Notes on Chapter 4 - Functions, Scoping and Abstraction

Swarup Tripathy *

February 2022

A curated list of important points for my reference.

1. Parameters inside functions provide something called ***Lambda Abstraction***, allowing programmers to write code that manipulates not specific objects , but instead whatever objects the caller of the function chooses to use as actual parameters.

2. In python, there are two ways that formal parameters get bound to actual parameters.

   - The most common method is called the **positional** – the first formal parameter is bound to the first actual parameter, the second formal to the second actual
   - python also supports **keyword arguments**, in which formats are bound to actuals using the name of the formal parameter.
   - positional arguments cannot appear after keyword arguments.

3. Most of the time you will find that you only want to use variables that are local to a function, and the subtleties of scoping will be irrelavant.

4. Experienced programmers know, however, that an investment in writing testing code often pays big dividends. It certainly beats typing test cases into the shell over and over again during debugging.

5. Writing help(function name), the system will display the help on the built-in-function.

6. Abstraction in Python is the process of hiding the real implementation of an application from the user and emphasizing only on usage of it.

*John V Guttag

7. Fibonacci's great contribution to European mathematics was his book *Liber Abaci*, which introduced european mathematicians many concepts already well known to Indian and arabic scholars. These concepts included Hindu-Arabic numerals and the decimal system. What we today call the Fibonacci sequence was taken from the work of the **Sanskrit mathematician Pingala**.

8. Fibonacci Numbers

    - The fib(k - n + 1) will give number of times fib(n) called when calculating fib(k) recursively, where k ¿ n and this works for n = 0 as well.