Seminario de Lenguajes
Opción Python
Práctica 1

La versión de Python que utilizaremos en la cursada es la 3.x, la cual puede ser descargada desde http://www.python.org/download/
Las prácticas presentadas en la cursada podrán ser realizadas utilizando cualquier editor de texto (EditPlus, NotePad++, gedit, etc) o IDE (IDLE, Eclipse, NetBeans, PyCharm, etc) que consideren conveniente sobre la plataforma que sea de su agrado (Windows, Linux, etc)
Importante: La evaluación de las entregas se realizará sobre el sistema Lihuen instalado en las salas de PC, por consiguiente es importante que prueben sus aplicaciones en ese sistema (aunque no es obligatorio que lo usen como entorno de desarrollo si es que no quieren)

Conceptos básicos

1.- ¿Qué quiere decir el hecho de que Python es un lenguaje interpretado? ¿Cuáles son sus ventajas y desventajas? Compare con Pascal o con otro lenguaje que haya utilizado.
2.- ¿A qué se refiere con que el lenguaje sea multiplataforma? ¿Cómo es esto posible en Python?
3.- ¿Por qué es software libre? ¿Qué ventajas ven en esta característica?
4.- Python es de tipado dinámico. ¿Qué significa eso?

Primero pasos...

5.- Ejecute las siguientes instrucciones:

```
print("Hola mundo!")
print("Este es mi primer script")
```

Ahora modifíquelo de la siguiente manera:

```
print("Hola mundo! Este es mi primer script")
```

a) ¿Qué hace falta para que esa línea tenga el mismo efecto que la anterior?

b) ¿Qué otros caracteres de control conoce? Muestre un ejemplo en el que con una única sentencia se genere la siguiente salida:

```
*************************
Uno         Dos         Tres
Cuatro      Cinco       Seis
*************************
```

6.- a) Escriba un script que almacene dos números en dos variables e imprima el resultado de la suma de ambos.

b) ¿Qué ocurre si asigna los números a las variables con **“”**, por ejemplo `num2 = "3"`? ¿Qué devuelve la suma?

c) ¿Qué sucede si en vez de la suma queremos hacer la multiplicación en a) y b)?

d) ¿Qué devuelve si al final de la sentencia de impresión de a) y b) agregamos `* 5`?

7.- Escriba un script que reciba desde la entrada de datos estándar (teclado) una letra o número y lo devuelva de la siguiente manera:

```
print("El caracter ingresado es: " + caracter)
```

Ayuda: investigue el comando **`input()`**

8.- a) Modifique el ejercicio 6) para que los valores a ser sumados sean datos ingresados por el usuario, indicando en la pantalla que debe ingresar un número por teclado y luego otro.

b) ¿El resultado fue el esperado? ¿Qué debería modificar para que dé lo esperado (suma de números)?

9.- a) Implemente una calculadora simple, en donde se ingrese (por entrada estándar) dos operandos y el operador (+, -, *, /) e imprima el valor de la operación resultante.

Nota 1: por el momento, no tenga en cuenta errores de tipos – ej: que el operando no sea número o que el operador no esté entre los enumerados.

Nota 2: El código no debe contener condicionales (**`if`**, **`elif`**, etc) ni bucles (**`for`**, **`while`**, etc)

Ayuda: Investigue el comando **`eval()`**

b) A la calculadora implementada en a) agréguele la capacidad de calcular la potencia, parte entera y resto de la división. ¿Tuvo que modificar el código realizado en a)?

c) Modifique la calculadora para que reciba dos números enteros y un operando, que pueden ser &, | o ^ y corresponden a AND, OR y XOR respectivamente, los transforme en números binarios y calcule la operación ingresada y devuelva el resultado del cómputo tanto en decimal como en binario.

10.- a) Escriba un script que reciba un carácter por entrada estándar e imprima su correspondiente código ASCII en pantalla.

b) Haga la inversa de a), reciba un número y devuelva a que carácter corresponde ese código ASCII

11.- Salida estándar – **`print`**

a) Explique qué hace la siguiente línea de código:

```
print("%15s %s Python" % ("Seminario", "de"))
```

b) Suponga que tengo la variable `numero_flotante = 7.55689245` y quiero que se

impriman sólo el número con 5 dígitos, ¿cómo debería formular la instrucción **print**?

c) Dados el nombre de un alumno y su promedio, investigue cómo realizar la impresión de los datos en pantalla sin tener que convertir el valor numérico a string.
Es decir, sin imprimirlo de la forma:

```
print(nombreAlumno + str(promedio))
```

ni `print(nombreAlumno, promedio)`

d) ¿Cómo podría lograr con la instrucción **print** la siguiente salida?

```
Fernandez, Gaston        5.56
```

Sin completar con blancos manualmente luego del nombre y con la información de la siguiente manera: `("Fernandez", "Gaston", 5.56)`

Listas, Tuplas y Diccionarios

12.- Listas http://banyut.obolog.es/python-listas-115312

Dada la siguiente lista:

```
lista = ['elemento 1', 2, 'elemento 3', 'elemento 4']
```

a) Imprima el primer y último elemento

b) Que retorna `lista[-1]`?

c) Imprima todos los elementos

d) Imprima los elementos en las posiciones 0, 1 y 2

e) Imprima ahora los últimos 3 elementos (sin saber a priori cuantos tiene la lista)

f) En la segunda posición ingrese el valor “elemento 2”

g) ¿Qué sucede si ejecutamos el siguiente código? ¿Por qué? ¿Qué debería hacer para agregar un nuevo elemento al final de la lista?

```
lista[4] = 'elemento 5'
```

h) Elimine el elemento creado en g)

i) ¿Cómo puedo hacer para retornar el valor del último elemento y eliminarlo de la lista en una sola instrucción? ver pop

j) Retorne el índice que tiene el elemento con el valor: **'elemento 4'**

k) Ordene en forma inversa la lista e imprima todos los elementos

l) Cree un string que contenga al menos cinco palabras, sepárelas y ordénelas alfabéticamente y en orden inverso. Imprima cada resultado y la cantidad de palabras del string. Por ejemplo, utilizar la siguiente frase como string: “Python es un lenguaje multiplataforma y de tipado dinamico”.

13.- Tuplas

a) ¿Qué son las tuplas? ¿Cómo se definen? ¿Cuál es la diferencia principal con las listas?

b) Sea la siguiente tupla:

Una tupla no puede modificarse de ningún modo después de su creación.

```
tupla = (1, [3, 4, 5])
```

I. Diga cómo está compuesta la tupla.

Las tuplas son objetos inmutables que se asemejan a las listas: se tiene acceso a sus componentes por indexación. Las tuplas son representadas por parentesis y, dentro de ellos, los valores de las componentes (sean o no del mismo tipo),Las tuplas también pueden ser representadas por los valores de las componentes de ellas separados por coma (es decir, omitiendo los parentesis),Por supuesto, al ser inmutable, su valor no puede ser cambiado y el cambio por indice no es permitido

II. ¿Cuántos elementos tiene? Demuéstrelo mediante un script

III. ¿Se pueden cambiar los valores de algún elemento? ¿Se puede agregar un elemento nuevo a la lista? Una tupla es una lista inmutable. Una tupla no puede modificarse de ningún modo después de su creación.

IV. ¿Para qué son especialmente útiles las tuplas?

Las tuplas son más rápidas que las listas. Si está usted definiendo un conjunto constante de valores y todo lo que va a hacer con él es recorrerla, utilice una tupla en lugar de una lista. Las tuplas pueden utilizarse como claves en un diccionario, pero las listas no.Las tuplas se utilizan para formatear cadenas, como veremos en seguida.

14.- Conjuntos

Python provee el tipo de dato set (conjunto) el cual se compone de una lista no ordenada de datos en la cual no hay ningún dato repetido. La forma de creación de un conjunto en python es la siguiente:

a) ¿Qué son los sets en Python?

b) ¿Qué tipo de operaciones se pueden hacer sobre ellos? incluida la unión (|), intersección (Y), diferencia (-), diferencia simétrica (^). Se trata de operaciones inusuales

c) ¿Qué particularidad tienen estas estructuras de datos? Los conjuntos son colecciones de elementos únicos pero no ordenados. Es posible convertir ciertos iterables en un conjunto. {"esto", "es", "un", "conjunto"}

d) Sea el siguiente código:

```
conjunto = set([1, 2, 3, 4, 5, 6])
```

I. Agregue el elemento 7 e imprima los elementos del conjunto

II. Agregue el elemento 4 e imprima los elementos del conjunto. ¿Hubo alguna modificación? ¿Por qué?

III. Realice la intersección, unión y diferencia del conjunto dado con este otro:

```
set([3, 4, 5, 10, 15])
```

15.- Diccionarios

a) ¿Qué son los diccionarios? ¿Qué diferencias tienen con las estructuras vistas anteriormente?

b) Sea el siguiente diccionario:

```
edades = {25: 'Juan', 33: 'Flor', 17: 'Pedro', 40: 'Alberto'}
```

En donde la clave identifica a la edad y el valor al nombre

I. Imprima la cantidad de elementos del diccionario

II. Imprima el nombre de la persona que tenga 33 años

III. Imprima todas las edades de las personas del diccionario

IV. Imprimir todas las personas del diccionario

V. Imprimir si una determinada edad está en el diccionario

VI. Modifique el nombre de Juan por Matias

VII. Agregue a Pedro con una edad de 44 años

VIII. Elimine el elemento ingresado en el punto anterior

IX. Elimine todos los elementos del diccionario

Un diccionario Python es una estructura de datos en la cual se asocia explícitamente un valor de un conjunto de índices (denominados claves) a un valor de otro conjunto de datos (llamado valor) constituyéndose un conjunto de pares (clave, valor). El conjunto de los primeros elementos del par es el conjunto de claves (keys) y sus elementos son únicos (no hay repetidos) y no modificables, es decir no mutables ; el conjunto de los segundos elementos del par es el conjunto de valores y pueden ser cualquier tipo Python.

La diferencia principal entre los diccionarios y las listas o las tuplas es que a los valores almacenados en un diccionario se les accede no por su índice, porque de hecho no tienen orden, sino por su clave, utilizando de nuevo el operador [].