APS106

UNIVERSITY OF TORONTO

Tutorial 10 – Week 11

We'll be starting at the 10 minute mark

if nothing else, write #cleancode

Agenda

- Lab 6 review
  - `score_hand()` function
- Lecture review
  - Intro to object-oriented programming
  - User-defined Classes and methods
- Practice questions

Learning Objectives

After this tutorial, learners should be able to:

- recognize / describe / create Python classes
  - recognize / describe / create data attributes
    - recognize / describe / create class data attributes
    - recognize / describe / create instance data attributes
  - recognize / describe / create methods
    - recognize / describe / create class initializers (__init__)
    - recognize / describe / create non-initializer imethods
- recognize / describe / create Python objects
- call methods on class / class instance objects

Review of Lab

`score_hand()` function

`score_hand` function – 1

Consider a card game where each player receives five cards, i.e., “a hand”. A “hand” is scored according to the following rules:

(R1) Each pair (i.e. two cards with the same card value) scores 2 points. If a hand contains three or four of a kind, 2 points are scored for every combination of pairs: three cards of a kind are worth 6 points and four cards of a kind are worth 12 points.

(R2) All combinations of cards that sum to 15 are worth 2 points. When summing card combinations, aces are counted as one and jacks, queens, and kings are all counted as 10.

Hint: iterate over all combinations of 2, 3, 4, and 5 cards to see if the values in these combinations sum to 15. Make sure to count any value larger than 10 as 10.

`score_hand` function - 2

(R3) A group of three cards with consecutive values (called a ***run*** or a ***straight)*** scores 3 points. A run of four consecutive values scores 4 points and a run of five consecutive values scores 5 points. (The suit of the cards does not matter.)

Hint:

Step 1: Get all combinations of 3, 4, and 5 cards

Step 2: Filter through all combinations and only keeps the ones that consist of consecutive values (you can use the `.sort()` method of class list or use the algorithm practiced in the tutorial a few weeks ago)

Step 3: Filter out the children subsets and only keep the maximal subsets (We did one example in last week's tutorial 😉).

Step 4: Assign scores based on the maximal subsets

`score_hand` function - 3

(R4) If all five cards in a hand are the same suit, i.e., one of **hearts, tiles, clovers** and **pikes**, 5 points are scored. If the hand has only four cards with the same suit, 4 points are scored. (A **suit** is one of the categories into which the cards of a deck are divided.)

Hint: iterate over all combinations of four and five cards to see if the values inside these combinations are all equal.

> An easy way to check if all values in a list are all equal is to **sets**. For example, if you have a list of cards that have the same suit, attempting to convert the list into a set will reduce its length to 1, e.g., `set(['hearts', 'hearts', 'hearts', 'hearts', 'hearts'])` will return the set `{'hearts'}`

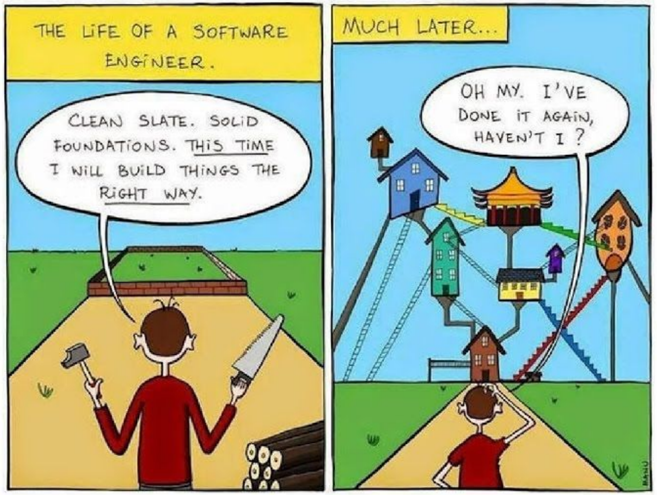
Review of Lecture

Intro to object-oriented programming

What is Object Oriented Programming (OOP)?

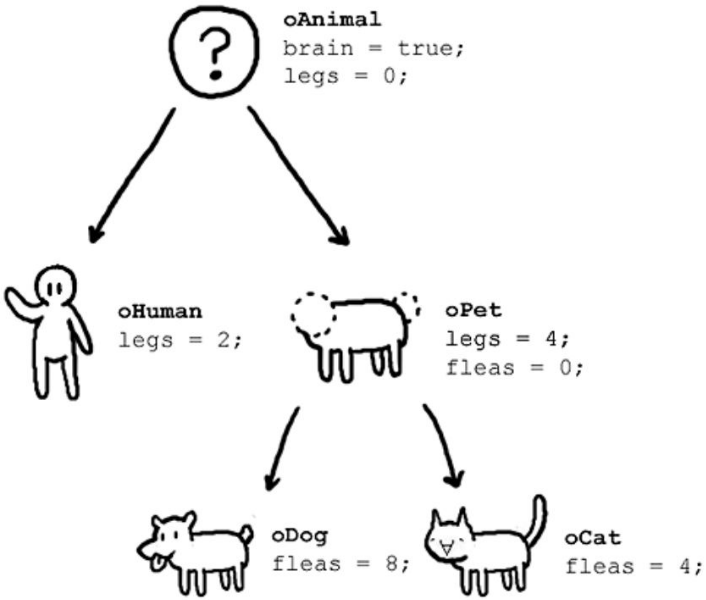
- OOP is a programming paradigm where programmers build abstract data types that resemble real world objects
- This enables the programmer to use the objects as they would outside the program to solve complex problems
- OOP focuses on the creation of objects which contain both **data** and **functionality** together and achieving the overall program functionality through the interaction of these objects.

Why use OOP?


THE LIFE OF A SOFTWARE ENGINEER.
CLEAN SLATE. SOLID FOUNDATIONS. THIS TIME I WILL BUILD THINGS THE RIGHT WAY.
MUCH LATER...
OH MY. I'VE DONE IT AGAIN, HAVEN'T I ?

- “How would you do it?” → “Translate this to Python”
- As programs get larger and need to handle increasingly complex problems, it gets harder to represent the data by simply composing built-in data-types, i.e., list, dictionary, int, string, etc.
  ⇒An option is to create new data types that can store both information (in data **attributes**) and behaviour (in **methods**).
  →In Python, methods are also referred to as **method attributes.**

This seems like more work, why would you do this?



- The real benefits of OOP manifest more clearly in programs that are complicated and large.

- Usually, in a large industry project, several classes will already be defined and new programmers who join a project I write code that uses the pre-existing classes.

APS106

UNIVERSITY OF TORONTO

Review of Lecture

User-defined Classes

if nothing else, write #cleancode

Review: Classes and Objects

- Classes are **templates** for generating objects
  Built-in classes: `int, list, str, dict,` …

- Each **object** is an **instance** of a class template

  Example: `x = [5,5,3]`

  `Y = list((5,6,7))`

  `X and Y` are objects, instances of class **`list`**

Note: – objects are the actual **values** in a program

– classes are how you would describe a type of object, and its capabilities.

User-Defined Classes – 1

- One can expand the set of available classes by defining new classes

- The general form of a class definition is:

```
class <class name>:
    #class data attributes
    …

    def __init__(self, parameters):
        #instance data attributes
       …

    def method_1(self, parameters):
       …
    def method_1(self, parameters):
       …
```

User-Defined Classes – 2

```
class <class name>:
        #class data attributes
        ...
        def __init__(self, parameters):
            #instance data attributes
            ...
```

- The `__init__` method is responsible for setting up the initial state of any new class instance. (The **initializer method** is automatically called whenever a new class instance is created by the class **constructor.**)
- The class **constructor,** in Python, is called **__new__**. It is called first after a class instantiation statement and returns an instance of the class. In general, you do not need to add a method **__new__** to your class. **It is already available by default.**

  ⇒ `__init__` is responsible for initializing a class instance. The class instance exists at the time `__init__` is called.

Example User Defined-Class

We can simulate multiple __init__ methods in a class by using optional parameters.

If multiple __init__ are present, the last one overrides the previous ones.

```
# define the class
class Point:

    def __init__(self,xx=0,yy=0,zz=0):
        self.x = xx
        self.y = yy
        self.z = zz
```

__init__ is called the "initializer method"

__init__ creates three instance attributes: x, y and z

The self parameter is automatically set to reference the newly created object

```
# instantiate some Point objects
q = Point()
```

__init__ sets the attributes x, y and z of an instance of class Point to 0.

```
p = Point(3,4,5)
```

__init__ sets the attributes x, y and z of an instance of class Point to 3, 4 and 5, respectively.

User-Defined Methods

- Classes have a set of functions (aka methods) that can only be applied to objects that are instances of the class

- The general form of a class with methods is:

```
class <class name>:

  #class data attributes
  ….

 def method_name1(self, param1):
   body1
 def method_name2(self, param2):
   body2
```

Method __init__ can be one of the methods in the body of a class, but it is not mandatory. Each class has a default initializer.

Example User Defined-Class

```
# define class Point
class Point:

    def __init__(self,x=0,y=0,z=0):
        self.x = x
        self.y = y
        self.z = z
```

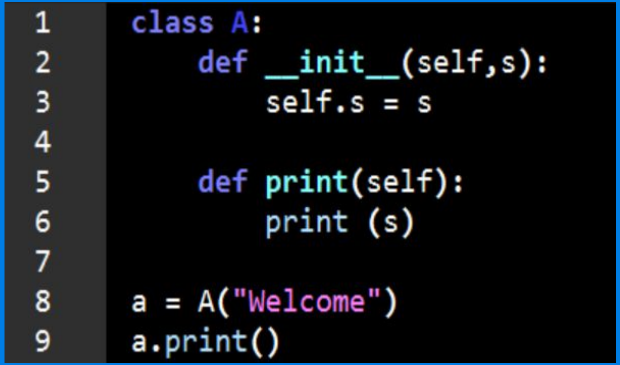
distance_from_origin is a method

```
    def distance_from_origin(self):
        return ((self.x**2) + (self.y**2) +(self.z**2))**0.5

# instantiate some 3d Point objects
p = Point(3,4,5)
q = Point(1,2,3)

# get the distance from origin (Note the two different ways to call methods!)

Distance_p = p.distance_from_origin()      # object.method()

Distance_q = Point.distance_from_origin(q) # Class.method(object)
```

Practice Problems

Review Practice Problem 1

Q1. Analyze the following code and select the appropriate statement



```
1  class A:
2      def __init__(self,s):
3          self.s = s
4
5      def print(self):
6          print (s)
7
8  a = A("Welcome")
9  a.print()
```

A. The program outputs “Welcome”
B. The program outputs “s”
C. The program has an error because class A does not have a constructor
D. The program would run if line 6 was `print(self.s)`
E. The program would run if line 6 was `print(self, s)`

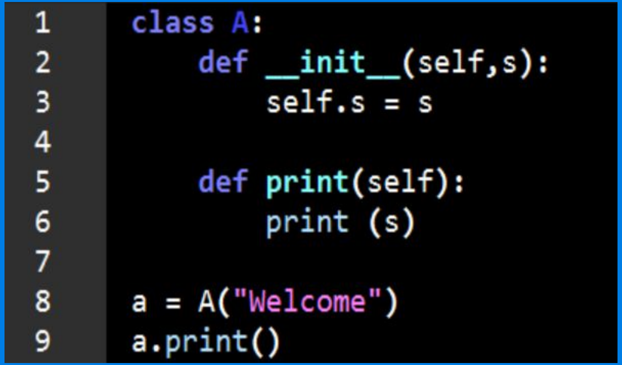
slido

Q1. Analyze the following code and select the appropriate statement

ⓘ Start presenting to display the poll results on this slide.

Review Practice Problem 1

Q1. Analyze the following code and select the appropriate statement



```
1  class A:
2      def __init__(self,s):
3          self.s = s
4
5      def print(self):
6          print (s)
7
8  a = A("Welcome")
9  a.print()
```

A. The program outputs “Welcome”
B. The program outputs “s”
C. The program has an error because class A does not have a constructor
D. The program would run if line 6 was `print(self.s)`
E. The program would run if line 6 was `print(self, s)`

Review Practice Problem 2

Q2. An object is an instance of …

A. a program
B. a method
C. data
D. a class
E. a function

slido

Q2. An object is an instance of ...

ⓘ Start presenting to display the poll results on this slide.

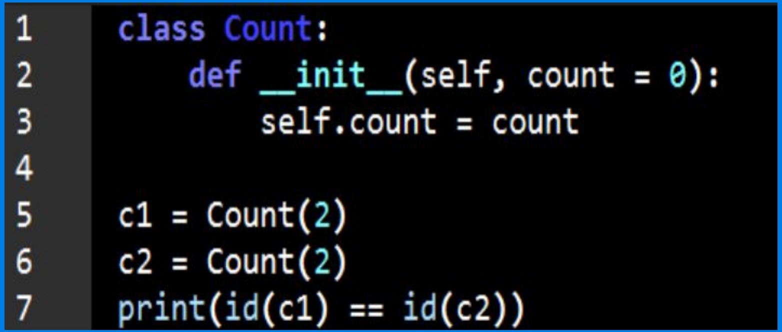
Review Practice Problem 2

Q2. An object is an instance of …

A. a program
B. a method
C. data
D. a class
E. a function

Review Practice Problem 3

Q3. What does the following code output?



```
class Count:
    def __init__(self, count = 0):
        self.count = count

c1 = Count(2)
c2 = Count(2)
print(id(c1) == id(c2))
```

A. True
B. False
C. An error

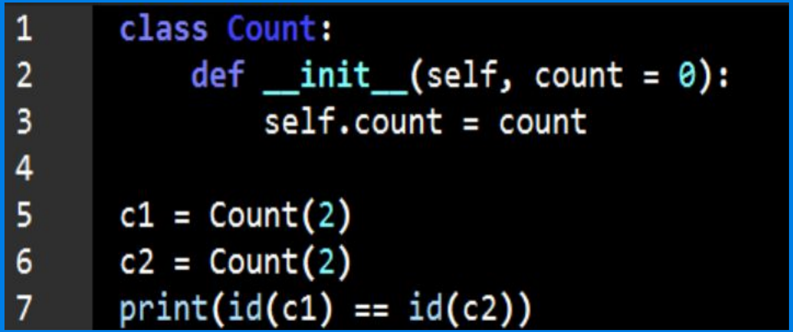
slido

Q3. What does the following code output?

ⓘ Start presenting to display the poll results on this slide.

Review Practice Problem 3

Q3. What does the following code output?

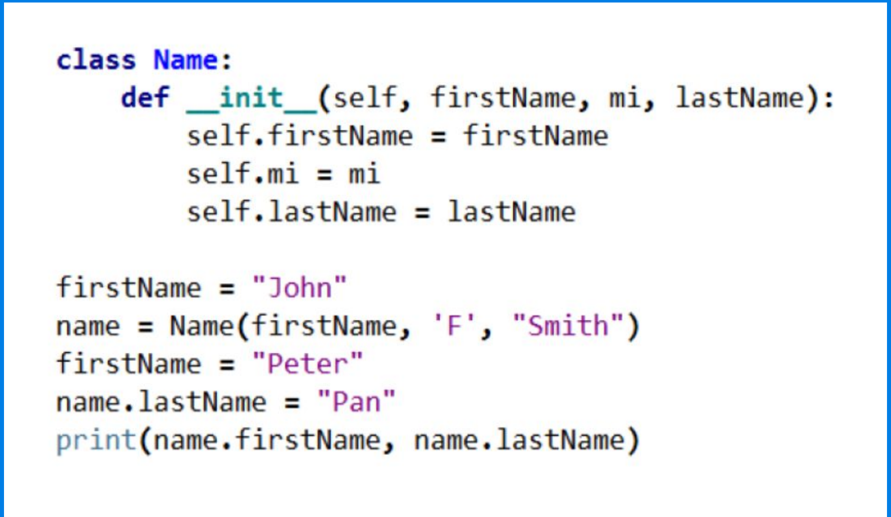


```python
class Count:
    def __init__(self, count = 0):
        self.count = count

c1 = Count(2)
c2 = Count(2)
print(id(c1) == id(c2))
```

A. True
B. False
C. An error

Review Practice Problem 4

Q4. What does the following code output?



```
class Name:
    def __init__(self, firstName, mi, lastName):
        self.firstName = firstName
        self.mi = mi
        self.lastName = lastName

firstName = "John"
name = Name(firstName, 'F', "Smith")
firstName = "Peter"
name.lastName = "Pan"
print(name.firstName, name.lastName)
```

A. `Peter Pan`
B. `Peter F Smith`
C. `John Pan`
D. `John F Smith`
E. None of the above

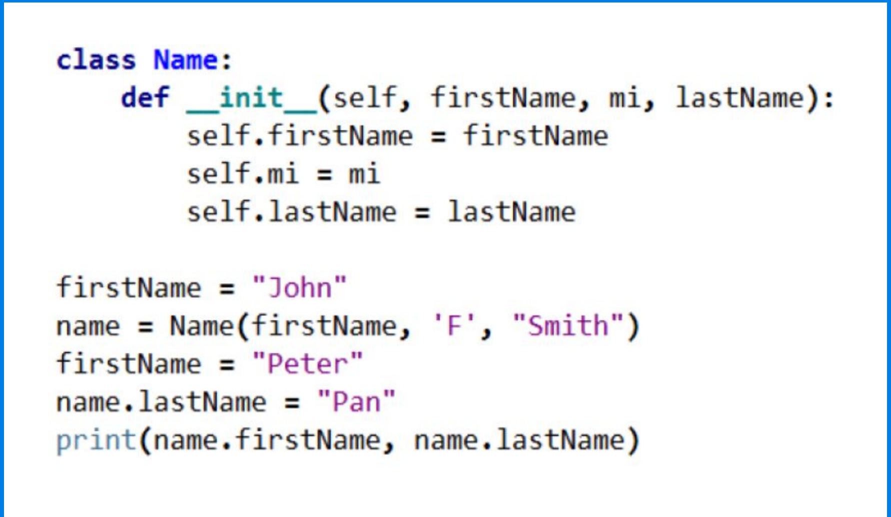
slido

Q4. What does the following code output?

ⓘ Start presenting to display the poll results on this slide.

Review Practice Problem 4

Q4. What does the following code output?



```
class Name:
    def __init__(self, firstName, mi, lastName):
        self.firstName = firstName
        self.mi = mi
        self.lastName = lastName

firstName = "John"
name = Name(firstName, 'F', "Smith")
firstName = "Peter"
name.lastName = "Pan"
print(name.firstName, name.lastName)
```

A. `Peter Pan`
B. `Peter F Smith`
C. `John Pan`
D. `John F Smith`
E. None of the above

Coding Question 1

Let's make a simple class called Car that has three data attributes:

- brand (stored as a string)
- model (stored as a string
- top_speed (stored as a floating point number)

1. Define a class **`Car`**.
2. Create a method `is_faster(self, other_car)`, which returns True if **`self`** is "faster" than the **`other_car`** and False if it is slower
3. Create two **`Car`** objects, `car1` and `car2`
4. Print out the attributes of each **`Car`** object.
5. Print out whether `car1` is "faster" than `car2`

Coding Question 2

Define a class called **`Rectangle`** and create methods to:

- compute its area
- compute its perimeter
- find its centre point
- compare two rectangles and return the rectangle with the largest area

Instructions:

- Use the starter code provided on Quercus. Go to your tutorial section and use the "open the practice problem" prompt.
- Use the **`Point`** class to solve this problem.
  - **Hint:** What are some logical attributes for a rectangle? What attributes would be useful within the methods that we're going to write?

slido

Any Questions?

ⓘ Start presenting to display the poll results on this slide.