Unidad 2

Tipo de dato booleano

Es un tipo de datos que representa los conceptos de verdadero o falso y permite almacenar **True** para representar un **valor verdadero y False** para representar un **valor falso.** También es conocido como tipo de dato lógico.

Ejemplo:

```
es_mayor = True
print(es_mayor)   # Muestra por pantalla: True
es_mayor = False
print(es_mayor)   # Muestra por pantalla: False
```

Operadores relacionales

Los operadores relacionales se utilizan para comparar valores y siempre devuelven un resultado **booleano.**

Operador	Descripción	Ejemplo	Resultado
==	Igual a	5 == 5	True
!=	Distinto de	5 != 5	False
>	Mayor que	7 > 4	True
<	Menor que	8 < 2	False
>=	Mayor o igual que	6 >= 6	True
<=	Menor o igual que	4 <= 5	True

Ejemplo:

```
print(5 > 3)       # Muestra por pantalla: True
print(2 == 10)     # Muestra por pantalla: False
```

Esto es muy útil si comparamos variables, por ejemplo, si quisiera saber si una persona es mayor de edad podría implementar el siguiente programa:

```
edad = int(input("Ingrese su edad: "))
print("¿el usuario es mayor de edad?: ", edad >= 18)
```

Si ejecutamos el programa e ingresamos 22, veremos lo siguiente por pantalla:

```
Ingrese su edad: 22
¿el usuario es mayor de edad?: True
```

Las operaciones que utilizan operadores relacionales se conocen como **expresiones lógicas**.

Estructuras condicionales o de selección

Este tipo de **estructuras** nos **permiten cambiar el flujo de la ejecución de un programa**, haciendo que ciertos bloques de código se ejecuten solo si cumplen una **condición**.

Las **expresiones lógicas** que vimos anteriormente son las que van a decidir si se cumple o no la **condición**.

La estructura condicional está formada por una **cláusula** y una **expresión lógica.**

Las cláusulas que veremos son:

- **if**
- **else**
- **elif**

Condicional if

```
if 5 > 3: # Si se cumple la condición
    print("Se cumplió la condición") # Entonces se ejecuta el bloque de código
```

Si ejecutamos el programa veremos por pantalla:

```
Se cumplió la condición
```

En este caso la expresión lógica 5 > 3 da como resultado True, por lo tanto, se cumple la condición y se logra mostrar el mensaje por pantalla.

Si el resultado de la expresión lógica fuese False no se cumpliría la condición, por lo tanto, no se lograría mostrar el mensaje por pantalla.

Ejemplo:

```
if 5 < 3: # No se cumple la condición
    print("Se cumplió la condición") # Entonces no se ejecuta el bloque de código
```

Como la condición no se cumple, no se puede ejecutar el bloque de código que contiene la función print().

Importante: Es obligatorio que la línea de código siguiente a la estructura condicional utilice **indentacion**. La **indentacion** es el espacio que se deja al comienzo de una línea de código, por lo general son 4 espacios o una tabulación.

Ejemplo:

```
if 5 < 3:  # No se cumple la condición
    print("Se cumplió la condicion") # No se ejecuta porque no se cumplió la condición
print("estoy fuera del bloque condicional") # Si se ejecuta porque esta fuera del bloque
condicional
```

También es importante que si la condición se cumple haya al menos una línea de código dentro del bloque condicional.

Ejemplo:

```
if 5 > 3: # La condición se cumple
print("Se cumplio la condicion") # Python indicara un error antes de llegar al print()
```

Veamos un ejemplo de uso utilizando variables.

```
edad = int(input("Ingrese su edad: "))
if edad >= 18:
    print("Eres mayor de edad")
```

En este caso la condición se cumplirá dependiendo del valor que almacene la variable edad.

Condicional else

Este condicional permite ejecutar un bloque de código solo si el condicional if no fue ejecutado.

Ejemplo:

```
edad = 15
if edad >= 18: # No se cumple la condición
    print("Eres mayor de edad") # No se ejecuta el print()
else:   # Se ejecuta el bloque del condicional else
    print("No eres mayor de edad") # Se ejecuta el print()
```

El bloque de código que se ejecuta luego del condicional else debe tener al menos una instrucción

Condicional elif

Este condicional solo se ejecutará si no se cumplió la condición en la estructura condicional if y siempre ira antes que el condicional else.

Ejemplo:

```
valor = 5
if valor == 4: # No se cumple la condición
    print("el valor es igual a 4")
elif valor == 5: # Se cumple la condición
    print("el valor es igual a 5")
```

Como el valor es igual a 5 entonces la condición elif se cumple y se ejecuta el print() dentro del bloque de código.

También podremos agregar más de un condicional elif o los que necesitemos de acuerdo a las condiciones que vamos a evaluar.

Ejemplo:

```
valor = 6
if valor == 4: # No se cumple la condición
    print("el valor es igual a 4")
elif valor == 5: # No se cumple la condición
    print("el valor es igual a 5")
elif valor == 6: # Se cumple la condición
    print("el valor es igual a 6")
```

Por último, también podremos agregar un condicional else para completar la estructura de selección.

Ejemplo:

```
valor = 99
if valor == 4: # No se cumple la condición
    print("el valor es igual a 4")
elif valor == 5: # No se cumple la condición
    print("el valor es igual a 5")
elif valor == 6: # No se cumple la condición
    print("el valor es igual a 6")
else:  # Como no se cumplió ninguna condición anterior se ejecuta el bloque else
    print("el valor no es 4, 5 o 6")
```

Operadores lógicos

Los **operadores lógicos** permiten **combinar condiciones** para que el programa tome decisiones más complejas. En Python existen 3 principales:

and (Y)

Devuelve verdadero si ambas expresiones son verdaderas.

Ejemplo:

```
edad = 25
tiene_dni = True
if edad >= 18 and tiene_dni:
    print("Puede votar")
```

Solo se ejecuta print(“Puede votar”) si la **edad es mayor o igual a 18 y tiene DNI**. Si una de las dos es falsa no se ejecuta el bloque de codigo.

or (O)

Devuelve verdadero si al menos una de las expresiones es verdadera.

Ejemplo:

```
es_estudiante = False
es_jubilado = True
if es_estudiante or es_jubilado:
    print("Tiene descuento")
```

Aunque no sea estudiante, como es jubilado se ejecuta el bloque de código porque al menos una de las dos condiciones fue valida.

not (no)

Invierte el valor de una expresión booleana.

```
lloviendo = False
if not lloviendo:
    print("Podemos salir sin paraguas")
```

Lloviendo invierte su valor a True por lo que se cumple la condición y se ejecuta el bloque de código.