Exercises for Lab 1

These exercises are to be worked on during lab hours. If you complete them beforehand you will be rather limited in what you can receive help with.

Classes

Exercise 1

Construct a class that represents a Person. A Person must have a name. The Person class must have at least one method which allows the Person to greet another Person.

Example:

```
alice = Person(“Alice”)
bob = Person(“Bob”)
alice.greets(bob)
>>> Alice: “Hello, Bob!”
```

Exercise 2

Create a class Employee that takes firstname and lastname as instance variables as well as a salary that is set to 10000 by default. Create the following methods:

- get_fullname(self), returns the employee’s first and last name as a string.
- print_email(self), prints the employee’s email in the format: firstname.lastname@company.com
- increase_salary(self, rate), multiplies the employee’s salary by the given rate.

Test the class by instantiating an employee and calling the different methods.

Binary Search

Exercise 1

a) Which of the following lists can you perform a binary search on?

1. `[1, 3, 5, 7, 9, 13, 19, 21, 25]`
2. `[5, 3, 7, 9, 0, 1 ,4, 3, 5]`
3. `[2000, 1996, 1994, 1989, 1969, 1952, 1945]`
4. `[“A”, “B”, “C”, “G”, “E”, “H”, “I”, “J”, “K”]`
5. `[“ANTMAN”, “BATMAN”, “BEAST BOY”, “CATWOMAN”, “HAWKGIRL”]`

b) Choose a value from one of the lists in exercise 1.a). Find the value using a binary search on the list. Illustrate the procedure by writing down the values that is searched through in each step of the search. If the size of the list is ***n***, how many steps did the search require?

Exercise 2

(a) Define a function **binary_search_big_o** that takes a list as an argument and returns how many steps binary search would use in the worst case. Hint: math.ceil and math.log could be useful for this, remember to import math.

(b) Define a similar function **simple_search_big_o** that instead returns the big o notation of performing a simple search on the list.

(c) Try out the two functions with various lists and compare the results.

Extra task

What is the big O notation of the following function:

```
def my_function(some_list):
    for item1 in some_list:
            for item2 in some_list:
                    print(item1, item2)
```

1. `O(log n)`
2. `O(n)`
3. `O(n^n)`
4. `O(n^2)`

Extra task 2

Write a Python class to reverse a string word by word.

Here are some example inputs and their desired outputs.

Input string : 'hello world'
Expected Output : 'world hello'

Input string : 'I really love python'
Expected Output : 'python love really I'