LE/EECS 1015 (Section A: LAB 04) Week 10: Lab #9 & #10 (Merged Session)

Shogo Toyonaga

York University

Lassonde School of Engineering

Goals of Lab 9 / 10

1. Write a script that solves problems using nested collections
2. Write a script that solves problems using classes & objects
3. Debugging
4. Software Engineering (Clean Code)

Concept Review

1. Nested Collections
    - Indexing
    - Shallow vs Deep Copying
2. Nested Loops
    - For Loops
    - Numerical Indexing
    - Time Complexity $\rightarrow O(n)\ vs\ O(n^2)$
3. Classes & Objects
    - Classes
    - Constructors
    - Methods: Getter & Setters
4. Input / Output
    - Reading, Writing, Creation, and Everything Else
    - Race-Conditions

Nested Collections (Declaration & Indexing)

- You can define collections within collections; we call this **nesting**.
- Accessing elements within a nested collection through numerical indexing will require n indexes for n nested items.
- We can also index nested items by using variable names in the for-loops.
- **Note:** Try to format nested collections for readability (e.g., multiline with proper indentation)

Nested Collections (Indexing Elements)

```python
points = [(1, 2), (3, 4), (5, 6)]
print(points)
# Access the x-value for each point in our collection
for point in points:
    x, y = point
    print(f'x = {x}\ny = {y}\n')
```

✓ 0.0s Python

```python
points = [(1, 2), (3, 4), (5, 6)]
print(points)
# Access the x/y values for each point in our collection
for i in range(len(points)):
    for j in range(len(points[i])):
        print(points[i][j])
```

Python

Challenge

Write a function $sum\_columns(input: List[List]) \rightarrow List[int]$ that sums the columns of a 2D integer list and returns the values in a 1D list.

Use a nested loop with numerical indexing.

Hint: While it is less efficient, you can pad each of the rows to have a similar length and then calculate the sums quickly.

Nested Collections (Copying)

- Shallow copy $(collection.copy())$: The reference addresses are copied.
- Deep copy $(copy.deepcopy(collection())$: The reference addresses are entirely different.

- **Homework:** See how the behaviour of 1D and 2D lists differ when creating shallow and/or deep copies.

Classes & Objects

- Classes are useful for creating your own data type with custom attributes and methods (aka behaviours)
- An object is an instantiation or realization of your class (data type). It has its own memory address.

Implementing a Class

1. Declare a Class (Name Convention: Camel Case)
2. Define the Constructor
    - Attributes should be private; In Python, these names should start with an underscore.
3. Define Methods
    - Getter: Gets the value of an objects' attribute
    - Setter: Assigns a value to the objects' variables
    - Other(s): Up to you to define the class methods; what is your end-goal?

Supporting Python Syntax with New Objects

- You can implement specific behaviour with Python syntax by overriding the default methods:

Operator	Magic Method
$+$	__add__(self, y)
$-$	__sub__(self, y)
$*$	__mul__(self, y)
$**$	__pow__(self, y)
$//$	__floordiv__(self, y)
$/$	__truediv__(self, y)
$-$	__neg__(self, y)
abs	__abs__(self, y)
$\sim$	__invert__(self, y)
$<$	__lt__(self, y)
<=	__le__(self, y)
==	__eq__(self, y)
!=	__ne__(self, y)
>	__gt__(self, y)
>=	__ge__(self, y)

Reading and Writing to Input/Output (I/O)

- When you open files for reading and writing, you need to ensure that you close them as soon as possible. This avoids race-conditions.
- We can read a file by using the following code:

```
with open(‘path', 'r') as file:
      pass
```

- Relative Path
  - Contingent on where you are with respect to a parent directory.
  - Using a relative path allows for more flexibility across devices
- Absolute Path
  - Hard-coded for your specific memory management hierarchy
  - Example: “C:\Users\Shogo\Desktop” will not work on your computer
- Modes
  - We can set the mode of open(..) to read, write, or both!

Open(..) Modes

Mode	Action
Read	Opens a text file for reading. If the file does not exist, an error is thrown.
Append	Opens a text file for writing. The data is written at the end of the existing data. The file is created if it does not exist.
Write	Opens a text file for writing. Existing data is over-written. If the file does not exist, it is created.
Create	Creates a text file and then writes to it dynamically.

- Note: Adding + to the mode ($e.g., x+$) enables reading and writing permissions simultaneously!

Lab 9 Objectives

1. Follow the Steps (Separate Numbers) (/30)
2. Debugging (Find Highest Correlation) (/30)
3. Implementation ($n \times n$ Tic-Tac-Toe) (/20)

Lab 10 Objectives

1. Follow the Steps (Rectangle Class) (/50)
2. Debugging (Inventory Management) (/30)


This is it, thank you for everything! 😊
I wish you all the best of luck in your future endeavours!
Shogo Toyonaga
Lassonde School of Engineering