Lab/Homework 7

Deadline: 23:59 pm, Sunday, Dec 17.

What to submit:

A report with answers to each exercise and corresponding python program (.py file), packaged into a zip file. Named zip as “class_name_HW7 ", for example, “AI1-jason-HW7”. Please submit to TA

Requirements on Coding:

1. Adding header to each .py file.

```
"""
xxxx.py
author:
date:
description:
"""
```

2. Please add a space around the operator and after the comma.
3. Add a blank line between code of different functions
4. Indent your code blocks with 4 spaces. Never use tabs or mix tabs and spaces.

Exercise 7.0 Person Class

Create person_class.py file and write the class Person so that the following test code passes:

```
def testPersonClass():

    print('Testing Person Class...', end='')

    fred = Person('fred', 32)

    assert(isinstance(fred, Person))

    assert(fred.getName() == 'fred')

    assert(fred.getAge() == 32)

    # Note: person.getFriends() returns a list of Person objects who

    #       are the friends of this person, listed in the order that

    #       they were added.

    # Note: person.getFriendNames() returns a list of strings, the

    #       names of the friends of this person.  This list is sorted!

    assert(fred.getFriends() == [ ])

    assert(fred.getFriendsNames() == [ ])



    wilma = Person('wilma', 35)

    assert(wilma.getName() == 'wilma')

    assert(wilma.getAge() == 35)
    assert(wilma.getFriends() == [ ])

    wilma.addFriend(fred)
    assert(wilma.getFriends() == [fred])
    assert(wilma.getFriendsNames() == ['fred'])
    assert(fred.getFriends() == [wilma]) # friends are mutual!
    assert(fred.getFriendsNames() == ['wilma'])

    wilma.addFriend(fred)
    assert(wilma.getFriends() == [fred]) # don't add twice!

    betty = Person('betty', 29)
    fred.addFriend(betty)
    assert(fred.getFriendsNames() == ['betty', 'wilma'])

    pebbles = Person('pebbles', 4)
    betty.addFriend(pebbles)
    assert(betty.getFriendsNames() == ['fred', 'pebbles'])
```

```
barney = Person('barney', 28)

barney.addFriend(pebbles)

barney.addFriend(betty)

barney.addFriends(fred) # add ALL of Fred's friends as Barney's friends

assert(barney.getFriends() == [pebbles, betty, wilma])

assert(barney.getFriendsNames() == ['betty', 'pebbles', 'wilma'])

fred.addFriend(wilma)

fred.addFriend(barney)

assert(fred.getFriends() == [wilma, betty, barney])

assert(fred.getFriendsNames() == ['barney', 'betty', 'wilma']) # sorted!

assert(barney.getFriends() == [pebbles, betty, wilma, fred])

assert(barney.getFriendsNames() == ['betty', 'fred', 'pebbles', 'wilma'])
print('Passed!')
```

Note that your solution must work in general, and not hardcode to these specific test cases.