

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 a la Sesión 3

Sintaxis Básica de Python

Refuerzo de Tipos y Variables

Ejemplos Interactivos

Actividad Colaborativa

Conclusiones

Introducción a la Sesión 3

- **Abordar** la sintaxis básica de Python de forma más sistemática.
- **Profundizar** en los tipos de datos, variables y operaciones aritméticas.
- **Realizar** ejemplos interactivos en Google Colab para afianzar la comprensión.
- **Fomentar** la colaboración y el trabajo grupal.

- Sesión 1 y 2 (Semana 1) se enfocaron en:
 - Introducción al entorno Google Colab.
 - Operaciones básicas y entrada/salida en Python.
 - Ejercicios simples de asignación y aritmética.
- Ahora profundizaremos en la **estructura y sintaxis** de Python.

Sintaxis Básica de Python

Sintaxis Básica de Python \ni Reglas Fundamentales de la Sintaxis de Python

- **Indentación:** Bloques de código se definen por la sangría (4 `espacios` usualmente).
- **Uso de dos puntos (:)** para iniciar bloques (if, while, for, funciones, etc.).
- **Sensibilidad a mayúsculas/minúsculas:** `var` es distinto de `Var`.
- **Comentarios:** Empiezan con `#` en una sola línea o con `""" ... """` para bloques.

Ejemplo de comentarios en Python:

```
1  # Esto es un comentario de una sola línea
2
3  """
4  Este es un comentario
5  que abarca múltiples
6  líneas de texto
7  """
8
9  x = 5  # Asigno 5 a x
10 print(x)
```

- **PEP8:** Recomendaciones oficiales de estilo (espacios, nombres de variables, etc.). <https://peps.python.org/pep-0008/>
- Facilita la lectura y el mantenimiento de código.

Sintaxis Básica de Python \ni Palabras Reservadas y Convenciones

- Palabras Reservadas: `if`, `elif`, `else`, `for`, `while`, `import`, `class`, `def`, `return`, ...
- Nombrado de variables:
 - Usar *snake_case* para variables y funciones: `mi_variable`, `calcular_area()`.
 - Mayúsculas para constantes: `PI = 3.14159`.
- **Longitud de línea:** Idealmente menor a 79 caracteres (conforme a PEP8).

Refuerzo de Tipos y Variables

Refuerzo de Tipos y Variables \ni Recordatorio: Tipos de Datos en Python

- **int**: números enteros (10, -3, 0, 9999).
- **float**: números con punto decimal (3.1415, 1.0, -2.5).
- **complex**: números complejos (3+4j).
- **bool**: True / False.
- **str**: cadenas de texto ("Hola", 'Mundo').

Refuerzo de Tipos y Variables \ni Conversiones Explícitas (Casting)

```
1  # De string a entero
2  edad_str = input("Ingresa tu edad: ")
3  edad = int(edad_str)
4
5  # De string a float
6  altura_str = input("Ingresa tu altura (m): ")
7  altura = float(altura_str)
8
9  # De int a float
10 numero_entero = 5
11 numero_flotante = float(numero_entero)  # 5.0
```

- **ValueError** si la conversión es inválida (ej.: `int("hola")`).

```
1  # Orden de precedencia:
2  # 1. Paréntesis
3  # 2. Exponente **
4  # 3. Multiplicación, División, Módulo
5  # 4. Suma y Resta
6
7  a = 2 + 3 * 4          # 2 + 12 = 14
8  b = (2 + 3) * 4        # 5 * 4 = 20
9  c = 2**3 * 3           # 2^3 * 3 = 24
```

- Evitar ambigüedades: usar paréntesis cuando sea necesario.
- Operadores compuestos: +=, -=, *=, /=.

Refuerzo de Tipos y Variables \ni Actividad 1: Ejercicios Rápidos en Colab

- Crea un nuevo **notebook** para “Semana 2, Sesión 1”.
- Implementa ejercicios:
 1. Calcular $(3 + 4) * 2^2$ y mostrar resultado.
 2. Ingresar una cadena y convertirla en **int** o **float**.
 3. Usar un **print** para concatenar texto con variables numéricas.
- **Tip:** Observa qué pasa si ingresas valores no válidos.

Ejemplos Interactivos

Ejemplos Interactivos ÷ Ejemplo 1: Manejo de Variables

```
1 nombre = input("¿Cuál es tu nombre? ")
2 edad_str = input("¿Cuál es tu edad? ")
3 edad = int(edad_str)
4
5 # Jugando con los datos
6 edad_futura = edad + 5
7 print("Hola,", nombre)
8 print("Hoy tienes", edad, "años.")
9 print("Dentro de 5 años tendrás", edad_futura, "años.")
```

Discusión: Manejo de `int` vs. `str`, impresiones múltiples.


```
1  # Aritmética directa en una celda Colab
2  result1 = (2 + 3) * (5 - 1)
3  result2 = 10 / 2 + 6 // 3 - 4
4  print("Result1 =", result1)
5  print("Result2 =", result2)
```

Punto a destacar:

- // es **división entera** (trunca el resultado).
- / es **división flotante** (da decimal).

Actividad Colaborativa

Enunciado

- Crear un programa que simule una “mini-calculadora”.
- Se piden dos números (float) y una operación (+, -, *, /).
- Se muestra el resultado de la operación.

Extensión: Manejar la división por cero con un mensaje de error.

- Equipos de 2-3 integrantes.
- Cada equipo edita un notebook compartido en Google Colab.
- Discutir la mejor forma de:
 - Pedir los datos al usuario.
 - Validar la operación.
 - Mostrar los resultados.
- Al final, compararán las soluciones y estrategias.

Actividad Colaborativa \ni Ejemplo de Solución (Mini-Calculadora)

```
1 num1_str = input("Ingresa el primer número: ")
2 num2_str = input("Ingresa el segundo número: ")
3 op = input("Ingresa la operación (+, -, *, /): ")
4
5 num1 = float(num1_str)
6 num2 = float(num2_str)
7
8 if op == "+":
9     res = num1 + num2
10 elif op == "-":
11     res = num1 - num2
12 elif op == "*":
13     res = num1 * num2
14 elif op == "/":
15     if num2 == 0:
16         res = "Error: división por cero"
17     else:
18         res = num1 / num2
19 else:
20     res = "Operación inválida"
21
22 print("Resultado:", res)
```

- **Uso de if/elif/else:** Estructura básica de control, veremos más en profundidad pronto.
- **Validaciones:** Cómo manejar entradas no numéricas o el caso de división por cero.
- **Buenas prácticas:** Comentar líneas clave, usar nombres de variables descriptivos.

- ¿Qué método usó cada grupo para manejar la división por cero?
- ¿Cómo presentar un mensaje de error informativo?
- ¿Se te ocurrió alguna manera de extender la mini-calculadora? (e.g. potencia, módulo).

Comparte tus conclusiones con la clase.

Conclusiones

- Reafirmamos la **estructura básica** y reglas de sintaxis en Python (indentación, mayúsculas, etc.).
- Exploramos con mayor detalle los **tipos de datos** y la **aritmética**.
- Realizamos ejercicios prácticos en Colab para afianzar la comprensión.
- Desarrollamos una pequeña **mini-calculadora** colaborativamente.

- **Sesión 4 (Semana 2):** Continuar con **estructuras de control** (`if`, `while`), y ejercicios más complejos.
- **Tarea sugerida:**
 - Practicar la creación de scripts con menús (mini-calculadora mejorada).
 - Revisar la documentación de Python sobre tipos numéricos y funciones matemáticas.

- **Python Official Tutorial** (apartados 1-3).
- **Real Python** (guías introductorias).
- **PEP8 Style Guide**: Recomendaciones de estilo y formato (útil para mantener código limpio).

¡Desafío!

- Crea un programa que pida el **radio de un círculo** y elija:
 - '1' para calcular el **diámetro**.
 - '2' para calcular la **circunferencia**.
 - '3' para calcular el **área**.
- Muestra el resultado correspondiente (usa `math.pi`).

Comentario: Requiere condicionales y entradas numéricas.

- Al finalizar la clase, escribe **notas de lo aprendido** en tu cuaderno.
- Identifica *dónde* has tenido más dudas:
 - Manejo de strings vs. números.
 - Operaciones matemáticas.
 - Orden de ejecución en un notebook.
- Prepara preguntas concretas para la siguiente sesión.

- ¿Te sientes más cómodo con la sintaxis de Python?
- ¿Ha quedado clara la importancia de la indentación?
- ¿Tienes inquietudes sobre el uso de Google Colab u otro entorno?

Expón tus dudas e impresiones.

¡Gracias y hasta la próxima sesión!

- Asegúrate de **guardar tu notebook** en Drive.
- Si tienes tiempo, aborda el ejercicio adicional.
- Nos vemos en la **Sesión 4 (Semana 2)** para profundizar en `if` y `while`.