SOC presents

Intermediate Python

Recap from last time

- Data Types
- If/Else Statements
- For/While Loops

Example #1

Let a user type numbers in one at a time, until “STOP” is input. Then, return a sum of the numbers

Example #2

Given a list of numbers, find all pairs that add up to 10

Example #3

Write a simple blackjack game

Break time!

Objects and Classes

What objects/classes do we already know?

Objects/Classes

- Encapsulates data, and functions centered around them (methods)
- Reduces code duplication
- With good design, can be reused, extended etc.
- Allows a “program” to be split into smaller components:
  - Easier to think about and write
  - Work can be split up easily

Example 1

- You want a class to represent a Person
  - What data do you want to have about a Person?
  - What methods should a Person be capable of?

Example 1

- You want a class to represent a Person
  - What data do you want to have about a Person?
    - Firstname, Lastname, Age……etc.
  - What kind of things should a Person be capable of?
    - Greet, Eat, Grow…..etc.

Person Class

```
class Person:

    def __init__(self, name):
        self.name = name
```

Example 1

By writing Person class, you can now instantiate as many Person objects as you want

Person Class

```
class Person:

    def __init__(self, name):
        self.name = name

    def greet(self):
        print("Hello! My name is", self.name)
```

Person Class

```
john = Person("John")
john.greet()
```

Person Class

```
class Person:
    Constructor
    def __init__(self, name):
        self.name = name  Field

    def greet(self):  Method
        print("Hello! My name is", self.name)
```

Class-level Properties

```
class Person:

    race = "Human"

    def __init__(self, name):
        self.name = name
```

Example 1

What else can we add to the Person class?

Default Object Methods

All Objects have default methods for certain tasks, for example, print(a) will call a.__str__(). See documentation for more details.

Person class

- We created a single class to encapsulate a Person, so we can create multiple people.
- Each Person object
  - can be created simply and easily
  - has the same functionality
- You should now **begin** to see why Object-Oriented Programming is useful

Example 2

- You want a simple, command line rock-paper-scissors game
- How would you do this? (TBC next lecture)