Version Control, Git, and GitHub

Software Engineering for Scientists

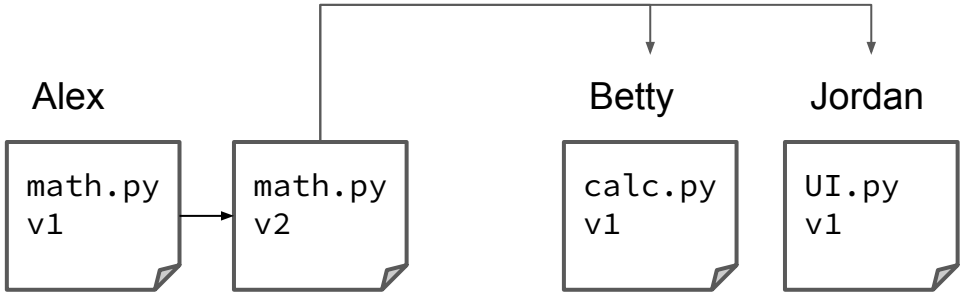
Version control manages changes to files for a project

```
def div(a, b):
  return a/b
```

```
def div(a, b):
  if b != 0:
    return a/b
  else:
    return None
```

```
def div(a, b):
  if b != 0:
    return a/b
  else:
    return Inf
```

- reverting back to an old version
- allowing developers to test changes without losing the original
- synchronizing code between developers and users



- tagging specific versions

Version control software

Software repository hosting company

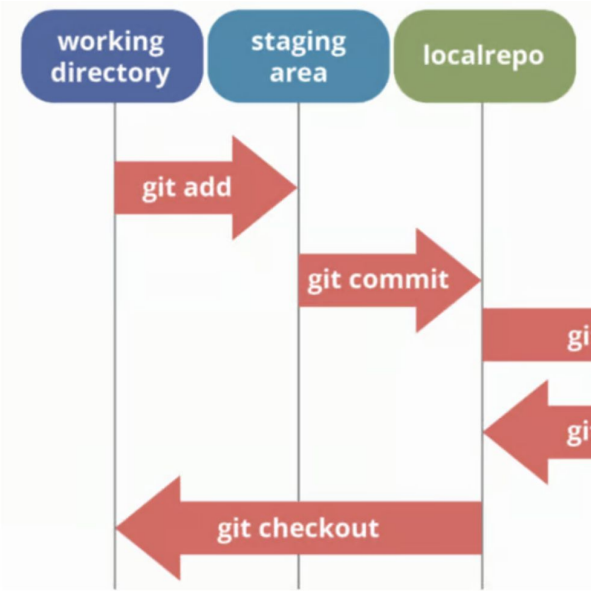
Git basics (locally)

git repository/repo: you can think of a git repository as a directory that stores all the files, folders, and content needed for your project along with the history of those folders and files

commit: a snapshot of the repo along with a unique identifier, message, and metadata about the changes since the last commit

working directory: the current state of your project files on your filesystem, where you make and edit changes.

staging area: a space where changes are prepared before committing, allowing you to decide what to include in the next commit.



DEMO

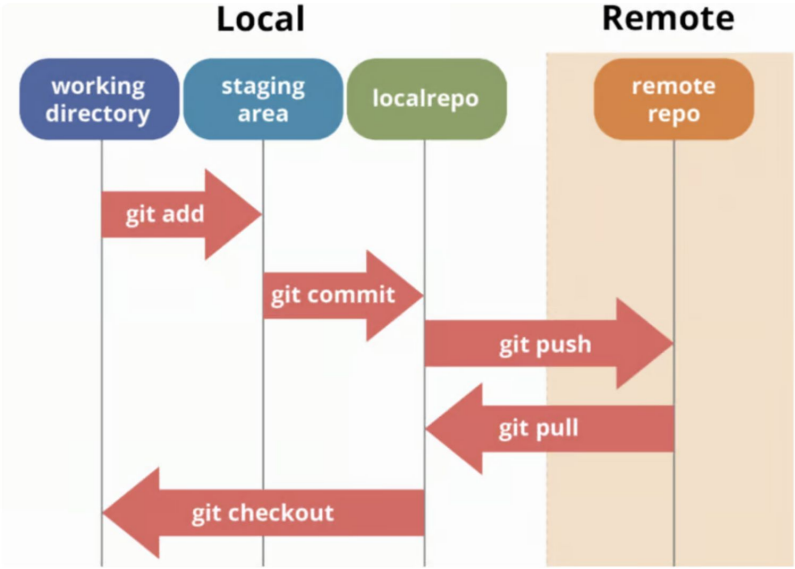
Git basics (full picture)

git repository/repo: you can think of a git repository as a directory that stores all the files, folders, and content needed for your project along with the history of those folders and files

commit: a snapshot of the repo along with a unique identifier, message, and metadata about the changes since the last commit

working directory: the current state of your project files on your filesystem, where you make and edit changes.

staging area: a space where changes are prepared before committing, allowing you to decide what to include in the next commit.

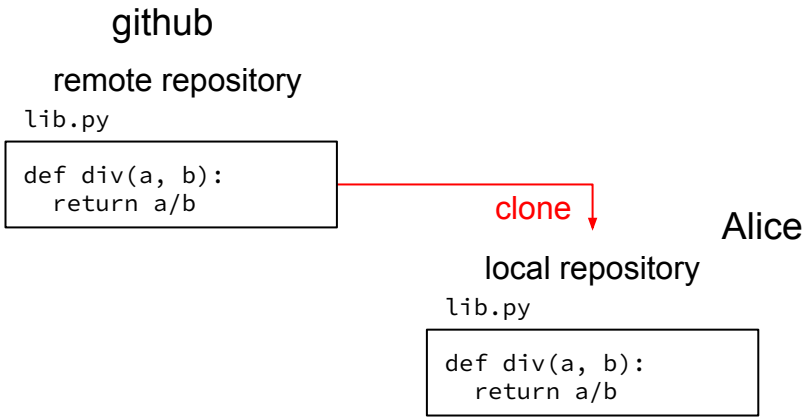


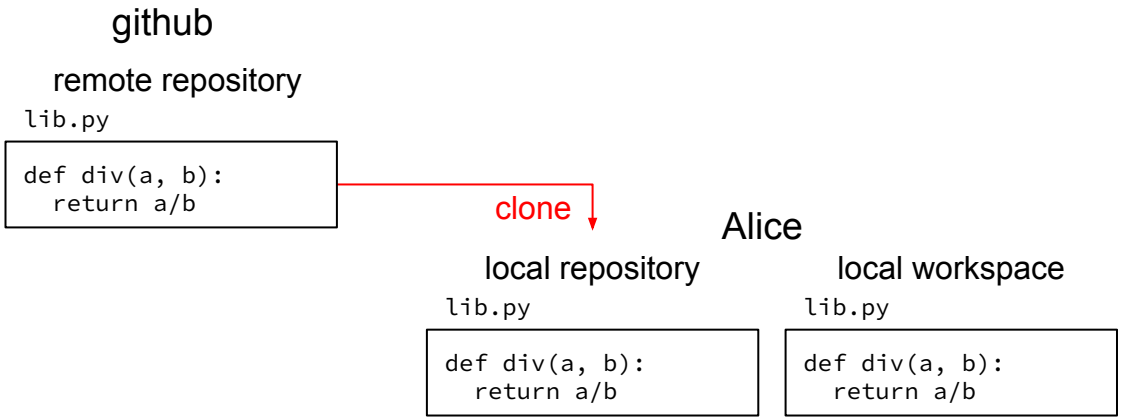
github

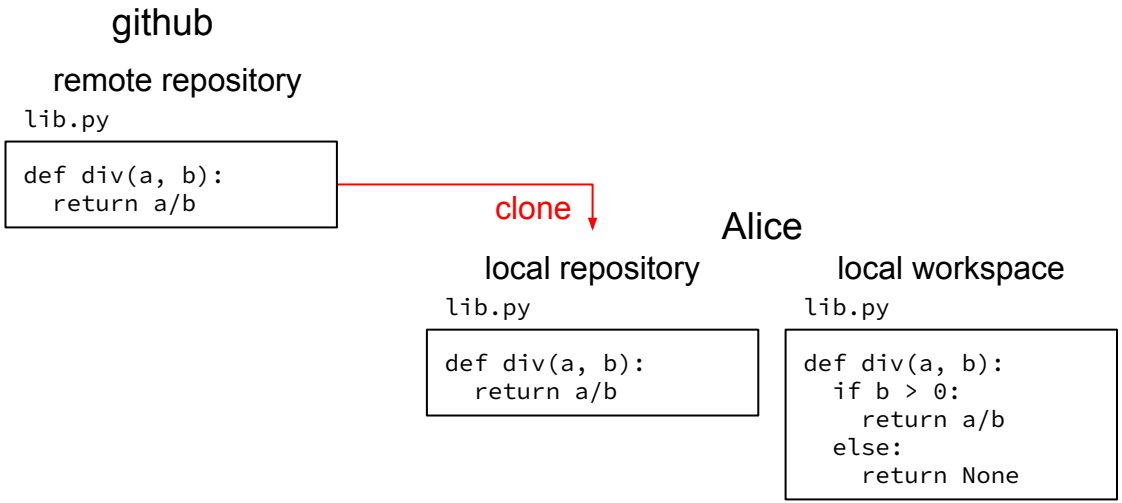
remote repository

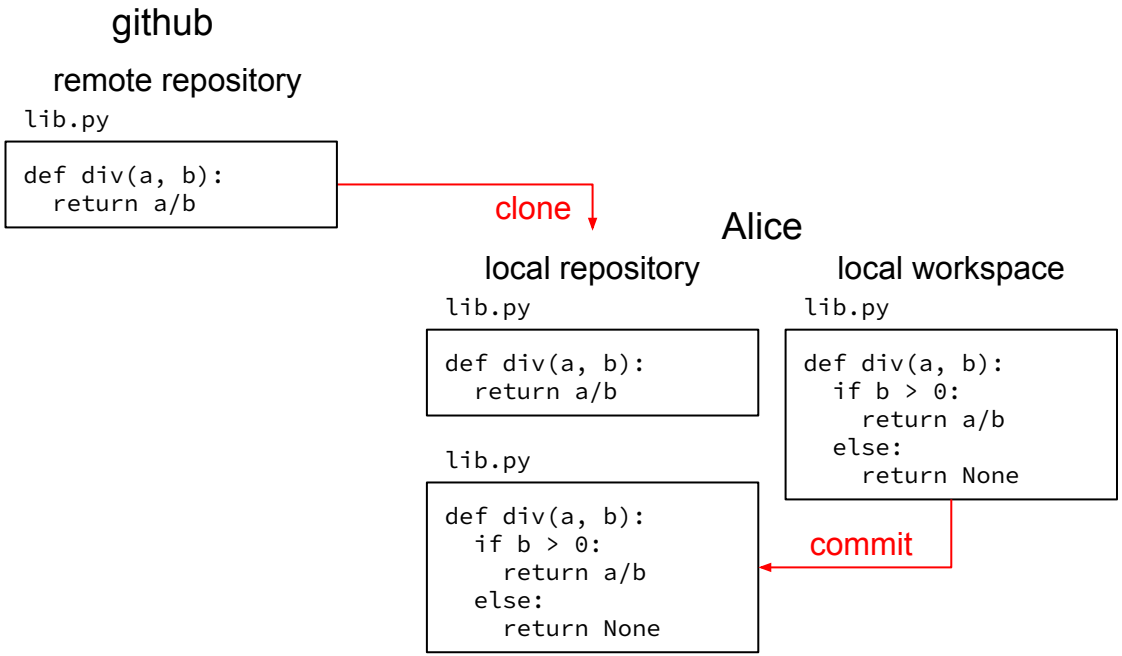
lib.py

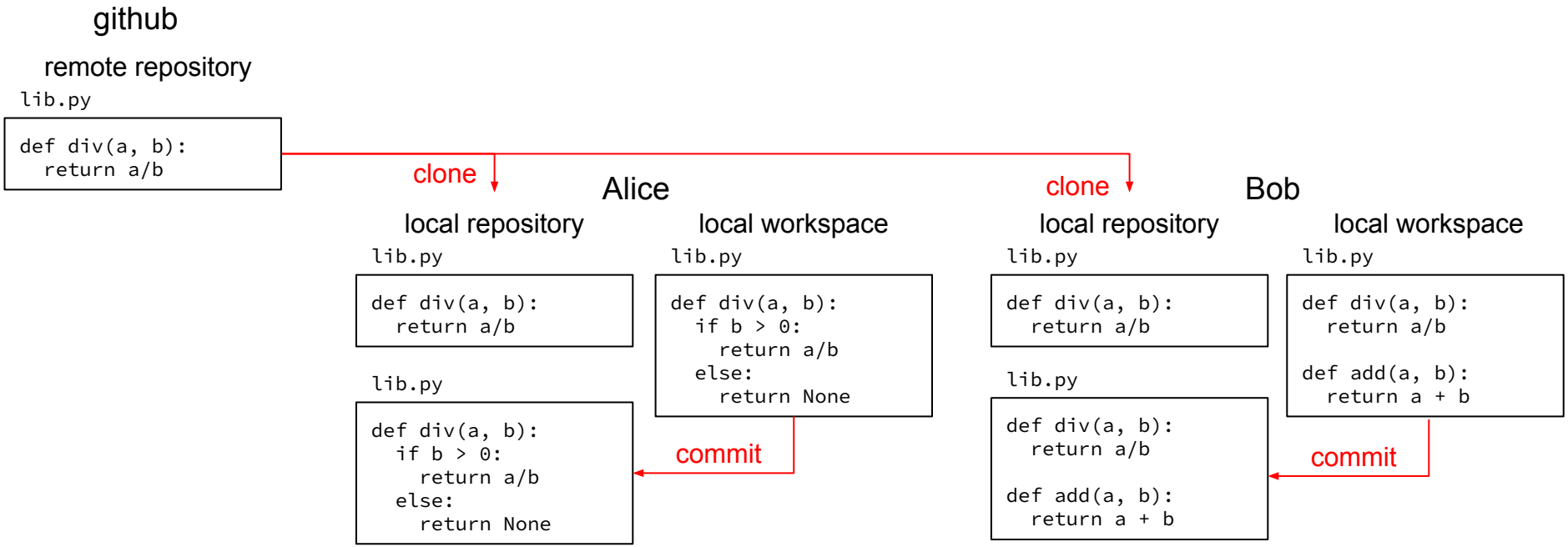
```
def div(a, b):
  return a/b
```

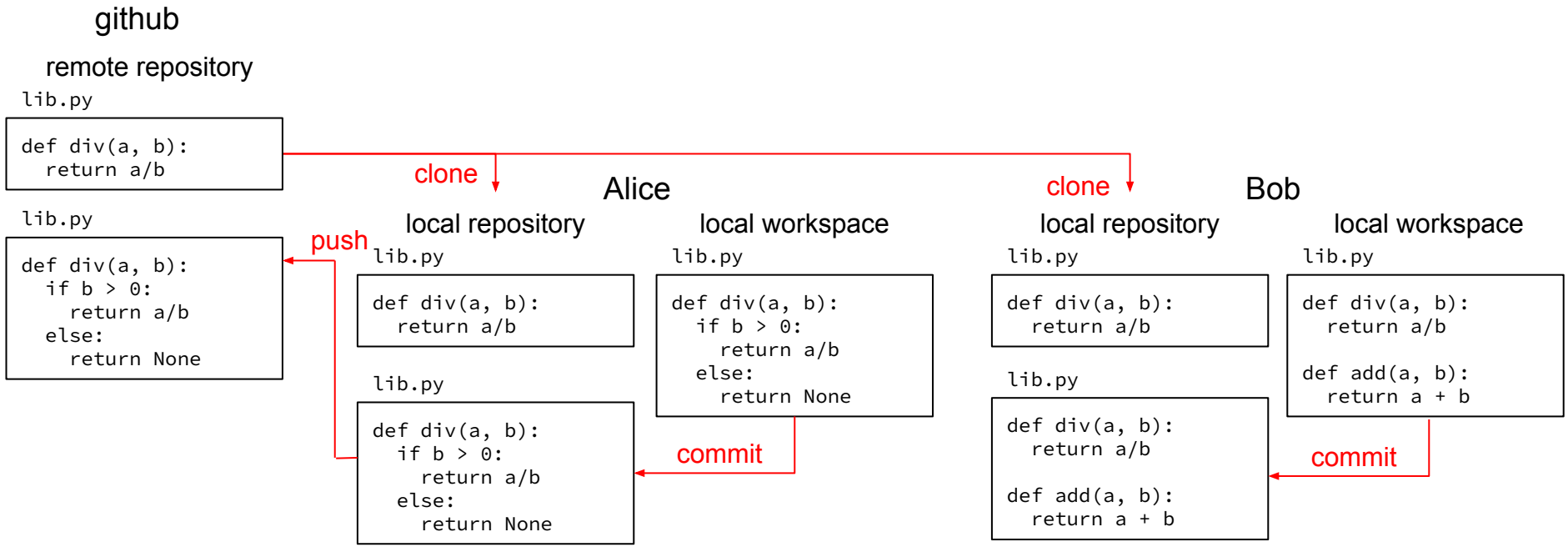

github
remote repository
lib.py
def div(a, b):
return a/b
clone
Alice
local repository
lib.py
def div(a, b):
return a/b

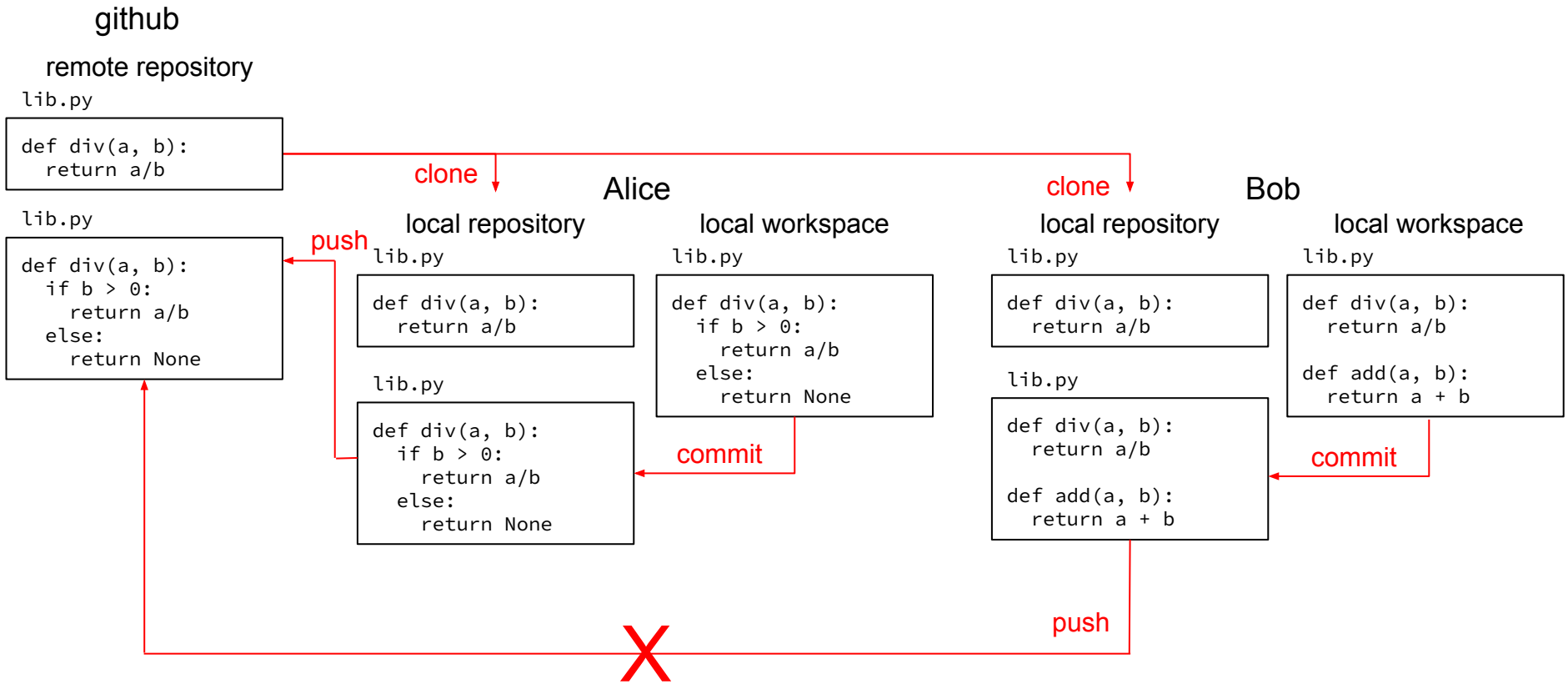

github
remote repository
lib.py
def div(a, b):
return a/b
clone
Alice
local repository
lib.py
def div(a, b):
return a/b
local workspace
lib.py
def div(a, b):
return a/b

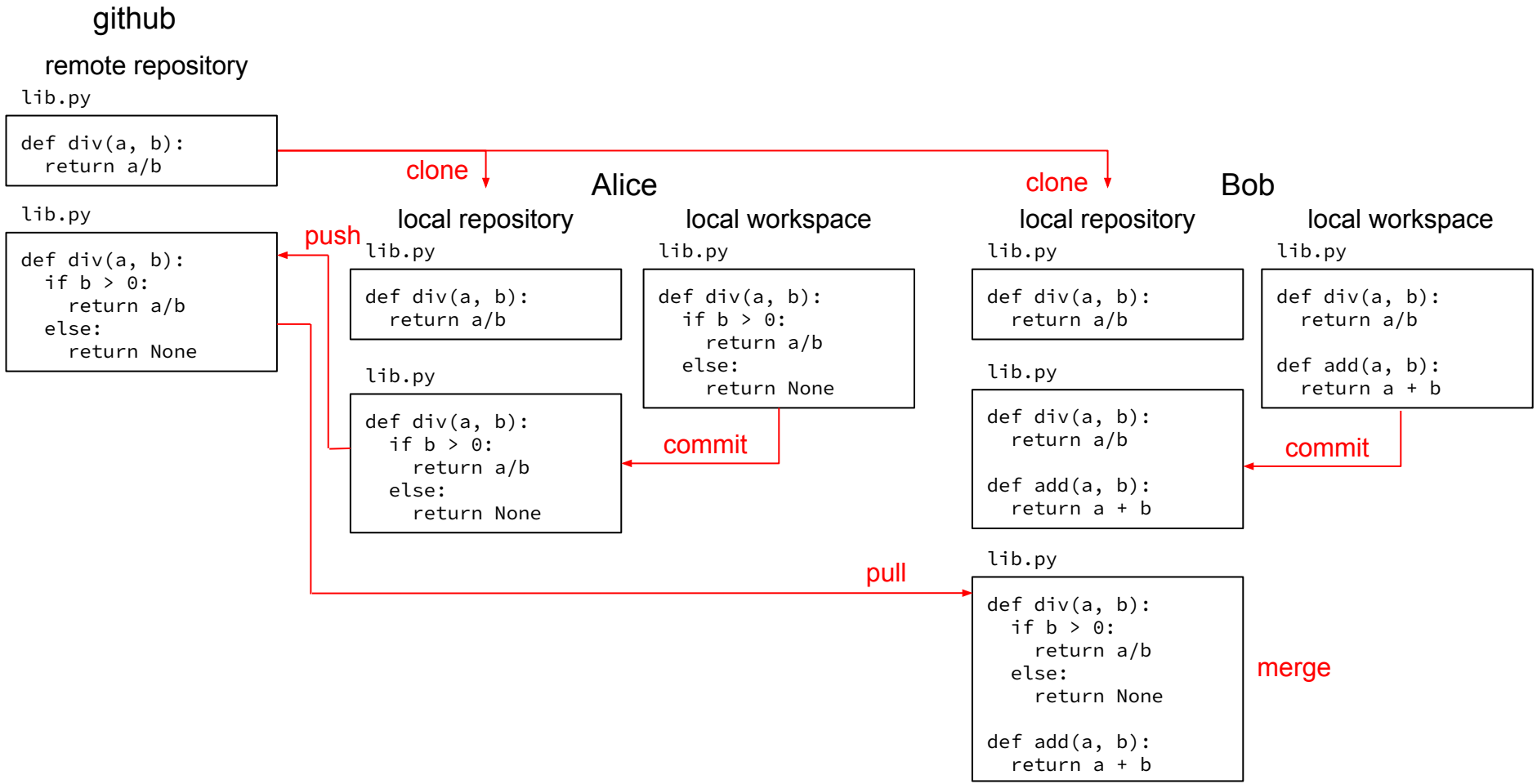

github
remote repository
lib.py
def div(a, b):
return a/b
clone
Alice
local repository
lib.py
def div(a, b):
return a/b
local workspace
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None

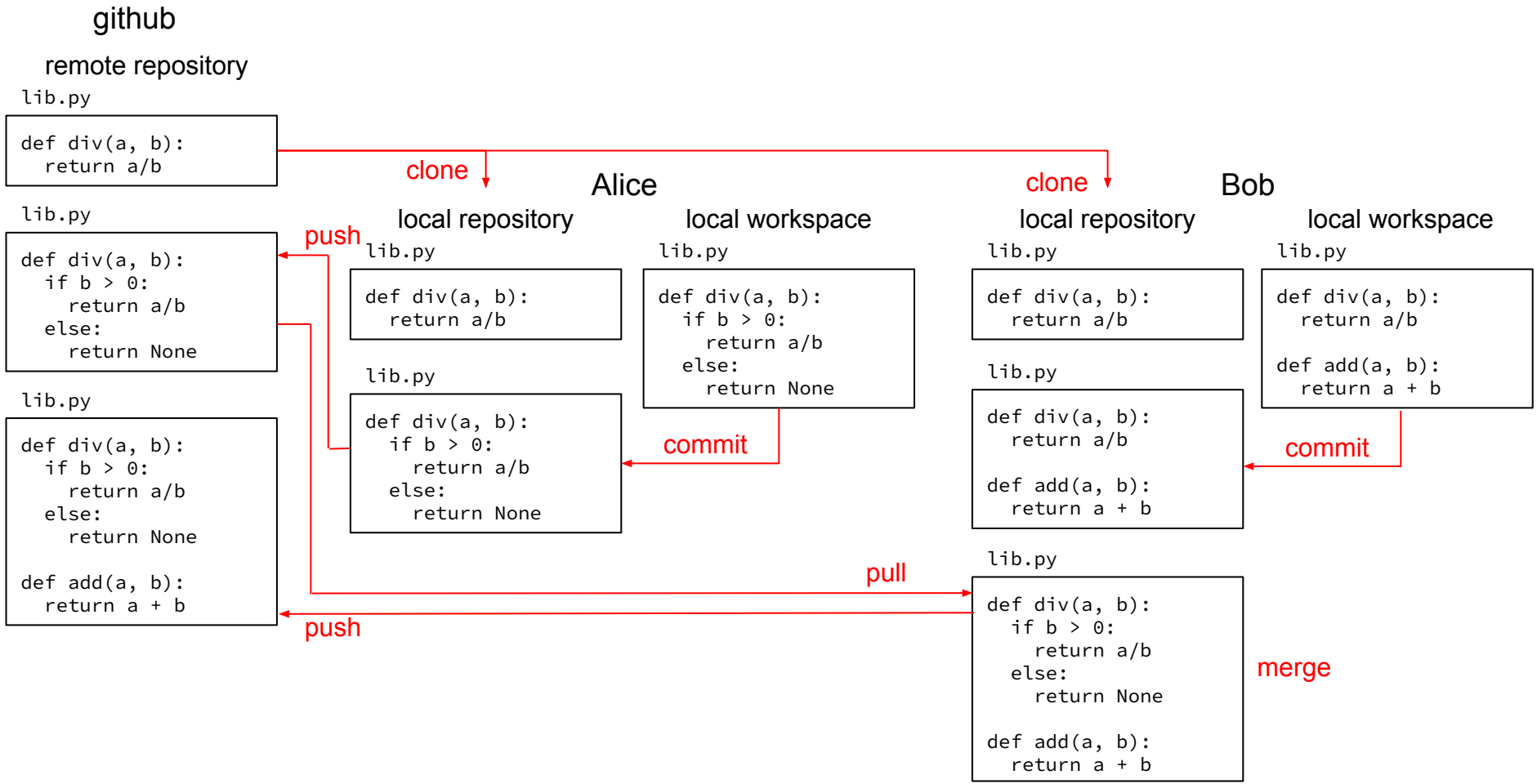

github
remote repository
lib.py
def div(a, b):
return a/b
clone
Alice
local repository
local workspace
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
commit

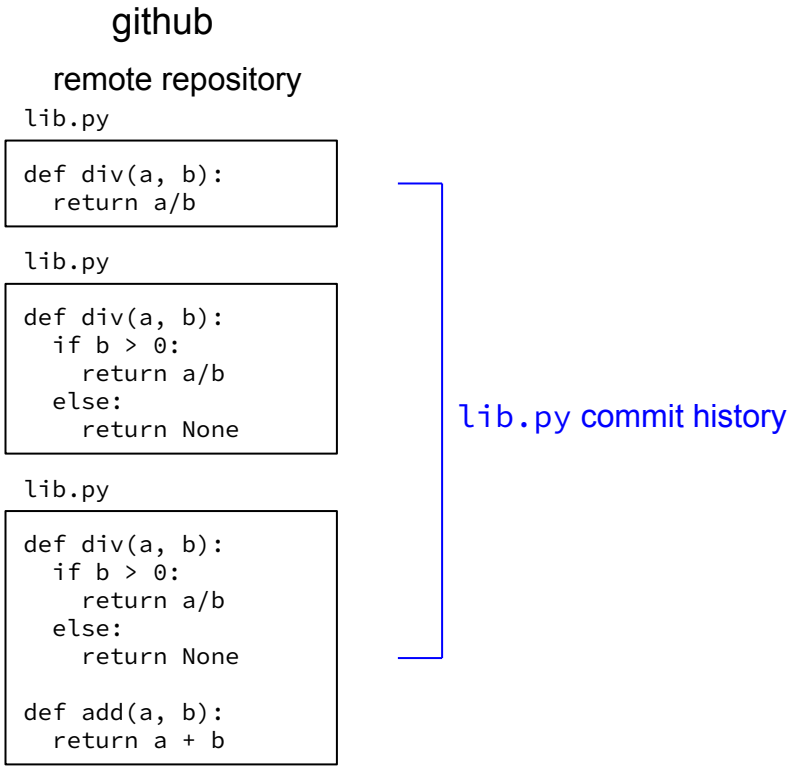

github
remote repository
lib.py
def div(a, b):
return a/b
clone
clone
Alice
Bob
local repository
local workspace
local repository
local workspace
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
commit
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
commit


github
remote repository
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
clone
push
Alice
local repository
local workspace
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
commit
clone
Bob
local repository
local workspace
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
commit


github
remote repository
lib.py
def div(a, b):
return a/b
clone
clone
Alice
Bob
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
push
local repository
local workspace
local repository
local workspace
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
commit
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
commit
push


github
remote repository
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
clone
clone
Alice
Bob
push
local repository
local workspace
local repository
local workspace
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
commit
lib.py
def div(a, b):
return a/b
def add(a, b):
return a + b
commit
pull
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
def add(a, b):
return a + b
merge


github
remote repository
lib.py
def div(a, b):
return a/b
clone
Alice
local repository
local workspace
clone
Bob
local repository
local workspace
push
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
def add(a, b):
return a + b
commit
commit
pull
push
merge


github
remote repository
lib.py
def div(a, b):
return a/b
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
lib.py
def div(a, b):
if b > 0:
return a/b
else:
return None
def add(a, b):
return a + b
lib.py commit history

github

remote repository

lib.py

```
def div(a, b):
  return a/b
```

lib.py

```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None
```

lib.py

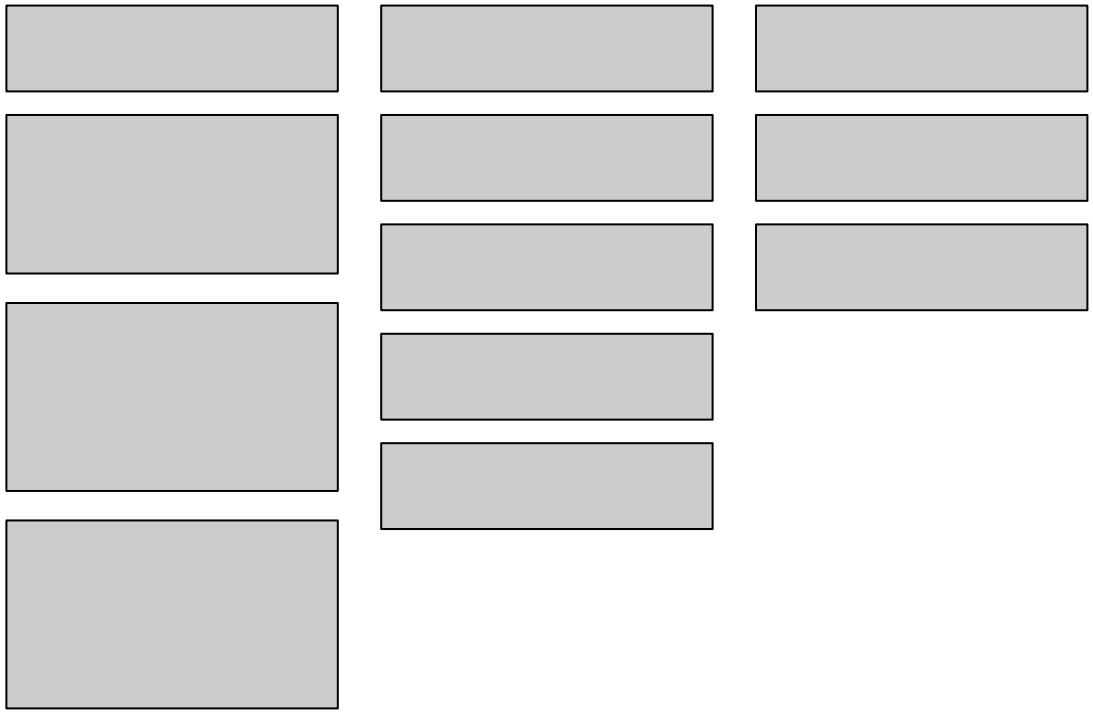
```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None

def add(a, b):
  return a + b
```

setup.py

viz.py

run.py



repo commit history

github

remote repository

lib.py

```
def div(a, b):
  return a/b
```

lib.py

```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None
```

lib.py

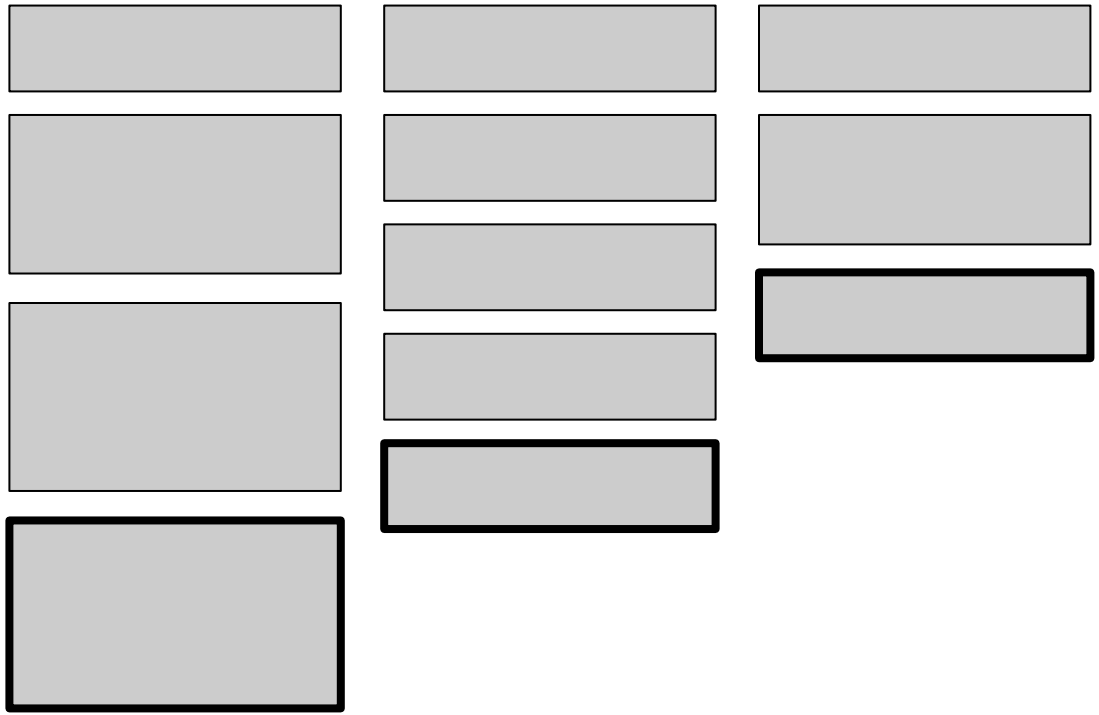
```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None

def add(a, b):
  return a + b
```

setup.py

viz.py

run.py



`$ git clone repo` grabs the most recent commit for every file

DEMO

github

remote repository

lib.py

```
def div(a, b):
  return a/b
```

lib.py

```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None
```

lib.py

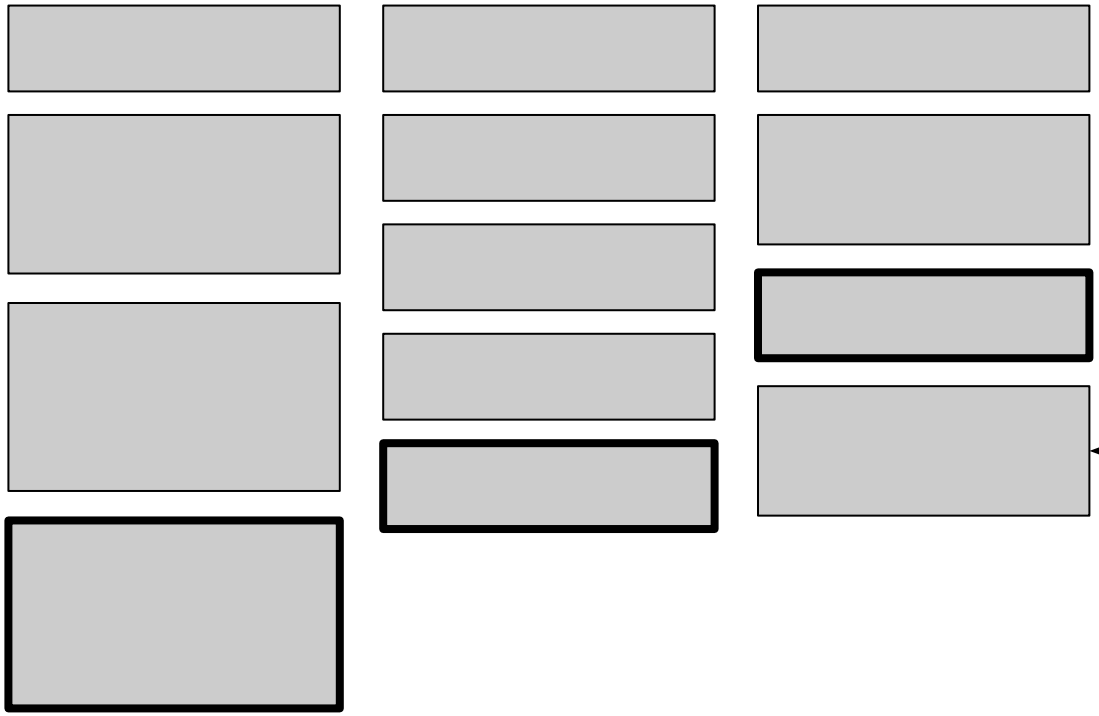
```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None

def add(a, b):
  return a + b
```

setup.py

viz.py

run.py



Recent commits cause stability issues

github

remote repository

lib.py

```
def div(a, b):
  return a/b
```

lib.py

```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None
```

lib.py

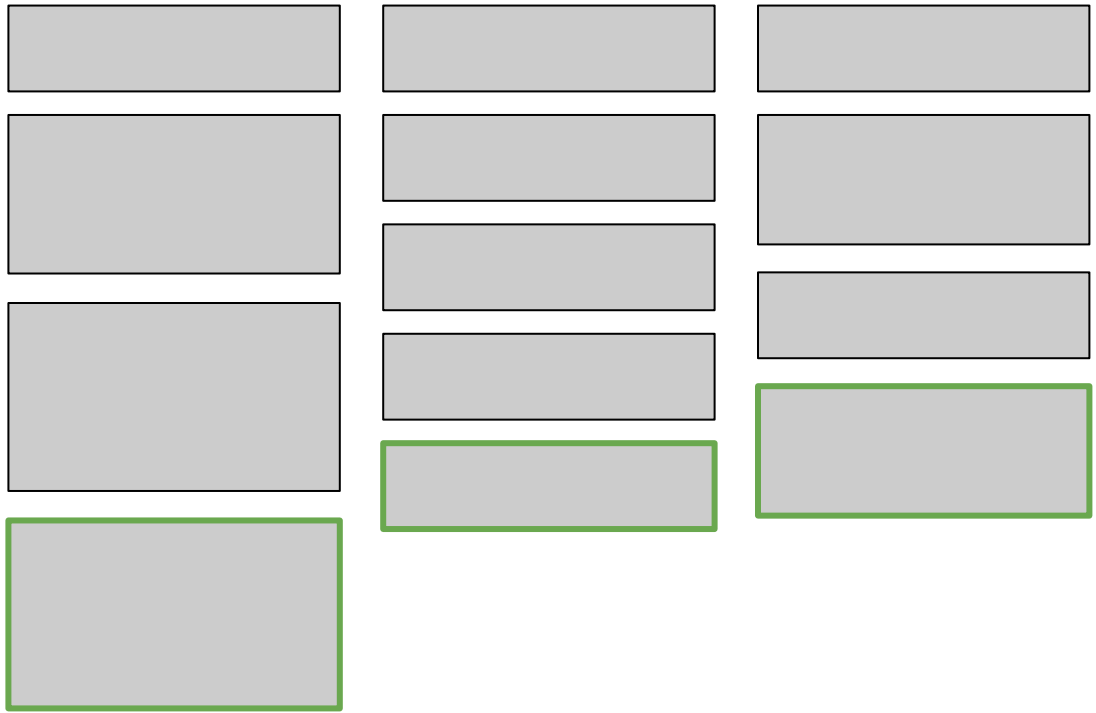
```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None

def add(a, b):
  return a + b
```

setup.py

viz.py

run.py



v1.0

Tag the commit state of the repo to provide consistency as development continues

```
$ git clone repo --branch v1.0
```

github

remote repository

lib.py

```
def div(a, b):
  return a/b
```

lib.py

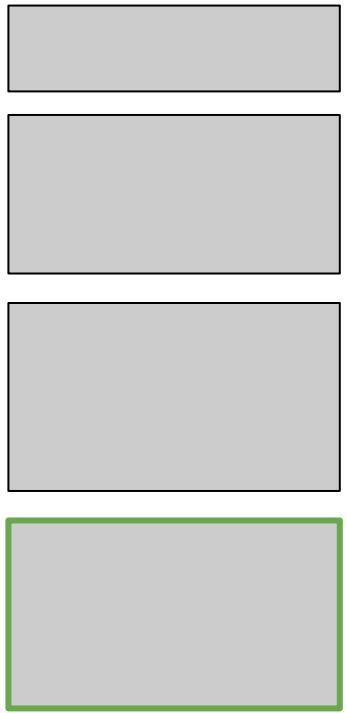
```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None
```

lib.py

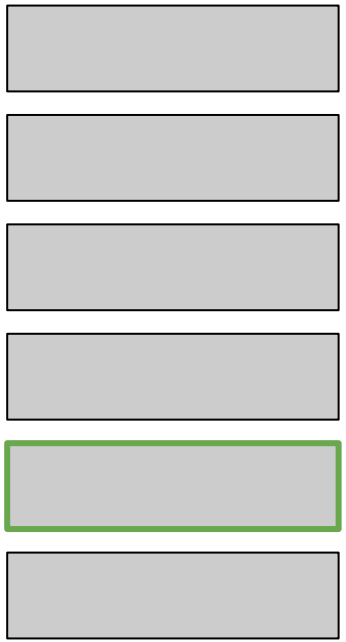
```
def div(a, b):
  if b > 0:
    return a/b
  else:
    return None

def add(a, b):
  return a + b
```

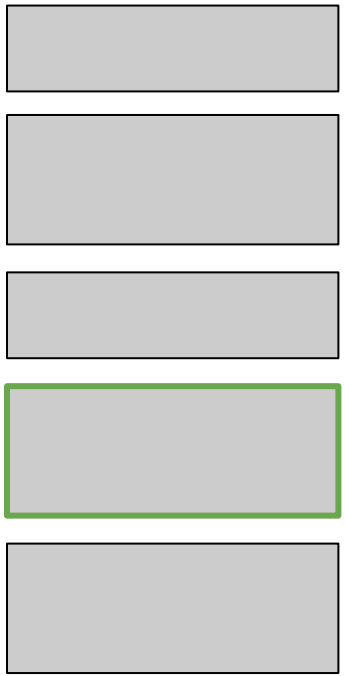
setup.py



viz.py



run.py



```
$ git clone repo --branch v1.0
```

Other stuff you can find in references on course webpage

- Authenticating with GitHub – how GitHub knows you're who you say you are
- git submodules – what if I need to use someone else's repo in my repo?