Dario Bonino, Luigi De Russis

LAB 1 – PYTHON BASICS

GETTING STARTED...

The goal of this set of exercises is to experiment with the basic concepts of Python and with the Eclipse/PyDev IDE.

If two or more of you share a single computer, adopt the pair programming[1] technique and change role at the beginning of each exercise.

Hint: to create a new Python project in with PyDev, select File > New > PyDev project from the Eclipse menu and assign the desired name to the project. Then, right-click on the newly created project, choose New > PyDev module, and fill the mandatory Name field.

EXERCISE 1 - PRIMES

Write down a program for detecting if a number is a prime.

EXERCISE 2 - FIBONACCI

Implement a program that given a number n, writes on the console the Fibonacci's series of order n. The number must be interactively inserted by the user.

In mathematical terms, the sequence Fn of Fibonacci numbers is defined by the recurrence relation:

$$F_n = F_{n-1} + F_{n-2}$$

with seed values

$$F_1 = 1, F_2 = 1.$$

EXERCISE 3 - BOTH ENDS

Given a string s, return a string made of the first two and the last two chars of the original string.

e.g., ***'spring'*** *yields* ***'spng'***

If the string length is less than two, return the empty string.

[1] In pair programming, two programmers work as a pair, together on one computer. One, the driver, writes code while the other, the navigator, reviews each line of code as it is typed in and helps plan and catch errors.

EXERCISE 4 - FIX START

Given a string s, return a string where all the occurrences of its first char have been changed to '*', except the first char. The string must be interactively inserted by the user.

e.g. ***'babble'*** *yields* ***'ba**le'***

Assume that the string length is 1 or more.

Hint: s.replace(stra, strb) returns a version of string s where all instances of stra have been replaced by strb.

EXERCISE 5 - MATCH ENDS

Given a list of strings, return the count of the number of strings where the string length is 2 or more and the first and last chars of the string are the same. *Note: Python does not have a ++ operator, but += works.*

Example:

Input = ['aba', 'xyz', 'aa', 'x', 'bbb']

Output = 3

EXERCISE 6 - FRONT X

Given a list of strings, return a list with the strings in sorted order, but grouping all the strings that begin with 'x' first.

Hint: this can be done by making two lists and sorting each of them (with the sort() method) before combining them.

Example:

Input = ['mix', 'xyz', 'apple', 'xanadu', 'aardvark']

Output = ['xanadu', 'xyz', 'aardvark', 'apple', 'mix']

EXERCISE 7 - FIND PLAYERS

Given a "2D dictionary" of people, like this one (i.e., people):

sam = {'age': 10, 'height': 42, 'weight': 175, 'instrument': 'fiddle' }
mary = {'age': 41, 'height': 70, 'weight': 160, 'instrument': 'piano'}
bertha = {'age': 32, 'height': 97, 'weight': 587, 'instrument': 'cello'}
david = {'age':100, 'height': 4, 'weight': 0.5, 'instrument': 'cello'}
people = {'Sam': sam, 'Mary': mary, 'Bertha': bertha, 'David': david}

return a new dictionary that contains only the people who play a certain instrument.

Example:

> find_players cello

{"Bertha": {"age": 32, "height": 97, "weight": 587, "instrument": "cello"}, "David": {"age":100, "height": 4, "weight": 0.5, "instrument": "cello"}}

EXERCISE 8 - WORD COUNT

Implement a "*print_words filenam*e" program that, given a text file read from command line, counts how often each word appears in the text and prints the results in the form:

word1 count1

word2 count2

You can use the provided *alice.txt* and *python-zen.txt* files for this exercise.