CSCi 1012 [Section 10]

Introduction to Programming with Python

Prof. Kartik Bulusu, CS Dept.

Course start date January 17, 2024
Lecture location 1957 E street Room 213
Lecture times Monday, 3:45 PM to 5:00 PM

Wednesday-lab
3:45 PM to 5:00 PM
Section-30: MON 352
Section-31: SEH 4040
Section-34: TOMP 310
Section-35: TOMP 204

Friday-lab
3:45 PM to 5:00 PM
Section-32: SEH 4040
Section-33: TOMP 309
Section-36: TOMP 306
Section-37: TOMP 107

GW School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

Spring 2024

Photo: Kartik Bulusu

Class Policy on Collaboration

- **You may *not* *discuss Modules, Assignments, Quizzes and Exams* among yourselves.**
- **Each student is expected to work out the course deliverables *independently*.**
- **Under *no circumstances* may you look at another student's *Modules, Assignments, Quizzes and Exams*, or look for answers to *Modules, Assignments, Quizzes and Exams* anywhere other than in the text in the course website.**
- **You are encouraged to discuss the class material on Ed-discussion board or in-person with the instruction team.**
- **You may *not* discuss *Modules, Assignments, Quizzes and Exams* nor give out hints for the same on problems on the Ed-discussion board or with other students in-person.**

All violations will be treated as violations of the Code of Academic Integrity.

Announcements

- **Please contact the offices concerned if you need any special accommodations**
- **We will support your request; you will need to contact them soon**
- **Office that supports you request will guide you on all special requests**

- **Students completed exam on Monday, 04/15 during the lecture.**
  - Grades will be released tonight

- **Remaining weeks of the course:**
  - **Make-up Quiz 2** on 04/22
  - **All Labs until last lecture will have regular quizzes**
  - **Last week of labs:** 04/24 (Wednesday) and 04/26 (Friday)
  - **Last lecture for the course:** 04/29
  - **Assignment 4: Only do this if you need it**

- **Frequency of announcements will be higher:**
  - Please keep track of all announcements made on the course-webpage and Ed.
  - Please respond to all emails sent by the staff.

- **Make-up Exam on Monday, 05/06 (exam week)**
  - Eligible students will be contacted directly
  - Location: 1957 E street Room 213
  - Duration: 50 minutes; Start time: 5:20 PM
  - Paper exam
  - Closed everything (no notes, no Thonny)
  - Multiple choice/ fill in the blanks/ write the output – similar to quizzes
  - 20 questions (no choice, 5 points each)
  - Total: 100 points
  - 60 % to Pass the Make-up exam and the course
  - Final grade (with grade scale) will not be higher than the lowest grade of the students who passed the exam on 04/15
- **If you didn't appear for the Exam on 04/15, you will NOT be permitted to take the Make-up Exam**

HWs
- Due dates
- Late work
- Extensions

- **48-hour extensions:**
  - After the 4th extension your subsequent late submissions will not be considered toward final grade.

Date	Topic(s)	Wednesday Lab Date	Friday Lab Date	Assignment(s)
Week 13 [04/15/2024]	**Examination** *Location:* **Lecture room 1957 E Room 213** *Start time:* **3:45 PM**	04/17/2024	04/19/2024	Unit 2 » Module 0 (Due **April 25, 2024** by **11:59 PM**)
Week 14 [04/22/2024]	Advanced Topics **Tuples, Sets & Dictionaries**	04/24/2024	04/26/2024	Unit 2 » Module 1 & Assignemnt 4 (Due **May 01, 2024** by **11:59 PM**)

CRN	Subj	Crse	Section	Building	Room		Instructor
93983	**CSCI**	**1012**	**10**	1957 E	213	**Make-up** Exam Date **Duration: 50 minutes** **Eligible students will be contacted directly** Monday, May 6, 2024 5:20pm-7:20pm	Bulusu

Late Work

- **Late work is not accepted, with the following exceptions:**
  - Every student many turn in **as many as four (in total, not each) assignments or modules 48 hours after the deadline with no penalty**. Requesting an extension is not necessary.
- **Extensions** will be **granted should there arise circumstances beyond your control** that impede your ability to complete coursework.
  - Notify your professor as soon as feasible in these cases.
    - Examples of such circumstances include (but are not limited to) illness, death in the family, and loss of housing. To ensure fairness toward all students, we will request documentation of such circumstances.

Submission Tips

- Pay attention to file names

extra spaces

assignment1.zip ≠ ***assignment 1.zip***

caesar_shift.py ≠ ***caesar_shift .py***

extra spaces

- Pay attention to small details

```
x is 3 and j is 2, x + j = 5
x is 3 and j is 3, x + j = 6
x is 3 and j is 4, x + j = 7
x is 3 and j is 5, x + j = 8
x is 3 and j is 6, x + j = 9
x is 3 and j is 7, x + j = 10
x is 3 and j is 8, x + j = 11
```

≠

missing commas

```
x is 3 and j is  2 x + j 5
x is 3 and j is  3 x + j 6
x is 3 and j is  4 x + j 7
x is 3 and j is  5 x + j 8
x is 3 and j is  6 x + j 9
x is 3 and j is  7 x + j 10
x is 3 and j is  8 x + j 11
```

extra space *missing =*

- Don't add extra endline spaces

print("++++", end = "") →

```
++++
+  +
+  +
++++
```

→ ***What Auto Grader Sees…***

'++++\n+ +\n+ +\n+ +\n++++'

≠

print("++++", end = " ") →

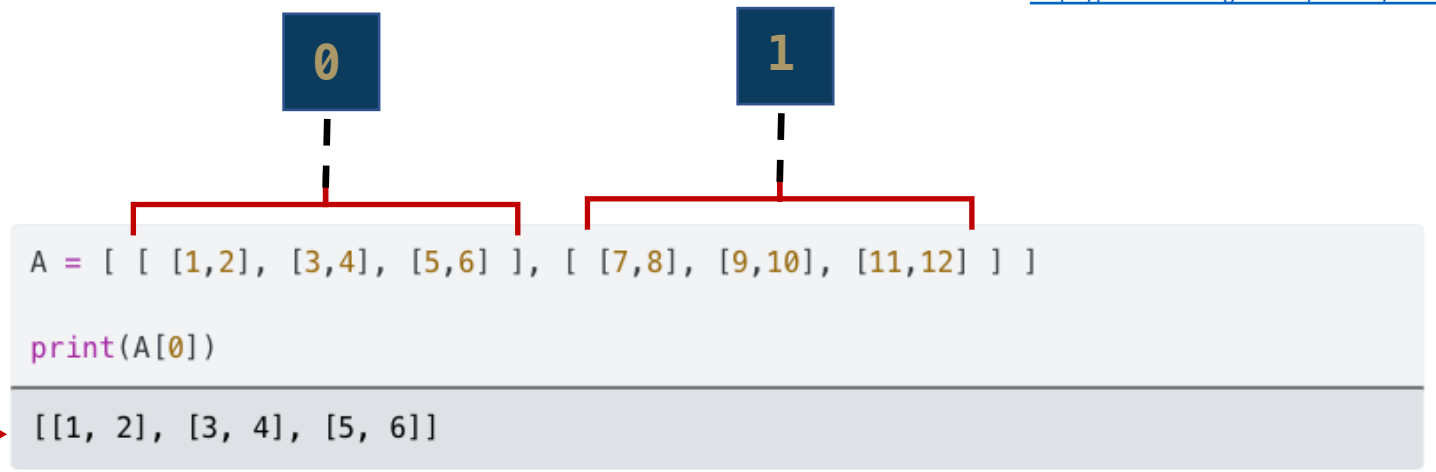
```
++++
+  +
+  +
++++
```

→ '++++ \n+ +\n+ +\n+ +\n++++ '

extra spaces

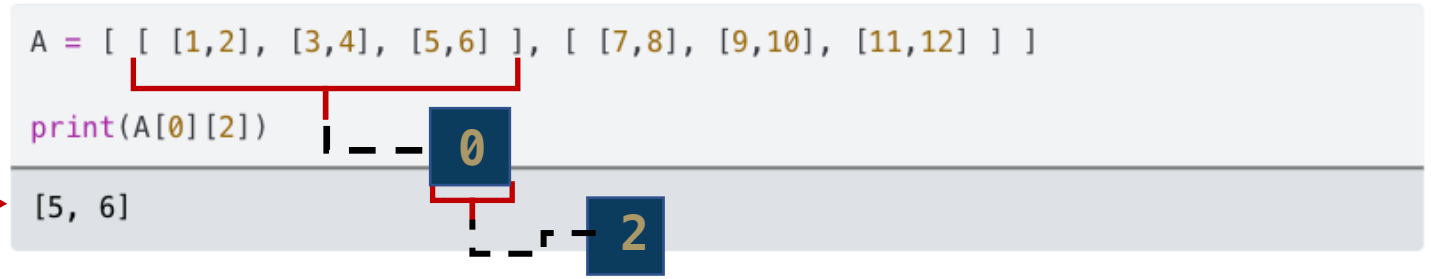
Sources:
https://www2.seas.gwu.edu/~cs4all/1012/unit2/module2.0.html

List of lists of lists

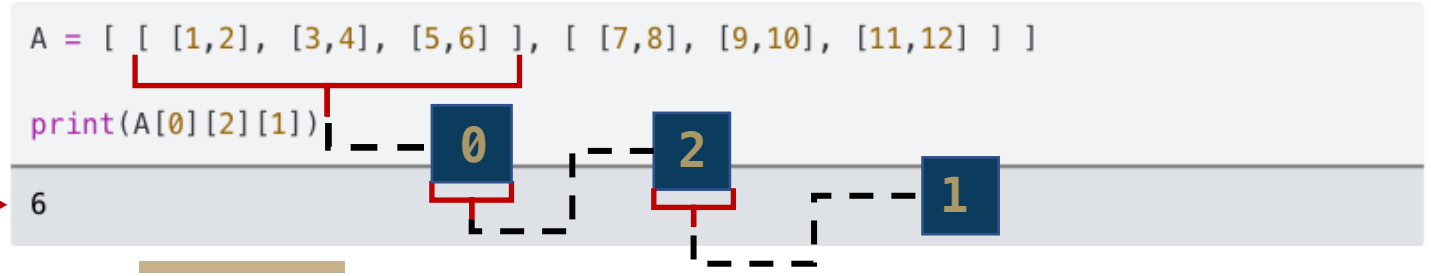


One needs to keep track of the square brackets !

Get the third element of the outermost element:



Get the second element of the third element of the outermost element:



Demo

Sources:
https://numpy.org
https://www2.seas.gwu.edu/~cs4all/1012/unit2/module2.0.html
https://numpy.org/doc/stable/user/absolute_beginners.html

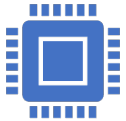
NumPy

```
import numpy as np
```

```
>>> import numpy as np
>>> a = np.array([1, 2, 3])
```

NumPy (**Numerical Python**) is an open source Python library that's used in almost every field of science and engineering.

Very large lists (million of elements or more) can slow down a program.



A list-of-lists is even slower for large sizes, and takes up a lot of memory.

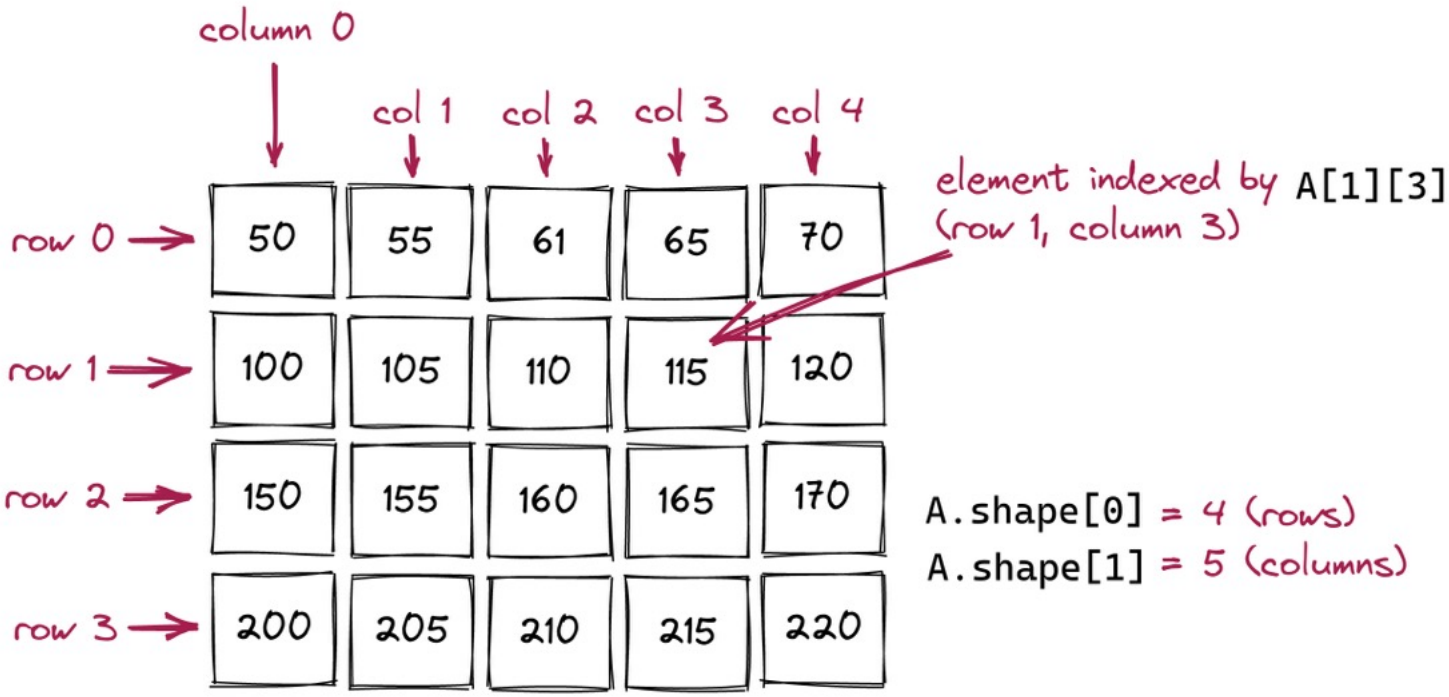
NumPy arrays (we will often call them simply "arrays") were created as a separate structure in Python to enable efficient processing of lists of numbers, especially multidimensional lists.

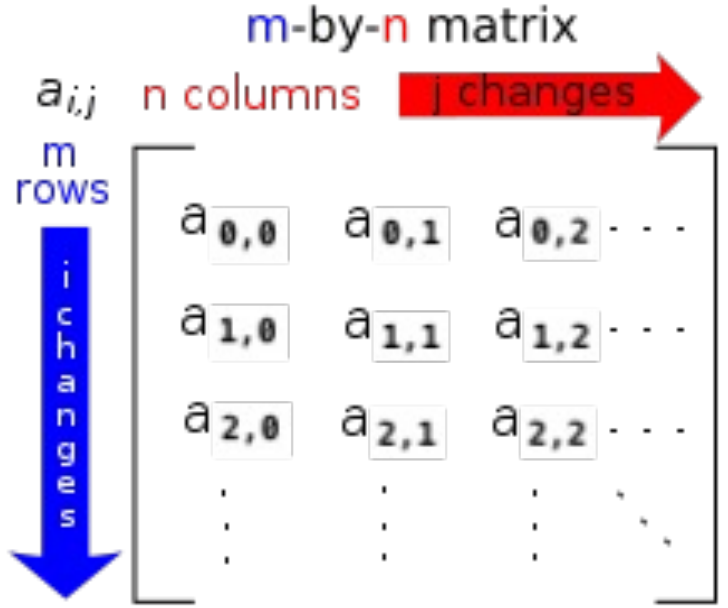
```
>>> import numpy as np
>>> A = np.array([[-1, 2],[3, 4]])
>>> B1 = A * 6
>>> B2 = A * (1/6)
>>> len(B1)
>>> np.shape(B2)
```

Source:
https://www2.seas.gwu.edu/~cs4all/1012/unit2/module2.0.html#objectives

Bookkeeping and slicing in 2D numpy arrays

```
some_numpy_array[row-0:row-M, column0:columnN]
```



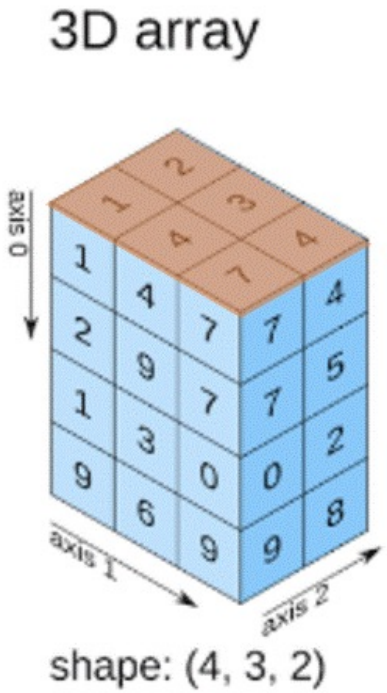


Source: http://en.wikipedia.org/wiki/Matrix_(mathematics)

Demo

Source:
https://www2.seas.gwu.edu/~cs4all/1012/unit2/module2.0.html#objectives
https://stackoverflow.com/questions/71640419/min-numpy-3d-array

Bookkeeping and slicing in 3D numpy arrays – Example of RGB images



- Each color is therefore a group of three numbers, for example:
  - `(255,0,0)` is all red (no green, no blue)
  - `(0,255,0)` is all green (no red, no blue)
  - `(0,0,255)` is all blue (no red, no green)
- Let's try a few more:
  - `(255,255,0)`
  - `(100,255,255)`
  - `(200,200,200)` (gray is R,G,B all equal)
  - `(0,0,0)`
  - `(255,255,255)` is white

Demo

Lists

Mutable

- Add
- Remove
- Change data

```
>>> A = [1, 2, 3, 4]
>>> type(A)
<class 'list'>
```

Data surrounded by square brackets

```
>>> print('# of list data points = ', len(A))
  # of list data points =  4
```

```
>>> for n in A:
        print("list data: ", n, end=" ")

  list data:  1 list data:  2 list data:  3 list data:  4
```

```
>>> print(dir(A))
```

Tuples

Immutable

Cannot be changed

```
>>> a = (1, 2, 3, 4)
>>> type(a)
<class 'tuple'>
```

Data surrounded by parenthesis

```
>>> print('# of tuple data points = ', len(a))
  # of tuple data points =  4
```

```
>>> for p in a:
        print("tuple data: ", p, end=" ")

  tuple data:  1 tuple data:  2 tuple data:  3 tuple data:  4
```

```
>>> print(dir(a))
```

lists have more methods in place than tuples

lists occupy more memory than tuples

Demo

Examples

```
def powers(x):
    square = x*x
    cube = square*x
    fourth = cube*x
    fifth = fourth*x
    return (square, cube, fourth, fifth)  # returns four grouped variables

x = 5
(a, b, c, d) = powers(x) # expects four grouped variables from the function
print(x, a, b, c, d)
```

```
def square_cube_list(x):
    square = x*x
    cube = square*x
    return [square, cube]

x = 5
[y, z] = square_cube_list(x)
print(x, y, z)
```

```
def square_cube(x)
    square = x*x
    cube = square*x
    return (square, cube)

x = 5
(y, z) = do_both(x)
print(x, y, z)
```

```
# List version:
L = square_cube_list(x)
print(L[0], L[1])      # L[0] has the square, L[1] has the cube

# Tuple version:
t = square_cube(x)
print(t[0], t[1])      # t[0] has the square, t[1] has the cube
```

```
# List version:
L = square_cube_list(x)
L[0] = 0               # This is allowed

# Tuple version:
t = square_cube(x)
t[0] = 0               # This is NOT allowed
```

Source:
https://www.simplilearn.com/tutorials/python-tutorial/set-in-python

Sets

The general mathematical term `set` means a "collection of like things but without duplicates".

- **Here are two `sets` being defined:**

```
A = {2, 4, 5, 6, 8}        # Curly brackets
B = {'hello', 'hi', 'hey', 'howdy'}
```

A set is mutable, i.e., we can remove or add elements to it. Set in python is similar to mathematical sets, and operations like intersection, union, symmetric difference, and more can be applied.

- **Python also organizes sets so that sets can be compared for equality:** Thus, printing the set

```
C = {8, 5, 4, 6, 2, 4, 5, 5}
```

actually results in

```
{2, 4, 5, 6, 8}
```

```
A = {2, 4, 5, 6, 8}
B = {1, 3, 5, 6}

D = A | B    # union
print(D)
```

Here, `D` contains every element across both sets.

Dictionaries

Source: https://www2.seas.gwu.edu/~cs4all/1012/unit2/module2.0.html#dictionaries

- A standard data structure in computer science is the ”associative array” which is also called the “map”
- In Python, this structure is called a `dictionary.`
- Python `dictionaries` allow you to store key/value pairs of data
  - Input (key) is mapped to an output (value)

```
d = {'apple': 3, 'banana': 3, 'pear': 1, 'kiwi': 2, 'orange': 4}
```

```
d = {'apple': 3, 'banana': 3, 'pear': 1, 'kiwi': 2, 'orange': 4}

d['plum'] = 0
```

```
print(d['apple'])  # Prints 3

d['banana'] = 0
# Which changes the value associated with 'banana' to 3
```

```
d = {'apple': 3}
print(d)
d['apple'] = 4
print(d)
e = {'peach': 2, 'peach': 5}
print(e)
```

```
{'apple': 3}
{'apple': 4}
{'peach': 5}
```

```
# Make an empty dictionary
counters = dict()
```

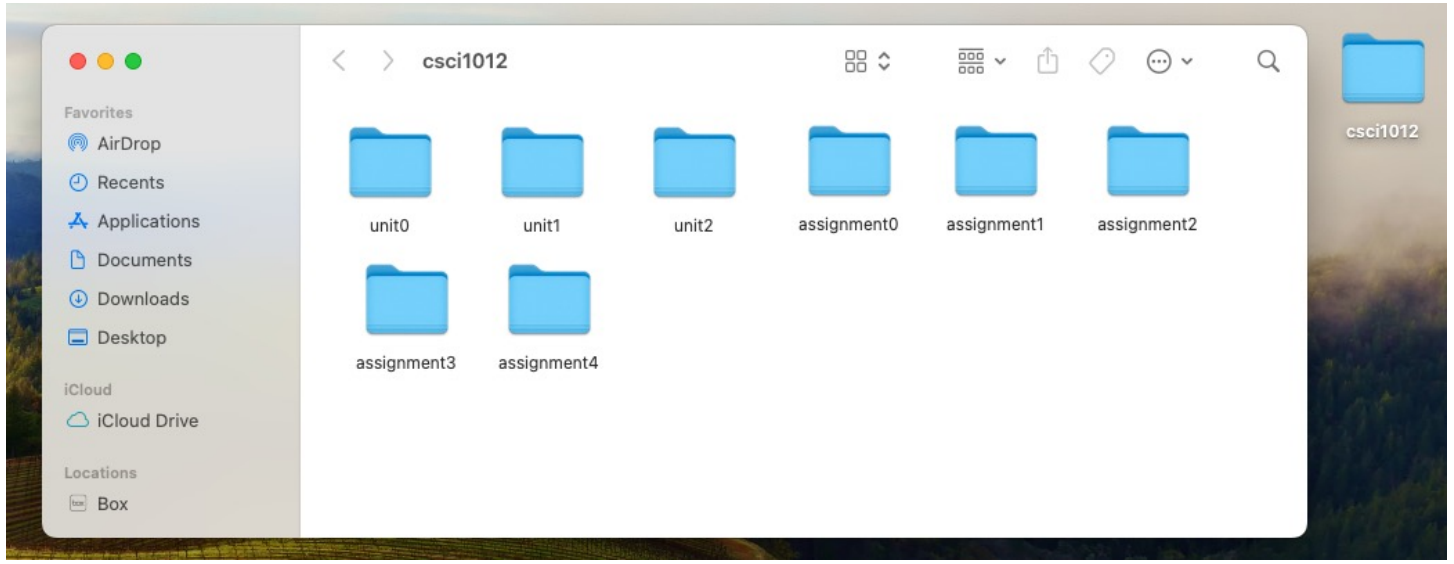
File-folder-structure

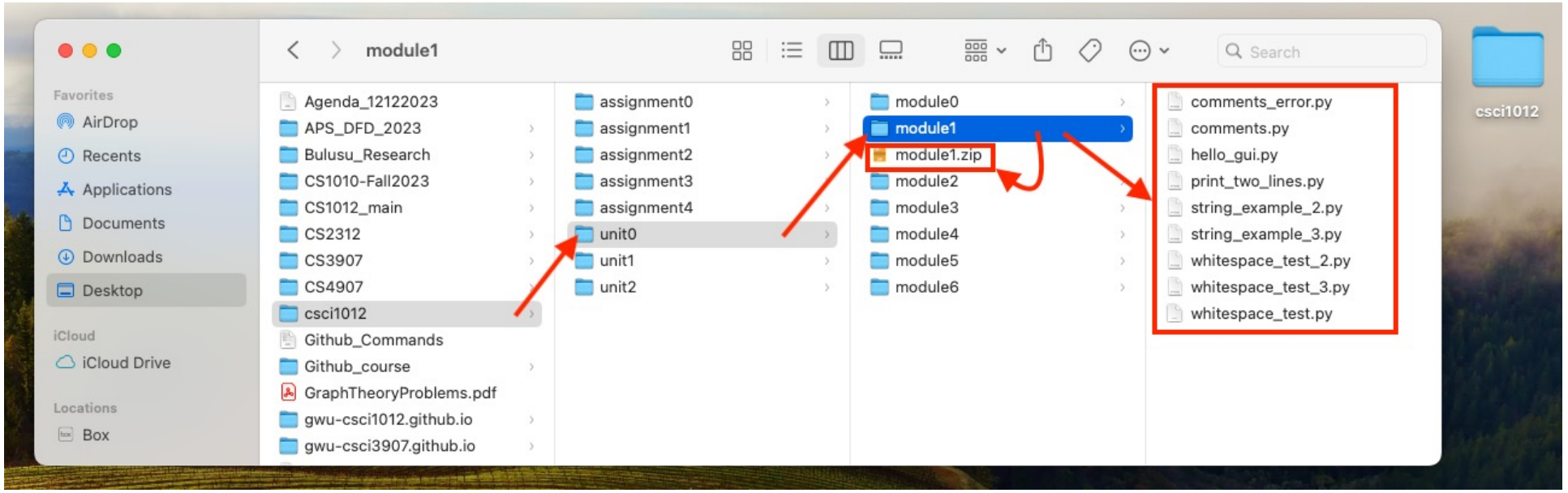
`module0.zip` (correct)

`Module0.zip` (wrong: starts with uppercase)

`module 0.zip` (wrong: space before 0)

`module0.docx` (wrong: not a zip).





See you all in the Wednesday and Friday Labs!