SI Session 12/05

By Cody Ptacek

Session Overview & Announcements

Overview

- In this session, we'll go over the material from the midterm onward. THE FINAL is COMPREHENSIVE, but we'll focus mainly on 2nd half material.
  - The course builds upon itself, so focusing on only midterm onward should also utilize stuff learned early on in the semester.
  - Here are things I would go back and review from the beginning of the semester:
    - Stand alone ifs vs if-elif blocks
    - string/list methods
    - File I/O
    - While loops
    - For-in loops
- We'll focus on code interpretation and conceptual questions today.

Announcements

- SI Survey
  - Remind me to present the QR code at the end of the session!
- Next Session
  - Focus on errors & writing classes (all magic methods [we've learned], some functions, & raising exceptions.
  - More interpretation and conceptual question review.

Activity: SI Leader Survey

Review: Memory Storage

Which of these are stored on the stack vs the heap?

- variable names
- variable information
- function calls
- lists
- information in lists

What was seemingly different/special about storing lists? What functionality does this provide?

Review: Memory Storage Cont.

Given the following code:

```
def random_pikachu(my_list):
    index = random.randint(0, len(my_list))
    my_list[index] = "Pikachu"

pokemon_list = ['Arcanine', 'Blastoise', 'Piplup',
'Gholdengo']
print(random_pikachu(pokemon_list))
```

Answer the following questions:

1) When we pass pokemon_list to the random_pikachu() function, what *specifically* is passed?
2) Assuming the random value is 3, what will be printed to the console?
3) If we were to print pokemon_list after the function call, what would be printed to the console?

Review: Memory Storage Cont.

BONUS

- Rewrite random_pikachu() to *return* a new copy of the list with a random pikachu in it, *without* modifying the original list.

Review: Parameters

Questions

- How do we pass parameters to a function?
- What are two ways of passing parameters that python supports?
- What are the three ‘kinds’ of parameters that python supports?
    - What symbols are used to specify which parameters are which ‘kind’?

Review: Parameters Cont.

Given the following code:

```
def learn_your_ifs(int1, int2, /, int3, int4, *, int5, int6):
    result = 0
    if int1 > int2:
        result += int2
    else:
        result += int1
    if int3 == int4:
        result += int3^2
    if int3 < int5:
        result += int3
    elif int3 < int6:
        return int6 - result
    elif int3 == int4:
        result -= int3^2
    else:
        result -= int3
    if int6 == 7:
        result = 7
    return result
```

Write the result (if an error is raised or the value returned) for the following questions:

1) `learn_your_ifs(1, 2, 3, 4, 5, 6)`
2) `learn_your_ifs(1, 2, int5=7, int6=7, int3=2, int4=2)`
3) `learn_your_ifs(int1=9, int2=6, int3=4, int4=2, int5=5, int6=1)`
4) `learn_your_ifs(6, 3, 7, int6=4, int5=2, int4=7)`
5) `learn_your_ifs(5,5,7,5,int5=7, int6=7)`

BONUS:

- Write a function call of learn_your_ifs that returns the value 24.

Write the result (if an error is raised or the value returned) for the following questions:

1) `learn_your_ifs(1, 2, 3, 4, 5, 6)`
2) `learn_your_ifs(1, 2, int5=7, int6=7, int3=2, int4=2)`
3) `learn_your_ifs(int1=9, int2=6, int3=4, int4=2, int5=5, int6=1)`
4) `learn_your_ifs(6, 3, 7, int6=4, int5=2, int4=7)`
5) `learn_your_ifs(5,5,7,5,int5=7, int6=7)`

BONUS:

- Write a function call of learn_your_ifs that returns the value 24.

Answers:

1) ERROR
2) 7
3) ERROR
4) 3
5) 7

Bonus:

- `learn_your_ifs(1, 27, 3, 4, int5=1, int6=1)`

Review: Comprehensions

Questions

- What is the general syntax for a list comprehension? Which elements are optional?
- Why do we use list comprehensions? What value do they add to programming?

Review: Comprehensions

Given the following lists:

```
my_closet = ['coat', 'gloves', 'guitar', 'scarf',
'mittens', 'snacks', 'shirt']
clothes = ['coat', 'hat', 'Scarf', 'gloves',
'mittens', 'shirt', 'pants']
```

Write the list returned by the following list comprehensions:

1) `[item.upper() for item in my_closet]`
2) `[item for item in my_closet if item in clothes]`
3) `[item[0] for item in my_closet if item in clothes]`

BONUS

4) `[item.upper() for index, item in enumerate(my_closet) if index % 2 == 0 and item in clothes]`

1) ['COAT', 'GLOVES', 'GUITAR', 'SCARF', 'MITTENS', 'SNACKS', 'SHIRT']
2) ['coat', 'gloves', 'mittens', 'shirt']
3) ['c', 'g', 'm', 's']
4) ['COAT', 'MITTENS', 'SHIRT']

Review: Dictionaries

Questions:

- What is unique about dictionaries as opposed to lists.
- How do we add new entries to a dictionary?
- Keys cannot be what type(s) that you have learned thus far?

Review: Dictionaries Cont

Boardwork 1

Write a function that takes a file name as a parameter. The file contains a list of friends and their b-days in the following form:

Joe 01-01
Tanner 05-06
Cody 12-15

Your function should store each entry in a dictionary with the key being the friend's name and the value being the birthday stored as a tuple of the form:

(month, day)
Ex. (12, 15)

Return this dictionary.

Boardwork 2

Create a function that takes a date and a dictionary as parameters. The dictionary will be of the form created in Boardwork 1:

ex. My_dict = {'Cody': (12, 15), …

Likewise the date will need to be of the form:

(month, day)

Your function will return a list of friends with b-days on the given date. If no friends have b-days on that date, raise an exception. (Use RuntimeError).

Review: Sets

Questions:

- What is the unique feature of sets?
- What are the set methods we've learned thus far? Explain what each does.

Boardwork:

- Create a function with the following functionality:
  - Takes a set as a parameter.The set contains five names of TV shows.
  - Collect five TV shows from a user and store them in a new set.
  - Then, using set methods, print the shows that were in both sets, and the shows that were unique to the parameter set.

Review: Classes

Questions

- Where do we make a new class?
- What is the syntax for making a class?
- What are the naming conventions for classes?
- What one magic method should all your classes have? What parameter should this method always have?
- How do we add new member variables to our classes?
- How do we add new methods to our classes?
- What are the nine magic methods you have learned and what does each do?

Questions Cont

- How do we specify a member variable as private?
- What is the point of specifying a member variable as private?
- How do we import a class in a new file?
- How do we create a new instance of the class?
- If we have not specified otherwise, what is the result of printing a class?

Review: Classes Cont

Given the following code for a Gold class:

```
class Gold:
    def __init__(self, weight, purity):
        self._weight = 0
        self._purity = 0.01
        self.set_weight(weight)
        self.set_purity(purity)
    def set_weight(self, weight):
        if weight >=0:
            self._weight = weight
        else:
            raise RuntimeError
    def set_purity(self, purity):
        if purity > 0 or purity <= 1:
            self._purity = purity
        else:
            raise RuntimeError
    def __eq__(self, other):
        return self._weight * self._purity == other._weight * other._purity
```

Answer the following questions

1) Why are we using ‘setter’ methods instead of directly passing values to self._weight and self._purity?:
2) How do we import functionality for the Gold class?
3) How do we initialize a new instance of the Gold class?
4) What would be the result of the following initializations (would each work? What values would the member variables for each contain?):
```
new_gold = Gold(77, 1)
newer_gold = Gold(0, 0)
newest_gold = Gold(22, 0.4)
```
5) Write all the remaining operator magic methods for the Gold class.

Closing Statements

Before you walk out the door, think back to all the topics we discussed today.

- What did you feel like you needed to study more?
- When are you planning to study more?
- What resources will you use to study?

If I don’t see you all again:

- Good luck on the final!
- Have a great winter break!

SI Survey

Activity: SI Leader Survey