**IS201 Fundamentals of Computing**

**HOP06 Classes & Intro to Object-Oriented Programming**

03/16/2020 Developed by Amrutha Vaidyanathan

10/25/2020 Reviewed by Kim Nguyen

Center for Information Assurance (CIAE) @City University of Seattle (CityU)

**Before You Start**

* The directory path shown in screenshots may be different from yours.
* Some steps are not explained in the tutorial**.** If you are not sure what to do:
  1. Consult the resources listed below.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

* Learn ways to pass information to functions
* How to write certain functions whose primary job is to display information and other functions designed to process data and return a value or set of values.
* Learn to store functions in separate files called modules to help organize your main program files.
* How to use debugger tools

**Resources**

Matthes, E. (2019). [Python Crash Course: A Hands-On, Project-Based Introduction to Programming, 2nd Edition](https://login.proxy.cityu.edu/sso/skillport?context=146803). No Starch Press. (ISBN 9781593279288)

**If you have completed all tasks in this document from previous weeks, move on to the Challenge document in Module 6 folder**

**Preparation**

In Visual Studio Code, open the private repository generated when you accepted the HOP06 assignment (If you cannot find that repository in your machine, you might have not cloned the repo, if so, please do before proceeding).

**Class**

Each thing or object is an instance of some class, it is a blueprint for any functional entity which defines its attributes and its methods.

Classes are used to create new user-defined data structures that contain arbitrary information about something.

**Object Oriented Programming**

It is a programming paradigm which provides a means of structuring programs based on a concept of “objects” which can include attribute and method.

For example, an object could represent a person with a name property, age, address, etc., with behaviors like walking, talking, breathing, and running. Or an email with properties like recipient list, subject, body, etc., and behaviors like adding attachments and sending. In the next module OOP is discussed in detail.

1. Create a class called Employee with the following and save it as **oop.py**

A picture containing meter

Description automatically generated

**pass**keyword means nothing happens yet when you have empty function without it, it will raise error. We will come back to edit this part later.

1. Next, we create an instance of the class as the following. Update the **oop.py** with following code.

A screenshot of a cell phone

Description automatically generated

We just created an instance of the Employee object. In the terminal type **python3 oop.py**



**Creating instance**

All classes create objects, and all objects contain characteristics called attributes (referred to as properties in the opening paragraph). Use the \_\_init\_\_() method to initialize an object’s initial attributes by giving them their default value.

This method must have at least one argument as well as the self-variable, which refers to the object itself.

Instance variables contain data that is unique. Below code shows how to create two instances in **oop.py**. **You don’t have to type the below code.**

A close up of text on a black background

Description automatically generated

1. If you were to add 1000 of employees, doing the above way might not be a good idea.

We can create what is called **\_\_init\_\_** method. It is a special method, which is called class constructor or initialization method that Python calls when you create a new instance of this class.

A screen shot of a computer

Description automatically generated

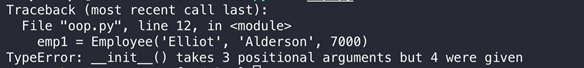
As you see here, a lot easier to add employee information. You might have spotted what is **self**?

The word "**self"** refers to the current object that you are working with. Basically, all methods in a class need the self-object as their first argument, so they can access any attribute that is part of the class.

Here is what happens if you remove self in the \_\_init\_\_. (Don’t type this one)

A screen shot of a smart phone

Description automatically generated



We passed 3 argument but why it said 4 were passed. Basically, when you call Employee(), Python creates an object for you, and passes it as the first parameter to the \_\_init\_\_ method.

1. Then, we will add first method for our class. Update the following in the red box to the **oop.py** file.

A screenshot of a cell phone

Description automatically generated

It will print full name and email address of emp1.

1. Add class variable to count how many employees we have so far. Update the file again with the following.

**A screenshot of a cell phone

Description automatically generated**

In the terminal type **python3 oop.py** to see the output.

A picture containing food, drawing

Description automatically generated

You will see the output show number 2 which means two employees have been created so far.

A little more explanation here:

* emp\_count is a class variable whose value is shared among all instances of this class
* You declare other class methods like normal functions with the exception that the first argument to each method is self. Python adds the self-argument to the list for you; you do not need to include it when you call the methods like following example.

**Push your work to GitHub**

Open the terminal from the VSCode by hitting the “control” + “~” key and type the following command:

>>> git add .

>>> git commit -m “Submission for Module 6 – Your Name”

>>> git push origin master