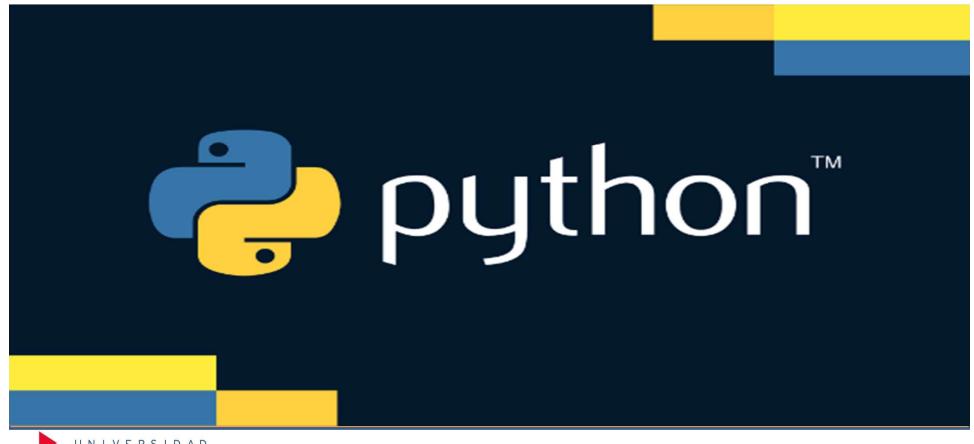
#### **OPTATIVO 1 – PYTHON I**

#### Prof. Ing. Aaron Zárate





### **Operadores en Python**

Los operadores son simbolos especiales que están diseñados para realizar operaciones especificas. Tenemos varios tipos que son:

- Operadores aritméticos: Permiten realizar cálculos matemáticos básicos, como suma, resta, multiplicación y división.
- Operadores de asignación: Se utilizar para asignar valores a variables.
- Operadores de comparación: Se utilizar para comparar un valore con otro.
- Operadores lógicos: Se utilizan para combinar expresiones condicionales o lógicas.
- Operadores de identidad: Se utilizan para comparar si dos variables son el mismo objeto.
- Operadores de membresía: Se utilizan para probar si una secuencia (Eje, una subcadena) se presenta en el objeto.



# Operadores Aritméticos en Python

Los operadores aritméticos nos permiten realizar cálculos matemáticos básicos. Por ejemplo.

- Suma(+): Suma dos operadores.
- Resta(-): Resta dos operadores.
- Multiplicación(\*): Multiplica dos operandos
- División(/): Divide el primer operando con el segundo operando, resulta un valor flotante.
- División(//): (/): Divide el primer operando con el segundo operando, resulta un valor tipo entero.
- Modulo(%): Regresa el residuo de la división.
- Exponente(\*\*): Eleva el primer operando a la potencia del segundo operando.



# Operadores Aritméticos en Python

```
operadores_aritmeticos.py ×
```

```
#Operadores aritmticos
       a = 10
       b = 3
       #Suma
 4
       suma = a+b
       print(f"Suma= {suma}")
 6
       #Resta
       resta=a-b
       print(f"Resta= {resta}")
 9
       #multiplicacion
10
       multiplicacion= a*b
11
        print(f"Multiplicacion= {multiplicacion}")
12
```



# Operadores Aritméticos en Python

```
division= a/b
14
       print(f"Division flotante= {division:.2f}")
15
       #Division entera
16
       division_entera = a//b
17
       print(f"Division entera= {division_entera}")
18
       #Operador Modulo
19
20
       modulo = a%b
       print(f"Residuo o modulo de la division: {modulo}")
       #Exponente (potencia)
22
       exponente= a**b # 10 elevando a la 3, 10*10*10 = 1000
23
       print(f"Exponente= {exponente}")
24
25
```



# Operadores de Asignación en Python

Los operadores aritméticos nos permiten realizar cálculos matemáticos básicos. Por ejemplo.

```
# Sintaxis operador asignacion
variable = valor
```

```
# Ejemplo Operador Asignación
numero = 10
texto = "Hola, mundo"
```

En Python también tenemos la asignación múltiple, lo que nos permite asignar valores a varias variables en una sola línea de código. El código es mas compacto y fácil de leer

```
# Sintaxis de Asignación Múltiple
variable1, variable2 = valor1, valor2
```

```
# Ejemplo de Asignación Múltiple
a, b, c = 10, 'Saludos', 14.5
```



# Operadores de Asignación en Python

```
operadores_asignacion.py ×
```

```
print("***Operadores de asignacion***")

numero = 5

print(f"Valor del numero= {numero}")

numero = 10

print(f"Nuevo valor del numero= {numero}")

cadena= "Saludos desde Python"

print(f"Valor de la cadena= {cadena}")

#Asiganacion multiple

x, y, z = 19, "Palmeiras", -9.25

print(f"Valor de x= {x}, El valor de y= {y}, El valor de z= {z}")
```



## Asignación encadenada

En Python también contamos con la asignación encadenada, esto permite asignar el mismo valor a múltiples variables.

```
# Sintaxis de Asignación Encadenada
variable1 = variable2 = ... = valor
```

```
# Ejemplo. Inicializar contadores
contador1 = contador2 = 0
```



## Asignación encadenada

```
#Asignacion encadenada
a=b=c=10
print(f"El valor de A={a}, El valor de B={b}, El valor de C={c}")
#Asignacion de valores en una cadena, sin utilizar variables temporales
x,y=5,10
print(f"Valores iniciales x= {x}, y = {y}")
#Aplicando concepto de asignacion multiple, intercambiamos valores
x,y= y,x
print(f"Invertir los valores de x={x} , y={y}")
#Recibir multiples valores de la entrada del usuario
nombre,apellido= input("Ingresa tu nombre y apellido separados por coma.").split(",")
print(f"Mi nombre es: {nombre.strip()} y mi apellido es : {apellido.strip()} ")
```



# Operadores de Asignación compuesto

Los operadores de asignación compuesto combinan una operación aritmética, haciendo las operaciones mas concisas.

Los operadores pueden ser: += , -= , \*= , /= , etc.

```
# Sintaxis Operador Asignación Compuesto 
variable OPERADOR= valor
```

```
# Ejemplo Operador Asignación Compuesto
contador = 0
contador += 1 # contador = contador + 1
```



# Operadores de Asignación compuesto

operadores\_asignacion\_compuestos.py ×

```
print("***Operadores ed asignacion compuestos***")
     a,b = 10,15
     print(f"Valore inicial de A: {a}, Valor inicial de B: {b}")
     #Operador compuesto suma =+
     a+=b #α=α+b
     print(f"Operador a+=b: {a}")
     # Operador compuesto de resta
     a = 10
     a-=b
     print(f"Operador a-=b: {a}")
10
     #Operador compuesto de multiplicacion *=
     a=10 #reiniciampos el valor de a
     a*=b
13
     print(f"Operador a*=b: {a}")
14
     a = 10
15
     #Operador compuesto de division /=
     a = 10
     a/=b # \alpha=\alpha/b
     print(f"Operador a/=b: {a:.2f}")
```

Los operadores condicionales se utilizan para comparar dos valores.

El resultado siempre es un valor booleano "True" o "False" dependiendo de si la condición se cumple o no.

Operador igualdad (==): Compara si dos valores son iguales

```
# Sintaxis operador igualdad —
a — b
```

```
# Ejemplo operador igualdad =
print(5 = 5) # True
print(5 = 6) # False
```

Los operadores condicionales se utilizan para comparar dos valores.

El resultado siempre es un valor booleano "True" o "False" dependiendo de si la condición se cumple o no.

Operador distinto (!=): Compara si dos valores son distintos

```
# Sintaxis operador distinto !=
a != b
```

```
# Ejemplo operador distinto !=
print(5 != 5) # False
print(5 != 6) # True
```



 Operador menor que (<): Compara si el valor de la izquierda es menor que el de la derecha, si es menor devuelve "True" si es mayor "False"

```
print(3 < 5) # True
print(5 < 3) # False
```

 Operador menor o igual que (<=): Compara si el valor de la izquierda es menor o igual que el de la derecha, si es menor o igual devuelve "True" si es mayor "False"

```
print(3 <= 5) # True
print(5 <= 5) # True
print(6 <= 5) # False</pre>
```



 Operador mayor que (>): Compara si el valor de la izquierda es mayor que el de la derecha, si es mayor devuelve "True" si es menor "False"

```
print(5 > 3)  # True
print(3 > 5)  # False
```

 Operador mayor o igual que (>=): Compara si el valor de la izquierda es mayor o igual que el de la derecha, si es mayor o igual devuelve "True" si es menor "False"

```
print(5 >= 3)  # True
print(5 >= 5)  # True
print(3 >= 5)  # False
```



```
operadores_comparacion.py ×
```

```
print("***Operadores de comparacion***")
a,b=7,5
print(f"El valor inicial de A: {a}, el valor inicial de B:{b}")

#operador de igualdad
resultado = a==b
print(f"Resultado a==b: {resultado}")

#Operdor diferente !=
resultado = a!=b
print(f"Resultado a!=b: {resultado}")
```



```
#Operador mayor que >
10
     resultado = a>b
11
      print(f"Resultado a>b: {resultado}")
12
13
     #Operador mayor o igual que >=
     resultado= a>=b
14
     print(f"Resultado a>=b: {resultado}")
15
16
     #operador menor que <
     resultado = a<b
17
     print(f"Resultado a<b: {resultado}")</pre>
18
     #operador menor que <=
19
     resultado = a<=b
20
      print(f"Resultado a<=b: {resultado}")</pre>
21
```



## **Operadores Lógicos**

Los operadores lógicos se utilizan para realizar operaciones lógicas con valores booleanos.

 Operador Lógico and (y):Devuelve True si ambos operandos son verdaderos.

```
exp1 = False
exp2 = True
print(exp1 and exp2) # False
```



### **Operadores Lógicos**

Operador Lógico or (o):Devuelve True si cualquiera de los operandos es verdadero.

```
exp1 = False
exp2 = True
print(exp1 or exp2) # True
```



### **Operadores Lógicos**

Operador Lógico not (o):Invierte el valor del operando. Es un operador unario.

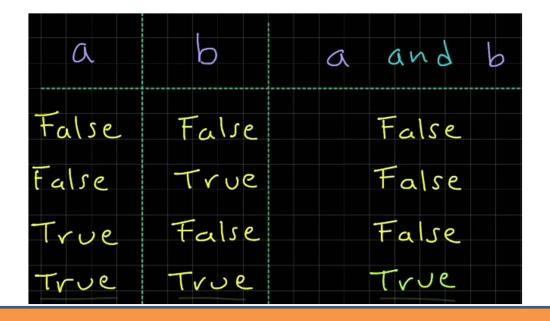
```
exp1 = False
print(not exp1) # True
```



#### **Operador and**

El operador and regresa verdadero si ambos operandos son verdaderos.

Tabla de la verdad





#### **Operador and**

```
operador_and.py ×
```

```
print(f"***Operador and***")

# Regresa verdadero sin ambos valores son verdaderos

condicion1 = False

condicion2 = False

resultado = condicion1 and condicion2

print(f"El resultado {condicion1} and {condicion2} es: {resultado}")
```



#### Sistema Descuentos VIP

Una tienda de supermercado ofrece un descuento especial a clientes que compren 10 o mas artículos por día y además sen miembros de la tienda.

El sistema debe solicitar al cliente que indique cuantos artículos ha comprado en el día y preguntarle si cuenta con la membresía de la tienda.

En caso de haber comprado 10 o mas productos y ser miembro de la tienda, entonces tendrá acceso al descuento VIP.



#### Sistema Descuentos VIP

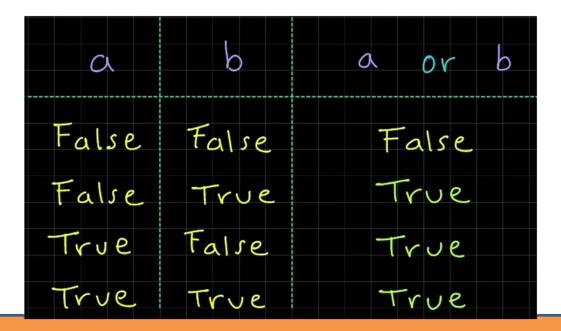
```
descuento_vip.py ×
```



#### **Operador or**

El operador or regresa verdadero si cualquiera de los operandos es verdadero.

Tabla de la verdad





#### **Operador or**

```
operador_or.py ×
```

```
print(f"****Operador or***")

# Regresa verdadero sin ambos valores son verdaderos

condicion1 = False

condicion2 = True

resultado = condicion1 or condicion2

print(f"El resultado {condicion1} or {condicion2} es: {resultado}")
```



#### Sistema Préstamo de libro

Se pide crear un sistema para una biblioteca, la cual desea prestar libros si cumple con cualquiera de las siguientes condiciones:

- El usuario tiene credencial de estudiante.
- El usuario no viva a mas de 3km a la redonda.

Si cumple con cualquier de estas condiciones se le puede prestar el libro.



#### Sistema Préstamo de libro

```
print("***Sistema de prestamo de libros***")

DISTANCIA_PREMITIDA_KM = 3

tiene_credencial = input("Cuentas con credencial de estudiante (Si/No: ")

distancia_biblioteca_km = int(input("A cuantos kilometros vives de la biblioteca?: "))

es_elegible_prestamo = (tiene_credencial.strip().lower()=="si"

or distancia_biblioteca_km <= DISTANCIA_PREMITIDA_KM)

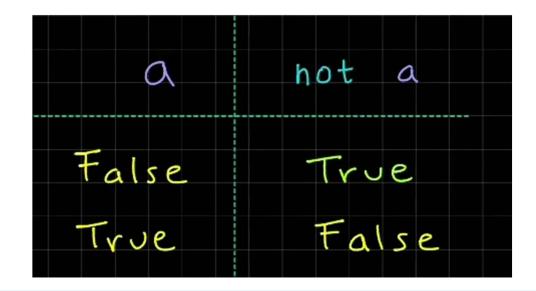
print(f"Eresa elegible para prestamos de libross?: {es_elegible_prestamo}")
```



### **Operador not**

El operador not invierte el valor del operando.

Tabla de la verdad





### **Operador not**

```
operador_not.py ×
```

```
print(f"***Operador not***")
                                                                         △ 5 ★ 16 ^
2
     condicion1 = False
 3
     resultado =not condicion1
 4
     print(f"Operador not sobre {condicion1} es {resultado}")
5
6
     #Revisar si es cadena vacia
7
     nombre =""
     es_cadena_vacia = not nombre
9
     print(f"\nLa variable no tiene ningun valor {es_cadena_vacia}")
10
11
     #Revisar si una variable no tiene ningun valor asignado
12
     varible = None
13
     es_variable_sin_valor = not varible
14
     print(f"\nLa variable no tiene ningun valor asignado? {es_variable_sin_valor}")
15
```



## Ejemplo fuera de rango not

# Revisar si una variable se encuentra dentro del rango entre 1 y 10

```
dentro_rango_not.py ×
```

```
#Revisar si esta dentro del rango entre 1 y 10

dato = int(input("Proporciona un dato entero: "))

#revisamos si esta dentro de rango

esta_dentro_rango = 1<=dato<=10

print(f"Variable esta dentro de rango (entre 1y 10)?: {esta_dentro_rango}")

#Revisamos la logica inversa, si el dato esta fuera der ango

#esta_fuera_rango = not(1<=dato<=10)

#print(f"Variable esta fuera de rango (entre 1 y 10)?: {esta_fuera_rango}")
```



#### Generación Ticket de venta

Supongamos que compramos varios artículos en el supermercado y queremos obtener el ticket de venta total incluyendo impuestos.

El sistema solicitara el precio de cada producto a comprar y el usuario deberá indicar su precio (valor tipo con punto decimal).

El sistema debe realizar la suma de cada producto, calcular el impuesto y finalmente imprimir el total de la compra.



#### Generación Ticket de venta

```
🚃 🥐 ticket_venta.py 🗵
```

```
print("***Generacion ticket de venta***")
1
     precio_leche = float(input("Precio de la leche: "))
     precio_pan = float(input("Precio del pan:"))
 4
     precio_lechuga = float(input("Precio de la lechuga: "))
     precio_banana = float(input("Precio de la banana"))
     #Calculo del subtotal sin impuestos
     subtotal = precio_banana + precio_pan + precio_leche + precio_lechuga
8
     #Calculo con impuestos 19%
     impuesto = subtotal * 0.19
10
     #calculo total de la compra
11
     costo_total = subtotal + impuesto
12
     print(f"""
13
     Subtotal: ${subtotal:.2f}
14
     Impuestos (19%): ${impuesto:.2f}
15
     Costo total de la compra: ${costo_total:.2f}""")
```



#### Sistema de autenticación

Crea un programa para validar el usuario y password proporcionados por el usuario.

Crea dos constantes con valores correctos y posteriormente compara que el usuario y password proporcionados por el usuario son validos

Debe solicitar el usuario y el password al usuario y si son iguales que los valores correctos almacenados en las constantes debe imprimir True, de lo contrario debe imprimir False.



### Valor dentro de rango

Solicitar al usuario un valor dentro del rango 0 y 5 si el valor proporcionado esta dentro del rango.

Se deben definir dos constantes. VALOR\_MINIMO = 0 y VALOR\_MAXIMO = 5.

Y debemos comprobar si el valor proporcionado se encuentra en el rango entre 0 y 5.

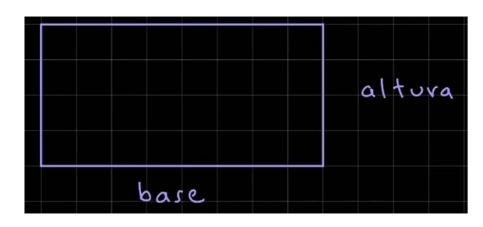
Finalmente se debe imprimir :

Valor dentro del rango: True/False



# Calculo del área y perímetro de rectángulo

Se solicita calcular el área y perímetro de un rectángulo aplicando las siguientes formulas:



Área =base \* altura

Perímetro = 2\*(base + altura)

Finalmente imprimir estos valores en pantalla.

