uniandes

Herramientas Computacionales
Tarea 5 - Python: Funciones y Recursividad
01-2018

uniandes

La solución a este taller debe subirse por SICUA antes de terminada la clase. Los archivos código fuente deben subirse en un único archivo `.py` con el nombre `NombreApellido_hw5.py`, por ejemplo yo debería subir el zip `JesusPrada_hw5.py`. Recuerden que es un trabajo individual y debe ser realizado en un script de python (.py).

Aclaraciones sobre python: El alcance(scope) de las variables

En python y en general en la mayoría de lenguajes de programación hay dos tipos de variables importantes: las variables **LOCALES** y las variables **GLOBALES**.

Las variables **globales** son declaradas por fuera de funciones. **Una buena manera de saber si una variable es global es si no está indentada**. Estas variables son accesibles por todos las funciones, ciclos, etc.

Las variables **locales** son declaradas adentro de una función, o como parámetro. Estas variables son accesibles únicamente adentro de la función en la que fueron declaradas.

Ahora, en python la sintaxis intuitiva y poco exigente, en donde **la asignación de variables automáticamente las declara**, se presta para confusiones con las variables locales y globales. Es decir, en python es posible definir una variable global y luego definir una variable local con el mismo nombre. En este caso, no es claro si la variable queda local o global. Demostrémoslo!

1. (0.5 points) **Comentarios**

   Por favor comenten todo su código. Específicamente, a cada variable importante o cuyo significado no es evidente deben comentarle su significado. A cada función deben comentarle su propósito principal y el significado de sus parámetros de entrada y sus retornos. A cada iteración deben comentarle su objetivo. Se acepta declarar variables con nombre evidente como `number` en lugar de comentar su significado.

2. (0.2 points) **Declarar variables globales**

   Declare una variable global llamada `var1` y otra llamada `var2`. Estas variables deben tener el contenido `'variable 1 global'` y `'variable 2 global'` .

3. (0.2 points) **Declarar una variable local (1)**

   Como vimos anteriormente, hay dos maneras de declarar una variable local. Una es asignándole un valor adentro de la función.

   Defina una función llamada `fun1()`. Adentro de la función declare una variable local llamada `var1` con el contenido `'variable 1 local'`. Inmediatamente después esta función debe **IM-**

PRIMIR la variable `var1`.

Cabe aclarar que definir una función **NO** la ejecuta. El contenido de una función es ejecutado únicamente al **LLAMAR** la función, no al **DEFINIRLA**.

4. (0.2 points) **Declarar una variable local (2)**

   Otra manera de declarar una variable local es declarándola como parámetro de una función.

   Defina una función llamada `fun2(var2)` que tome como parámetro una variable llamada `var2` y la **IMPRIMA**.

5. (0.5 points) **Globales o locales?**

   Imprima `'Ejecutando fun1()'`, luego, en el siguiente renglón, **EJECUTE** la función `fun1()`. Imprima `'Imprimiendo var1'`, luego, en el siguiente renglón, imprima la variable `var1`.

   Imprima `'Ejecutando fun2()'`, luego, en el siguiente renglón, **EJECUTE** la función `fun2('variable 2 local')`. Imprima `'Imprimiendo var2'`, luego, en el siguiente renglón, imprima la variable `var2`.

   Ejecute el código.

   Hemos declarado una variable global y luego hemos declarado una variable local con el mismo nombre (de dos maneras distintas). Imprima un mensaje en el que comente si las variables son locales, globales o ambas, según lo que usted concluyó al correr el código.

6. (0.4 points) **Se puede asignar un valor a una variable global dentro de una función?**

   Como vimos anteriormente, al asignarle el valor a una variable adentro de una función, se está declarando como local. Esto significa que esa asignación sólo es válida dentro de la función. Esta confusión se puede arreglar trabajando con nombres diferentes de variables. Es decir, si se declara una variable (global) por fuera de una función, es mejor NO declarar/asignar una variable (local) con el mismo nombre adentro de la función.

   Sin embargo, a veces es necesario asignar o cambiar el valor de una variable global adentro de una función. Recordando que asignar el valor de una variable adentro de una función la declara como local, esto es un problema que no se puede solucionar cambiando el nombre de las variables. En caso de que se quiera modificar una variable global **adentro** de una función, es necesario, antes de modificarla, recordarle a python que se modificará la variable global y no una versión local. Para esto, se usa la siguiente línea de código para la variable `var` **ADENTRO** de la función:

   ```
   global var
   ```

   Declare una función llamada `fun3()`. Adentro de la función modifique la variable global llamada `var1` con el contenido `'variable 1 global modificada'`. Inmediatamente después esta

función debe imprimir la variable `var1`.

Imprima `'Ejecutando fun3()'`, luego, en el siguiente renglón, **ejecute** la función `fun3()`. Imprima `'Imprimiendo var1'`, luego, en el siguiente renglón, imprima la variable `var1`. Note la diferencia con el punto anterior y déjela expresada en un mensaje que imprimirá.

Ejecute el código.

Cabe aclarar que establecer una variable `var` como praámetro de una función automáticamente la hace local. Dado que `global var` tiene que ser ejecutado antes de que la variable sea declarada local, modificar una variable global llamada `var` dentro de una función que tiene como parámetro la variable `var` es imposible.

Recursividad

7. (2.5 points) **La conjetura de Collatz**

   Podemos generar una secuencia de números a partir de un número `n` de la siguiente manera:

$$a_0 = n$$

$$a_{i+1} = \begin{cases} \frac{a_i}{2} & \text{si } a_i \; es \; Par \\ 3a_i + 1 & \text{si } a_i \; es \; Impar \end{cases}$$

   La conjetura establece que no importa cuál sea el `n` que escojamos, la secuencia eventualmente caerá en el ciclo `4,2,1,4,2,1,4,2,1.....`

   Cree una función **RECURSIVA** llamada `Collatz()` que imprima la secuencia asociada a un número `n`. La función debe parar cuando $a_i = 1$.

8. (0.5 points) **Las primeras 20 secuencias**

   Imprima la secuencia de los primeros 20 enteros, una debajo de otra. Para evitar confusiones y por orden, antes de llamar la función `Collatz()` imprima un mensaje que especifique sobre qué número `n` se calcula la secuencia.