MIT AITI
Android Software Development

Lab 05: Python Classes

Download and install Python version 2.7.5, and write your solutions to the following questions by modifying the solution template which can be found on the AITI website. Submit your answers to the box!

1. Creating the Person class

In the next two problems you will be building an address book using object-oriented programming in Python. To begin, you'll define a Person class. You'll use this class in the next problem to build your address book – each entry in your address book will be an instance of the Person class.

To begin, define the __init__ and __str__ methods for the Person class. Read the Python documentation if needed to help you understand what __init__ and __str__ need to do.

Here's an example in the Python Shell demonstrating how your class should work after implementing these two methods:

```
>>> Markus = Person("vonRudno", "Markus", "87839437045",
"mv331@mit.edu")
>>> print Markus
vonRudno, Markus -- Phone Number: 87839437045 -- Email
Address(es): mv331@mit.edu
>>> Lynn = Person("Yu", "Lynn", "6531111111",
['yeeey@mit.edu', 'cupcakelove@gmail.com'])
>>> print Lynn
Yu, Lynn -- Phone Number: 6531111111 -- Email
Address(es): yeeey@mit.edu, cupcakelove@gmail.com
```

2. Creating the Address Book

In this problem you will use the Person class from the previous problem to create an address book for storing the contacts of your friends and family. It should allow you to search the address book for a friend, and return their contact information.

Begin by defining an __init__ method. Don't forget about Python's built in dictionary datatype.

Next write a method add_contact that allows you to add a new person to the address book.

Finally write a method lookup_contact that looks up a contact by last name. The method should accept the last name as an argument, and print each contact that matches the last name on a new line. As an additional challenge, extend this method to allow users to optionally specify a first name to narrow down the results when multiple contacts have the same last name. Refer to the documentation about variable arguments for more information.

For example, suppose the contact book contains entries for both Markus vonRudno and his wife Nicole vonRudno. The following is an example output:

```
>>> a = AddressBook()
>>> a.add_person(Person("vonRudno", "Markus", … ))
>>> a.add_person(Person("vonRudno", "Nicole", … ))
>>> a.lookup_contact("vonRudno")
vonRudno, Markus -- Phone Number: …
vonRudno, Nicole -- Phone Number: …
>>> a.lookup_contact("vonRudno", "Markus")
vonRudno, Markus -- Phone Number: …
```

```
# Use the following template:
# MIT AITI Indonesia Summer 2013
# File: Python2lab.py
# Below are templates for your answers to Lab 5

# INSTRUCTIONS: Write your complete name in student_name and age in student_age
# Complete the implementation of functions and classes as described in the handout.
# Delete the pass statements below and insert your own code.
student_name = 'Markus von Rudno'
student_age = 22


class Person:
  def __init__(self, lastName, firstName, phoneNumber, emailAddress):
   '''
   Takes in a persons last name, first name, phone number,
    and email address(es).
    '''

  def __str__(self):
    pass


class AddressBook:
  def __init__(self):
    pass

  def add_person(self, person):
    pass

  def lookup_contact(self, lastname):
    '''

    '''
    pass

# Once you have the previous part working, use this function definition
# as a model for the challenge of using optional arguments in Python.
'''
  def lookup_contact(self, lastname, firstname=None):



    pass
'''
```