COMP 142 Project 3: Waiting in Line

Assigned: Friday, September 26
Due: Monday, October 6, 2014 by 11:55pm (on Moodle)

You are already to do some crazy holiday shopping and get the best bargains around! It's 4AM the day after Thanksgiving, and the store doesn't open until 6AM, but already there are people lining up. To keep yourself entertained for the next 2 hours, you decide to implement a program to keep track of what happens in line.

Program Description/Specification

1. You must implement two classes: a class `PersonList` representing a node in a linked list of people, and a class `Line`, representing a line of people as a linked list.
2. `PersonList` class must have:
    a. Two private instance variables: `name` (a string representing the person's name), and `next`, a link to the person who is behind them
    b. `__init__`: the constructor should be called as follows `PersonList(name)`, and it should initialize both instance variables (set next to None)
    c. Two getter methods, one for `getName` that returns the name of the person and one for `getNext` that returns a PersonList object with the next person in line (None if person is last in line)
    d. Two setter methods, one for `setName` and one for `setNext`
3. `Line` class must have:
    a. A single private instance variable called `head`, that points to the first person in the line
    b. `__init__`: the constructor initializes an empty linked list
    c. `__str__`: overload the str method so that when you print a Line object, it prints the people currently in line, in order, with a number indicating their position (starting at 1).
    d. `add(name)`: adds a person to the END of the line
    e. `remove(name)`: removes a person from the line
    f. `joinFriend(first, second)`: modifies the list to:
        1. add person second at the position right after person first. If first is
        2. not in the line, the function must print an error message
4. You should put both classes in the same file.

Testing your Program

You should test your classes using various calls and inputs. However, I will be testing your code by importing your python file into my program and making sure the output is correct. To test your classes and make sure that they are working properly, you can write a main function and call it. For instance, if you add the following code to your file:

```
def main():
    ln = Line()
    ln.add("John")
```

```
    ln.add("Mary")
    ln.add("Alec")
    ln.add("Celia")
    ln.remove("Mary")
    ln.joinFriend("John", "Mike")
    ln.joinFriend("Celia", "Jim")
    print(ln)

main()
```

Your output should be:

```
1 John
2 Mike
3 Alec
4 Celia
5 Jim
```

When you hand in your code file, be sure to remove the main method first! Your file should only contain the 2 classes as described above. Please use the exact names and cases as shown above as well. This will allow me to test your code more efficiently!

Notes

- I would implement the methods of Line starting with the constructor and the add and __str__ functions, and then proceeding to the remove and joinFriend functions once the first three are working.
- You are welcome to implement additional member functions or non-member functions (though you shouldn't need to); however, do not add any additional member variables to either class.
- Please make sure to thoroughly test your functions; this means thinking of what kinds of input might cause your functions to break.

Language

You should write this program in Python. If you are still unsure of writing Python code, please come see me so we can discuss options.

Coding style, comments, and pledge

Please see Program 1 for details on proper coding style, comments and including the pledge in your program header.
In particular, every Class you write should have a high-level comment describing what it does, and each of its method functions should have a comments similar to those of regular functions.

What to turn in

Through Moodle, turn in your code as a file called `wait_yourLastName_yourFirstName.py`.