Programando con Python

Clase1 - Ejercicios

Introducción al IDE Spyder: http://ligdigonzalez.com/ide-spyder-para-python/

Tutorial de Pycharm: https://www.tutorialspoint.com/pycharm/pycharm_keymaps.htm

Python Tutor: http://www.pythontutor.com/

1. Iteración-Control de Flujo

A continuación se presentan algunos programas cortos de Python. Responde cual es la salida de cada programa. Intenta responder las preguntas sin ejecutar el código.

a. Usando while

Qué imprimen estos *loops* simples?

```
num = 0
while num <= 5:
    print(num)
    num += 1

print("Fuera del loop")
print(num)
```

```
numberOfLoops = 0
numberOfApples = 2
while numberOfLoops < 10:
    numberOfApples *= 2
    numberOfApples += numberOfLoops
    numberOfLoops -= 1
print("Número de manzanas: " + str(numberOfApples))
```

```
num = 10
while num > 3:
    num -= 1
    print(num)
```

```
num = 10
while True:
    if num < 7:
        print('Saliendo del loop')
        break
    print(num)
    num -= 1
print('Fuera del loop')
```

```
num = 100
while not False:
    if num < 0:
        break
print('num is: ' + str(num))
```

b. Usando for

Convierta los siguientes códigos en uno que usa el *loop* for

```
prints 2
prints 4
prints 6
prints 8
prints 10
prints "Adios!"
```

```
prints "Hola!"
prints 10
prints 8
prints 6
prints 4
prints 2
```

2. Usando Strings

a. Escribir un programa que cuente la cantidad de vocales contenidas en el string **s**. Por ejemplo. Si **s** = **'azcbobobegghakl'**, el programa debe imprimir Número de vocales: 5
b. Escribir un programa que compare dos *strings* **s1** y **s2** e imprima las letras que tienen en común. Ej. **s1** = 'curso de python' y **s2**= 'programando con python'.
c. Cargar por teclado una cadena de caracteres. Mostrar después los dos primeros, los dos últimos caracteres de dicho *string* y por último mostrar el *string* sin el primer y último carácter.
d. Realizar un programa que pida la carga por teclado de una cadena de texto. Informar después si la palabra es capicúa o no (son aquellas que leyéndolas en ambos sentidos son iguales, por ejemplo: elle, erre o acurruca).

3. Funciones

a. Sean las variables x=1 e y=2, realizar el intercambio de sus valores de forma a obtener x=2 e y=1.

b. Para cada una de las siguientes funciones, especifique el tipo de su devolución. Puede asumir que cada función se llama con un argumento apropiado, según lo especificado por su cadena de documentación (docstring).

```
1. def a(x):
    '''
    x: int or float.
    '''
    return x + 1

2. def b(x):
    '''
    x: int or float.
    '''
    return x + 1.0

3. def c(x, y):
    '''
    x: int or float.
    y: int or float.
    '''
    return x + y

4. def d(x, y):
    '''
    x: Can be of any type.
    y: Can be of any type.
    '''
    return x > y

5. def e(x, y, z):
    '''
    x: Can be of any type.
    y: Can be of any type.
    z: Can be of any type.
    '''
    return x >= y and x <= z

6. def f(x, y):
    '''
    x: int or float.
    y: int or float
    '''
    x + y - 2
```

c. A continuación se muestra una transcripción de una sesión con el shell de Python. Proporcione el tipo y valor de las expresiones que están siendo evaluadas. Si evaluar una expresión causa un error, seleccione Ninguno y escriba 'error' como respuesta. Si el valor de una expresión es una función, seleccione la función como tipo y escriba "función" como respuesta.

1. `a(-5.3)`
2. `a(a(a(6)))`
3. `c(a(1), b(1))`
4. `d('apple', 11.1)`
5. `e(a(3), b(4), c(3, 4))`
6. `f`

4. Listas y Tuplas

a. Cree una lista con números del 1 al 100.
b. Realice un programa que contenga una lista con 10 valores enteros. Informar cuántos de ellos son superiores a 100. Ej. `lista=[213,44,578,123,100,56,78,1000,2345,1278]`
c. Realice un programa que permita crear una lista y almacenar los nombres de n países. Ordenar alfabéticamente la lista e imprimirla.
d. Realice un programa que permita cargar una lista de n elementos enteros. Ordenarla de mayor a menor e imprimir los resultados obtenidos. Informar también del número mayor y del número menor.
e. Realice un programa que cree una lista de 50 elementos. El primer elemento es una lista con un entero, el segundo es una lista con dos enteros, etc. La lista debería tener la siguiente estructura:
`[[1], [1,2], [1,2,3], …]`
f. Cree una tupla con los meses del año. Pide un número al usuario, si el número está entre 1 y la longitud máxima de la tupla, muestre el contenido de esa posición, sino muestre un mensaje de error. El programa termina cuando se introduce cero.
g. Ingrese un número por teclado y guarde en una lista su tabla de multiplicar hasta 10.
h. Escriba una función que reciba una lista de *tuplas* (Apellido, Nombre, Inicial_segundo_nombre) y devuelva una lista de *strings* donde cada una contenga primero el nombre, luego la inicial con un punto, y luego el apellido.

5. Diccionarios

a. Realizar un programa que permita crear un diccionario inglés / castellano. La clave es la palabra en inglés y el valor es la palabra en castellano. Crear funciones para cargar el diccionario, listar el diccionario. Crear una tercera función que solicite la carga por teclado de una palabra en inglés, mostrar después el valor en castellano si este existe.
b. Crear un diccionario que defina como clave el número de Cédula de identidad (CI) de una persona y como valor un *string* con el nombre. Cargar los datos de 4 personas, listar

el diccionario completo y realizar una tercera función para consultar el nombre de la persona introduciendo su CI.

c. Realizar un programa que permita cargar un código de producto como clave en un diccionario. Guardar para esta clave el nombre del producto, el precio y el total disponible en stock. Después de ingresar un producto, preguntar si se desea cargar otro, el límite de productos lo dispondrá el usuario. Mostrar el listado de productos, una tercera para consultar un producto mediante su clave y otra para mostrar el listado de productos con stock 0.

d. Realizar un programa de una agenda. En el diccionario la clave sería la fecha. Para cada actividad ingresar la hora y el nombre de dicha actividad y permitir la introducción de varias actividades para cada fecha. Implementar funciones para cargar datos, listar la agenda al completo y consultar una fecha.

e. Desarrollar un programa que permita almacenar los datos de n alumnos. Definir un diccionario cuya clave será el número de alumno en el centro. Como valor almacenar una lista con el nombre, la materia a la que está apuntado y la nota obtenida en dicha materia. Un alumno puede estar apuntado a diferentes materias. Implementar funciones para cargar los datos de los alumnos, listar todos los alumnos y consultar un alumno a través de su id.