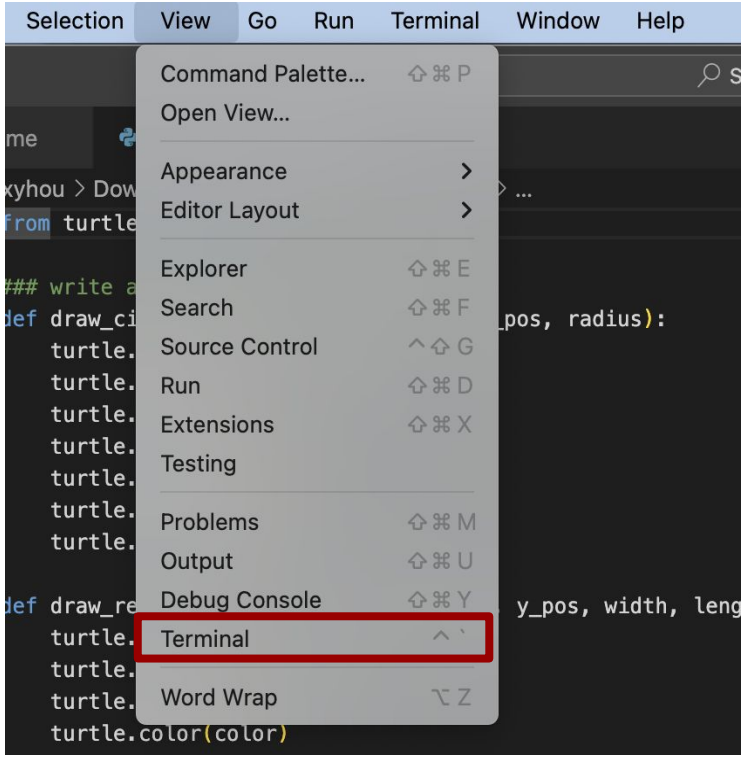
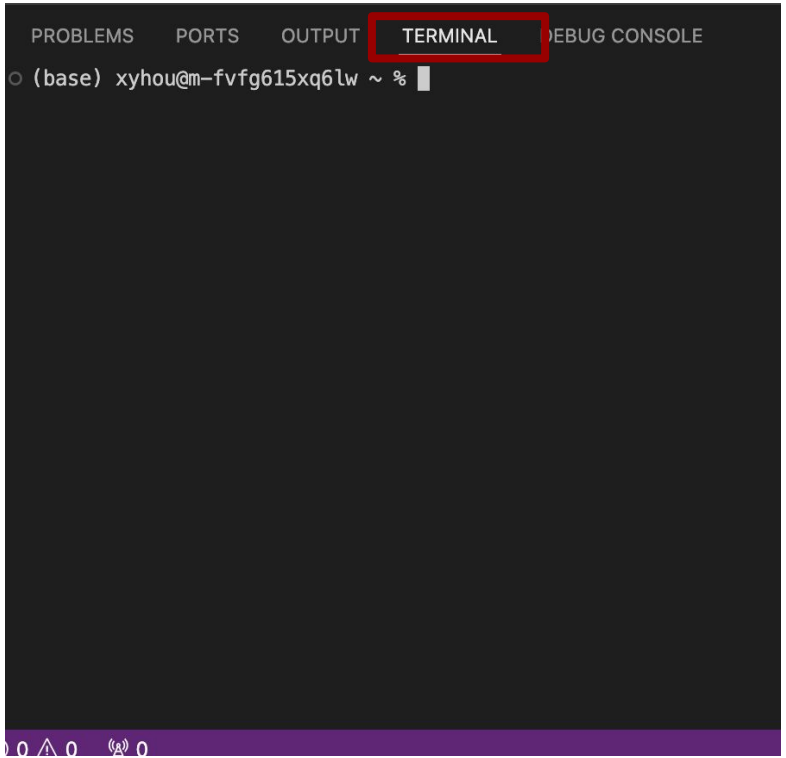
SI 206 Discussion 4:

The Terminal, Git, and Rectangles

The Terminal

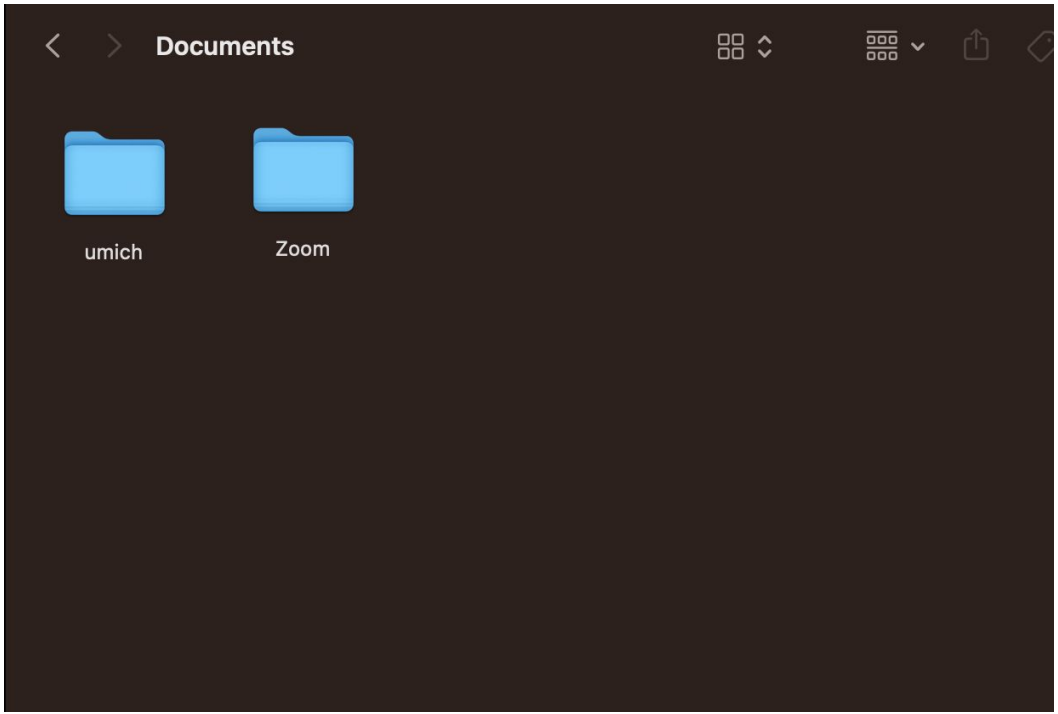
The Terminal - Command Line Interface (CLI)





VS Code Integrated Terminal

The Terminal - Graphical User interface (GUI)



Git & Github

Typical Git Flow - will need it for future assignments

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

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

2. Git : Commit code after each method and push to GitHub in the end
    c. 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.

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 <width> and height <height>`**"
for example, "**`A rectangle with width 3 and height 6`**"

Rectangle class

Problem 3. Create the "**`calculate_area`**" method

It returns the area of the rectangle (float)

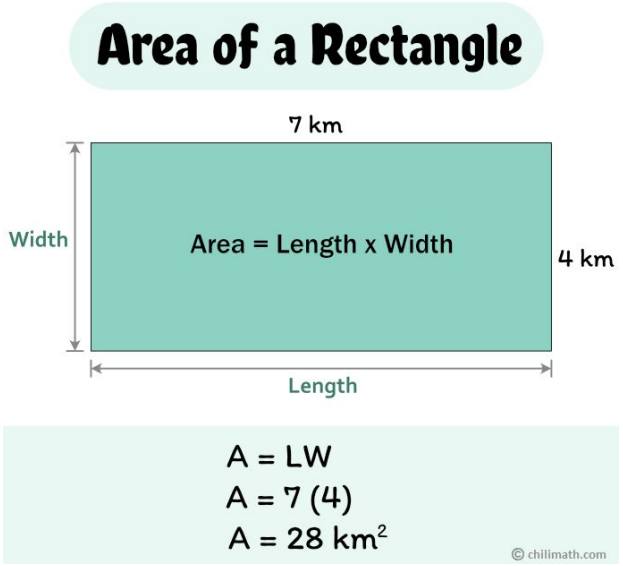
Area of rectangle = length × width.

Problem 4. Create the "**`__eq__`**" method

It returns a boolean

`True` if the two rectangles have the same width and the same height

`False` otherwise

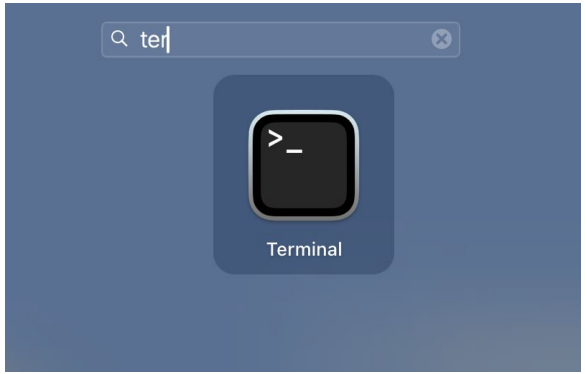


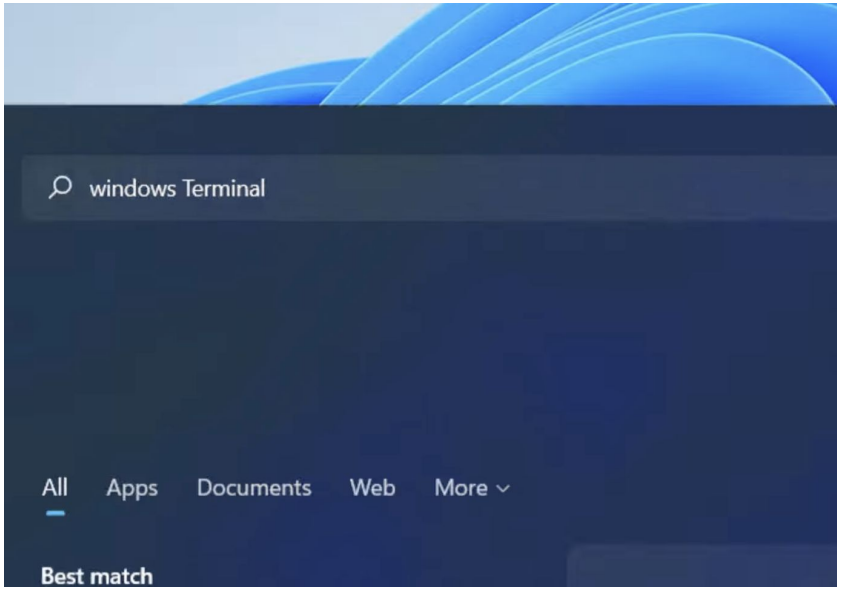
Sample output

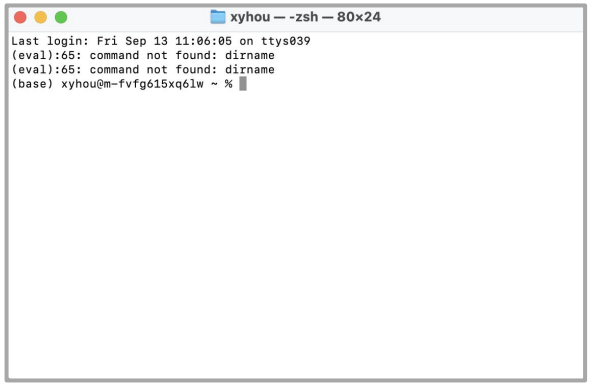
```
r1: A rectangle with width 10 and height 10
Area: 100

r2: A rectangle with width 10 and height 15
Area: 150
Equal: r1 == r2? False

r3: A rectangle with width 10 and height 15
Area: 150
Equal: r2 == r3? True
```

Additional Tips about Git & Terminal

The Terminal - Command Line Interface (CLI)







Laptop Terminal

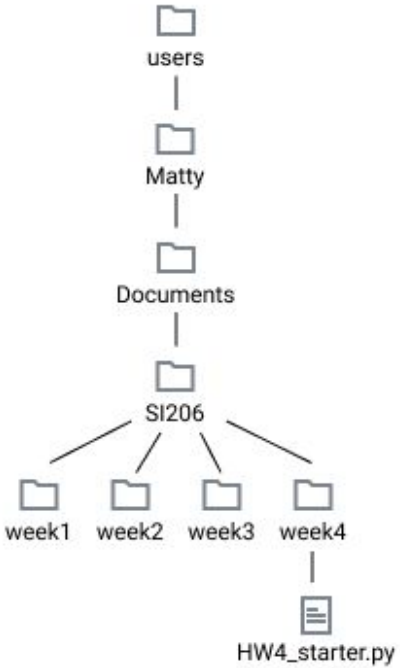
Basic Commands - Try it on your terminal!

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

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

Paths

`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`

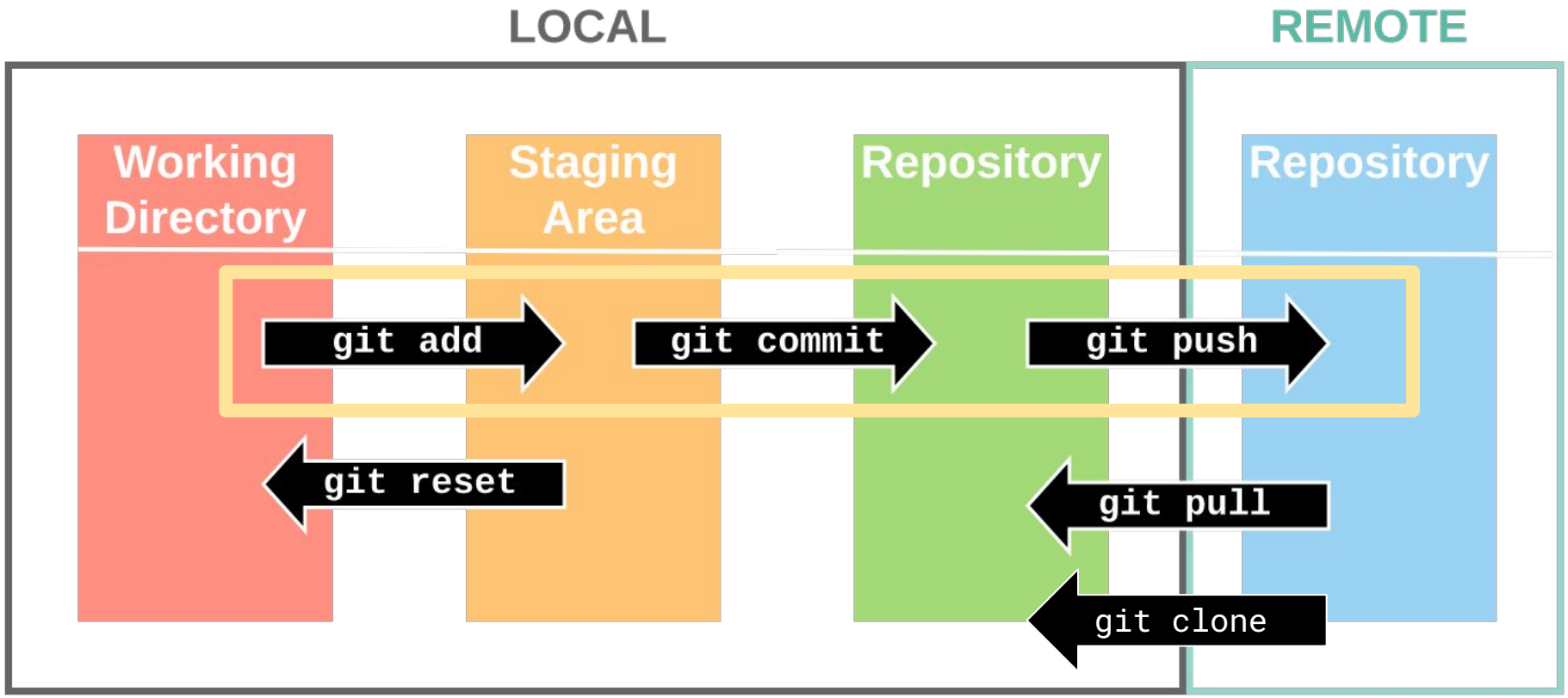
Absolute path

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`


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