

# Programación para Física y Astronomía

Departamento de Física.

---

Corodinadora: C Loyola

Profesores C Femenías / F Bugini / D Basantes

Primer Semestre 2025

Universidad Andrés Bello

Departamento de Física y Astronomía



Introducción y Repaso

Estructuras de Control: Condicionales

Estructuras de Control: Bucles While

Ejemplos Prácticos

# Introducción y Repaso

---

- **Semana 2, Sesión 2 (Sesión 4)** se enfocó en:
  - Integrar y aplicar conceptos básicos de Python en problemas más elaborados.
  - Ejercicios prácticos sobre ecuaciones de movimiento, varianza y conversiones de unidades.
  - Manejo de pequeños bucles **for** y condicionales sencillos **if/elif**.
- **Meta de hoy:** Introducir y profundizar en las estructuras de control fundamentales (condicionales y bucles).

- **Comprender** la sintaxis y el uso de las estructuras de control en Python (`if`, `elif`, `else` y `while`).
- **Analizar** ejemplos prácticos que requieran decisiones condicionales y repeticiones.
- **Diseñar** programas sencillos que apliquen bucles `while` con criterio de parada.
- **Relacionar** estas estructuras con problemas físicos o astronómicos básicos.

## Estructuras de Control: Condicionales

---

# Estructuras de Control: Condicionales ∈ La Estructura `if`, `elif`, `else`

- Permiten ejecutar un bloque de código si se cumple una condición.
- Sintaxis básica:  

```
if <condicion1>:  
    <bloque1>  
elif <condicion2>:  
    <bloque2>  
else:  
    <bloque3>
```
- Se pueden tener *múltiples* `elif` y *opcionalmente* un `else`.

# Estructuras de Control: Condicionales ∈ Ejemplo: Determinar si una Nota Aprueba o No

```
1 nota_str = input("Ingresa tu nota (0.0 - 7.0): ")
2 nota = float(nota_str)
3
4 if nota >= 4.0:
5     print("Aprobado")
6 else:
7     print("Reprobado")
```

## Discusión:

- ¿Qué pasa si la nota está fuera del rango esperado?
- ¿Podríamos manejar **elif** para rangos (ej. “sobresaliente”, “suficiente”, etc.)?



# Estructuras de Control: Condicionales ∈ Actividad 1: Crear un Menú Sencillo

## Enunciado

- Pedir al usuario una opción entre:
  - 1 - Calcular el área de un cuadrado.
  - 2 - Calcular el área de un triángulo.
  - 3 - Salir.
- Usar **if/elif/else** para procesar la opción.
- Si se elige **1** o **2**, pedir dimensiones, calcular y mostrar el resultado.
- Si se elige **3**, finalizar el programa.

**Consejo:** Manejar opciones inválidas (mensaje de error).

## Estructuras de Control: Bucles While

---

- Repite un bloque de código **mientras** una condición sea verdadera.
- Sintaxis:  
`while <condicion>:`  
    <bloque>
- Importante asegurarse de que la condición **cambie** para evitar bucles infinitos.

## Estructuras de Control: Bucles While ∈ Ejemplo: Contador Descendente

```
1 count = 5
2 while count > 0:
3     print("Cuenta atrás:", count)
4     count -= 1
5
6 print("¡Fin de la cuenta!")
```

### Análisis:

- `count -= 1` evita que el bucle sea infinito.
- Se ejecuta el bloque mientras `count > 0`.

# Estructuras de Control: Bucles While ∈ Actividad 2: Suma Iterativa

## Enunciado

- Pide repetidamente al usuario ingresar un número.
- Suma todos los números ingresados.
- Si el usuario ingresa **0**, terminar el bucle y mostrar la suma final.

**Tip:** Usa un **while True** y **break** al detectar 0, o controla la condición en **while**.

## Ejemplos Prácticos

---

## Ejemplos Prácticos ∈ Ejemplo 1: Búsqueda de Raíz (Método Ingenuo)

- **Problema:** Encontrar  $n$  tal que  $n^2 \approx m$  para un  $m$  dado.
- **Idea:**
  - Partir de  $n = 0$ .
  - Incrementar  $n$  en 1 hasta que  $n^2 \geq m$ .
  - Al final,  $n$  está cerca de  $\sqrt{m}$ .
- **Uso:** `while` para la búsqueda.
- Posteriormente: Mejorar con métodos numéricos (Newton-Raphson).

```
1 m_str = input("Ingresa un número (m): ")
2 m = float(m_str)
3
4 n = 0
5 while (n**2) < m:
6     n += 1
7
8 print("Resultado aproximado:", n)
9 print("n^2 =", n**2, ", m =", m)
```

### Discusión:

- Exactitud vs. tiempo de ejecución.
- Controlar **n** o implementar un **break** si **n** se vuelve muy grande.



## Ejemplos Prácticos ∈ Ejemplo 2: Verificación de Input

```
1 clave_correcta = "astronomia"
2 intentos = 3
3
4 while intentos > 0:
5     clave = input("Ingresa la clave: ")
6     if clave == clave_correcta:
7         print(";Bienvenido/a!")
8         break
9     else:
10        intentos -= 1
11        print("Clave incorrecta. Intentos restantes:", intentos)
12
13 if intentos == 0:
14     print("Has agotado todos los intentos. Acceso denegado.")
```

### Comentario:

- Ejemplo de `if + while` para autenticar con un número limitado de intentos.

### Enunciado

- Repetir un menú hasta que el usuario seleccione la opción `"Salir"`.
- Opciones posibles:
  - (1) Calcular área de círculo.
  - (2) Calcular energía potencial:  $E_p = mgh$ .
  - (3) Salir.
- Usa `if/elif/else` dentro de un `while True` y un `break` cuando sea `Salir`.

**Tip:** Cada operación requiere datos distintos (masa, altura, radio, etc.).

- **Formen equipos** de 2-3 personas.
- **Objetivo:** Implementar el menú iterativo con `while`.
- **Desafío:** Agregar validaciones (por ej. valores negativos en masa/altura).
- Comparar soluciones y resaltar las diferencias de implementación.

- ¿Surgieron bucles infinitos? ¿Cómo se detectaron y resolvieron?
- ¿Alguna validación extra para datos imposibles o nulos?
- **Buenas prácticas:** Comentar el código y usar nombres de variables descriptivos.

# Ejemplos Prácticos ∈ Ejemplo de Solución para el Menú Interactivo

```
1 import math
2
3 while True:
4     print("=== MENÚ ===")
5     print("(1) Área de círculo")
6     print("(2) Energía potencial (mgh)")
7     print("(3) Salir")
8     opcion = input("Elige una opción: ")
9
10    if opcion == "1":
11        r_str = input("Radio del círculo (m): ")
12        r = float(r_str)
13        area = math.pi * r**2
14        print("Área =", area, "m^2")
15    elif opcion == "2":
16        m_str = input("Masa (kg): ")
17        h_str = input("Altura (m): ")
18        g = 9.8
19        m = float(m_str)
```

- Uso de **while True** + **break** para controlar el flujo.
- Estructura **if/elif/else** para manejar opciones.
- Se separan las variables y cálculos de cada opción.
- **Posible mejora:** Manejo de errores al convertir **str** a **float**.

### Enunciado

- El programa genera un número al azar entre 1 y 10.
- El usuario intenta adivinar el número ingresando valores.
- Indicar si el número es **mayor**, **menor** o **igual**.
- Terminar cuando el usuario acierte o supere los 5 intentos.

**Tip:** Usa `import random` y `random.randint(1,10)` para generar el número.

- Implementa el juego de **Adivinar un Número** en un **notebook** de Colab.
- Añade mensajes claros para el usuario (“**El número es menor**” / “**El número es mayor**”).
- Verifica que se detenga en 5 intentos (o en el acierto).
- Al terminar, muestra cuántos intentos usó el jugador.



- ¿Lograste implementar la lógica de `if` dentro de un `while` sin errores?
- ¿Qué ocurrió cuando el usuario ingresa valores fuera del rango?
- **Extensión:** Permitir varios rangos de adivinanza o puntajes.

- **Simulaciones en Física:** Muchas implican decisiones (condiciones de borde, colisión, etc.).
- **Procesamiento de Datos:** Filtrar registros según criterios (`if` para descartar outliers).
- **Automatización:** Scripts que se repiten hasta cumplir una condición.

**Conclusión:** Las estructuras de control son la base de la lógica en programación.

- Python Docs: Control Flow.
- W3Schools: Python Conditions.
- Automate the Boring Stuff (capítulos sobre `if` y `while`).

- **Combina** estructuras de control: Un `while` que usa `if/elif/else` internamente.
- **Gestiona** valores inválidos con `try/except` (si deseas profundizar en manejo de errores).
- **Experimenta** con ejemplos de la vida real (cálculo de impuestos, clasificación según edad, etc.).

# ¡Gracias y hasta la próxima sesión!

- Continúen practicando `if`, `elif`, `else`, `while`.
- En la siguiente sesión, profundizaremos en `for` y prácticas de control de flujo (`break`, `continue`).
- ¡No olviden guardar sus notebooks en Drive!