TALLER DE OPERADORES RELACIONALES Y LÓGICOS EN PYTHON

PARTE # 1

1. **¿Qué devuelve el operador and cuando se usa entre dos valores booleanos?**

   Devuelve True solo si ambos valores son True; en cualquier otro caso, devuelve False.

2. **¿Qué devuelve el operador or cuando se usa entre dos valores booleanos?**

   Devuelve True si al menos uno de los valores es True; solo devuelve False si ambos son False.

3. **¿Cuál es la diferencia entre or y xor en lógica booleana?**

   El operador or devuelve True si al menos una de las condiciones es verdadera. El operador xor (o "o exclusivo") devuelve True solo si exactamente una de las condiciones es verdadera, pero no ambas a la vez.

4. **¿Qué valor devuelve True and False?**

   Devuelve False.

5. **¿Qué valor devuelve False or True?**

   Devuelve True.

6. **¿En Python, existe un operador lógico xor nativo? Si no, ¿cómo se representa?**

   No existe un operador lógico xor (o exclusivo) como palabra reservada. Se puede representar utilizando el operador ^ (bit a bit), o con la expresión:

   (cond1 and not cond2) or (not cond1 and cond2)

7. **¿Cuál es la diferencia entre == y is en Python?**

   == compara si los valores de dos objetos son iguales, mientras

que is comprueba si dos variables apuntan al mismo objeto en memoria (identidad).

8. **¿Qué operador lógico usarías para comprobar si ambas condiciones son verdaderas?**

   Usaría el operador and.

9. **¿Qué operador lógico se usa para verificar si al menos una condición es verdadera?**

   Se usa el operador or.

10. **¿Cómo se utiliza el operador ^ en Python y qué significa cuando se usa entre dos valores booleanos?**

    El operador ^ es el operador "xor" bit a bit. Entre dos valores booleanos, devuelve True solo si los valores son diferentes (uno True y el otro False). Por ejemplo:

    True ^ False # Devuelve True

    True ^ True # Devuelve False

PARTE # 2

11. **Escribe una expresión que devuelva True si x es mayor que 10 y menor que 20**

    x > 10 and x < 20

12. **Condición que devuelva True si a es igual a b o c es mayor que 100:**

    a == b or c > 100

13. **¿Qué devuelve esta expresión y por qué?**

    True or (False and False)

    True or False

    True // operadores tienen un orden de precedencia

14. ¿Qué resultado tiene la siguiente operación?

(5 > 3) ^ (2 < 1)

- (5 > 3) es True
- (2 < 1) es False
- True ^ False (xor) es True **Resultado:** True

15. Reescribe la siguiente expresión usando operadores relacionales y lógicos: not (a < b or b < c)

not (a < b or b < c) ≡ (a >= b) and (b >= c)

16. Condición para x = 10, y = 20, que devuelva True solo si x y y son diferentes y al menos uno de los dos es mayor que 15:

(x != y) and (x > 15 or y > 15)

Para x = 10 y y = 20:

- x != y: True
- x > 15: False, y > 15: True → or: True
- Resultado: True and True → True

17. Evalúa el resultado de:

(10 != 5) and (4 == 4) or (2 > 3)

- (10 != 5): True
- (4 == 4): True
- (2 > 3): False
- True and True → True
- True or False → True

Resultado: True

18. Dado el siguiente código, ¿cuál es el valor de resultado?

a = True
b = False
resultado = a ^ b

- True ^ False → True

  Valor de resultado: True

19. ¿Qué devuelve esta expresión en Python y por qué?

(3 > 2) and (2 > 1) ^ False

Orden de precedencia: ^ se evalúa antes que and.

- (2 > 1): True
- True ^ False → True
- (3 > 2): True
- True and True → True

Resultado: True

Porque: Primero se evalúa el xor, luego el and.

20. Escribe un programa que reciba dos números y diga si exactamente uno de ellos es positivo (usa xor para resolverlo).

```
a = int(input("Introduce el primer número: "))
b = int(input("Introduce el segundo número: "))
if (a > 0) ^ (b > 0):
    print("Exactamente uno de los dos es positivo.")
else:
    print("No se cumple la condición.")
```