = 0;
>>>>>> 5468697320697320435335302e
int c = 3;
int d = 4;
int e = 5;
```

Merge Conflicts

- when two different commits can't be automatically merged
- need to be resolved



your
changes


remote
changes

```
int a = 1;
<<<<<<< HEAD
{ int b = 2;
  =====
  { int b = 0;
    >>>>>>> 5468697320697320435335302e
    int c = 3;
    int d = 4;
    int e = 5;
```

conflicting commit
↓

Merge Conflicts

- when two different commits can't be automatically merged
- need to be resolved




```
git pull
```

```
int a = 1;
<<<<<<< HEAD
int b = 2;
=====
int b = 0;
>>>>>> 5468697320697320435335302e
int c = 3;
int d = 4;
int e = 5;
```


Merge Conflicts

- when two different commits can't be automatically merged
- need to be resolved



```
git pull
```

```
int a = 1;
```

```
int b = 2;
```


```
int c = 3;
```

```
int d = 4;
```

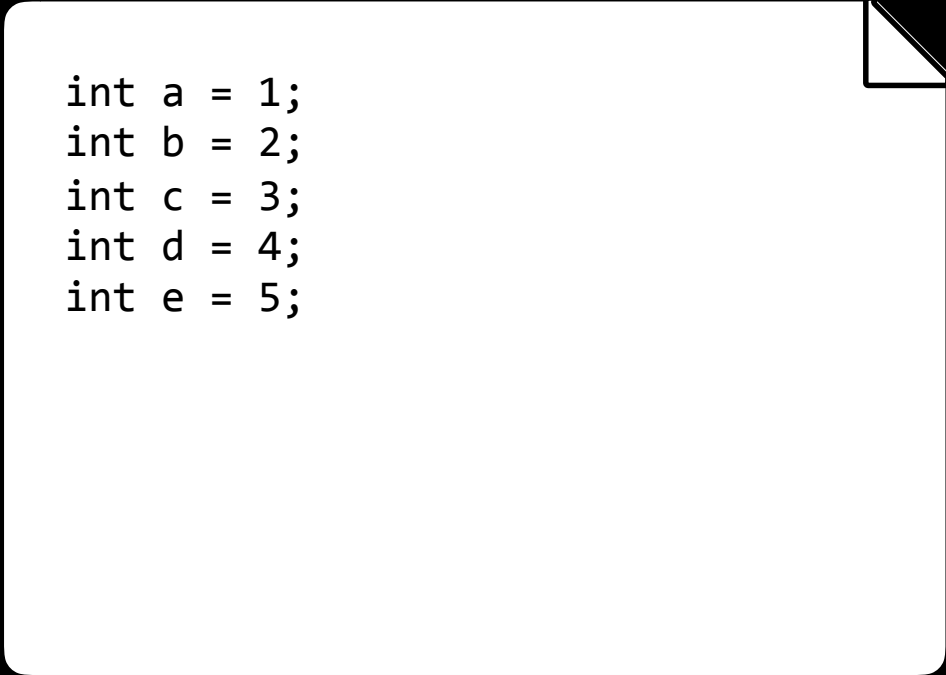
```
int e = 5;
```

Merge Conflicts

- when two different commits can't be automatically merged
- need to be resolved

A white-outlined icon of a terminal window or laptop screen. Inside the screen, the text "git pull" is written in a monospaced font.

```
git pull
```

A white-outlined icon of a code block or document with a folded top-right corner. Inside, five lines of C code are written in a monospaced font.

```
int a = 1;  
int b = 2;  
int c = 3;  
int d = 4;  
int e = 5;
```

```
git log
```

git log

- shows a history of commits and messages



git log

- shows a history of commits and messages



git log

- shows a history of commits and messages

```
commit 5468697320697320435335302e
Author: Brian Yu <brianyu@college.harvard.edu>
Date:   Tue Oct 11 21:09:37 2016 -0400
```

Remove a line

```
commit 4920746f6f6b20435335302e
Author: Brian Yu <brianyu@college.harvard.edu>
Date:   Tue Oct 11 21:05:28 2016 -0400
```

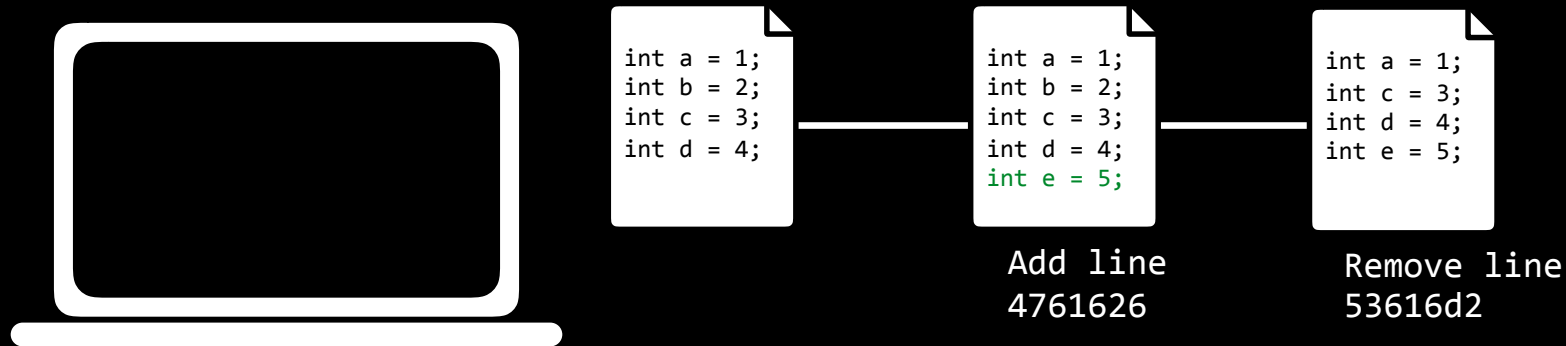
Add a line



```
git reset
```

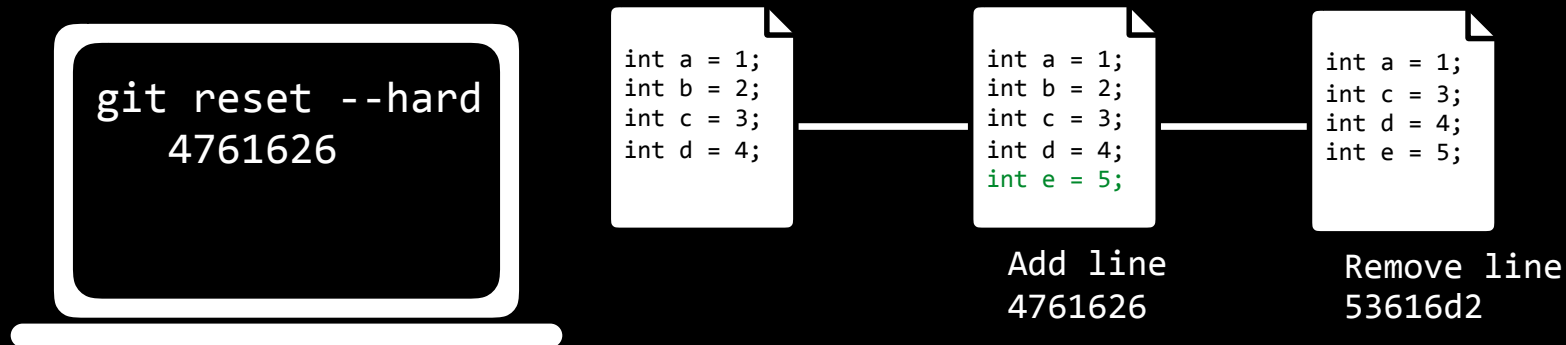
git reset

- `git reset --hard <commit>`
reverts code back to a previous commit
- `git reset --hard origin/master`
reverts code back to remote repository version



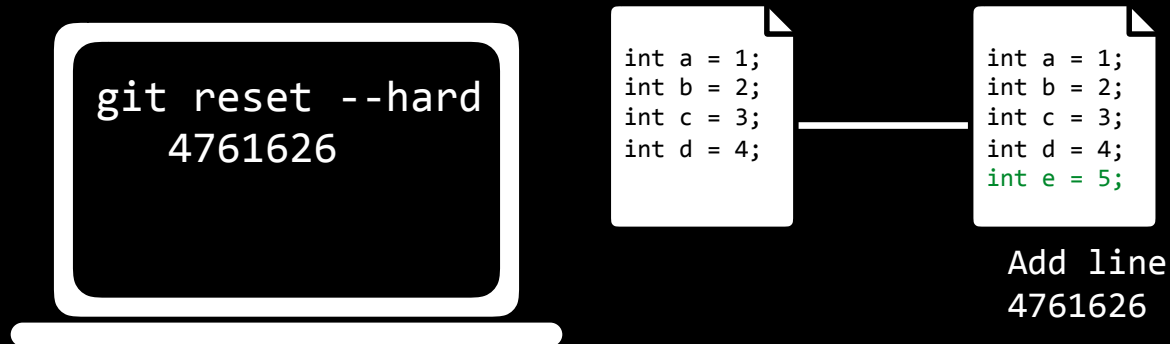
git reset

- `git reset --hard <commit>`
reverts code back to a previous commit
- `git reset --hard origin/master`
reverts code back to remote repository version



git reset

- `git reset --hard <commit>`
reverts code back to a previous commit
- `git reset --hard origin/master`
reverts code back to remote repository version



Branching

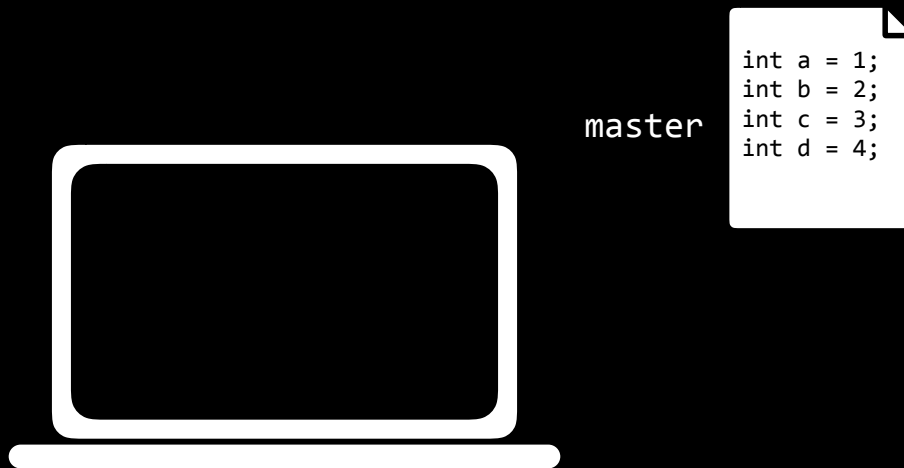
Branching

- Branch is a version of the repository.
- Each branch has its own commit history and current version.

git branch

git branch

- shows all branches of code
- create a branch with `git branch <branch_name>`
- switch to ("checkout") a new branch with `git checkout <branch_name>`



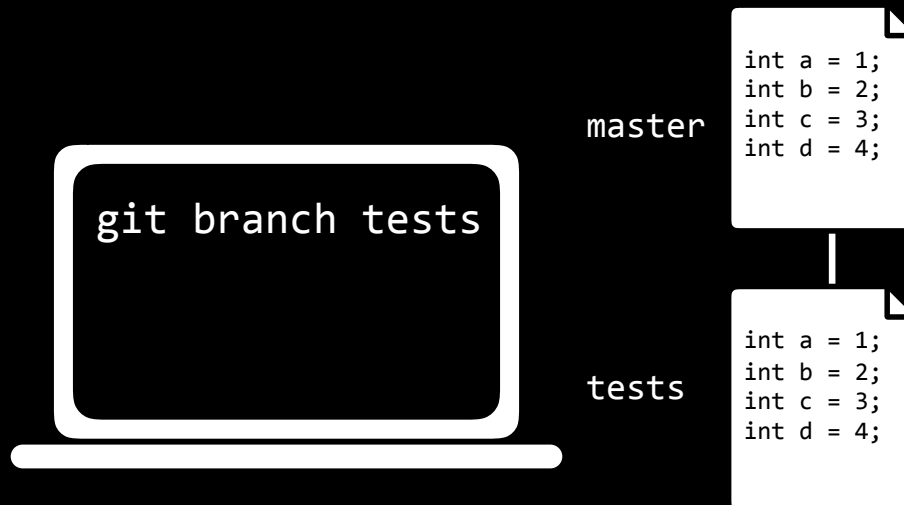
git branch

- shows all branches of code
- create a branch with `git branch <branch_name>`
- switch to ("checkout") a new branch with `git checkout <branch_name>`



git branch

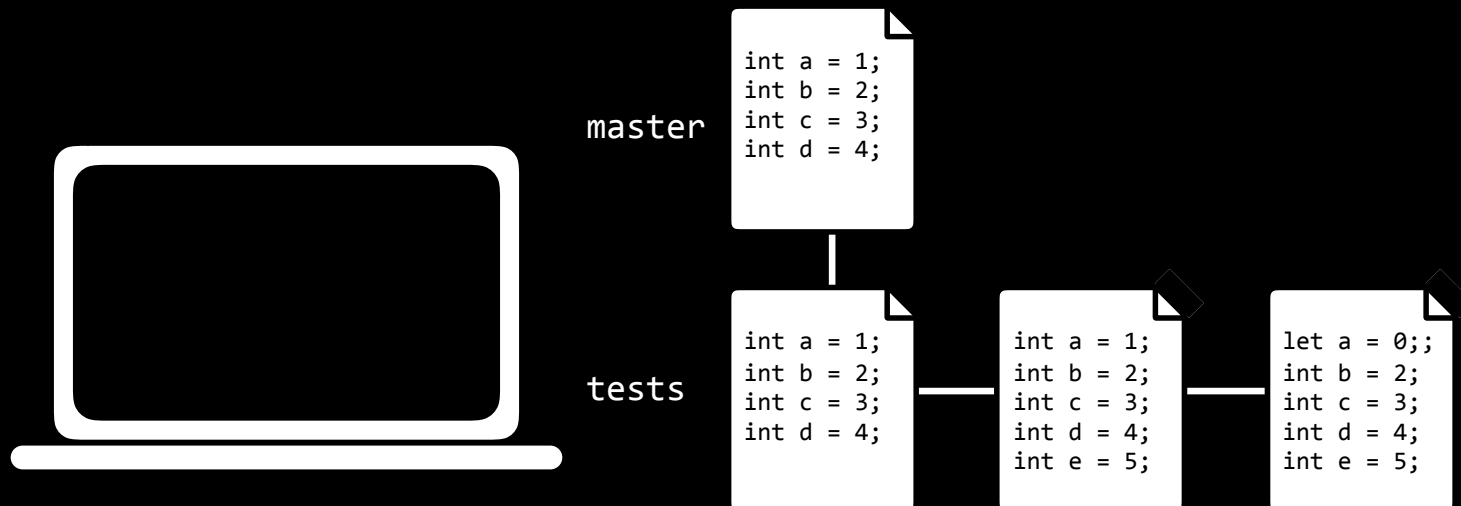
- shows all branches of code
- create a branch with `git branch <branch_name>`
- switch to ("checkout") a new branch with `git checkout <branch_name>`




```
git merge
```

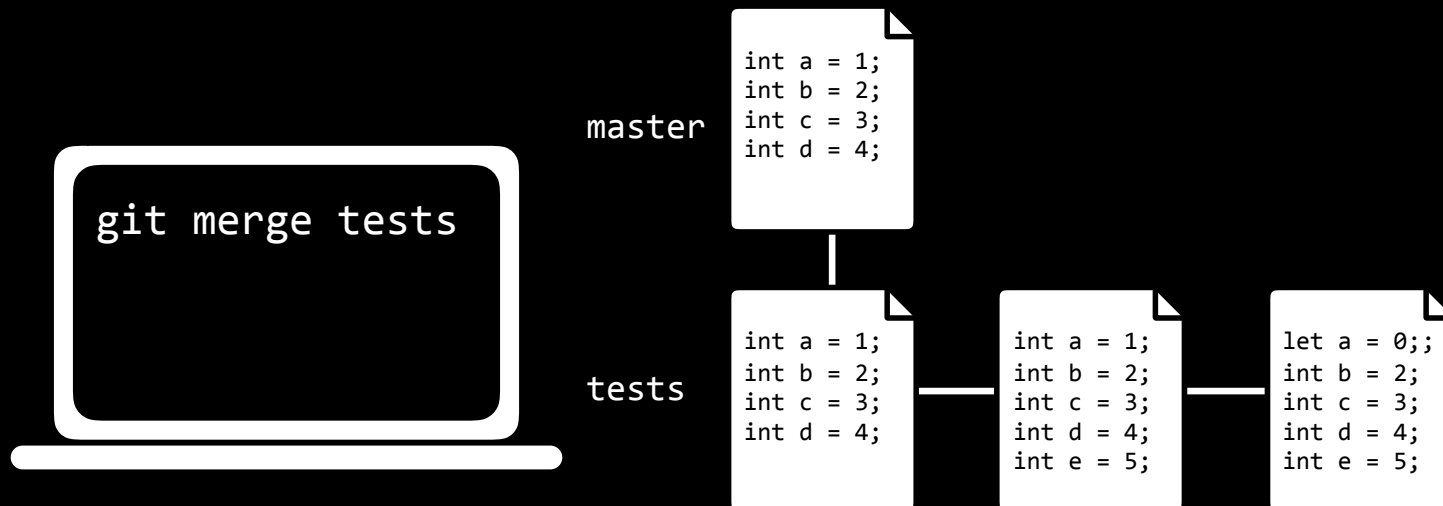
git merge

- `git merge <branch_name>` merges the branch `branch_name` with current branch



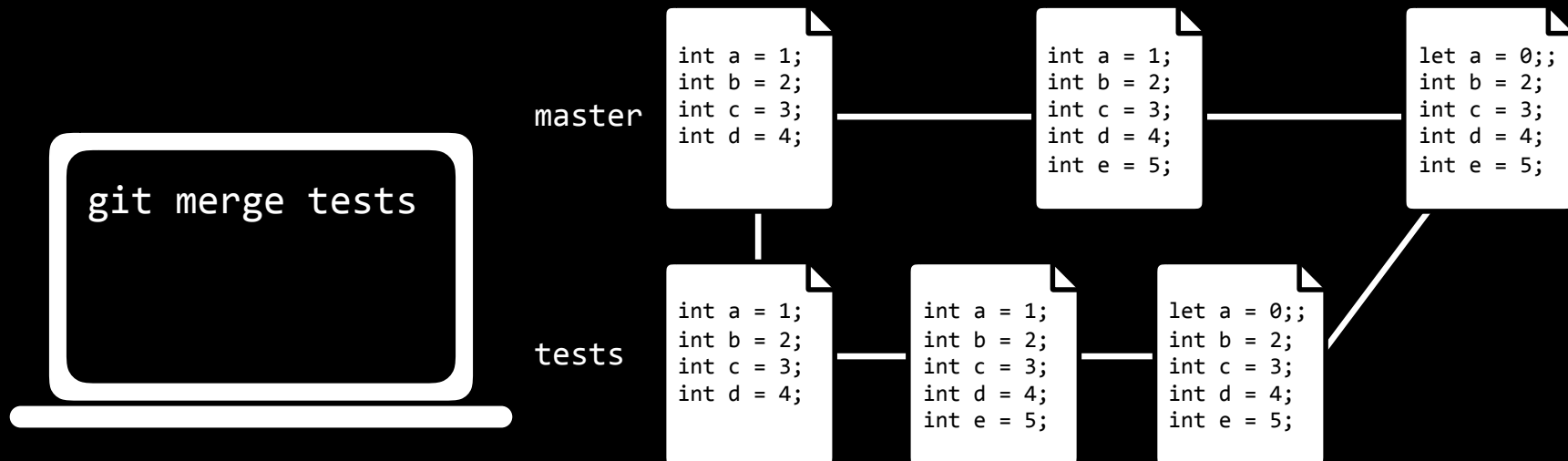
git merge

- `git merge <branch_name>` merges the branch `branch_name` with current branch



git merge

- `git merge <branch_name>` merges the branch `branch_name` with current branch



Pull Requests

Git

- Keeps track of changes to code.
- Synchronizes code between different people.
- Test changes to code without losing the original.
- Revert back to old versions of code.

Git and GitHub

CS50 Seminar
Brian Yu