Recordando python

Hecho por Ricardo De la Cruz. Mi github y mi linkedin para consultas.

1. Imprimir mensaje en pantalla

```
print("Hello world")
```

2. Hacer comentarios en python

```
#Comentario en linea
```

3. Declarar variables y tipos de datos

```
soy_un_int = 0
soy_un_string = "0"
soy_un_float = 0.23
soy_una_lista = ["a", 0, 32.3]
soy_un_diccionario = {
    "nombre": "Ricardo",
    "edad": 22,
    "promedio notas": 15,
    "cursos matriculados": ["mate", "ingles"]
}
```

El string puede ser declarado con doble comillas " o con una '.

Las variables son sensibles a las mayusculas, por lo que la variable "x" es diferente a "X".

4. Cambio de tipo de dato

A veces, para hacer un if, necesitas que las variables sean del mismo tipo de dato, para esto se hace un "casting".

```
x = 3
```

```
y = str(x) # y valdra "3" como string, no como int

z = float(x) #z valdra 3.0 como float, no como int

a = "3"

b = int(a) # b valdra 3 como int, no como string
```

5. Obtener tipo de dato

Cuando no sepas que tipo de dato sea una variable, puedes utilizar

```
type(variable) #Si colocas un print, tendras el tipo de dato
```

6. Funciones

Las funciones son bloques de codigo que realizan operacions. Estas pueden retornar valores o no. En caso no lo haga, normalmente, imprimirán algo.

```
def miFuncion():
    print("Soy una funcion")

miFuncion() #Esta funcion no retorna nada, pero imprime un mensaje

x = 3
def miSegundaFuncion(a):
    print(a)

miSegundaFuncion(x) #Esta funcion no retorna nada, pero imprime el valor de la
variable x.
#En este caso, la funcion tiene un parametro de entrada denominado "a",
# la cual tendra que ser escrita al hacer el llamado de la funcion, de lo contrario
no funcionaria.

y = 10
z = 20
def miTerceraFuncion(a, b):
    return a + b

miTerceraFuncion(y,z) #Esta funcion retorna la suma de la variable y + z.
```

7. Operaciones básicas

```
#Suma
10 + 3

#Resta
10 -5

#Multiplicacion
5 * 3

#Division
6/3

#Modulo (Residuo)
10 % 2 #Da residuo 0
10 % 3 #Da residuo 1

#Exponente
10 ** 2 #10*10
```

8. Condiciones IF

```
a = 10
b = 15

if a>b:
    print("A es mayor a B")
else if (a == b):
    print("Valen lo mismo")
else:
    print("B es mayor")

#Tambien existen:
#  != si no es lo mismo
#  <= menor o igual
#  >= mayor o igual
```

9. Bucles while

Son bloques de codigo que se repetiran hasta que no se cumpla la condicion

```
i = 1
while i < 6: #Si es mayor o llega a ser 6, acaba
  print(i)
  i += 1 #Esto equivale a i = i + 1

#######
```

```
i = 1
while i < 6:
  print(i)
  if i == 3: #Si i es igual a 3, se acaba el while
    break #Break acaba el while (tambien hay continue
          #continue hace que pase la iteracion, es decir
          #no pasaria por la linea i+= 1
  i += 1
```

10. Bucles for

Los bucles for son los más faciles de utilizar y los más dinamicos.

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

En este caso x valdra, apple, banana y cherry.

```
for x in "banana":
  print(x)
```

En este caso x valdra: b, a, n ... n, a.

```
for x in range(6):
  print(x)
```

En este caso x valdra: 0, 1, 2, 3 ... 5

```
for x in range(2, 6):
  print(x)
```

En este caso x valdra: 2,3,4,5