



HERRAMIENTAS PARA LA LÓGICA CON PYTHON



Analice y memorice la sintaxis de las siguientes herramientas que le ayudarán a ser competente en la lógica de programación. Cada ciertos días, a medida que su proceso formativo avanza, revise este documento y marque con una x los estados de cada herramienta: la conoce y la maneja, la conoce pero no la maneja, no la conoce.

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Funciones: sintaxis

```
def sumar(a, b):  
    suma = a + b  
    return suma
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Procedimientos: sintaxis

```
def sumar(a, b):  
    suma = a + b  
    print(suma)
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Ciclo for: sintaxis

```
for i in range(10):  
    print(i)
```



Variables y tipos: sintaxis

```
#variable tipo entero  
edad = 23  
#variable tipo flotante  
estatura = 167.5  
#variable tipo string  
ciudad = "Armenia"  
#variable tipo boolean  
estado = True  
#variable sin tipo  
sin_tipo = None  
#conocer el tipo de dato  
type(edad)
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Entrada de datos: sintaxis

```
edad = input("Ingrese su edad")
```



La conozco y la manejo ☒

La conozco pero no la manejo ☐

No la conozco ☐



Comentarios: sintaxis

```
#este comentario es de una linea  
"""  
Este comentario es de bloque  
Permite comentar varias lineas  
"""
```

La conozco y la manejo ☒

La conozco pero no la manejo ☐

No la conozco ☐



Conversiones: sintaxis

```
edad = int("15")  
estatura = float("165.7")  
cedula = str(15746564)
```

La conozco y la manejo ☒

La conozco pero no la manejo ☐

No la conozco ☐



Operadores Relacionales: sintaxis

```
5 < 1 #menor que (<)  
10 > 1 #mayor que (>)  
20 <= 20 #menor o igual (<=)  
10 >= 8 #mayor o igual (>=)  
15 == 15 #igual que  
10 != 8 #diferente que
```

La conozco y la manejo ☒

La conozco pero no la manejo ☐

No la conozco ☐



Imprimir: sintaxis

```
print("Hola mundo")  
print("Hola, ", "mundo")  
edad = 16  
print("Edad: ", edad, " ok")
```



La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Condicionales: sintaxis

```
edad = 17
if edad > 18:
    print("Mayor de edad")
elif edad == 18:
    print("Mayor de edad")
else:
    print("No es mayor de edad")
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Operadores Lógicos: sintaxis

```
edad = 6
#operador and(Y)
5 < 1 and 10 == 10
#operador or(0)
10 == 10 or edad > 4
#operador not(No)
not edad == 8
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Operadores Aritméticos: sintaxis

```
5 * 10 #multiplicación *
20 / 10 #division /
10 // 3 #division entera //
10 + 5 #suma +
20 - 8 #resta -
10 % 3 #modulo %
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Ciclo while: sintaxis

```
x = 0
while x < 10:
    print(x)
    x += 1
```



La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Eliminar de la memoria: sintaxis

```
edad = 9
#elimina edad de la memoria
del edad
nombres = ["Juan", "Pedro", "Maria"]
#elimina nombres de la memoria
del nombres
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Diccionarios: sintaxis

```
#declaración e inicialización
datos = {"nombre": "Pepe", "edad": 30}

#accede al valor cuya clave es nombre
datos["nombre"]#devuelve Pepe
#si la llave nombre no existe,
#ocurre un error

#accede al valor cuya clave es nombre
datos.get("nombre")#devuelve Pepe
#si la llave nombre no existe,
#no ocurre un error, devuelve None

#elimina la llave nombre
#junto con su valor
datos.pop("nombre")
```

La conozco y la manejo ☐

La conozco pero no la manejo ☐

No la conozco ☐



Arreglos: sintaxis

```
#declaración e inicialización
datos = ["Pepe", 32, True]
#accede al elemento 0 de la lista
datos[0]#devuelve Pepe
#accede al elemento 1 de la lista
datos[1]#devuelve 32
#accede al elemento 2 de la lista
datos[2]#devuelve True
#elimina el elemento 2 de la lista
datos.pop(2)
#agrega el 50 al final de la lista
datos.append(50)
```