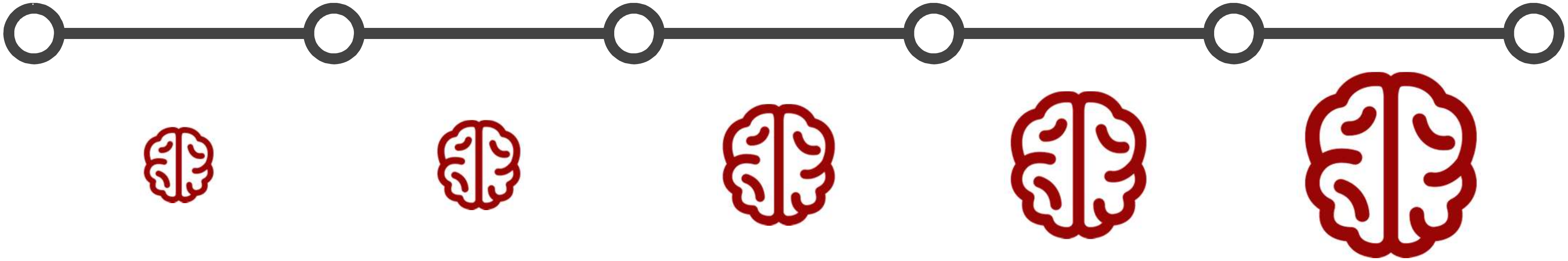
Git Basics

Learning Objectives

Version Control Systems

Installing Git and GitHub

Git Basics



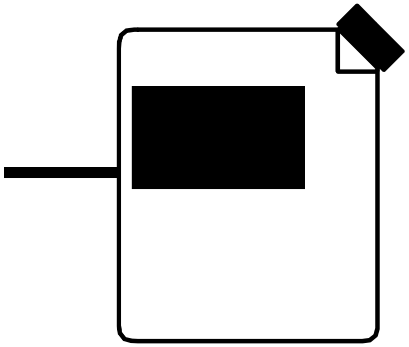
Version Control Systems

What is Version Control?

- A system that keeps records of your changes
- Allows for collaborative development
- Allows you to know who made what changes and when
- **Allows you to revert any changes and go back to a previous state**

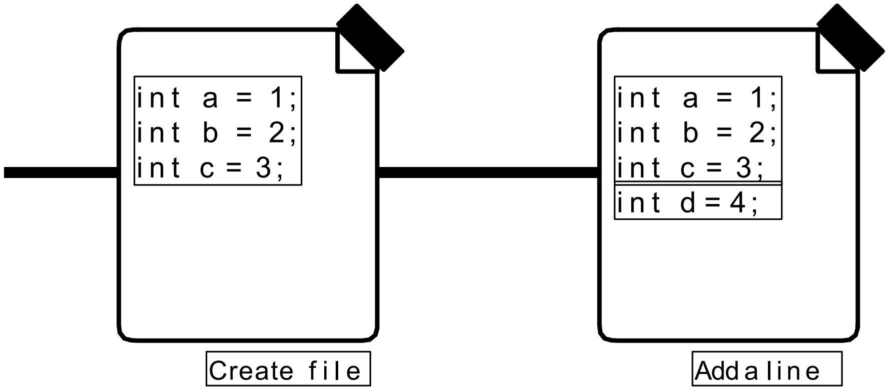
Git Basics

Keep track of changes to code.



Create file

Keep track of changes to code.



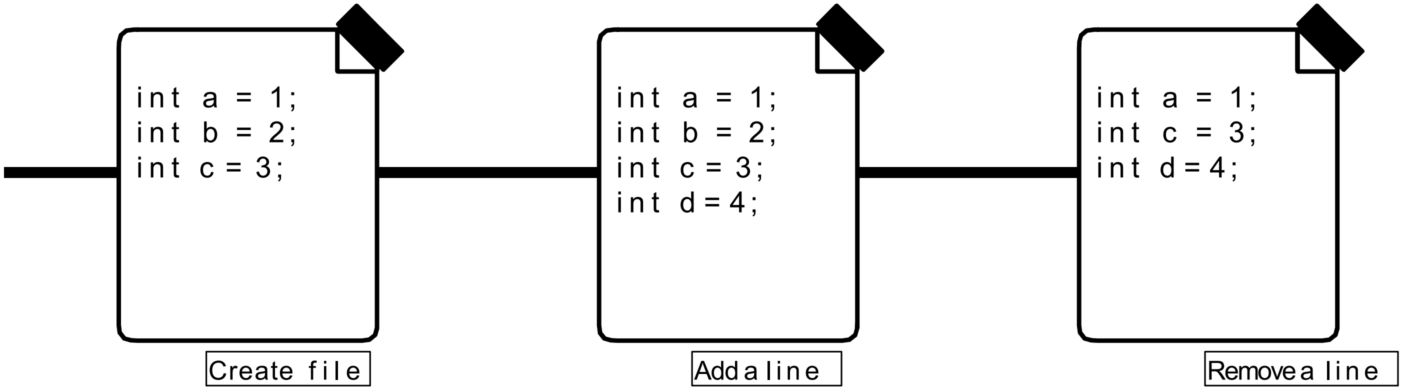
```
int a = 1;
int b = 2;
int c = 3;
```

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

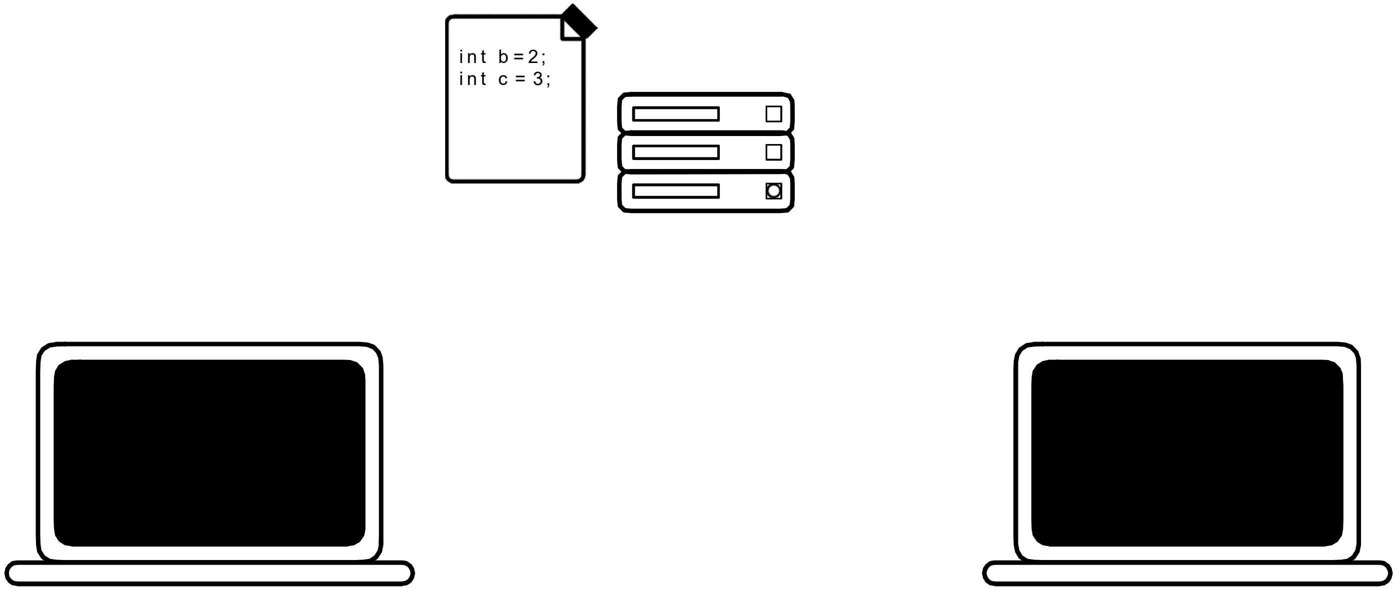
```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```

Add a line

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

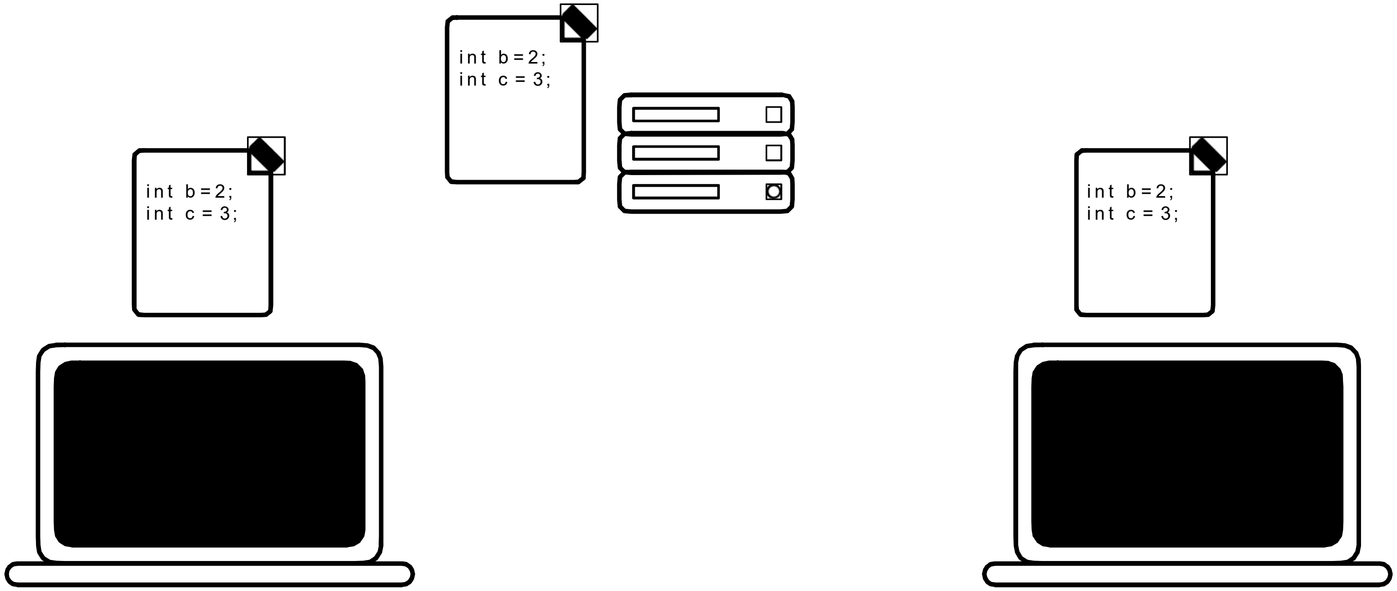
Remove a line

Synchronizes code between different people.



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

Synchronizes code between different people.

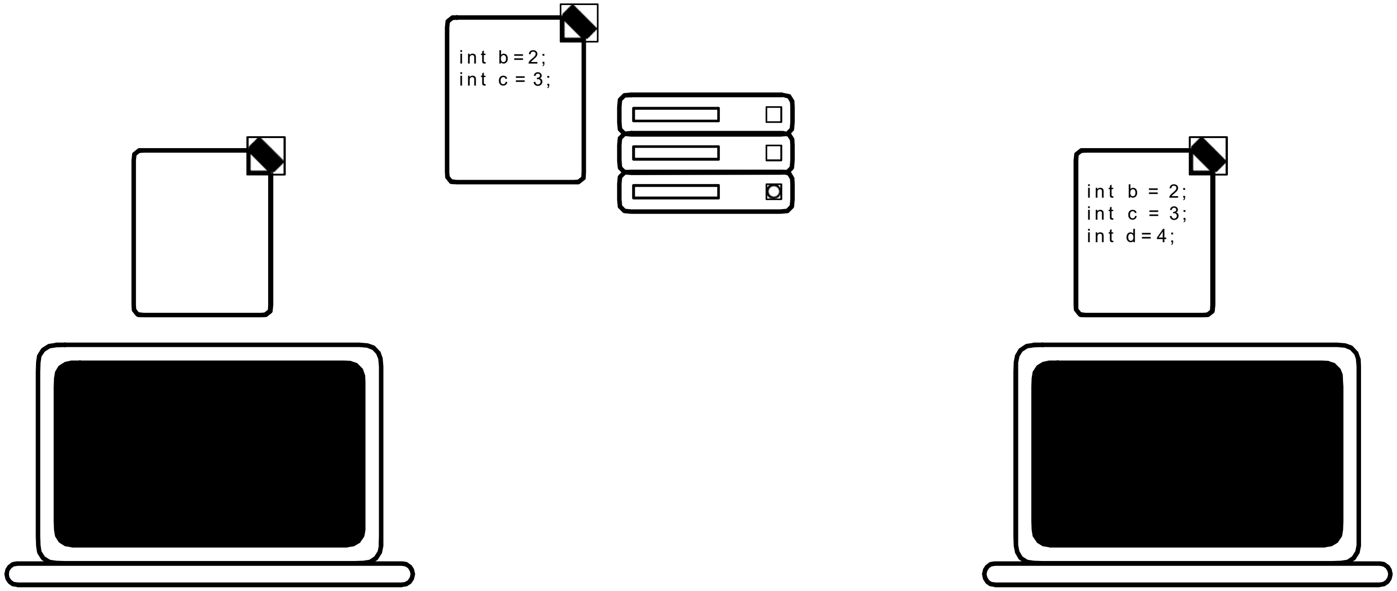


```
int b=2;
int c = 3;
```

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

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

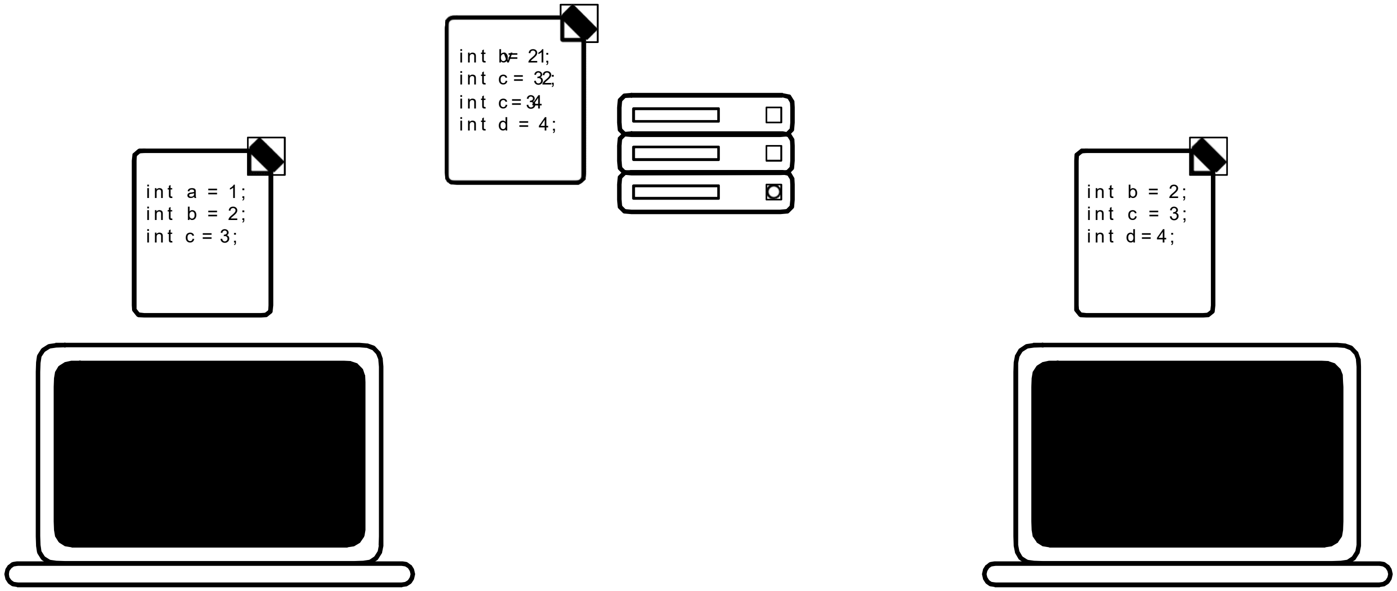
Synchronizes code between different people.



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

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

Synchronizes code between different people.

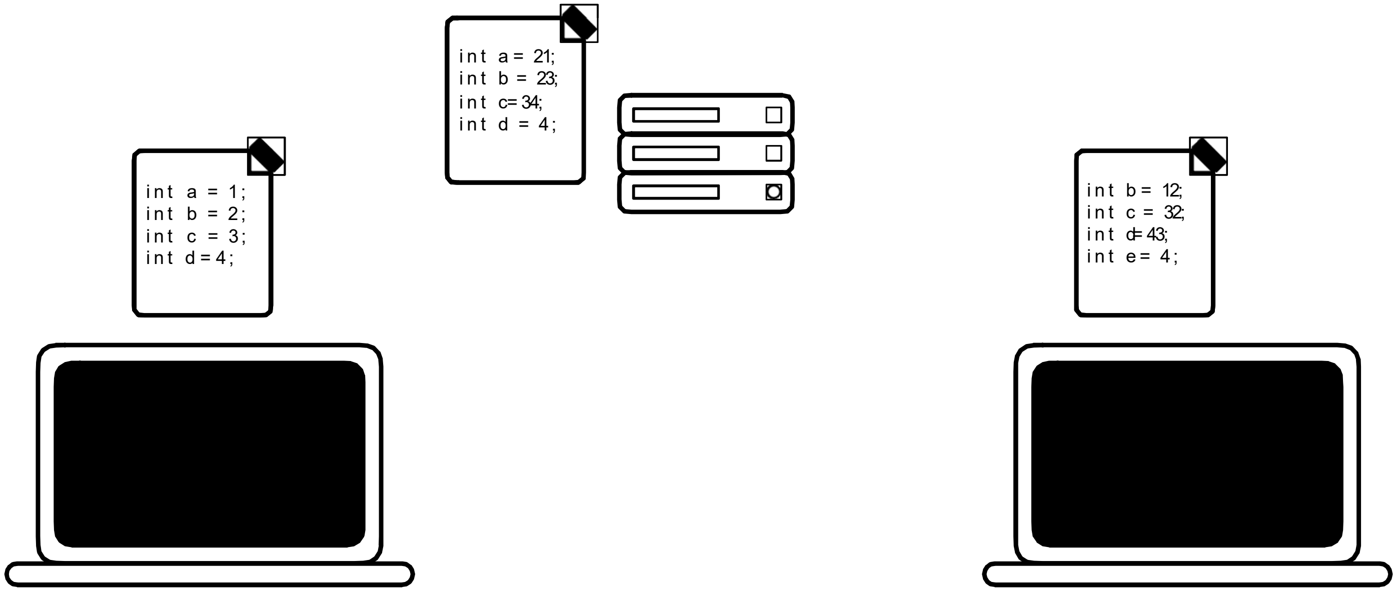


```
int b= 21;
int c = 32;
int c=34
int d = 4;
```

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

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

Synchronizes code between different people.

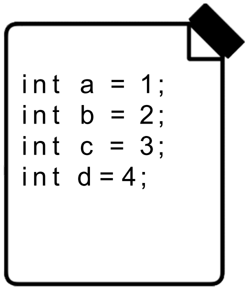


```
int a = 21;
int b = 23;
int c= 34;
int d = 4;
```

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

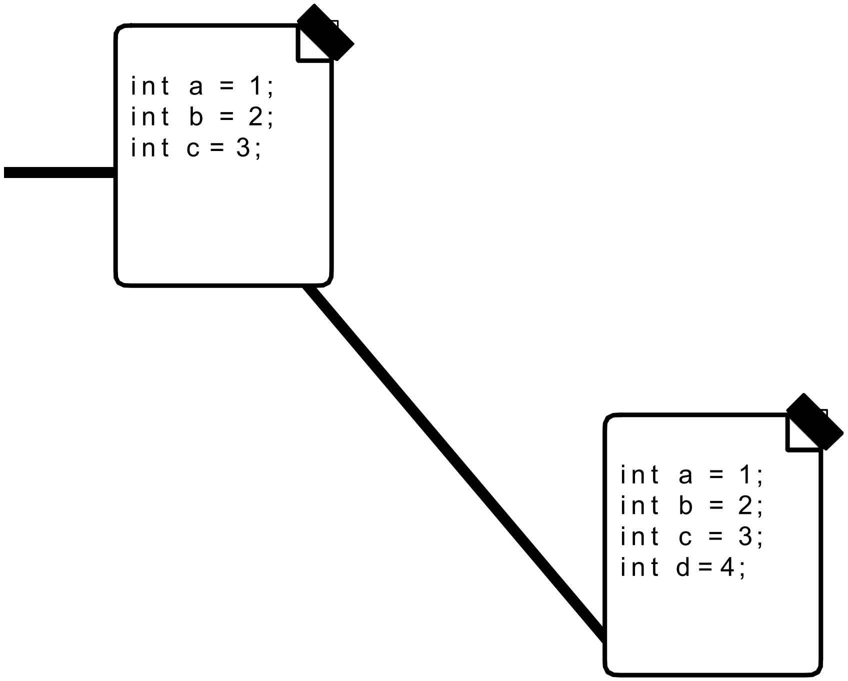
```
int b= 12;
int c = 32;
int d= 43;
int e= 4;
```

Test changes to code without losing the original.



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

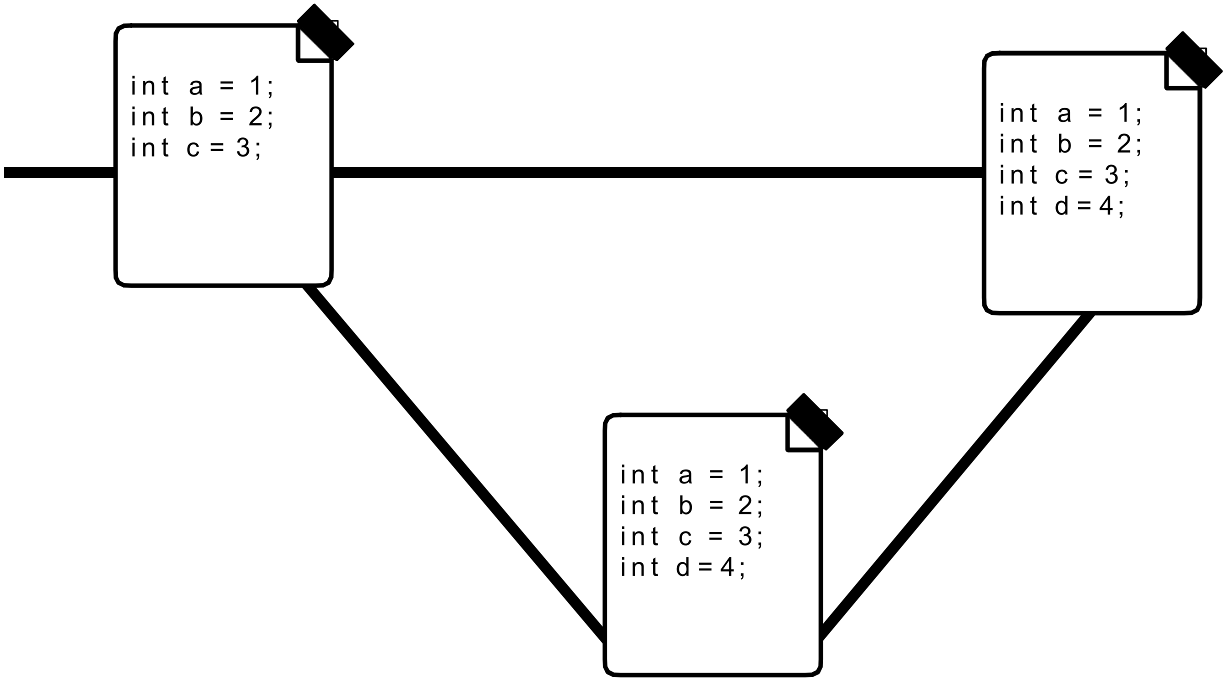
Test changes to code without losing the original.



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

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

Test changes to code without losing the original.

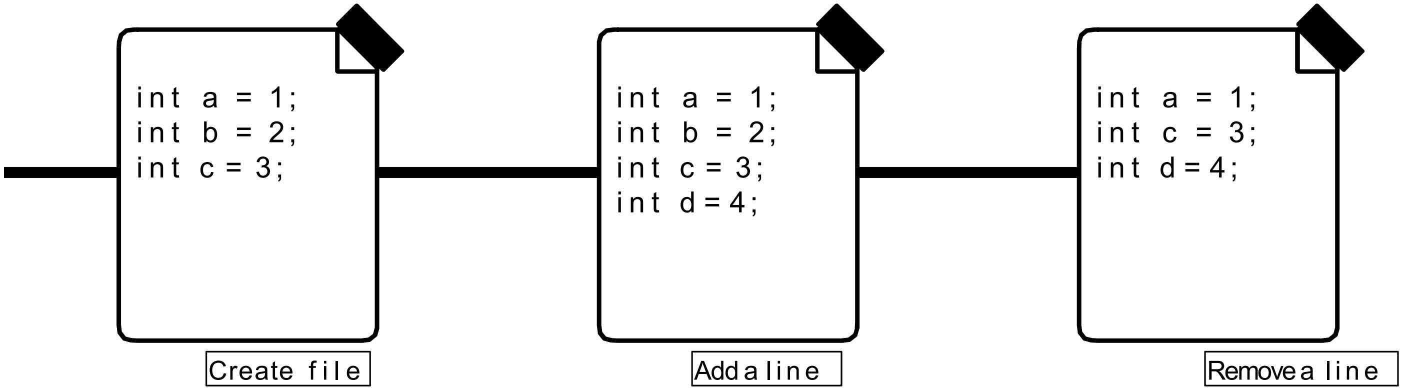


```
int a = 1;
int b = 2;
int c = 3;
```

```
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

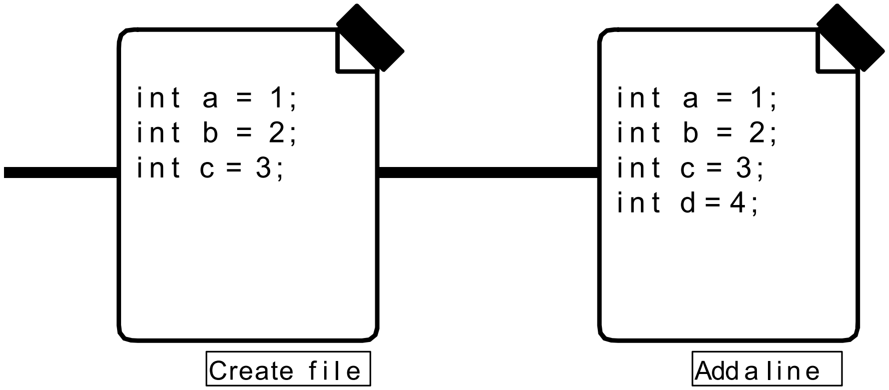
```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
```

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.

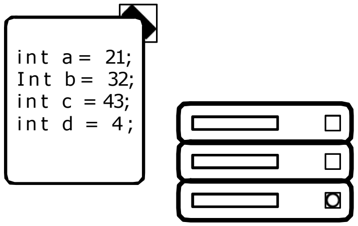
Test changes to code without losing the original.

Revert back to old versions of code.

git clone <url>

makes a copy of a repository stores it on your computer

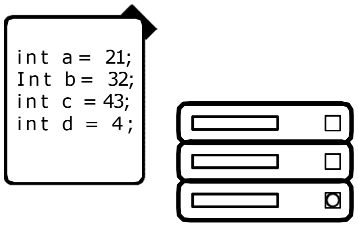
a "fork" creates your own copy of someone else's repository



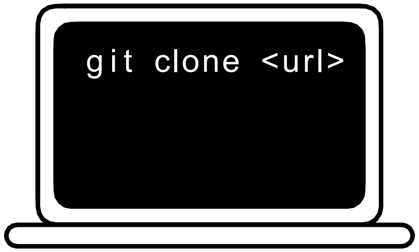
```
int a = 21;
Int b = 32;
int c = 43;
int d = 4;
```

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



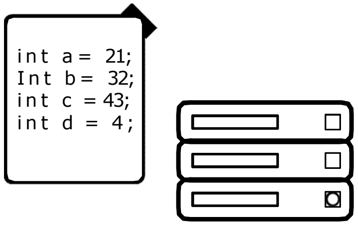
```
int a= 21;
Int b= 32;
int c = 43;
int d = 4;
```



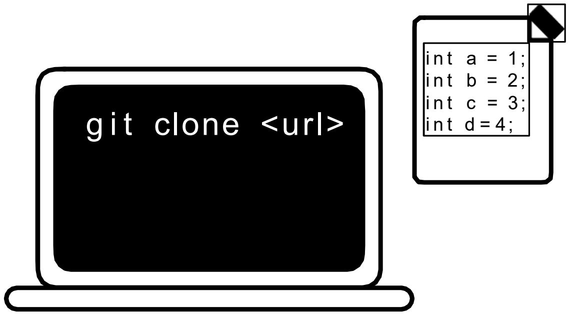
```
git clone <url>
```

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= 21;
Int b= 32;
int c = 43;
int d = 4;
```



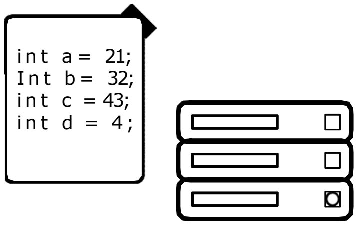
```
git clone <url>
```

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

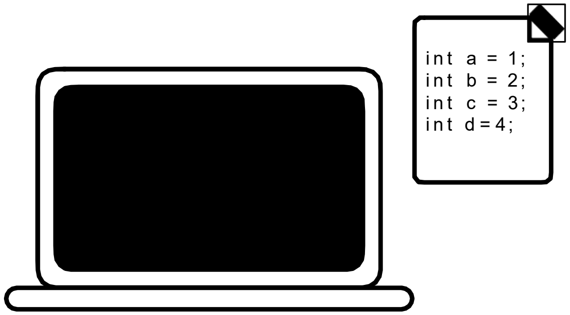
git add <filename>

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= 21;
Int b= 32;
int c = 43;
int d = 4;
```



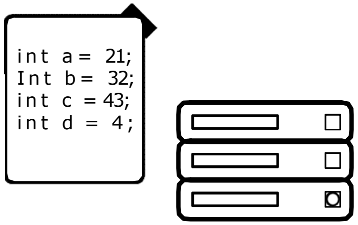
```
int a = 1;
int b = 2;
int c = 3;
int d=4;
```

GREYATOM
learn = do real work

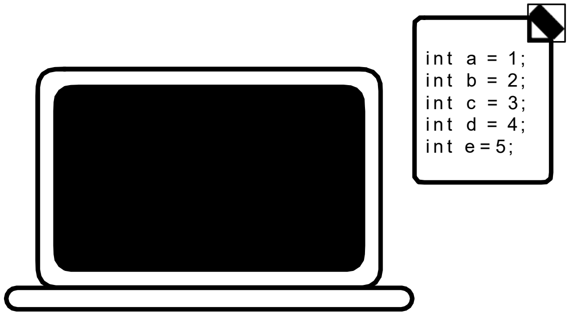
git add <filename>

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= 21;
Int b= 32;
int c = 43;
int d = 4;
```



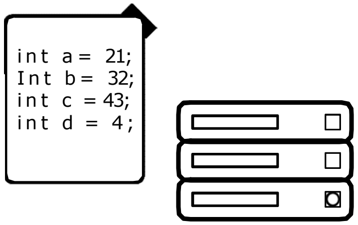
```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e=5;
```

GREYATOM
learn = do real work

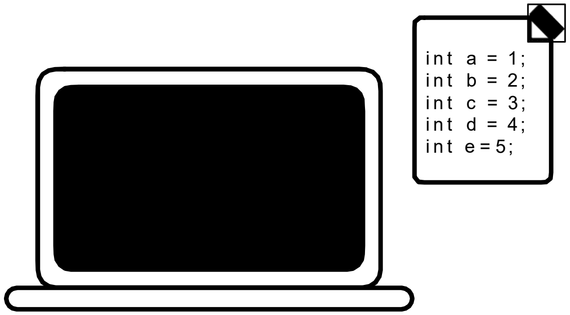
git add <filename>

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= 21;
Int b= 32;
int c = 43;
int d = 4;
```

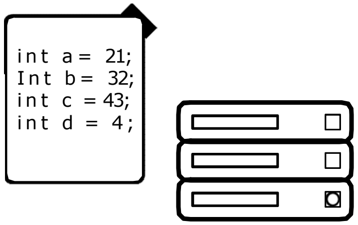


```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e=5;
```

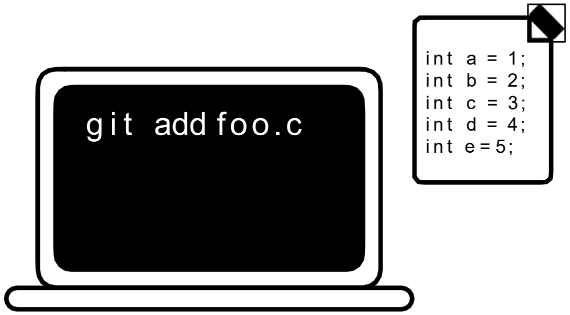
git add <filename>

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 = 21;
Int b = 32;
int c = 43;
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:

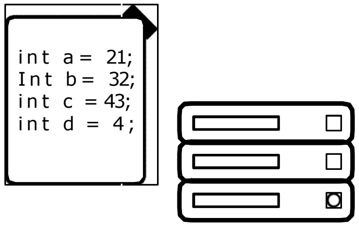
modified: foo.c

git commit -m "message"

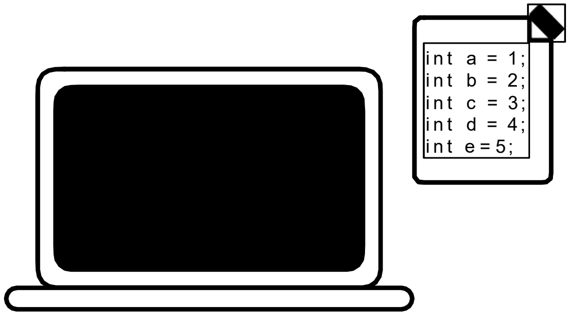
saves the changes to repository as a new revision (a "commit") records a message

Git commit –am "message"

Adds and commits in same step



```
int a= 21;
Int b= 32;
int c = 43;
int d = 4;
```



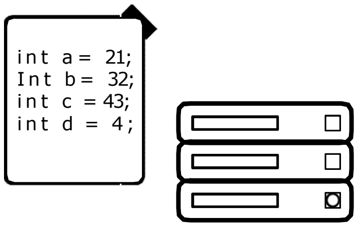
```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e=5;
```

git commit -m "message"

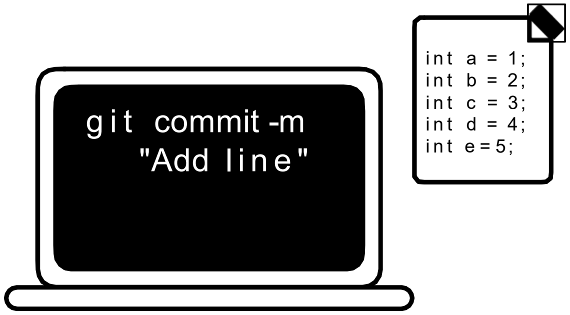
saves the changes to repository as a new revision (a "commit") records a message

Git commit –am "message"

Adds and commits in same step



```
int a= 21;
Int b= 32;
int c = 43;
int d = 4;
```



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

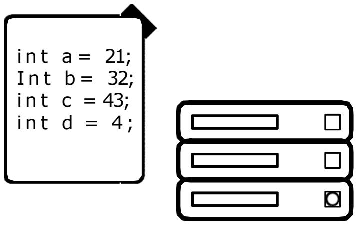
```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
int e=5;
```

git commit -m "message"

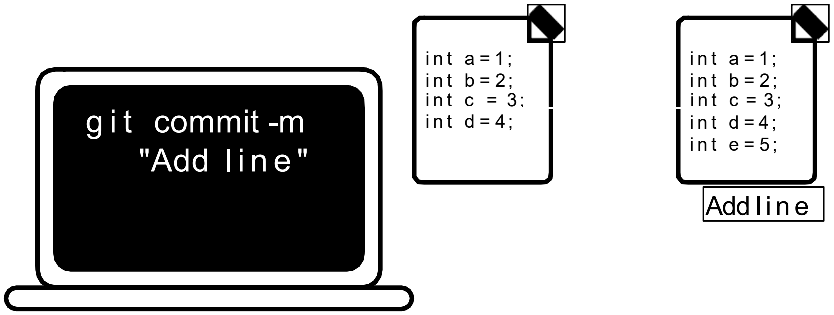
saves the changes to repository as a new revision (a "commit") records a message

Git commit –am "message"

Adds and commits in same step



```
int a= 21;
Int b= 32;
int c = 43;
int d = 4;
```



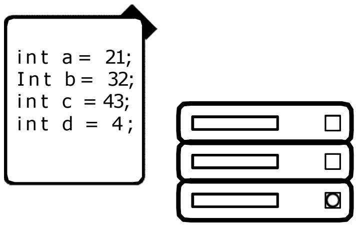
```
git commit -m
   "Add line"
```

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

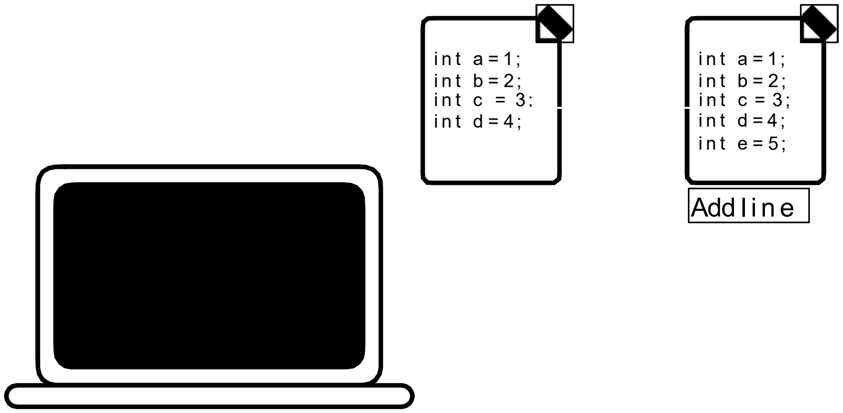
```
int a=1;
int b=2;
int c=3;
int d=4;
int e=5;
```

git status

-shows current status of repository



```
int a= 21;
Int b= 32;
int c =43;
int d = 4;
```

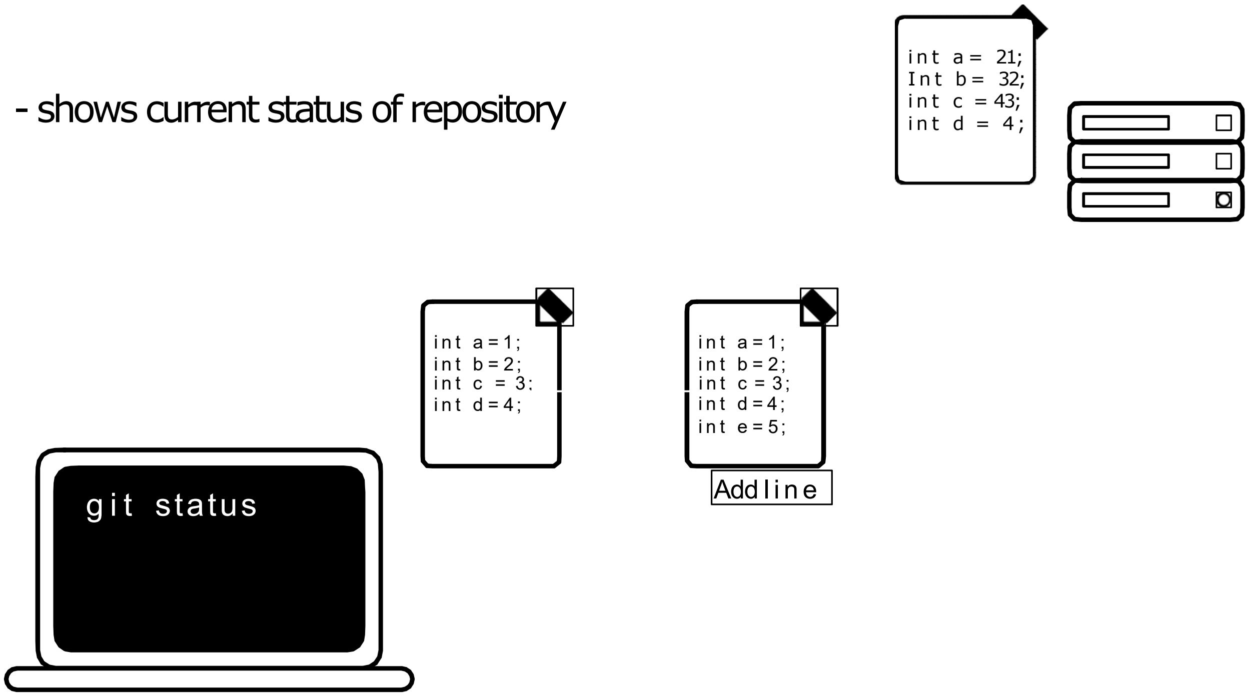


```
int a=1;
int b=2;
int c = 3;
int d=4;
```

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

git status

- shows current status of repository



```
int a= 21;
Int b= 32;
int c = 43;
int d = 4;
```

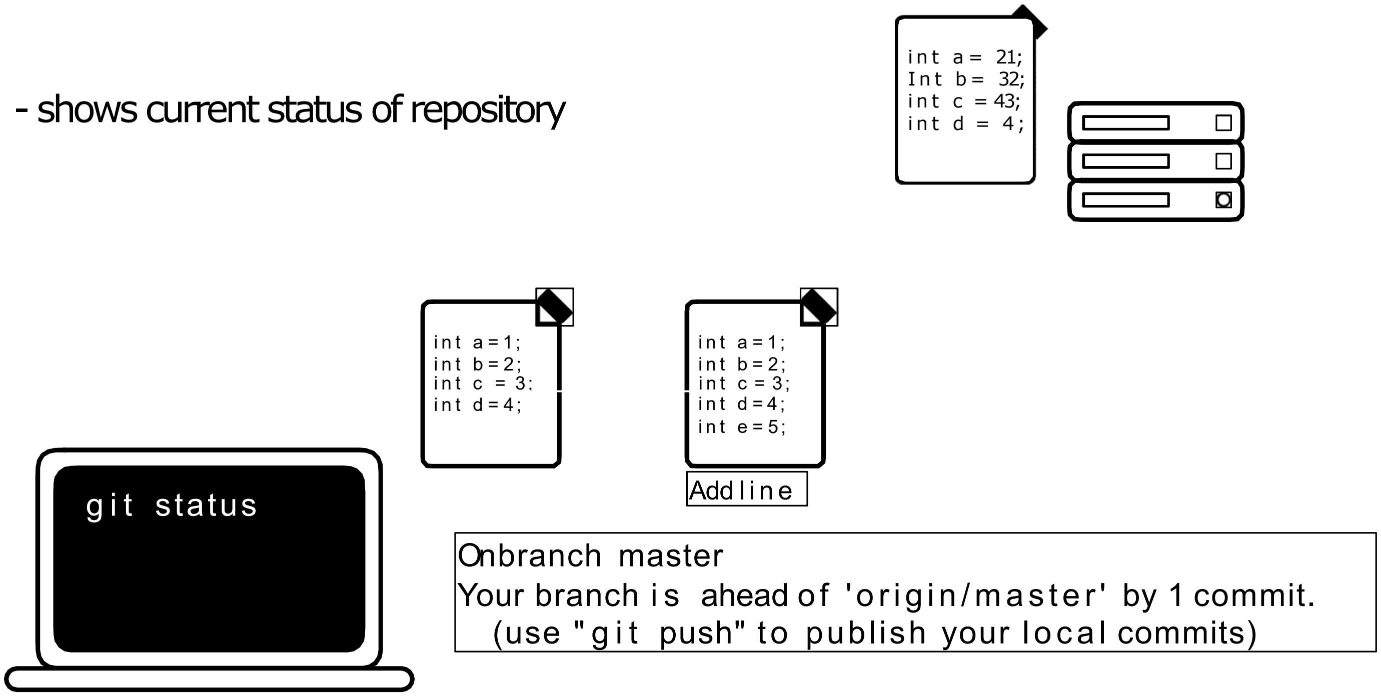
```
git status
```

```
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 status

- shows current status of repository



```
int a = 21;
Int b = 32;
int c = 43;
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;
```

```
git status
```

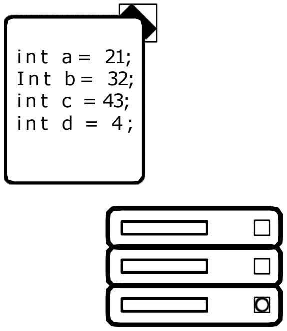
```
Onbranch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
```

git push

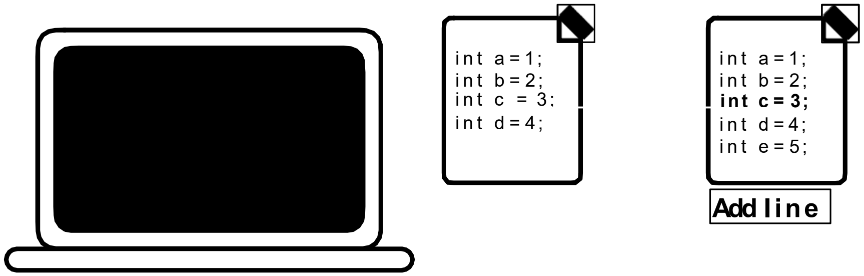
- sends committed changes to remote repository

more explicitly, could write

Git push origin master



```
int a= 21;
Int b= 32;
int c =43;
int d = 4;
```



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

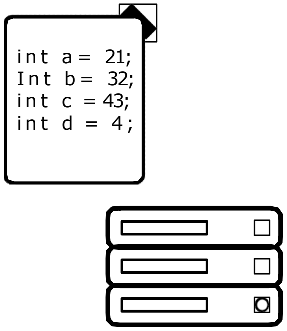
```
int a=1;
int b=2;
int c=3;
int d=4;
int e=5;
```

git push

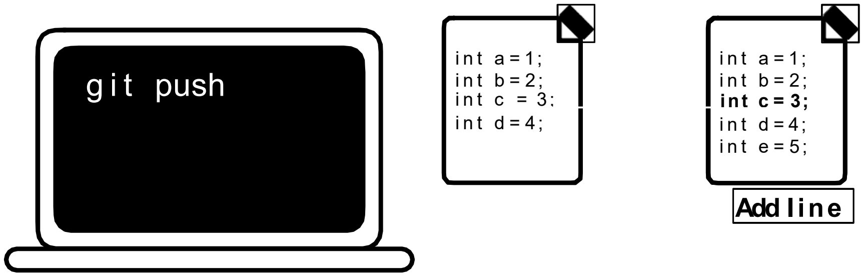
- sends committed changes to remote repository

more explicitly, could write

Git push origin master



```
int a = 21;
Int b = 32;
int c = 43;
int d = 4;
```



```
git push
```

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

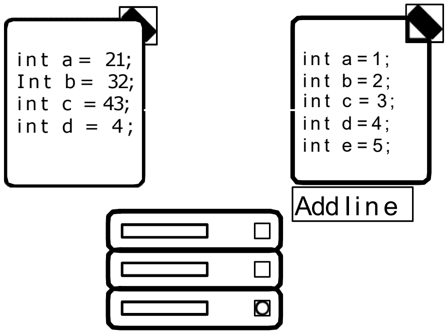
```
int a=1;
int b=2;
int c=3;
int d=4;
int e=5;
```

git push

- sends committed changes to remote repository

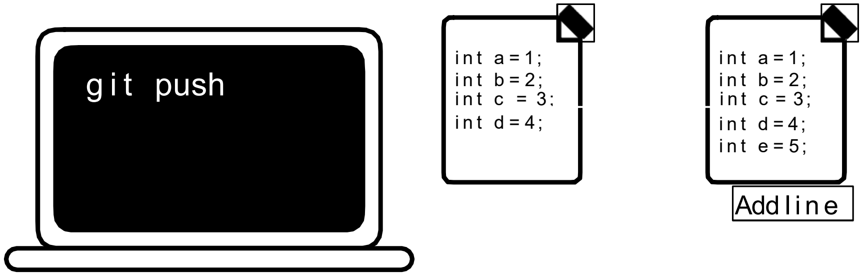
more explicitly, could write

Git push origin master



```
int a = 21;
Int b = 32;
int c = 43;
int d = 4;
```

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



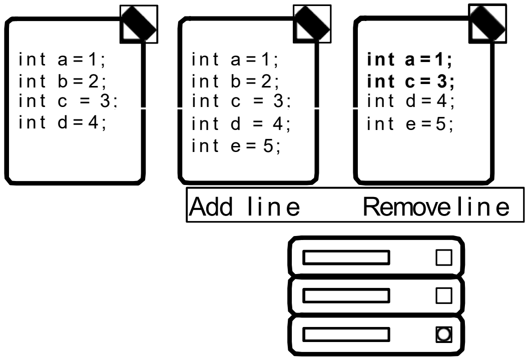
```
git push
```

```
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 pull

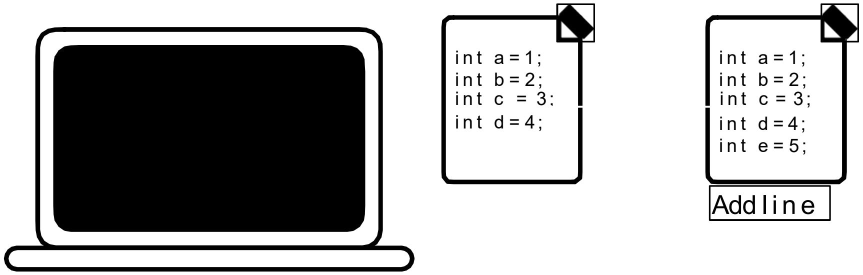
- retrieves changes from remote repository



```
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 e=5;
```

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



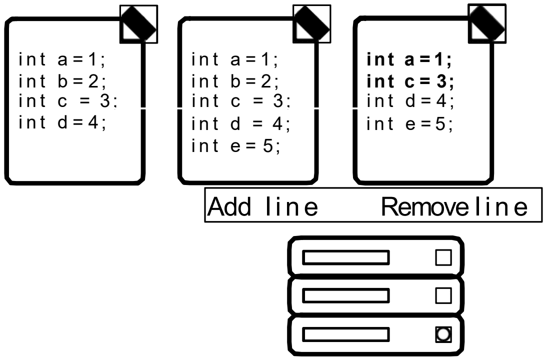
```
int a=1;
int b=2;
int c = 3:
int d=4;
```

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

GREYATOM
learn = do real work

git pull

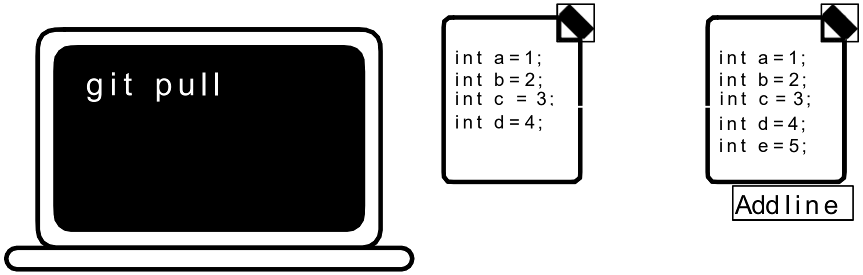
- retrieves changes from remote repository



```
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 e=5;
```

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



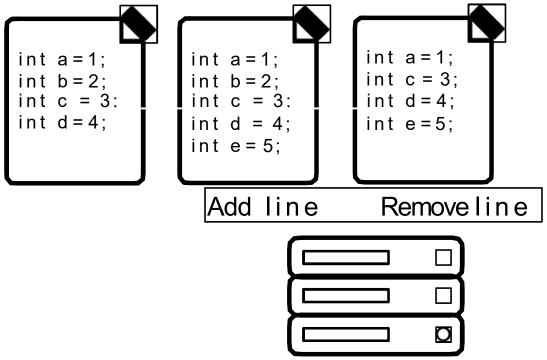
```
git pull
```

```
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 pull

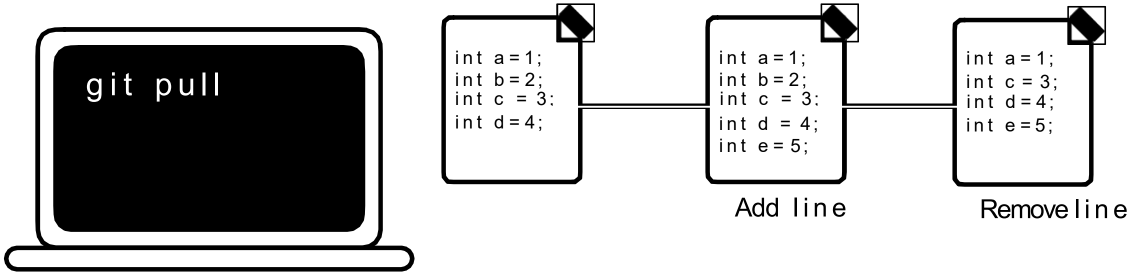
- retrieves changes from remote repository



```
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 e = 5;
```

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



```
git pull
```

```
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;
```

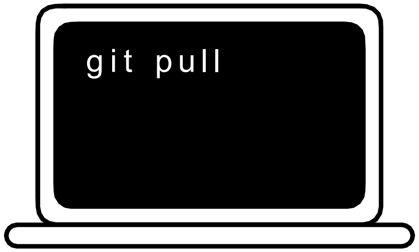
```
int a = 1;
int c = 3;
int d = 4;
int e = 5;
```

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
```

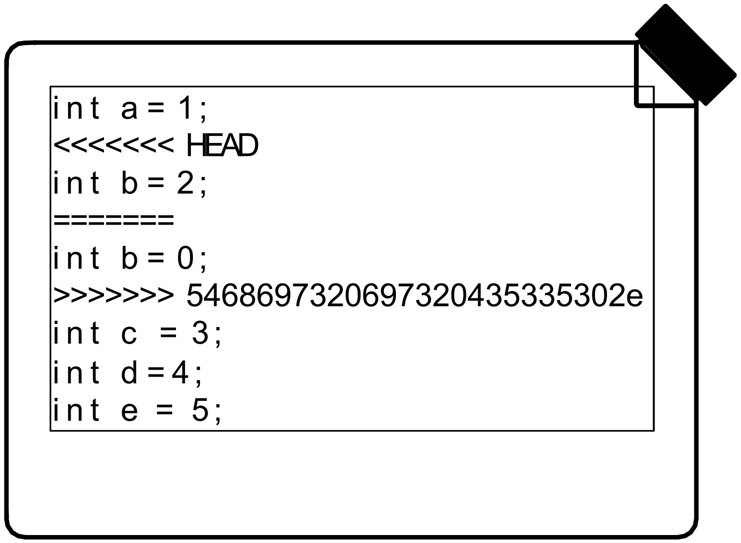
Merge Conflicts

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

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

Merge Conflicts

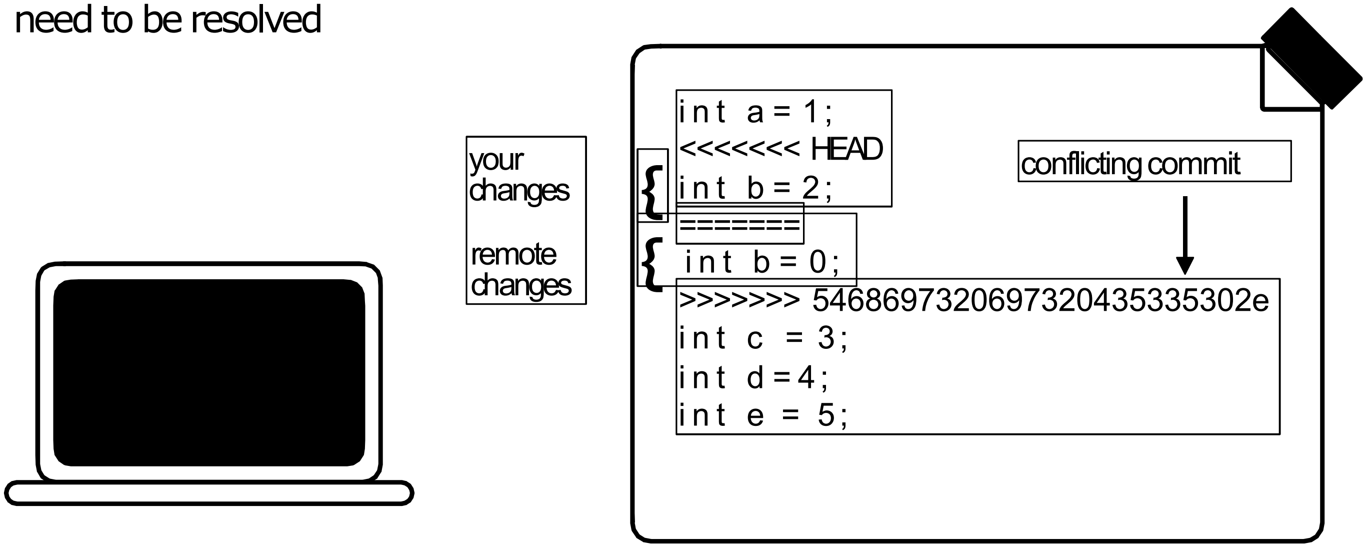
- when two different commits can't be automatically merged need to be resolved



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

Merge Conflicts

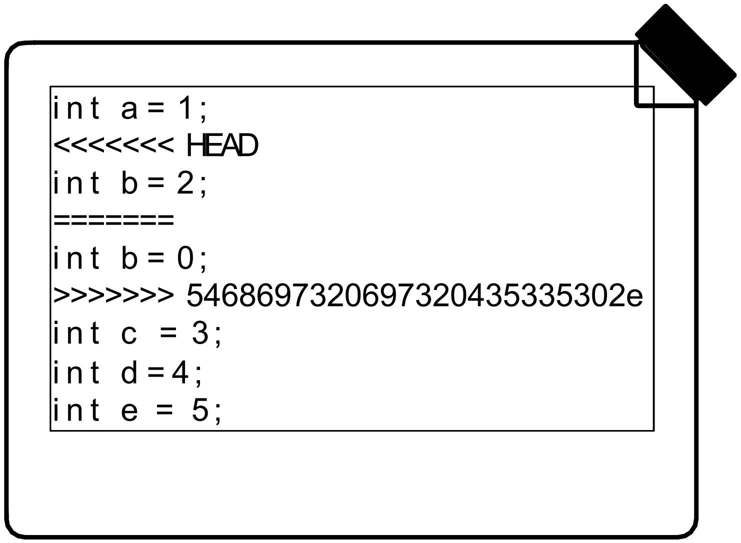
- when two different commits
can't be automatically merged
need to be resolved



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

Merge Conflicts

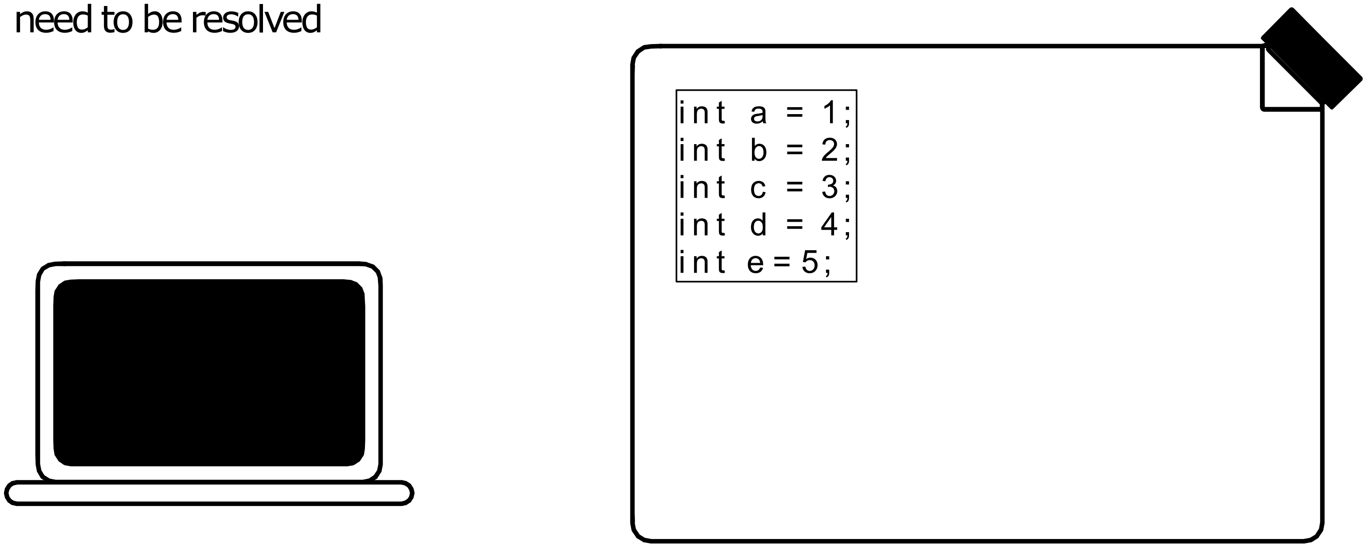
- when two different commits
can't be automatically merged
need to be resolved



```
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



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

Git log

- shows a history of commits and messages

```
git log
```

Git reset

- Git reset –hard <commit>

reverts code back to a previous commit

- Git reset –hard origin/master

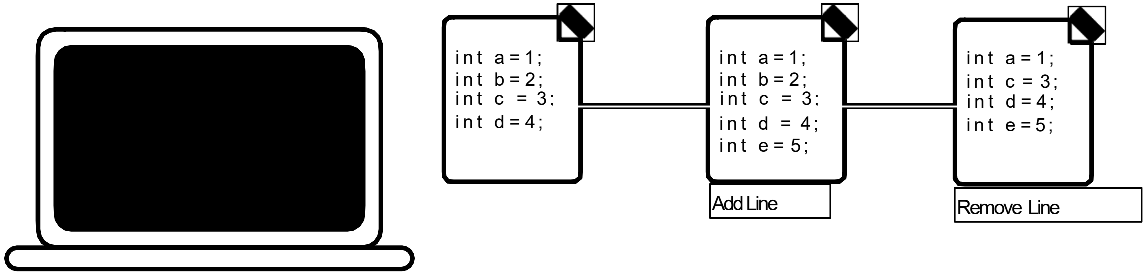
reverts code back to remote repository version

```
$ git log
commit dfb8c46953598dfd57124b11027b29f21b818dad (HEA
origin/master, origin/HEAD)
Author: Varun Panicker <varunppanicker@gmail.com>
Date:   Fri Apr 6 20:03:32 2018 +0530

    changed to 77777

commit f22f947ff36701c845e67e9079ff8d12abdfeda7
Author: Varun Panicker <varunppanicker@gmail.com>
Date:   Fri Apr 6 19:58:22 2018 +0530

    Changed a to 77
```



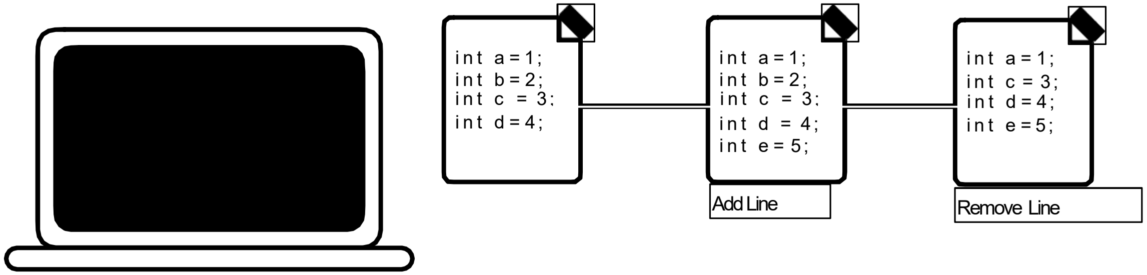
```
int a=1;
int b=2;
int c = 3;
int d=4;
```

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

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

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



```
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;
```

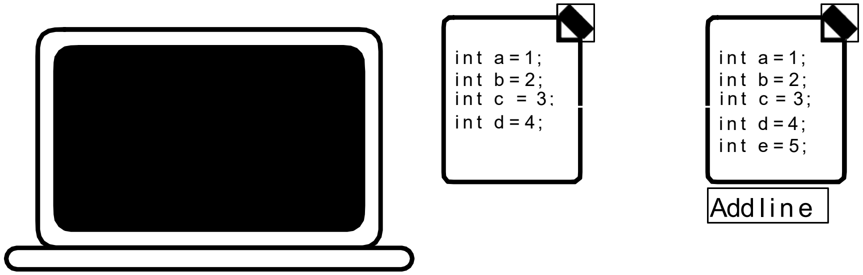
```
int a=1;
int c=3;
int d=4;
int e=5;
```

GREYATOM
learn = do real work

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



```
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;
```

GREYATOM
learn = do real work

Branching

Branch is a version of the repository.

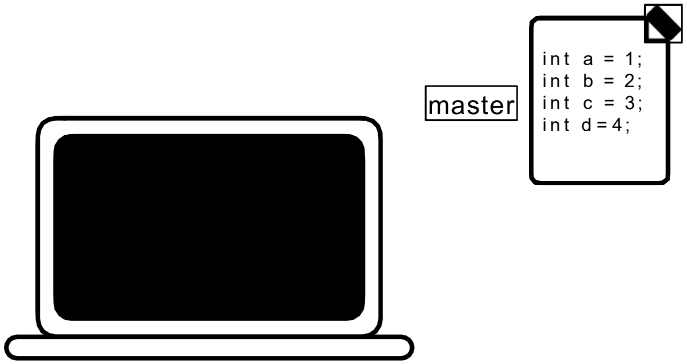
Each branch has its own commit history and current version.

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>**



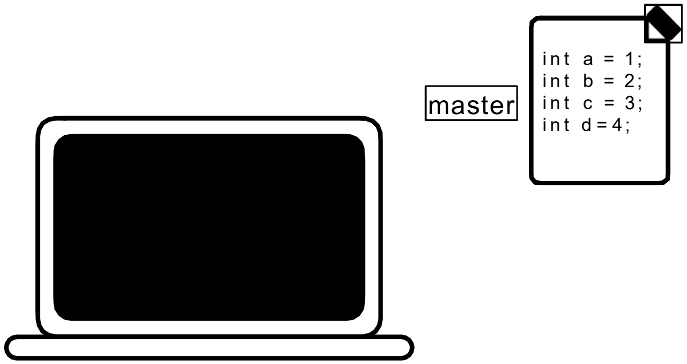
```
int a = 1;
int b = 2;
int c = 3;
int d=4;
```

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>



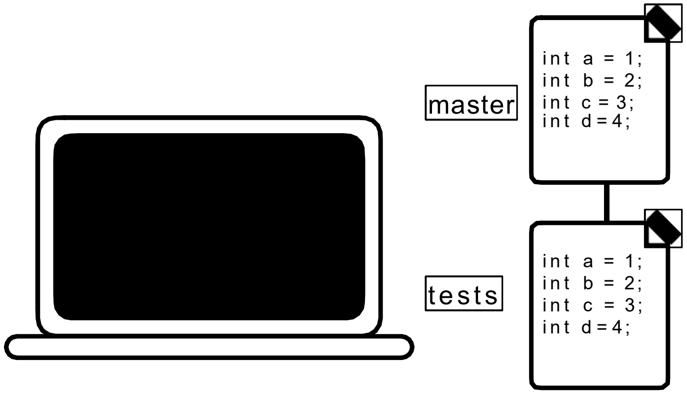
```
int a = 1;
int b = 2;
int c = 3;
int d=4;
```

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>



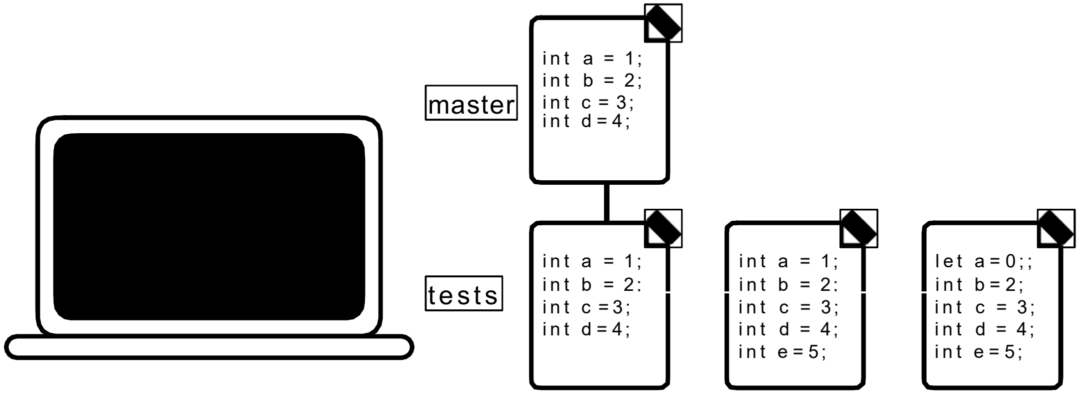
```
int  a = 1;
int  b = 2;
int  c = 3;
int  d = 4;
```

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

git merge

- Git merge <branch_name>

Merges the branch branch_name with current branch



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

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

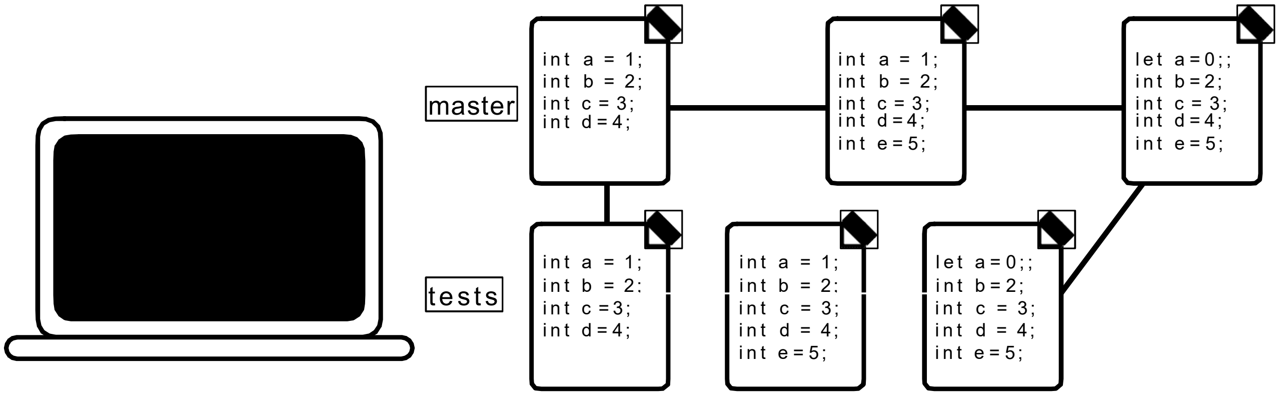
```
int a = 1;
int b = 2:
int c = 3;
int d = 4;
int e=5;
```

```
let a=0;;
int b=2;
int c = 3;
int d = 4;
int e=5;
```

git merge

- Git merge <branch_name>

Merges the branch branch_name with current branch



```
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;
```

```
let a=0;;
int b=2;
int c = 3;
int d=4;
int e=5;
```

```
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;
```

```
let a=0;;
int b=2;
int c = 3;
int d = 4;
int e=5;
```

Thank You!