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: ~~1957 E 310~~ MON 352
Section-31: SEH 4040
Section-34: TOMP 310
Section-35: TOMP 204

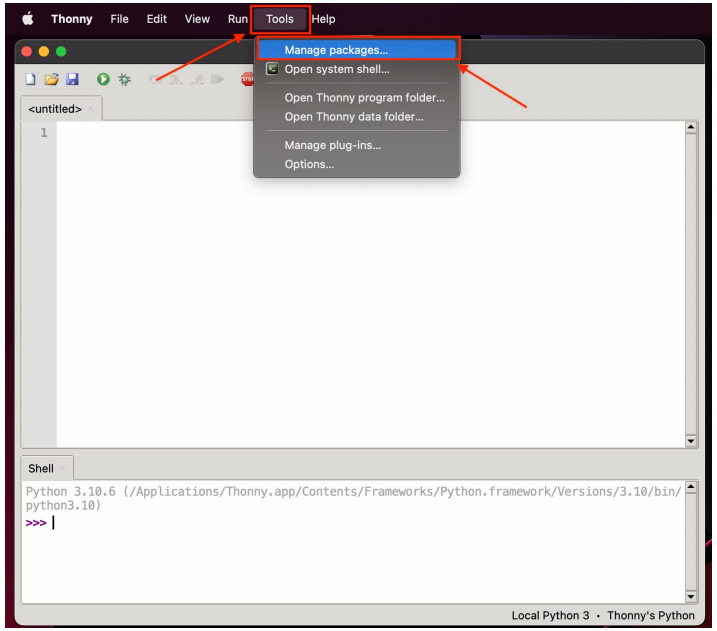
Friday-lab
3:45 PM to 5:00 PM
Section-32: SEH 4040
Section-33: TOMP 309
Section-36: TOMP 306
Section-37: TOMP 107

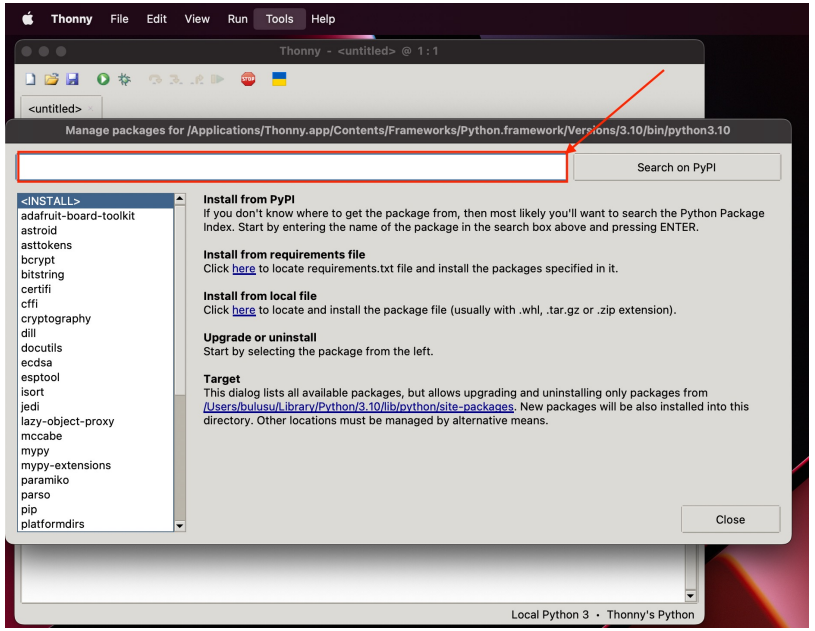
GW School of Engineering & Applied Science
THE GEORGE WASHINGTON UNIVERSITY

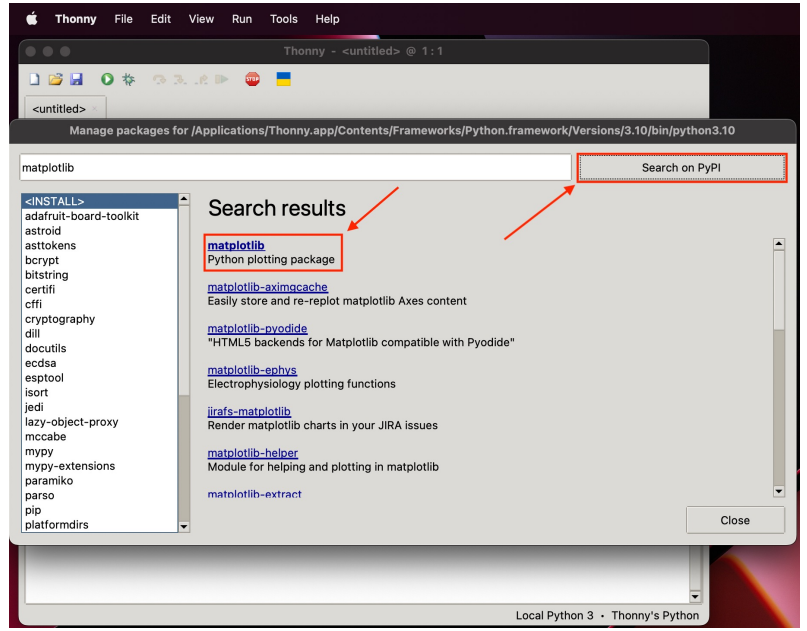
Spring 2024

Photo: Kartik Bulusu

Installing packages in Thonny







Demo

Python programs

Sources:
What is computation? By Ana Bell: https://youtu.be/nykOeWgQcHM
Bilbo Baggins: https://en.wikipedia.org/wiki/Bilbo_Baggins
Bilbo Baggins: https://heroes-and-villain.fandom.com/wiki/Bilbo_Baggins

- Sequences of definitions and commands
- Everything in `Python` is an object
  - number 9, is an object
  - range() is an object
- Manipulate objects
- Objects:
  - have a `type()`, that determine what `python` program can do with them

Analogy

`type`

`human`

- `Python` has two kinds of objects
  - scalar: are very basic and other objects can be made with them
  - non-scalar: have an internal structure

Source:
What is computation? By Ana Bell: https://youtu.be/nykOeWgQcHM

Recap: Our focus was on scalar objects

`int`	Represent integers including 0 (whole numbers)

```
>>> type(9)
<class 'int'>

>>> type(9.0)
<class 'float'>

>>> type(range(9))
<class 'range'>
```

Recap: Integer operators

`i+j`	Sum of integers
`i-j`	Difference
`i*j`	Product
`i/j`	Division
`i%j`	Remainder of `i/j`
`i**j`	`i` raised to the power `j`
`i//j`	Integer division

```
>>> 4*2
8
```

```
>>> 4/2
2.0
```

```
>>> 4**2
16
```

`i` and `j` are integers. **Result is an integer.**

`i` and `j` are integers. **Results in decimal. (float)**

`i` and `j` are integers. **Result is an integer.**

Ungraded In-class Concept Check #1

Some rules to observe:

- 5 minutes restriction
- You are limited to ONE response per question
- You can discuss with your colleagues in-class
- Watch real-time results

http://tinyurl.com/2sp5y37d

Strings

- Sequences of characters
  - Characters can be letter, spaces, numbers, special characters
- Enclosed in quotation marks
- Can be concatenated
- Can be repeated

Strings allow us to create interactive programs

```
>>> type(9)
<class 'int'>

>>> type(9.0)
<class 'float'>

>>> type(range(9))
<class 'range'>

>>> type("Hello")
<class 'str'>
```

```
>>> my_FirstName = "Kartik"
>>> my_LastName = 'Bulusu'
```

0.1.2 - Strings

A *string* in Python is a sequence of letters, digits, or symbols (such as & or @) surrounded by either:

- A pair of double quotes, as in "Hello world!"
- A pair of single quotes, as in: 'Hello world!'

Source: https://www2.seas.gwu.edu/~cs4all/1012/unit0/module0.1.html

String operations

```
>>> my_FirstName = "Kartik"
>>> my_LastName = 'Bulusu'
```

Concatenation of Strings Puts the strings together

```
>>> my_FullName2 = my_FirstName +' '+ my_LastName
```

***-operator on Strings** Repeats the string

```
>>> greeting = my_FullName2 + ('Hello')*3
```

Demo

Ungraded In-class Concept Check #2

Some rules to observe:

- 5 minutes restriction
- You are limited to ONE response
- You can discuss with your colleagues in-class
- Watch real-time results

http://tinyurl.com/ytbse7ca

Recap: print()

We have been printing strings for a while now!

- By default, Python's `print()` function ends with a newline
- Commas between each entry outputs a space between each entry
- The entry of arguments of `print()` can be strings

sep

```
>>> print("Hello", "World!" ,"I", "love", "Python", sep=",")
Hello,World!,I,love,Python
```

end

```
>>> print("Hello", "World!" ,"I", "love", "Python", end=" ")
Hello World! I love Python
```

An escape character is a backslash \ followed by the character you want to insert.

`\'`	Single Quote
`\\`	Backslash
`\n`	New Line
`\r`	Carriage Return
`\t`	Tab
`\b`	Backspace
`\f`	Form Feed
`\ooo`	Octal value
`\xhh`	Hex value

We can now treat strings as objects and binding variables to them.

Source:
What is computation? By Ana Bell: https://youtu.be/nykOeWgQcHM

Nuance of `print()` with Strings

Concatenation puts strings together.

Cannot add a number to a string

```
>>> n = 10
>>> m = 'I am '
>>> k = m + n
>>> print(k)
```

Printing a concatenation requires all entries to be strings

Cast number as a string

Demo

```
n = 10
m = "I am "

# Need to cast the integer as a string to print
n_cast = str(n)

print(m + n_cast + ' years old.')
```

```
I am 10 years old.
```

Source:
What is computation? By Ana Bell: https://youtu.be/nykOeWgQcHM

`input(" ")` Strings

- Requires some string within quotes from the user
  - Single or double quotations as long as they are consistent
- Returns a string
- Can be bound to a variable

```
>>> name = input("Enter your name: ")
Enter your name: Kartik Bulusu

>>> print(name)
Kartik Bulusu
```

Within the quotations inside the parenthesis
- User types in the text that is prompted
- The program will stop until the user enters **something and hit enter**
- **Whatever is typed at the prompt becomes a string**
  - **Numbers, characters, strings, special characters**

```
>>> print(name*3)
Kartik BulusuKartik BulusuKartik Bulusu
```

→ **`*`-operator on Strings** Repeats the string

Demo

Source
Strings and Characters: https://www2.seas.gwu.edu/~cs4all/1012/unit0/module0.5.html

len() of Strings

```
# Make a string and print it:
s = 'The quick brown fox jumps over the lazy dog'
print(s)

# Extract the length of the string and print that:
k = len(s)
print(k)
```

ord() of Strings

```
first_letter = 'a'
last_letter = 'z'

k = ord(first_letter)
print(k)

n = ord(last_letter)
print(n)
```

"some string".count('g')

```
s = 'The quick brown fox jumps over the lazy dog'
n = s.count('a')      # How many a's occur in the string s?
print(n)
```

```
1
```

chr() of Strings

```
for i in range(97, 123):
    s = chr(i)
    print(s, end='')
print()
```

Source:
https://www2.seas.gwu.edu/~cs4all/1012/unit0/module0.5.html

Example: Loops in strings with tracing

```
n = 3
s = ''

for i in range(1, n+1):
        for j in range(0, i):
                s = s + '*'
        s = s + '\n'

print('A triangle with base=' + str(n))

print(s)
```

Iteration #	Value of n	Value of i	Value of j	Value of s	Comment
0	3	0	0	Empty space	
1	3	1	0	*	
		1	0	*\n	end of inner for-loop
2	3	2	0	*\n*	
		2	1	*\n**	
		2	1	*\n**\n	end of inner for-loop
3	3	3	0	*\n**\n*	
			1	*\n**\n**	
			2	*\n**\n***	
			2	*\n**\n***\n	end of inner for-loop
					A triangle with base=3
					prints the triangle using *

Output →

```
A triangle with base=3
*
**
***
```

Ungraded Concept Check

http://tinyurl.com/fnswjee9

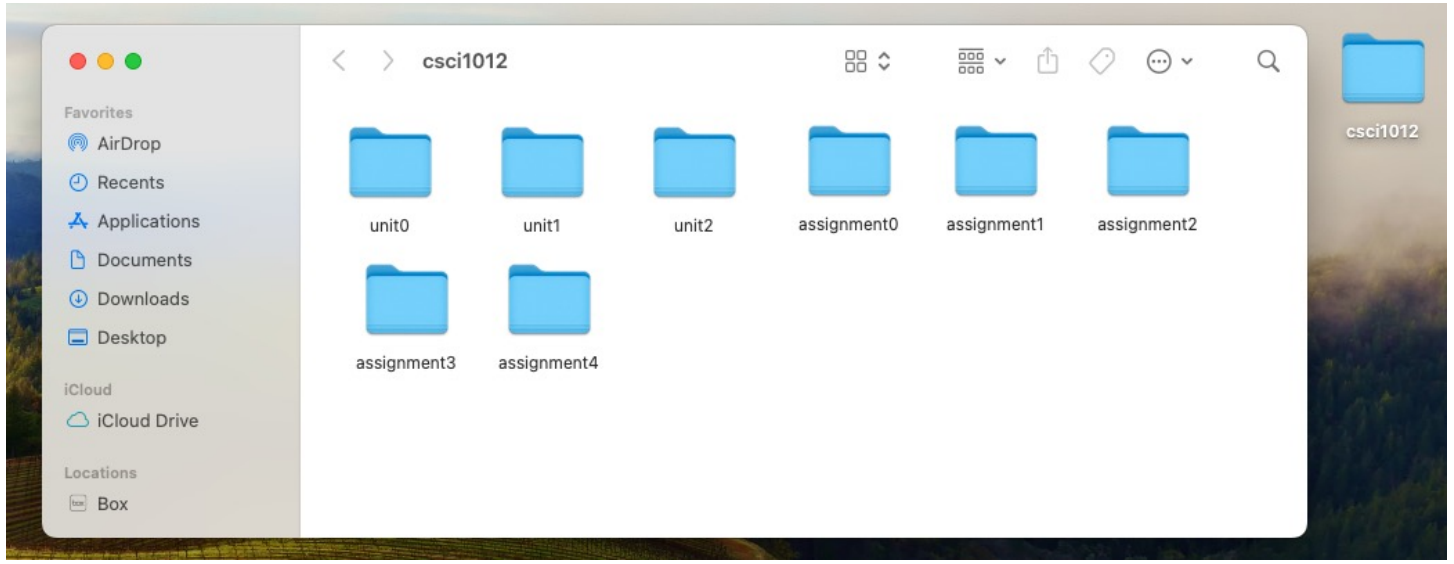
File-folder-structure

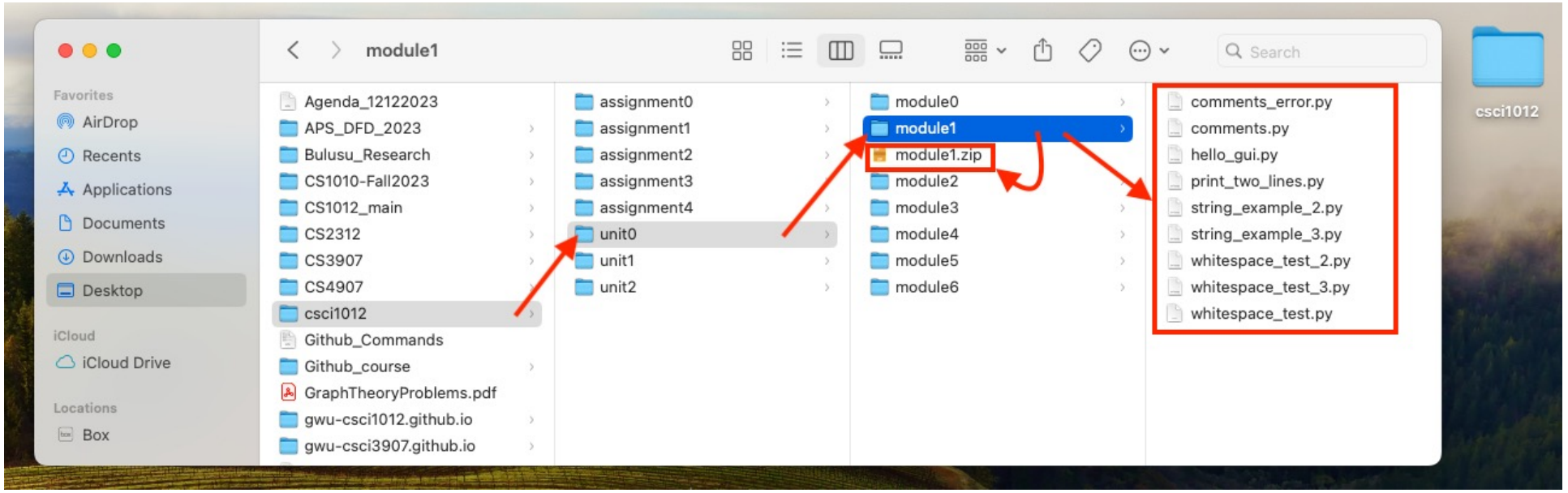
`module0.zip` (correct)

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

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

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





HWs
- Due dates
- Late work
- Extensions

Date	Topic(s)	Wednesday Lab Date	Friday Lab Date	Assignment(s)
Week 3 [02/05/2024]	Integers	02/07/2024	02/09/2024	Unit 0 » Module 4 (Due **February 12, 2024** by **11:59 PM**) & Assignment 0 (Due **February 16, 2024** by **11:59 PM**)
Week 4 [02/12/2024]	Strings	02/14/2024	02/16/2024	Unit 0 » Module 5 (Due **February 19, 2024** by **11:59 PM**)

- **CSCI 1012.30 (CRN: 94165)** - Moved to MONROE 352
- **Office hours location change:** Friday 10:00 AM to 2:00 PM is SEH B1280
- **IMPORTANT:** Please attend the ONLY lab that you registered into.

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.

See you all in the Wednesday and Friday Labs!