UNIVERSIDAD
DE GRANADA

Python: Primeros programas en Python
Práctica 1 – Guión 6

Informática Aplicada a la Biología

Dpto. Ciencias de la Computación e Inteligencia Artificial
E.T.S. de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Índice de contenido

1. Preliminares.

“Lo que tenemos que aprender a hacer, lo aprendemos haciéndolo”.
Aristóteles

“In theory, there is no difference between theory and practice. But, in practice, there is”.
Jan L. A. van de Snepscheut

“The gap between theory and practice is not as wide in theory as it is in practice”.

“Theory is when you know something, but it doesn’t work. Practice is when something works, but you don’t know why. Programmers combine theory and practice: Nothing works and they don’t know why”.

2. Introducción

El objetivo de esta práctica es que el alumno se familiarice con el software Anaconda Python 3.7 y aprenda a construir expresiones y pequeños scripts.

El alumno utilizará el *Seminario de Introducción al entorno y el lenguaje* (en un principio) y las *transparencias de teoría*, así como la ayuda integrada en el entorno como guía para investigar los elementos básicos de la interface y resolver la relación de ejercicios propuesta.

Todos los ejercicios deben realizarse con el editor de Spyder, en el correspondiente fichero-py con nombre P1_<num_ejercicio>.py para no perder lo hecho (Ejemplo: P1_1.py P1_2.py P1_3.py para los ejercicios 1, 2 y 3 respectivamente y así en adelante).

Por tanto, para el correcto desarrollo de esta sesión de prácticas, se deben realizar los siguientes pasos:

- **Debemos abrir Spyder.**
- **Elegimos como Carpeta de Trabajo el escritorio o una carpeta que hayamos creado para trabajar (este paso es importante).**
- **Recuerde usar `print` para que los cálculos (o resultados) sean visibles al ejecutar los scripts.**
- **Subiremos todos los archivos generados (incluso los no acabados) en la tarea creada en PRADO para cada sesión antes de irnos.**

Muy importante:

- La resolución de los problemas y actividades puede hacerse en grupo, pero la entrega durante las horas de prácticas es individual.
- La nota final de cada sesión dependerá del número de ejercicios entregados. Sin embargo, es preferible entregar solamente aquéllos que se sepan resolver.
- En caso de detectar algún tipo de copia, o una implementación que no se sepa explicar con claridad, se evaluará con un 0.
- Es muy importante que la asignatura se lleve al día para poder realizar los ejercicios propuestos en estos guiones.
- Debemos tener instalado *Anaconda* en casa tanto para realizar trabajo autónomo, como para la resolución del trabajo autónomo de Python.

3. Ejercicios obligatorios

3.1 Bloque 1: Ejercicios Básicos

1. Prueba a utilizar la sentencia **`print()`** de Python mostrando un mensaje de tu elección en la terminal. Cuidado con la sintaxis, utiliza siempre las mismas comillas al abrir y cerrar la cadena.
2. Escriba en Python las siguientes expresiones. Utilice el valor de las constantes PI y e de la biblioteca matemática correspondiente. Fíjese que el símbolo decimal es el punto.

   a) $\frac{2^5*(-4)^3*6^3}{9^2*12^2*(-20)^2} = -0.09481481481481$

   b) $\frac{3-\sqrt[3]{3}}{3+\sqrt[3]{3}} = 0.3506670224259358$

   c) $PI^{3e} = 11328.708312840092$

3. Pruebe a ejecutar la expresión 2.c) con distintos valores aproximados para PI. Copie la sentencia inicial (utilizando `mt.pi`) en otro fichero. Añada hasta 3 ó 4 sentencias adicionales modificando "`mt.pi`" por valores numéricos como 3.14, 3.1415, etc. Comente las diferencias en los resultados obtenidos (Use """ para varias líneas ó # [teclado alt-gr + 3] para una línea)

3.2 Bloque 2: Cálculos usando variables

4. Multiplique dos números 1398 y 11 y asigne el resultado a una variable llamada "producto". Calcule el resto de 1398 entre 11 y guarde el resultado en una variable "resto".
5. Imprima un mensaje con las dos variables numéricas calculadas anteriormente. Puede que necesite usar la función "**`str`**`(variable)`". Comente en una línea el porqué.
6. Convierta las siguientes fórmulas a expresiones en Python. Asigne valores previamente a las variables (*x*, *y*) y haga que los resultados salgan por pantalla al ejecutar el script.

   a) $\frac{(x-y)*(x+y)}{2}$

   b) $hipotenusa = \sqrt{cateto1^2 + cateto2^2}$

   c) $polinomio = -x^9 - 2.7x^6 - 5x^4 + 10x^3 - x + 3.3$

   Puede intentar inicializar las variables utilizando la orden "input". Recuerde que para guardar valores numéricos debes "convertirlos" a enteros o reales. Un ejemplo:

   ```
   x = int(input("Introduce el valor de X: ") )
   ```

   `x` es la variable que queremos utilizar
   `int` es para que el valor introducido se convierta de "`texto`" a un número `entero`

"`introduce el valor de X:`" es el mensaje que se muestra al usuario como "pista" para que sepa qué es lo que tiene que escribir.

Cuando se utiliza "**`input`**" debemos acceder al terminal de "**`Ipython`**" para escribir el valor (y pulsar "`Enter`"). En ocasiones se puede quedar "atascado" y no reaccionar. Pulsa el botón "rojo" de stop que aparece en la parte superior derecha de la terminal.

7. Escriba un script que pida una distancia en milímetros y muestre su equivalente en metros, centímetros y milímetros. Ejemplo: 1033 milímetros serían equivalentes a 1 metro, 3 centímetros y 3 milímetros. Pista: debe encadenar operaciones de división y módulo sobre el mismo número. Trate siempre de pensar cómo lo obtendría manualmente (lápiz y papel).

8. Construya un programa que simule un juego inspirado en el de los triles (del que procede el nombre de trilero). Suponemos dos participantes y cada uno tiene una caja etiquetada con su nombre. Dentro de cada caja hay una cantidad de dinero y el objetivo es intercambiar las cantidades que hay dentro. Se pide construir un programa que haga lo siguiente:

   Debe leer desde teclado dos variables `caja_izda y caja_dcha`
   A continuación, debe intercambiar sus valores y finalmente, mostrarlos en pantalla

   Recuerda, el siguiente código no es válido:
   `print``("la caja derecha es", caja_izda)`

9. Extienda el ejercicio anterior para intercambiar el valor de tres variables (caja_izda, caja_dcha, caja_central). El resultado final será que el valor de la 1ª acabe en la segunda, el de la segunda en la tercera y el de la tercera en la primera. Muestre el contenido final de las tres variables.

10. Verifique las siguientes identidades trigonométricas asignando previamente los valores indicados a las variables correspondientes:
    - Para x = $(2/5)\,\pi$, y = 3
      a) `sen(x)* cos(y) = (sen(x+y)+sen(x-y))/2`
      b) `sen(x)* sen(y) = (cos(x-y)+cos(x+y))/2`
    - Para $\alpha = 2\pi/3, \beta = \pi/2$
      a) $\cos\alpha + \cos\beta = 2\cos\frac{1}{2}(\alpha+\beta)\cos\frac{1}{2}(\alpha-\beta)$

    Recuerde utilizar expresiones "relacionales" para comprobar si se cumple la igualdad (la salida será `True`) o no (`False`)

4. Ejercicios voluntarios

En esta última sección, se plantean diversos ejercicios que a priori NO serán puntuables para las prácticas.

El objetivo de los mismos es servir como refuerzo mediante trabajo autónomo o mediante resolución por el profesor en pizarra.

1. Consulte los valores predefinidos para el máximo, mínimo, etc., valor real representable en su sistema para Python. Para ello, importe la biblioteca "**sys**" y consulte **`sys.float_info`**

2. Utilice las funciones de redondeo *round, ceil, floor* con varios valores reales. Observe las diferencias.

3. Dados los escalares a = 5, b = 2, c = -4, determinar los valores de verdad de las proposiciones.

   a. a>b
   b. not (b<c)
   c. (a>b) or (c>b)
   d. not ((a<b) and (c<b))
   e. (a < b and a > c) or (b > c)
   f. (a ≥ b and b ≤ c) or (a > c)
   g. (a > b) or (a > c)
   h. (a > b) or ((a > c) and (b < a))
   i. (a < b and a > c) or (b > c)

4. ¿Cuáles de estos identificadores son válidos como nombres de variables y cuáles no? En el caso de no serlo, ¿por qué?

```
Total
total acumulado
resultado-1
resultado_1
resultado 1
Resultado
resultaDo
1resultado
el_valor_total_de_la_suma_de_los_operandos_es
el_valor_total_de_la_suma_es
el_valor_total_de la_suma_es
resul*1
int8
double
integer1
válido
```