Departamento de Física
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Programación para laboratorio
1^{er} cuatrimestre 2015
`Talleres FIFA`

Trámites

Por favor, inscribite al curso, toma 3 segundos (cronometrados)

`http://goo.gl/EfXGFB`

Luego, para comenzar, en caso de que el IDE de las pc no funcionen, se puede probar programar online en la web: `https://cloud.sagemath.com/#settings`

Ahora sí.

Ejercicios propuestos

Unidad 1. Comandos y variables

Las variables son formas de almacenar información. La diversidad de esta información da origen a la diversidad de tipos de variables. Los comandos son operaciones que se pueden hacer sobre las variables para transformarlas, leerlas u obtener otras cosas de ellas. Recomendamos utilizar el comando *type(variable)* al final de cada item para que Python nos devuelva el tipo de variable con la que estamos trabajando.

- Comando *print* y *raw_input*

  Haga que Python realice las siguientes tareas:

  1. Escriba el texto 'Hola mundo'.
  2. A continuación deje un espacio de tabulación (pruebe con $\backslash t$).
  3. Guarde en una variable el texto 'Me tiemblan los dedos de la emoción de saber que estoy programando'.
  4. Guarde en dos variables dos textos, y que imprima ambas variables separadas por un espacio de tabulación. (¿Se puede usar el + para eso?)
  5. Pruebe escribir a=int(raw_input('Dale un valor a a: ')) ¿Qué tipo de variable es a? ¿Por qué sería necesario el comando *int*?
  6. Aplique el comando *len(variable)* sobre una de las variables con texto.

- Variables *integer* y *float*

  ¿Qué diferencia hay entre ambos tipos de variables?

1. Guarde en una variable a el valor 7, en otra variable b el valor 9, y que los sume.
2. Pruebe ahora restarlas, multiplicarlas y dividirlas. ¿Qué otras operaciones matemáticas se pueden hacer con Python?
3. Guarde en una variable el resultado de la división de dos variables *integer*. ¿Qué tipo de variable es este resultado?
4. Averigue: ¿Cuántas cifras puede almacenar una variable tipo *float*?
5. ¿Qué tipo de variable devuelve Python al aplicar *round*(x,n)? (siendo x un *float* y n la cantidad de decimales que uno desea redondear).

- Variables *boolean*

1. ¿Qué valor obtendrá de $5 > 5$? ¿Y de $5 >= 5$?
2. ¿Qué obtendrá de escribir $4 == 5 == 3$ =!2? ¿Son necesarios los paréntesis?
3. ¿Valen los operadores = y $\neq$ para *strings*?

- Variables tipo *list* y tipo *tuple*

  ¿Qué diferencia hay entre ambos tipos de variables?

1. Arme una lista con las materias en las que debe el examen final.
2. Pasó el cuatrimestre y nos colgamos todos. Agregue un elemento a la lista anterior (*append*) y pedile que te muestre el segundo elemento.
3. ¿Cuál es la diferencia entre *extend* y *append*? ¿Y con *insert*?
4. ¿Qué diferencia hay entre los comandos *remove* y *pop*? ¿Qué hace el comando *sort*?
5. Armada una lista, aplíquele la función *len*.
6. Pruebe *range*(10), *range*(3,13), *range*(2,26,2). ¿Qué tipo de variable es el resultado?

Unidad 2. Librerías

Las librerías contienen funciones pensadas y distribuidas para extender las posibilidades de Python. Lo primero que debemos hacer es cargar la librería con el comando *import*. Para resolver este ejercicio usaremos la librería *math*.

Probar luego de eso resolver los siguientes ejercicios.

1. Calcular el seno de π en radianes. Si no te gusta, cambialo a grados (funciones *degrees* o *radians*)
2. Calcular $arctan(1/2)$.
3. Calcular 2^{3^4} y e^{π}
4. Calcular $\log_3(25)$.
5. Calcular $e^{ln(x)}$ siendo x el número que quiera.

Pregunta aislada: ¿Qué pasa si se aplica la función *type* sobre la respuesta a *type* sobre una variable?

Unidad 3. Condicionales y funciones

Una función es un paquete aislado de acciones a realizar. Pueden ser definidas y nunca utilizadas. Definir funciones como serie de acciones a realizar con una o más variables nos da la posibilidad de aplicar ese paquete más de una vez sin necesidad de repetir código.

Implemente las siguientes funciones (*def* o *lambda*) en Python:

1. **doble**(n): muestre el doble de n.
2. **promedio3**(n1, n2, n3): calcule y muestre el promedio de tres números pero redondeando para arriba.
3. **iguales**(n1,n2): devuelve True si n1=n2.
4. **divisible**(n,d): devuelve True si n es divisible por d.
5. **factorial**(n): devuelve el valor factorial de n.
6. **primo**(n): devuelve True si n es un número primo.
7. **norma**(x,y,z): devuelve la distancia de un punto al cero de un espacio euclidiano.

Unidad 4. Gráficos

Para encaminarnos al próximo encuentro, resulta básico y muy útil saber graficar funciones de una variable respecto a otra. Para esto hay funciones ya armadas en las librerías *Numpy* y *Matplotlib*.

Para experimentos numéricos, procederemos siempre de la misma manera:

1. Crearemos un dominio homogéneo con *numpy.linspace()*
2. Crearemos una imagen asociada a una función $f(x)$ definida por nosotros
3. Aportaremos ruido a los datos, simulando "mediciones", con funciones como *numpy.random.rand()*
4. Graficaremos x vs y en un *scatter* o en un *plot*, aportando una grilla, dándole límites al dominio y la imagen mostrados, dándole nombres tanto a los ejes como al gráfico

Interesante sería ver cómo hacer varios gráficos juntos (*subplot*) o como hacer gráficos de tipo histograma, cosas que abarcaremos en el próximo encuentro.

Unidad 5. Juegos en Python

Esta sección tiene un par de desafíos de aplicación de lo visto hasta ahora. Son totalmente resolubles a esta altura del aprendizaje.

Intente crear los siguientes juegos en Python:

1. ¡Adivina un número del uno al diez! (que el usuario pruebe números del uno al diez y el programa le diga si adivinó o no)
2. Piedra, papel o tijera (Variante para valientes: Piedra, papel, tijera, lagarto y spock).
3. Ta-te-tí.