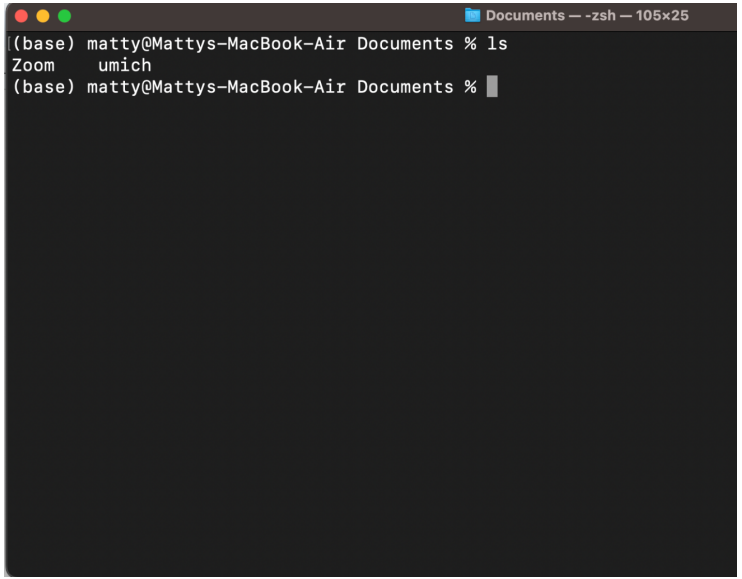
SI 206 Discussion 4:

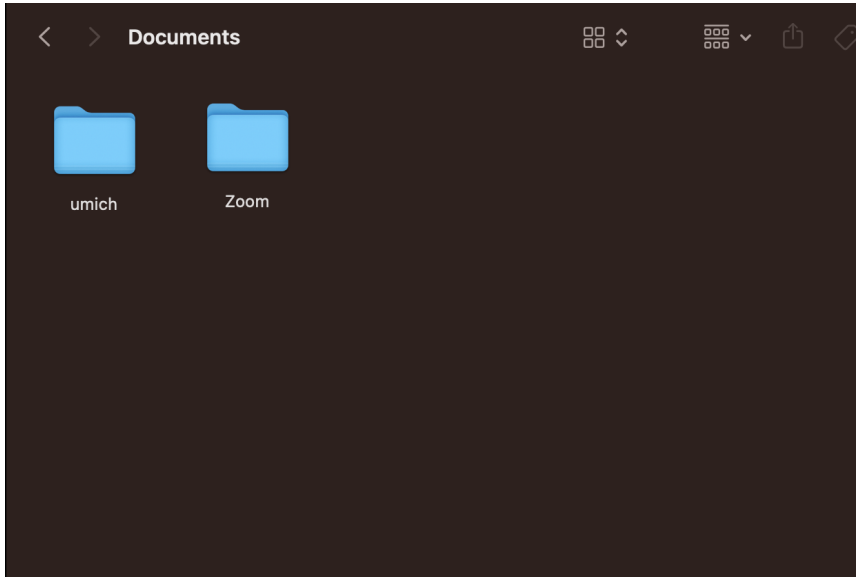
The Terminal, Git, and Rectangles

The Terminal

The Terminal



Command Line Interface (CLI)



Graphical User interface (GUI)

Basic Commands

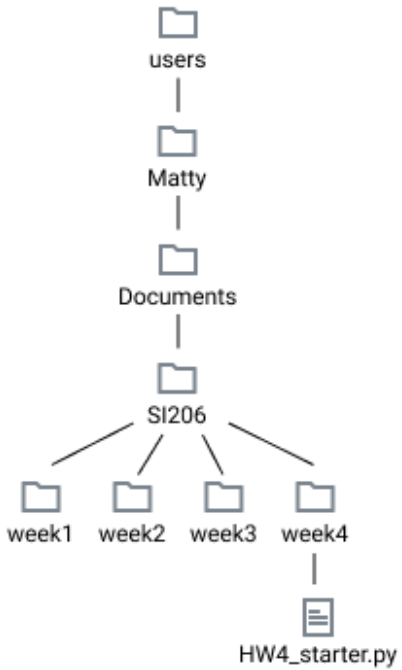
GUI	CLI Command	Example
* current folder	pwd	pwd
* display folder contents	ls	ls
navigate/ change location	cd	cd SI206
make a new folder	mkdir	mkdir my_new_folder

N.B.
GUI = “folder”
CLI = “directory”

Paths

Special Characters:
current directory = .
parent directory = ..
home directory = ~
root directory = /

cd takes a *path* as an argument of which there are two kinds:



Relative path

If you are in "SI206"
cd week4

If you are in "Matty"
cd Documents/SI206/week4

If you are in “week3”
cd ../week4

Absolute path

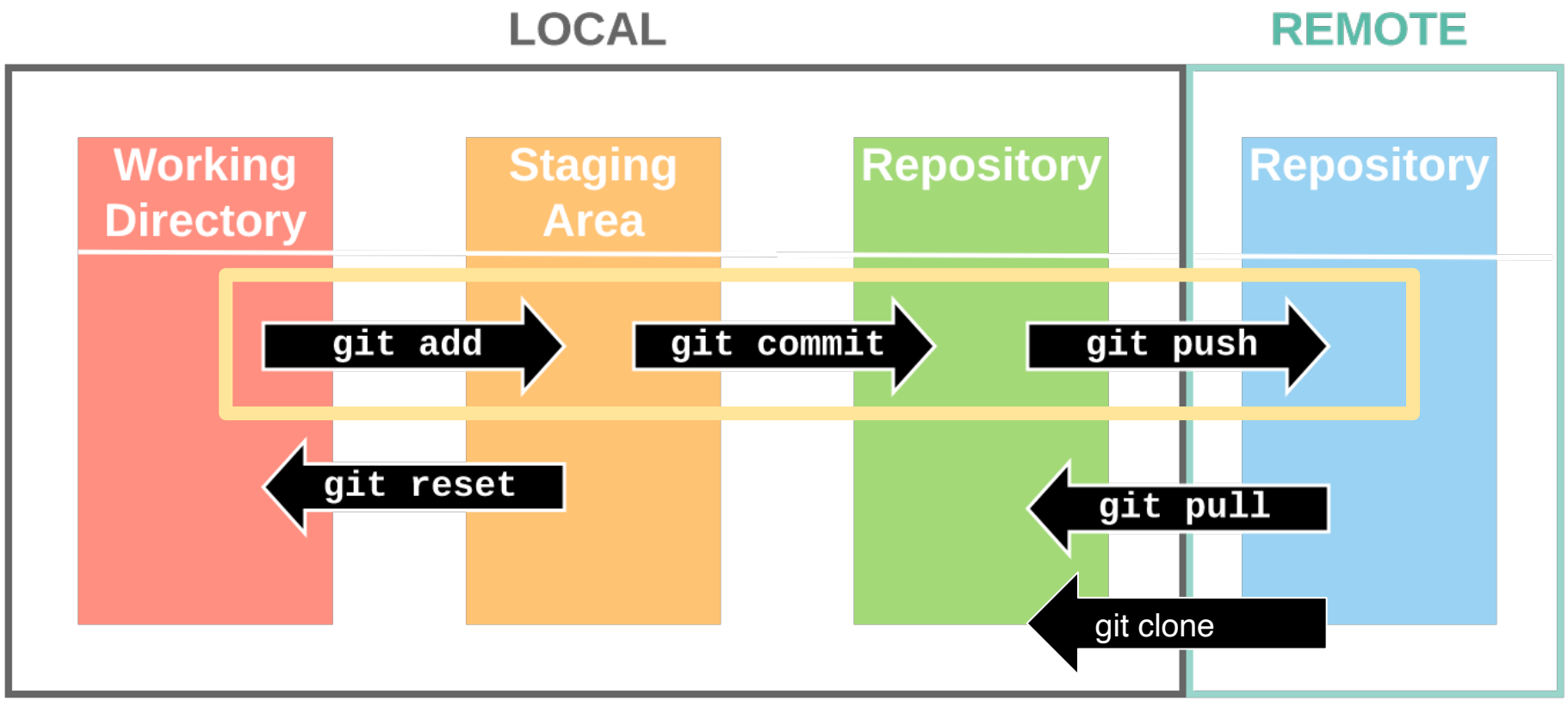
Path from root directory
cd /users/Matty/Documents/SI206/week4

Path from home directory
cd ~/Documents/SI206/week4

Command's Purpose	MS-DOS	Linux	Basic Linux Example
Copies files	`copy`	`cp`	`cp` *`thisfile.txt`* `/home/`*`thisdirectory`*
Moves files	`move`	`mv`	`mv` *`thisfile.txt`* `/home/`*`thisdirectory`*
Lists files	`dir`	`ls`	`ls`
Clears screen	`cls`	`clear`	`clear`
Closes shell prompt	`exit`	`exit`	`exit`
Displays or sets date	`date`	`date`	`date`
Deletes files	`del`	`rm`	`rm` *`thisfile.txt`*
"Echoes" output to the screen	`echo`	`echo`	`echo` *`this message`*
Edits text files	`edit`	`gedit`([a])	`gedit` *`thisfile.txt`*
Compares the contents of files	`fc`	`diff`	`diff` *`file1 file2`*
Finds a string of text in a file	`find`	`grep`	`grep` *`word or phrase`* *`thisfile.txt`*

Command's Purpose	MS-DOS	Linux	Basic Linux Example
Formats a diskette	`format a:` (if diskette is in `A:`)	`mke2fs`	`/sbin/mke2fs /dev/fd0` (`/dev/fd0` is the Linux equivalent of `A:`)
Displays command help	*`command /?`*	`man` or `info`	`man` *`command`*
Creates a directory	`mkdir`	`mkdir`	`mkdir` *`directory`*
Views contents of a file	`more`	`less`([b])	`less` *`thisfile.txt`*
Renames a file	`ren`	`mv`([c])	`mv` *`thisfile.txt thatfile.txt`*
Displays your location in the file system	`chdir`	`pwd`	`pwd`
Changes directories with a specified path (*absolute path*)	`cd` *`pathname`*	`cd` *`pathname`*	`cd` */directory/directory*
Changes directories with a *relative path*	`cd..`	`cd ..`	`cd ..`
Displays the time	`time`	`date`	`date`
Shows amount of RAM in use	`mem`	`free`	`free`

Git & Github


LOCAL
REMOTE
Working Directory
Staging Area
Repository
Repository
git add
git commit
git push
git reset
git pull
git clone

Typical Git Flow

1. **git clone** <link>
2. **git add** <file(s) you are modifying>
3. make your changes
4. **git commit -m** <message>
5. **git push**

use **git status** before, after, and throughout to keep track

Good Commit Messages

Commit message should give a short description of what changes you made

Writing good git commits is an important part of writing clear, professional code

This will also help you if you need to reference an earlier version of your code

Good Commit Messages

Bad Commit Messages

- “Fixed stuff”
- “Added code”

Better Commit Messages

- “Fixed bug in draw function”
- “Implemented shape class”

Time to Practice!

Practice

1. Object oriented programing
    a. Create a rectangle class and methods to calculate the area and perimeter.
    b. Create the rectangle instances, and call the methods

1. Git : Commit code after each method and push to GitHub in the end
    a. Please commit at least 4 times while working on your project; you might commit each time you finish writing a new a function or method.

Discussion 4 Assignment

Accept the github classroom assignment and clone the repo

https://classroom.github.com/a/zJlXkgCQ

if you are having issues:

Files -> Discussions ->

Discussion 4 -> discussion4_starter.py

Rectangle class

Problem 1. Create the constructor "__**init**__" method
with arguments **width** (an integer), **height** (an integer)

(1) It sets an instance variable, "**width**" to the passed argument, width
(2) It sets an instance variable, "**height**" to the passed argument, height

Problem 2. Create the "__**str**__" method
It returns a string, "**A rectangle with width ____ and height ____**"

Rectangle class

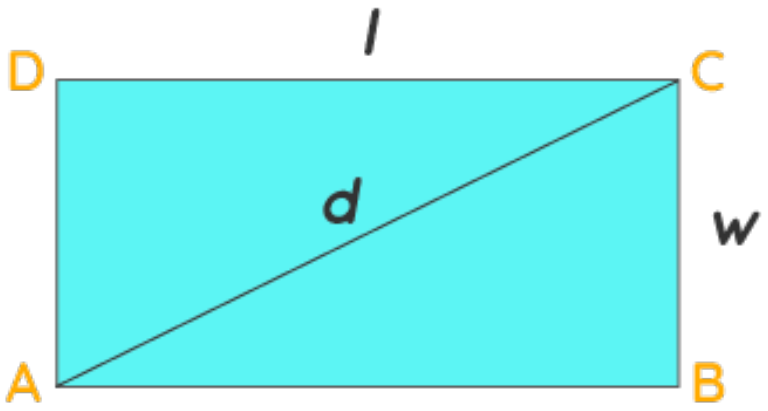
Problem 3. Create the “**describe shape**" method
It returns a string
(1) Describes the shape of the rectangle

Problem 4. Create the “**calculate_area**" method
(1) Returns the area of the rectangle.

Problem 5. Create the “**calculate_diagonal**" method
(1) Returns the length of the rectangle’s diagonal
(2) Hint: We have imported the math module for you in the starter code. Use the math module’s built in

Diagonal Formula

The formula is also included in the starter code



Diagonal of a rectangle, $d = \sqrt{(l^2 + w^2)}$

Sample output

```python
def main():
    r = Rectangle(10, 10)
    print(r)
    print("Shape Description:", r.describe_shape())
    print("Area:", r.calculate_area())
    print("Diagnol Length:", r.calculate_diagnol_length())
    print()
```

```
A rectangle with width 10 and height 10
Shape Description: width and height are equal, it is a square
Area: 100
Diagnol Length: 14.142135623730951
```