Task: Introduction to Functions

www.hyperiondev.com

Introduction

Welcome to the Introduction to Functions Task of the Certified Software Developer Course!
Overview:

This task is aimed to ensure that you have a concrete understanding of Strings, Functions and basic List manipulations, as these will be needed for upcoming more advanced tasks. In example.py, you will see examples that deal with functions and operations that can be applied to elements in lists. The Task also re-introduces Functions and how they can be used to compute certain values on list elements as well as printing these results to text files.

-The Hyperion Team

A note from the Hyperion Team...

Functions make life easier! They do this in many ways such as code reusability, or simply to create a better code structures that could compare to that of a veteran programmer.

The code below is an example of how you could compare the absolute values of two numbers. What you'll soon realise is that to repeat the comparison, you'd have to create all that code again, which is simply a waste of time.

```
a = 23
b = -23

if a < 0:
    a = -a
if b < 0:
    b = -b
if b > 0:
  elif:
    print("ok")
    print("Nope")
```

Below is code for a program to carry out the exact same comparison. You can immediately notice key points such as a reduction in length, infinite repetitions allowed, and improved code structure. All of this is achieved by simply implementing a function which took four lines to create!

```
def absolute_value(n):
    if n < 0:
        n = -n
    return n

a = 23
b = -23

if absolute_value(a) == absolute_value(b):
    print "The absolute values of", a, "and", b, "are equal"
else:
    print "The absolute values of", a, "and", b, "are different"
```

By now you're most likely beginning to see the picture. Functions can be compared to technology, it makes your life easier!

Below is a basic function based program using a few basic functions to create a program that can operate with different inputs.

```
def hello():
    print "Hello"

def area(w, h):
    return w * h

def print_welcome(name):
    print "Welcome", name

hello()
hello()

print_welcome("Fred")
w = 4
h = 5
print "width =", w, "height =", h, "area =", area(w, h)
```

A further application of functions can be found in the example programs folder of this task called Temperature.py which is a program that converts temperate from Celsius to Farenheit and vice versa. Once you've read through example.py, take a look at this example program and see if you can understand how it works!

Instructions

First read example.py, open it using Notepad++ (Right click the file and select 'Edit with Notepad++').

- Example.py should help you understand some simple Python. Every task will have example code to help you get started. Make sure you read all of example.py and try your best to understand.
- You may run example.py to see the output. The instructions on how to do this are inside the file. Feel free to write and run your own example code before doing this task to become more comfortable with Python.
- You are not required to read the entirety of Additional Reading.pdf, it is purely for extra reference.

Compulsory Task 1

Follow these steps:

- Create a Python file called “functionPractice.py” in this folder.
- Define a function sumAll(n) which will use a for loop to returns the sum of all numbers from 1 to n.
- You’ll also need to use a sum variable that increases in value over each loop iteration

Compulsory Task 2

Follow these steps:

- Create a Python file called “amazon.py” in this folder.
- Write code to create user-defined functions for the following operations: Minimum, Maximum, Sum, Average, and Standard Deviation. These functions should run on a defined list, and write the outputs to a text file called output.txt.
- Define a list of values as follows:
  list = [1, 5, 8, 77, 24, 95] (you can make this list as long as you want, just remember the commas)

  Your program should generate output.txt as follows:
  The min of [1, 5, 8, 77, 24, 95] is 1.
  The max of [1, 5, 8, 77, 24, 95] is 95.
  The avg of [1, 5, 8, 77, 24, 95] is 35.
- Your program should handle any length of input numbers. You can assume that the list of input numbers are always valid integers and the list is never empty.

Things to look out for

1. Make sure that you have installed and setup all programs correctly. You have setup **Dropbox** correctly if you are reading this, but **Python or Notepad++** may not be installed correctly.
2. If you are not using Windows, please ask your tutor for alternative instructions.

Still need help?

Just write your queries in your comments.txt file and your tutor will respond.

Task Statistics

Last update to task: 23/12/2015.
Authors: Riaz Moola and Jared Ping.
Main trainer: Umar Randeree.
Task Feedback link: Hyperion Development Feedback.