Tipos de Datos de python (1ª parte):

Cuando almacenamos datos en una variable, lo que podemos hacer con ellos depende del tipo de datos que sean.

Si tengo los datos 1 y 2 y hago 1+2 obtendré como resultado 3 siempre y cuando estos datos sean de tipo numérico. Si son de tipo "texto", obtendré 12 igual que cuando tengo los datos a y b y hago a+b obtengo ab.

Un ejemplo:

```
>>> 1 + 2
3

>>> "1" + "2"
'12'
```

En este ejemplo, podemos ver que he hecho dos veces lo mismo, pero la primera vez no utilicé comillas y en la segunda si. Esto ha hecho que python se de cuenta de que tipos de datos quiero manejar y los ha convertido automáticamente a los tipos correspondientes devolviéndome el resultado correcto.

Por esto es necesario conocer cuáles son los tipos de datos que python maneja y como tenemos que hacer nosotros para indicarle que tipo de datos queremos utilizar, así no corremos el riesgo de que python convierta equivocadamente nuestros datos a tipos que no queremos.

Hoy veremos sólo algunos de estos tipos, que son los que más utilizaremos por ahora.

Tipos de Datos:

int
Una variable de tipo integer o entero sólo puede guardar números enteros.
Es decir sin coma.

float
Una variable de tipo float sólo puede almacenar números decimales.
Tomar en cuenta que en python, los números decimales se escriben con punto en lugar de coma.

chr
Una variable de tipo character sólo puede guardar un caracter.
O sea, un símbolo tipográfico que puede ser una letra, un número, un espacio, una coma, etc . . . *(Más adelante en este documento te quedará más claro que es un caracter).*

str
Una variable de tipo string o cadena sólo puede almacenar una cadena de caracteres, pueden ser letras y números, puntos, comas, espacios, etc . . .

bool
Una variable booleana sólo puede guardar uno de los siguientes valores: True o False (verdadero o falso).

¿Como convertimos valores a tipos de datos específicos para operar correctamente con ellos?

int

```
>>> 1
1
```

python convierte cualquier número sin coma en un entero en forma automática.

```
>>> int(12.35)
12
```

De esta forma convertimos un número decimal en un entero.
Fíjate que en python, utilizamos el punto en lugar de la coma para escribir un número decimal.

```
>>> int("325  . ")
325
```

python puede convertir algunas cadena de caracteres, (un texto), a entero.

Esto es un error !!

```
>>> int("hola")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'hola'
```

float

```
>>> 12.35
12.35
```

python convierte cualquier número con coma en un float o decimal en forma automática.

```
>>> float("325")
325.0
```

python puede convertir algunas cadenas en un dato de tipo float.

Esto es un error !!

```
>>> float("Hola")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for float(): Hola
```

chr

```
>>> chr(90)
'Z'
```

En informática, un caracter o character, representa al código correspondiente a una de las teclas en el teclado de la computadora, por eso, nos imprime 'Z' ya que el caracter 90 del teclado es la letra Z.

Para aclarar esta idea, podemos hacer:

```
>>> chr(128)
'\x80'
```

El resultado es raro para nosotros, pero si hacemos:

```
>>> print chr(128)
€
```

Ahora vemos que el carácter 128 corresponde en el teclado al símbolo €

Esto pasa porque siempre que utilizamos la función print de python para mostrar un dato en pantalla, este se ve obligado a convertir el dato en un tipo string o cadena de texto (un texto) para poder mostrarlo.

De modo que, un caracter es cualquier signo tipográfico, puede ser una letra, un número, un signo de puntuación, un espacio, etc.

Nota:
Para hacer lo contrario, utilizamos la función ord(), que a partir de un carácter, nos devuelve su código chr.

```
>>> chr(90)
'Z'
>>> ord("Z")
90
```

str

```
>>> str(150)
'150'

>>> str(150.25)
'150.25'
```

Un string es una cadena de caracteres, no es número.

```
>>> str(1500) + " Km"
'1500 Km'
```

En este último ejemplo queda claro que str(1500) no es un número ya que no se puede sumar números más letras, las operaciones siempre se hacen entre datos del mismo tipo.
El tipo de datos string es tan importante y utilizado en informática, que python cuenta con una clase especial para manejar cadenas de caracteres, más adelante trabajaremos con ella.

bool

```
>>> a = bool(1)
>>> print a
True
```

Para python, sólo 0 es False, todo lo demás es True.

```
>>> a = bool("x")
>>> print a
True
```

Cualquier cosa que no esté vacía es True.

```
>>> a = "hola que tal?"
>>> b = bool(a)
```

```
>>> print a
hola que tal?
>>> print b
True
```

Cualquier cosa vacía es False.

```
>>> a = ""
>>> b = bool(a)
>>> print a

>>> print b
False
```

Ejemplo final del Tema:

```
>>> a = 90

>>> int(a)
90
>>> float(a)
90.0
>>> chr(a)
'Z'
>>> str(a)
'90'
>>> bool(a)
True
```

Referencias:

http://docs.python.org/library/functions.html
http://docs.python.org/library/types.html
http://es.wikipedia.org/wiki/Unicode