Python Learning Program

python -c "import this"

Abstract

This document gives you a set of resources that you can study to improve your Python knowledge. It also lists some exercises meant to give you some hands-on practice. The exercises not only test your understanding but should also generate valuable discussions with your mentor. Because of this you should try to get feedback on your progress as often and as early as possible. This goes without saying but you should also incorporate the feedback you get on the rest of the exercises.

Online Resources

The Python Tutorial is the best resource for diving into Python and should be enough to solve the exercises in this document. Another excellent resource for understanding and visualizing your program execution is Python Tutor. For advanced topics you can check out The Python Language Reference - more precisely the Data Model and Execution Model chapters. If you need in-depth documentation about a certain feature checkout the Python PEPs. The standard library documentation may also come handy for some exercices. There are many more other great resources online so feel free to do your own research - the one listed here should be enough to build a solid foundation.

Exercices

Intro

1. Write a script that reads your name from the console and prints it reversed in upper-case letters.
2. Write a script that checks if a given number is palindrome (eg. 2, 33, 13231). Do not use strings.
3. Write a script that prints all the palindrome numbers lower than a given number.

Functions

1. Write a function that reads a number from the console and checks if it's prime.
2. Write a function that finds all the prime numbers lower than a given number.
3. Write a function that finds the fibonacci sequence up to n elements iteratively and another one recursively.
4. Write a function that emulates the "map" functionality: it should take another function **`func`** and an iterable **`iterable`** as arguments and should return an iterable of the results of applying **`func`** over each value in **`iterable`**.
5. Write a function that emulates the "filter" functionality: it should take another function **`func`** and an iterable **`iterable`** as arguments and should return an iterable of those values in **`iterable`** that returned a truthy value when passed to **`func`**.
6. Write a function that emulates the "reduce" functionality: it should take another function **`func`**, an iterable **`iterable`** and an initial value **`initializer`** as arguments. It should return the final result of repeatedly applying **`func`** over the

previous result (or **`initializer`** for the first time) and the next element in **`iterable`**.

7. Use map and filter and reduce to find the sum of all numbers **`n`** lower than a given value which satisfy the following constraint: **`n`** * **`n`** - 1 divisible by 3 (eg. 4 * 4 - 1 == 15 and 15 div with 3).
8. Write a function that sorts a list of values.
9. Write a function that searches for a value in a list of already sorted values.
10. Implement stateful function that starts from 0 and returns successive values each time it's called without using global variables (eg. a() == 0; a() == 1; a() == 2).
11. Write a decorator that can be used to measure the execution time of functions. It should print the duration on the console.
12. Write a decorator that can be used to measure the total execution of the decorated functions.
13. Write a memoization decorator to speed up the recursive fibonacci function.

Input/Output

1. Compute the sum of all the numbers in a file. The file contains a number on each line.
2. Write all the prime numbers lower than a given number in a file, one number per line.
3. Find all the words used only once in a file containing text.
4. Write a program that reads a number of files containing sorted numbers (one number per line) and outputs a large file with all the numbers from all the files sorted.
5. Extract all the links in a website on a given URL.

OOP

1. Implement a Queue.
2. Design a configuration reader library API and implement it. The reader should read a configuration file from disk and expose its values through the API you choose.
3. Design a logging library API and implement it.