

Python Básico

Prof. Ariel Palazzesi

Comenzamos a grabar la clase

Módulo 2:

Control de flujo

Clase 03

Ingresos de datos

1. Entrada de datos: función `input()`.
2. Funciones de conversión de tipos de datos: `int()`, `float()`, `str()`, etc.
3. Determinación de tipos: función `type()`.
4. Comentarios en el código.

Clase 04

Condicionales I

1. Operadores relacionales.
2. Toma de decisiones: `if`, `else`, `elif`.
3. Operadores Lógicos.

Clase 05

Bucles while

1. Bucle `'while'`.
2. Contadores.
3. Acumuladores.

Operadores condicionales

Operadores relacionales

✓ Los **operadores relacionales** en programación son herramientas clave para comparar valores y evaluar condiciones.

Incluyen los símbolos “==”, “>”, “<”, “>=”, “<=” y “!=”, que permiten establecer relaciones entre variables y valores.

Estos operadores son fundamentales en estructuras condicionales, donde determinan la ejecución de bloques de código según si una condición dada es verdadera o falsa.

Operadores relacionales

| Operador | Significado | Ejemplo |
|----------|-------------------|------------------------|
| < | menor que | $5 < 7$ (Verdadero) |
| <= | menor o igual que | $5 <= 4$ (Falso) |
| > | mayor que | $3 > 4$ (Falso) |
| >= | mayor o igual que | $13 >= 13$ (Verdadero) |
| != | distinto que | $15 != 15$ (Falso) |
| == | igual que | $10 == 10$ (Verdadero) |

Operadores relacionales

Código Python

```
a = 5
b = 10
print(a>b)
print(a==b)
print(a!=b)
print(a==a)
```

Terminal

```
False
False
True
True
```


Bloques de código (Indentación)

Bloques de código (Indentación)

✓ No todos los lenguajes de programación necesitan de una **indentación**, aunque sí se estila implementarla a fin de otorgar mayor legibilidad al código fuente. Pero **en el caso de Python, la indentación es obligatoria**, ya que de ella dependerá su estructura.

Código Python

```
def suma_numeros (numeros): # Bloque 1
    suma = 0                 # Bloque 2
    for n in numeros:       # Bloque 2
        suma += n           # Bloque 3
        print(suma)         # Bloque 3
    return suma              # Bloque 2

print()                     # Bloque 4
```

Estructuras de control: Condicionales

Estructuras de control

✓ En programación, las **estructuras de control** permiten modificar el flujo de ejecución de las instrucciones de un programa.

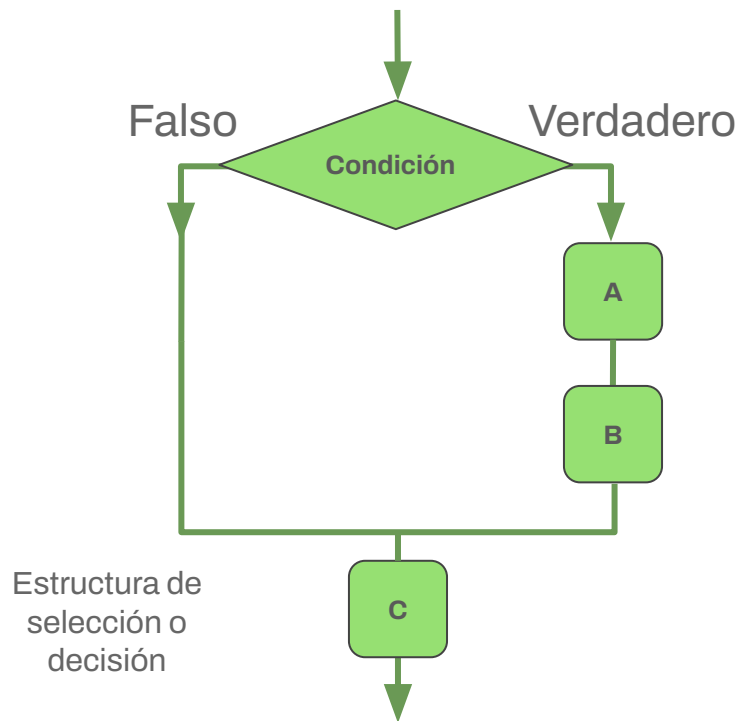
Con ellas se puede ejecutar un grupo u otro de sentencias, según se cumpla o no una condición.

También existen estructuras de control que permiten ejecutar un grupo de sentencias mientras se cumpla una condición o repetir un grupo de sentencias un número determinado de veces.

Estructuras de control - Condicionales

✓ Las estructuras condicionales ejecutan un bloque de instrucciones u otro, o saltan a otro según se cumpla o no una condición.

Esta condición solamente puede ser verdadera o falsa.



Estructuras de control - Condicionales

✓ La palabra clave asociada a esta estructura es **if**. Si la condición es **True** se ejecuta el **bloque** dentro del **if**. Luego, independientemente del valor de verdad de la condición, el programa continúa con la ejecución del resto del programa.

Código Python

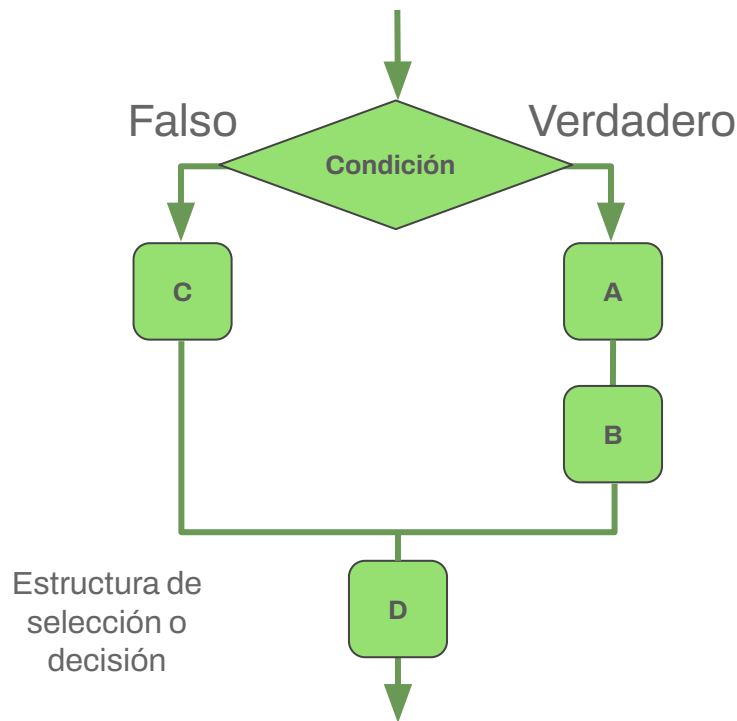
```
nota = float(input("Ingrese la calificación: "))

if nota >= 7:
    print("Aprobado.") # Bloque de instrucciones que se ejecuta
                        # si la condición es verdadera
```

Estructuras de control - Condicionales

✓ Podemos utilizar la cláusula **else** para incluir otro bloque de instrucciones que se ejecutará en caso de que la evaluación de la condición resulte ser **falsa**.

Es decir, podemos ejecutar un bloque de instrucciones u otro, dependiendo de si la condición es **verdadera** o **falsa**.



Estructuras de control - Condicionales

Código Python

```
if edad >= 18:
    print("Puedes pasar.") # Bloque de instrucciones que se
                          # ejecuta si la condición es verdadera
else:
    print("No admitido.") # Bloque de instrucciones que se
                          # ejecuta si la condición es falsa
```

Operadores lógicos

Operadores lógicos



Se pueden utilizar **operadores lógicos** para tomar una decisión basada en múltiples condiciones, reduciendo la cantidad de if.

Las **tablas de verdad** muestran los valores de verdad de una proposición en función del valor lógico de sus operadores:

| Cond 1 | Cond 2 | Y (and) |
|--------|--------|---------|
| V | V | V |
| V | F | F |
| F | V | F |
| F | F | F |

| Cond 1 | Cond 2 | O (or) |
|--------|--------|--------|
| V | V | V |
| V | F | V |
| F | V | V |
| F | F | F |

| Cond | NO (not) |
|------|----------|
| V | F |
| F | V |

Desafíos

Desafíos de la clase:

✓ Desafío 1: Tope máximo de extracción

Escribe un programa para un cajero automático de un banco que permita ingresar un importe y determine si la extracción puede hacerse. El tope máximo para una extracción es de \$ 200.

✓ Reto:

Modifica el código para que tenga almacenado el saldo disponible, y permita hacer la extracción si el saldo disponible es suficiente. Si no se puede realizar la extracción, indicar el motivo con un mensaje.

Desafíos de la clase (Resolución):

✓ Desafío 1: Tope máximo de extracción

Código Python

```
# Programa para verificar tope máximo de extracción
importe = float(input("Ingrese el importe que desea extraer: "))

if importe > 200:
    print("Error: El importe excede el tope máximo de extracción de $200.")
else:
    print("Extracción permitida. Puede retirar $", importe)
```


Desafíos de la clase (Resolución):

Código Python

```
saldo_disponible = 500 # Saldo inicial de ejemplo
print("Saldo disponible:", saldo_disponible)
importe = float(input("Ingrese el importe que desea extraer: "))
if importe > 200:
    print("Error: El importe excede el tope máximo de extracción de $200.")
else:
    if importe > saldo_disponible:
        print("Error: No tiene suficiente saldo para realizar esta extracción.")
    else:
        saldo_disponible -= importe # Descontamos el importe del saldo
        print("Extracción exitosa. Ha retirado $", importe)
        print("Saldo restante: $", saldo_disponible)
```

Desafíos de la clase (Resolución):

✓ Desafío 1 resuelto con elif:

Código Python

```
saldo_disponible = 500 # Saldo inicial de ejemplo
print("Saldo disponible:", saldo_disponible)
importe = float(input("Ingrese el importe que desea extraer: "))

if importe > 200:
    print("Error: El importe excede el tope máximo de extracción de $200." )
elif importe > saldo_disponible:
    print("Error: No tiene suficiente saldo para realizar esta extracción." )
else:
    saldo_disponible -= importe # Descontamos el importe del saldo
    print("Extracción exitosa. Ha retirado $" , importe)
    print("Saldo restante: $" , saldo_disponible)
```

Desafíos de la clase:

✓ Desafío 2: Control de acceso a un evento

Escribe un programa para un sistema de control de acceso a un evento. El programa debe pedir al usuario que ingrese su edad y si tiene una entrada válida. Solo se permite el ingreso si la persona tiene **18 años o más** y posee una **entrada válida**. El programa debe imprimir uno de los siguientes mensajes según corresponda:

1. **"Acceso permitido"** si cumple ambos requisitos.
2. **"Acceso denegado: debe tener al menos 18 años"** si no cumple con la edad.
3. **"Acceso denegado: no tiene una entrada válida"** si no cumple con la entrada.
4. **"Acceso denegado: no cumple con los requisitos"** si no cumple con ninguno de los dos.

Desafíos de la clase (Resolución):

Código Python

```
edad = int(input("Ingrese su edad: "))
entrada_valida = input(";Tiene entrada válida? (si/no): ")
if edad >= 18 and entrada_valida == "si":
    print("Acceso permitido")
else:
    if edad < 18 and entrada_valida != "si":
        print("Acceso denegado: no cumple con los requisitos")
    else:
        if edad < 18:
            print("Acceso denegado: debe tener al menos 18 años")
        else:
            print("Acceso denegado: no tiene una entrada válida")
```


Desafíos de la clase (Resolución):

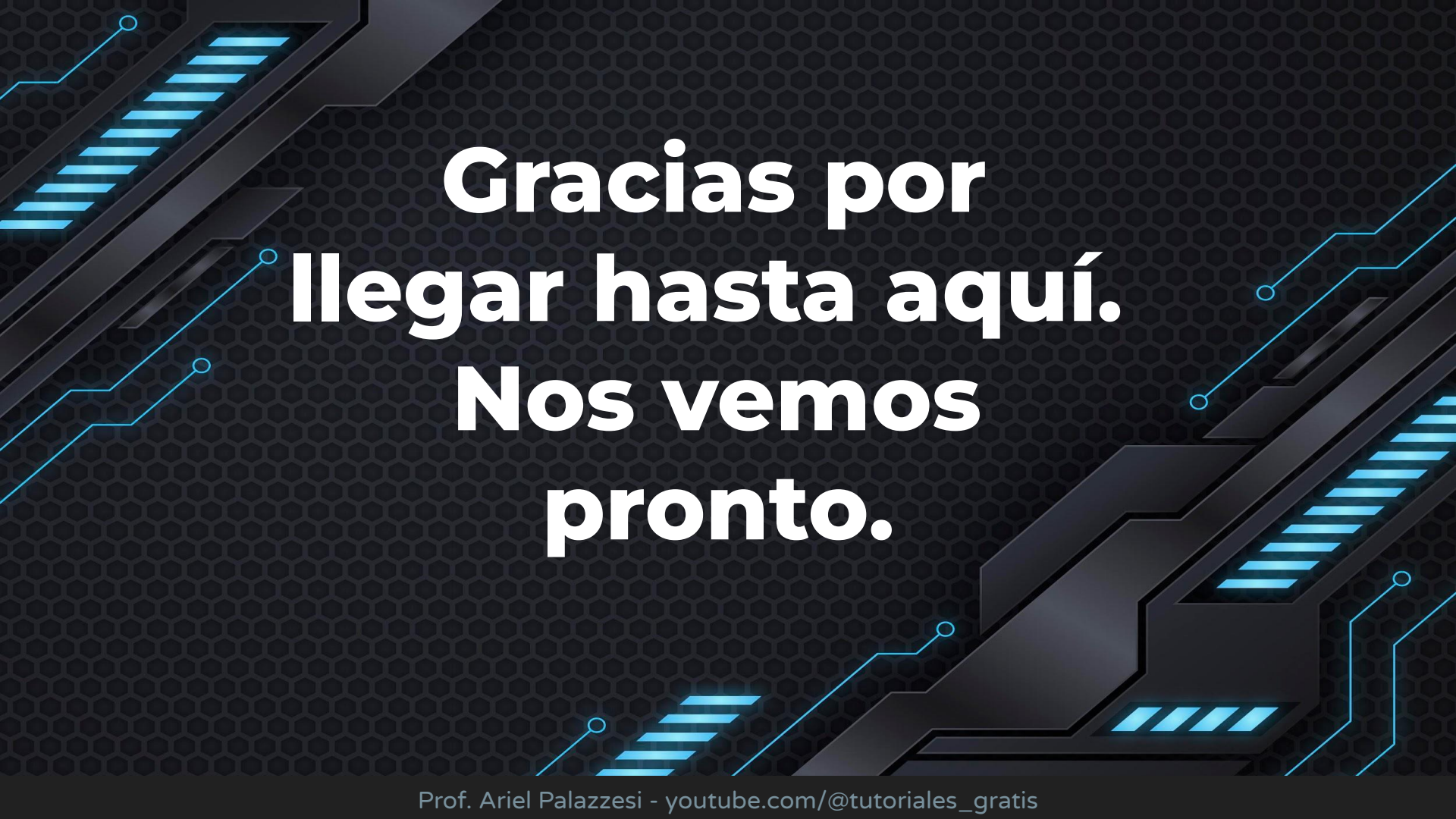


Desafío 2 resuelto con elif:

Código Python

```
edad = int(input("Ingrese su edad: "))
entrada_valida = input("¿Tiene entrada válida? (si/no): ")

if edad >= 18 and entrada_valida == "si":
    print("Acceso permitido")
elif edad < 18 and entrada_valida != "si":
    print("Acceso denegado: no cumple con los requisitos")
elif edad < 18:
    print("Acceso denegado: debe tener al menos 18 años")
elif entrada_valida != "si":
    print("Acceso denegado: no tiene una entrada válida")
```



**Gracias por
llegar hasta aquí.
Nos vemos
pronto.**