Cheatography

Python 3 Español Cheat Sheet

by Ismael Mercado (torerohk) via cheatography.com/23626/cs/5397/

print()

print()

print al ser funcion siempre se utiliza con parentesis

Variable String

y = "a"
z = 'Hola'
multilinea = """cadena de texto
con mas de una linea"""

variables de tipo string van entre comillas dobles o sencillas
para cadenas de texto de multiples lineas se utiliza """ texto """

Operadores Matematicos

suma	a+b
resta	a-b
multiplicacion	a*b
division_real	a/b
division_entera	a//b
resto	a%b
potencia	a**b

los operadores matemáticos principales pueden utilizarse combinados respetando la jerarquía al resolverlas
1. Resolver ( ) [ ] { }
2. Resolver exponentes.
3. Resolver * y / de izquierda a derecha
4. Resolver + y - de izquierda a derecha

Operadores Logicos

Igual a	==
Diferente a	!=
Menor que	<
Menor o igual que	<=
Mayor que	>
Mayor o igual que	>=

Devolverán un valor boleano

Metodos para Strings

len() retorna longitud de caracteres en string:
len(string)
lower() retorna string en minúsculas:
string.lower()
upper() retorna string en mayúsculas:

Metodos para Strings (cont)

string.upper()
capitalize() retorna primer carácter de string en mayúsculas
string.capitalize()
str() retorna conversión explícita de strings:
str(string)

Literales
variable.lower()
variable.upper()
No Literales
len(variable)
str(variable)

String Inmutable a Flexible

print("%s" % (variable)) o print("%s" % ("string"))
nom= "Ismael"
ape = "Mercado"
# variables
print ("mi nombre %s. mi apellido %s ." % (nom, ape))
strings
print ("mi nombre %s. mi apellido %s ." % ("Ismael", "Mercado"))

Comparadores guia

	AND	
True	True	True
True	False	False
False	True	False
False	False	False
	OR	
True	True	True
True	False	True
False	True	True
False	False	False
	NOT	
True		False
False		True

Diccionarios

Estructura de datos que almacena valores utilizando otros como referencia para su acceso y almacenamiento, es iterable, mutable y puede contener elementos de diferente tipo; se declara entre llaves *{clave:valor}*
diccionario={'a':1, 'b':2, 'c':3}
Podemos utilizar la funcion **dict()**
diccionario=dict(a=1, b=2, c=3)

Diccionarios (cont)

Acceder a un elemento utilizamos el indice
diccionario['c']
Modificar un valor
diccionario['b']=28
Nuevos elementos añadimos una clave no existente
diccionario['d']=4
Iterar con un diccionario
items() *Acceso a claves y valores*
diccionario.items()
values() *Acceso a valores*
diccionario.values()
keys() *Acceso a claves*
diccionario.keys()
Ordenar un diccionario
sorted(diccionario)
Ordenar un diccionario en inverso
sorted(diccionario, reverse=True)

Matrices

Anidando listas construimos matrices de elementos
matriz=[[1,2,3],[4,5,6]]
para acceder a los elementos utilizamos
matriz[0][1]
sustituir un elemento
matriz[1][0]=33

crear, modificar y leer archivos en disco

Función para crear un archivo

```
def crearArchivo():
  archivo=open('datos.txt', 'w')
   archivo.close()
```

Función para escribir en un archivo

```
def escribirArchivo():
   archivo=open('datos.txt', 'a')
   archivo.write('prueba de texto\n')
   archivo.close
```

Función para leer un archivo

```
def leerArchivo():
```

crear, modificar y leer archivos en disco (cont)

```
  archivo=open('datos.txt', 'r')
  linea = archivo.readline()
  while linea!="":
     print(linea)
     linea=archivo.readline()
  archivo.close()
```

Modos de apertura de archivos

Indicador	Modo de apertura	Ubicación del puntero
`r	Solo lectura	Al inicio del archivo
rb	Solo lectura en modo binario	Al inicio del archivo
r+	Lectura y escritura	Al inicio del archivo
rb+	Lectura y escritura en modo binario	Al inicio del archivo
w	Solo escritura. Sobreescribe el archivo si existe. Crea el archivo si no existe	Al inicio del archivo
wb	Solo escritura en modo binario. Sobreescribe el archivo si existe. Crea el archivo si no existe	Al inicio del archivo
w+	Escritura y lectura. Sobreescribe el archivo si existe. Crea el archivo si no existe	Al inicio del archivo
wb+	Escritura y lectura en modo binario. Sobreescribe el archivo si existe. Crea el archivo si no existe	Al inicio del archivo
a	Añadido (agregar contenido). Crea el archivo si éste no existe	Si archivo existe, al final. Si no, al comienzo

Cheatography

Python 3 Español Cheat Sheet

by Ismael Mercado (torerohk) via cheatography.com/23626/cs/5397/

Modos de apertura de archivos (cont)

ab	Añadido en modo binario (agregar contenido). Crea el archivo si éste no existe	Si archivo existe, al final. Si no, al comienzo
a+	Añadido (agregar contenido) y lectura. Crea el archivo si éste no existe.	Si archivo existe, al final. Si no, al comienzo
ab+	Añadido (agregar contenido) y lectura en modo binario. Crea el archivo si éste no existe	Si archivo existe, al final. Si no, al comienzo

indicado a la función open() como una string en su segundo parámetro.

Funciones integradas

__import__()	abs()	all()
any()	ascii()	bin()
bool()	bytearray()	bytes()
callable()	chr()	classmethod()
compile()	complex()	delattr()
dict()	dir()	divmod()
enumerate()	eval()	exec()
filter()	float()	format()
frozenset()	getattr()	globals()
hasattr()	hash()	help()
hex()	id()	input()
int()	isinstance()	issubclass()
iter()	len()	list()
locals()	map()	max()
memoryview()	min()	next()
object()	oct()	open()
ord()	pow()	print()
property()	range()	repr()
reversed()	round()	set()
setattr()	slice()	sorted()
staticmethod()	str()	sum()
super()	tuple()	type()

Funciones integradas (cont)

vars()	zip()

Python incluye las siguientes funciones y siempre están disponibles

type()

```
x = 3.1415
print(type(x))
>><class 'float'>
```

La función type permite comprobar el tipo de variable

Variables Numericas

```
num_entero = 5
num_negativo = -7
num_real = 3.14
num_complejo = 3.2 + 7j
num_binario = 0b111
num_octal = 0o10
num_hex = 0xff
```

puedes crear variables del tipo Enteros, Reales, Complejos y los puedes representar en Decimal, Binario, Octal y Hexadecimal

Conjunto Matematico funcion set()

```
conjunto = set('246')
conjunto2 = {2, 4, 6}
```

se pueden utilizar los métodos *add()* y *remove()* para añadir o eliminar elementos.
si se crea un conjunto con valores repetidos, estos se eliminan automáticamente.

Operadores Comparadores

and	compara 2 elementos y devuelve True si ambos son verdaderos
or	compara 2 elementos y devuelve True si uno de ellos es verdadero
not	devuelve el valor opuesto de un boleano

primero se calcula *not*
después se calcula *and*
por último se calcula *or*

Definiciones

Iteración	Término general para la toma de cada elemento de algo, una después de la otra. Usar un bucle, explícita o implícita, al pasar sobre un grupo de elementos

Metodos Especiales para Strings

find() Retorna el indice del primer carácter que coincide con el buscado
cad = "ABC"
cad.find("B")
>>1
replace() reemplaza un carácter por otro
cad.replace("B", "Z")
>>AZC
split() divide una cadena basado en un caracter y retorna una lista
cad.split(";")
join() retorna una cadena donde los valores son separados por un caracter
lista = ["Hola", "Mundo"]
print ("+".join(lista,))
lista2 = "Hola"
print ("-".join(lista2))
strip(), **lstrip()**, **rstrip()** eliminan los espacios en blanco, a la izquierda y a la derecha respectivamente
cad.strip()
cad.lstrip()
cad.rstrip()

Tabla Basica

Tupla	( )	Inmutable
Lista	[ ]	Mutable
Diccionario	{ }	Mutable

Tupla

Arreglo de objetos definido entre paréntesis es inmutable puede contener diferentes tipos de objetos.
tupla = (1, 'a', 3.5)
Se puede anidar una tupla dentro de otra
tupla2 = (1, (4, 'B'), 3.5)
Se puede acceder a los valores a través del indice.
tupla[1]

Lista

Arreglo de objetos definido entre corchetes es mutable puede contener diferentes tipos de objetos.
lista = [2, 'B', 4.5]
Se puede acceder a los valores a través del indice y reemplazarlos.
lista[1] = 'A'
Podemos comprobar si un valor existe en una lista usando**in**.
'B' in lista
se insertan valores al final de la lista con**.append()**
lista.append('nuevo')
insertar en una posición definida se utiliza el indice y**.insert()**
lista.insert(2, 'C')
borrar un elemento usamos **del()**
del(lista[1])
ordenar sin alterar **sorted()** y para orden inverso argumento**reverse**
sorted(lista)
sorted(lista, reverse=True)
ordenar con criterio como argumento
sorted(lista, key=str.lower)
ordenar alterando usamos **sort()**
lista.sort()

Comprensión de Listas y Diccionarios

Compresión Lista
lista= [x for x in (1,2,3)]
Compresión Diccionario
diccionario= {k: k+1 for k in (1,2,3)}

La comprensión es una construcción sintáctica de python, permite declarar una lista o diccionario a través de la creación de otra.

For y While

El bucle **while** (mientras) ejecuta un fragmento de código mientras se cumpla una condición.

```
edad = 0
while edad < 18:
    edad = edad + 1
    print “Felicidades, tienes “ + str(edad)
```

Permiten ejecutar un mismo fragmento de código un cierto número de veces, mientras se cumpla una determinada condición.

If, Else y Elif

Evalúan la condición indicada y ejecutan una instrucción u otra
if condicion1:
si condicion1 es True realiza esto
elif condicion2:
si condicion2 es True realiza esto
else:
si ambas condiciones son False realiza esto
se pueden anidar
if condicion1:
si condicion1 es True realiza esto
    if condicion3:
    si condicion3 es True realiza esto
    else:
    si es False realiza esto
else:
en caso contrario realiza esto

Clases, Objetos, Propiedades y Metodos

*Clases
 *Objetos
  *Propiedades
   *Métodos
class Clase(): # La clase
    varClase=0 # Variables de Clase
    def __init__(self): # Método de Instancia (constructor)
        self.varInstancia=0 # Variable de Instancia
objeto=Clase()
objeto.metodoinstancia()
@classmethod # Decorador Metodo de Clase
def clsmet(cls): # Obligatorio (cls)
Clase.clsmet

Self hace referencia a si mismo
__init__ constructor para inicializar los objetos a un valor
al colocar *(clase)* se habilita la hereda los objetos de la clase Persona

Palabras reservadas

and	as	assert
break	class	continue
def	del	elif
else	except	False
finally	for	from
global	if	import
in	is	lambda
None	nonlocal	not
or	pass	raise
return	True	try
while	with	yield

Estas palabras no pueden utilizarse para nombrar variables.

Patrones caracteres

\n	Nueva Linea
\r	Retorno de carro
\t	Tabulador Horizontal
\w	Caracter minuscula
\W	Caracter Mayuscula
\s	Engloba minusculas y mayusculas
\S	cualquier caracter que no es espacio en blanco
\d	numero entre 0 - 9
\D	cualquier carácter que no es un numero
^	Inicio de cadena
$	Fin de cadena
``	Escape caracter especial
[]	rango de caracteres dentro de corchetes
^[]	cualquier caracter fuera de corchetes
\b	separacion entre numero y/o letra
{{Metacaracter}}	repeticiones
+	una o mas veces
*	cero o mas veces
?	cero o una vez
{n}	n numero de veces