Tema 0 - Programación básica con Python II

Curso de Física Computacional

M. en C. Gustavo Contreras Mayén

Contenido

Contenido

Instrucciones de entrada y salida

Entrada de datos:

- `raw_input("entrada")`: lee una línea de entrada que es convertida a `string`.
- `eval(string)` : convierte `string` en un valor numérico.

```
>>> a = raw_input("Ingrese a: ")
Ingrese a: 2
```

```
>>> print a
2
```

```
>>> b = eval(a)
```

```
>>> a
'a'
```

```
>>> type(a)
<type 'str'>
```

Instrucciones de entrada y salida

Entrada de datos:

- `raw_input("entrada")`: lee una línea de entrada que es convertida a `string`.
- `eval(string)` : convierte `string` en un valor numérico.

```
>>> a = raw_input("Ingrese a: ")
Ingrese a: 2
```

```
>>> print a
2
```

```
>>> b = eval(a)
```

```
>>> a
'a'
```

```
>>> type(a)
<type 'str'>
```

Instrucciones de entrada y salida

Entrada de datos:

- `raw_input("entrada")`: lee una línea de entrada que es convertida a `string`.
- `eval(string)` : convierte `string` en un valor numérico.

```
>>> a = raw_input("Ingrese a: ")
Ingrese a: 2
```

```
>>> print a
2
```

```
>>> a
'a'
```

Instrucciones de entrada y salida

Entrada de datos:

- `raw_input("entrada")`: lee una línea de entrada que es convertida a `string`.
- `eval(string)` : convierte `string` en un valor numérico.

```
>>> a = raw_input("Ingrese a: ")
Ingrese a: 2
```

```
>>> print a
2
```

```
>>> a
'a'
```

```
>>> type(a)
<type 'str'>
```

Instrucciones de entrada y salida

Entrada de datos:

- `raw_input("entrada")`: lee una línea de entrada que es convertida a `string`.
- `eval(string)` : convierte `string` en un valor numérico.

```
>>> a = raw_input("Ingrese a: ")
Ingrese a: 2
```

```
>>> print a
2
```

```
>>> b = eval(a)
```

```
>>> a
'a'
```

```
>>> type(a)
<type 'str'>
```

```
>>> print b, type(b)
2 <type 'int'>
```

```
>>> s=eval(raw_input("Ingrese s :"))
Ingrese s: 2*3
```

```
>>> print s, type(s)
6 <type 'int'>
```

```
>>> m=eval(raw_input("Ingrese m :"))
Ingrese m: hola
Marca un error, por qué?
```

```
>>> print b, type(b)
2 <type 'int'>
```

```
>>> s=eval(raw_input("Ingrese s :"))
Ingrese s: 2*3
```

```
>>> print s, type(s)
6 <type 'int'>
```

```
>>> m=eval(raw_input("Ingrese m :"))
Ingrese m: hola
Marca un error, por qué?
```

```
>>> print b, type(b)
2 <type 'int'>
```

```
>>> s=eval(raw_input("Ingrese s :"))
Ingrese s: 2*3
```

```
>>> print s, type(s)
6 <type 'int'>
```

```
>>> m=eval(raw_input("Ingrese m :"))
Ingrese m: hola
Marca un error, por qué?
```

```
>>> print b, type(b)
2 <type 'int'>
```

```
>>> s=eval(raw_input("Ingrese s :"))
Ingrese s: 2*3
```

```
>>> print s, type(s)
6 <type 'int'>
```

```
>>> m=eval(raw_input("Ingrese m :"))
Ingrese m: hola
Marca un error, por qué?
```

```
>>> print b, type(b)
2 <type 'int'>
```

```
>>> s=eval(raw_input("Ingrese s :"))
Ingrese s: 2*3
```

```
>>> print s, type(s)
6 <type 'int'>
```

```
>>> m=eval(raw_input("Ingrese m :"))
Ingrese m: hola
Marca un error, por qué?
```

```
>>> print b, type(b)
2 <type 'int'>
```

```
>>> s=eval(raw_input("Ingrese s :"))
Ingrese s: 2*3
```

```
>>> print s, type(s)
6 <type 'int'>
```

```
>>> m=eval(raw_input("Ingrese m :"))
Ingrese m: hola
Marca un error, por qué?
```

Salida de datos

La mayoría de programas requiere mostrar un resultado, en ocasiones a la pantalla y en otras, como un conjunto de datos que se enviarán a un archivo.

Para facilitar la lectura del resultado conviene elegir el respectivo formato de salida.

- `objeto1, objeto2, ...`
- %formato1, %formato2, ..., %tupla

Entero	d
Punto flotante	f
Notación científica	e

Formato de salida de datos

```
>>> u=6543
```

```
>>> v=1234.56789
```

```
>>> print u, v
6543 1234.56789
```

```
>>> print "u = %6d" % u
u=6543
```

```
>>> print "u = %06d" % u
u=006543
```

Formato de salida de datos

```
>>> u=6543
```

```
>>> v=1234.56789
```

```
>>> print u, v
6543 1234.56789
```

```
>>> print "u = %6d" % u
u=6543
```

```
>>> print "u = %06d" % u
u=006543
```

Formato de salida de datos

```
>>> u=6543
```

```
>>> v=1234.56789
```

```
>>> print u, v
6543 1234.56789
```

```
>>> print "u = %6d" % u
u=6543
```

```
>>> print "u = %06d" % u
u=006543
```

Formato de salida de datos

```
>>> u=6543
```

```
>>> v=1234.56789
```

```
>>> print u, v
6543 1234.56789
```

```
>>> print "u = %6d" % u
u=6543
```

```
>>> print "u = %06d" % u
u=006543
```

Formato de salida de datos

```
>>> u=6543
```

```
>>> v=1234.56789
```

```
>>> print u, v
6543 1234.56789
```

```
>>> print "u = %6d" % u
u=6543
```

```
>>> print "u = %06d" % u
u=006543
```

Formato de salida de datos

```
>>> print "v= %7.2f" % v
v=1234.57
```

```
>>> print "v= %9.2f" % v
v =    1234.57
```

```
>>> print "v= %7.8f" % v
v=1234.56789000
```

```
>>> print "v= %.2e" % v
v=1.23e+03
```

```
>>> print "u= %6d y v=%8.4e" %(u, v)
u = 6543 y v=1.2346e+03
```

Formato de salida de datos

```
>>> print "v= %7.2f" % v
v=1234.57
```

```
>>> print "v= %9.2f" % v
v =    1234.57
```

```
>>> print "v= %7.8f" % v
v=1234.56789000
```

```
>>> print "v= %.2e" % v
v=1.23e+03
```

```
>>> print "u= %6d y v=%8.4e" %(u, v)
u = 6543 y v=1.2346e+03
```

Formato de salida de datos

```
>>> print "v= %7.2f" % v
v=1234.57
```

```
>>> print "v= %9.2f" % v
v =    1234.57
```

```
>>> print "v= %7.8f" % v
v=1234.56789000
```

```
>>> print "v= %.2e" % v
v=1.23e+03
```

```
>>> print "u= %6d y v=%8.4e" %(u, v)
u = 6543 y v=1.2346e+03
```

Formato de salida de datos

```
>>> print "v= %7.2f" % v
v=1234.57
```

```
>>> print "v= %9.2f" % v
v =    1234.57
```

```
>>> print "v= %7.8f" % v
v=1234.56789000
```

```
>>> print "v= %.2e" % v
v=1.23e+03
```

```
>>> print "u= %6d y v=%8.4e" %(u, v)
u = 6543 y v=1.2346e+03
```

Formato de salida de datos

```
>>> print "v= %7.2f" % v
v=1234.57
```

```
>>> print "v= %9.2f" % v
v =    1234.57
```

```
>>> print "v= %7.8f" % v
v=1234.56789000
```

```
>>> print "v= %.2e" % v
v=1.23e+03
```

```
>>> print "u= %6d y v=%8.4e" %(u, v)
u =  6543 y v=1.2346e+03
```

Formato de salida de datos

```
>>> print "v= %7.2f" % v
v=1234.57
```

```
>>> print "v= %9.2f" % v
v =    1234.57
```

```
>>> print "v= %7.8f" % v
v=1234.56789000
```

```
>>> print "v= %.2e" % v
v=1.23e+03
```

```
>>> print "u= %6d y v=%8.4e" %(u, v)
u = 6543 y v=1.2346e+03
```

Ejercicio

A continuación se presenta el código que calcula el promedio de dos números que se proporcionan por el usuario.

Código

```
A = eval(raw_input('Ingresa A: '))
Ingresa A: 10
B = eval(raw_input('Ingresa B: '))
Ingresa B: 20
PROM = (A+B)/2.0
print 'El promedio de %f y %f es %f' % (A,B,PROM)
```

Estructuras de control

En cualquier lenguaje de programación se incluye una serie de estructuras de control para ampliar las posibilidades de ejecución de un programa.

En Python, manejaremos las más comunes, que son relativamente sencillas de usar, cuidado siempre la sintaxis respectiva.

Condicionales

Una sentencia condicional permite evaluar si se cumple cierta condición, es decir, si su valor es `True`, se ejecuta una instrucción, en caso de que el valor de la condición no se cumpla, valor `False`, no se ejecuta la instrucción contenida y se sigue a la siguiente línea de código.

Ejemplo de condicional

```
1 a = input('Introduce el valor de a')
2 if a > 0:
3     print "a es positivo"
4     a = a + 1
5 elif a == 0:
6     print "a es 0"
7 else:
8     print "a es negativo"
```

Bucles

Un bucle es una sentencia que evalúa inicialmente una condición, en caso de que se cumple (valor `True`) se ejecuta(a) un conjunto de instrucciones, posteriormente, se revisa el valor de la condición, mientras sea verdadero, las instrucciones se ejecutan nuevamente.

Hay que considerar que se puede conocer de antemano, el número de repeticiones, hay que evitar los bucles infinitos, es decir, sentencias que no modifican el valor de la condición y por tanto, siempre se mantendrá sin salir del bucle.

```
1 nMax = 5
2 n = 1
3 a = [] # Crea una lista vacia
4 while n < nMax:
5       a.append(1.0/n) # agrega un elemento a la lista
6       n = n + 1
7 print a
```

Sentencia for

```
1 lista = ['Hugo', 'Paco', 'Luis', 'McPato']
2 nombre = eval(raw_input('Teclea un nombre: '))
3 for i in range(len(lista)):
4     if lista[i] == nombre:
5     print nombre, ' es el numero ', i + 1, ' en la lista
          '
6     break
7 else:
8     print nombre, ' no esta en la lista'
```