Programación para Física y Astronomía

Departamento de Física.

Coordinadora: C Loyola
Profesores C Femenías / F Bugini / D Basantes

Primer Semestre 2025

Universidad Andrés Bello
Departamento de Física y Astronomía

Universidad Andrés Bello®

python powered

Resumen - Sesión 1 (Semana 1)

Contexto del curso e Introducción

Herramientas Principales

Fundamentos de Python

Ejercicios Guiados

Actividad Práctica

Curiosidades y Recursos

Conclusiones

Contexto del curso e Introducción

- **Asignatura:** Programación para la Física y Astronomía.
- **Período:** 1er semestre (de acuerdo al Syllabus).
- **Objetivo principal de esta sesión:**
  - Presentación de Syllabus Oficial
  - Introducir las herramientas fundamentales del curso.
  - Familiarizarnos con el entorno de programación (Python, Google Colab).
- Corresponde a la **Sesión 1, Semana 1**.

Contexto del curso e Introducción ∋ ¿Por qué Programar en Física y Astronomía?

- Muchas áreas de la Física y Astronomía requieren simulaciones y análisis de grandes volúmenes de datos.
- Python se ha vuelto esencial para:
  - Resolver problemas numéricos complejos.
  - Procesar y visualizar datos (observacionales o experimentales).
  - Facilitar la reproducibilidad de la investigación.
- Además, tiene una enorme comunidad científica activa.

- **Unidad I:** Elementos Básicos (GNU/Linux, Google Colab, Python).
- **Unidad II:** Programación en Python (tipos, aritmética, funciones).
- **Unidad III:** Controladores y arreglos (if, while, for, listas, slicing).
- **Unidad IV:** Gráficas con Matplotlib.
- **Unidad V:** Manejo de datos (clases, estadística, NumPy/Pandas).
- **Unidad VI:** Algoritmos y Performance (sorting, recursividad, hilos, etc.).

Syllabus 2025

Vamos entonces a revisar el detalle del Syllabus de este período.

Herramientas Principales

Herramientas Principales ∋ Google Colab

- Plataforma online gratuita de Google para programar en Python.
- **Ventajas:**
  - No requiere instalación local.
  - Integrada con Google Drive (colaboración sencilla).
  - Ejecución en la nube (libera recursos locales).
- Sólo necesitas una cuenta de Google.

Nota: Otras alternativas incluyen Jupyter, VSCode, etc.

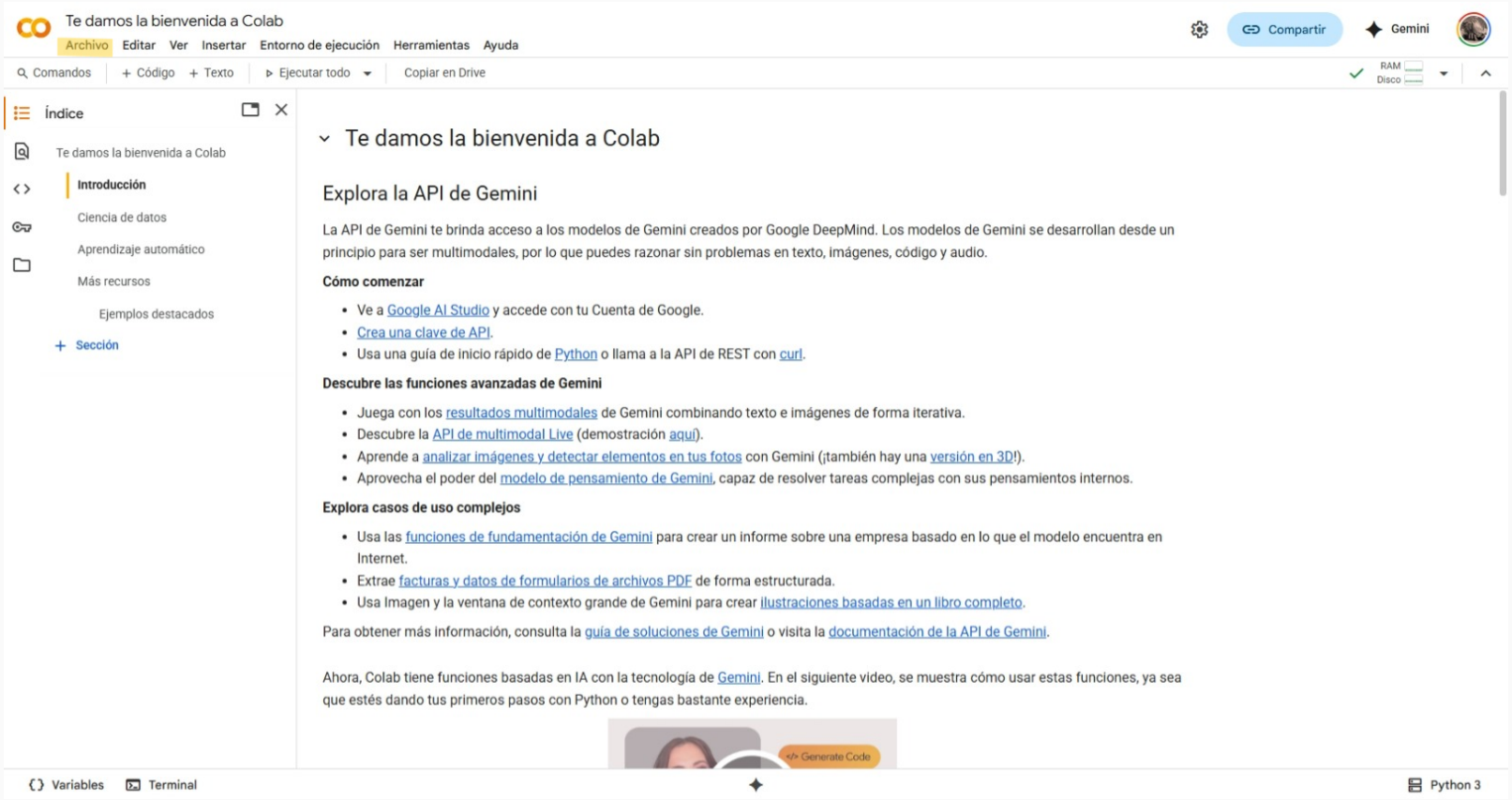
IMPORTANTE!!!

El que nuestro curso sea de programación, es **altamente recomendado** que cada estudiante tenga su propio cuaderno, en donde vamos a anotar cosas importante a medida que las clases avancen.

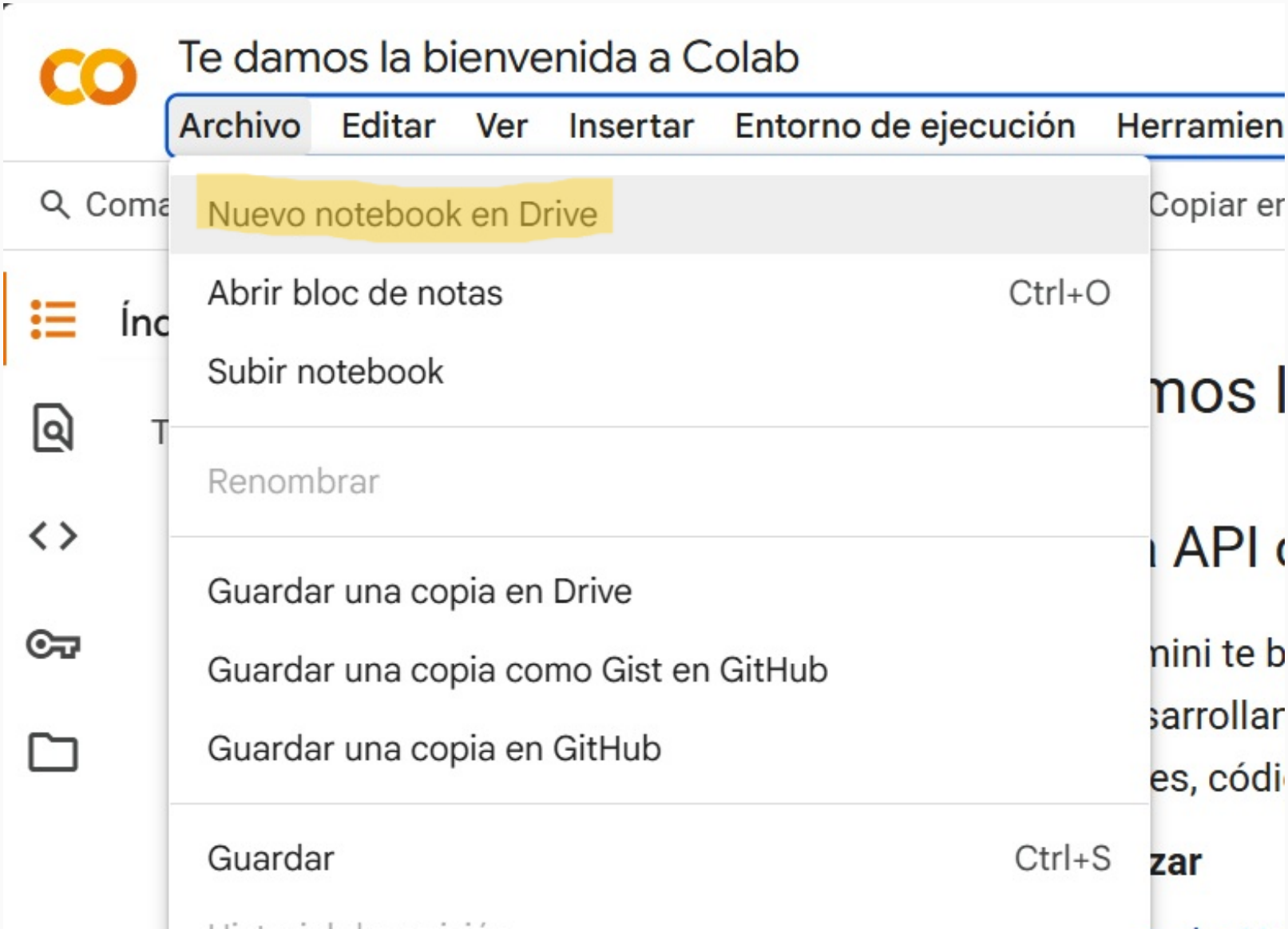
1. Visita: `https://colab.research.google.com`
2. Inicia sesión con tu cuenta de Google.
3. Crea un nuevo cuaderno (*New Notebook*).
4. Almacena el archivo en tu Google Drive.

Sugerencia: Organiza tus carpetas en Drive para mantener un buen orden.

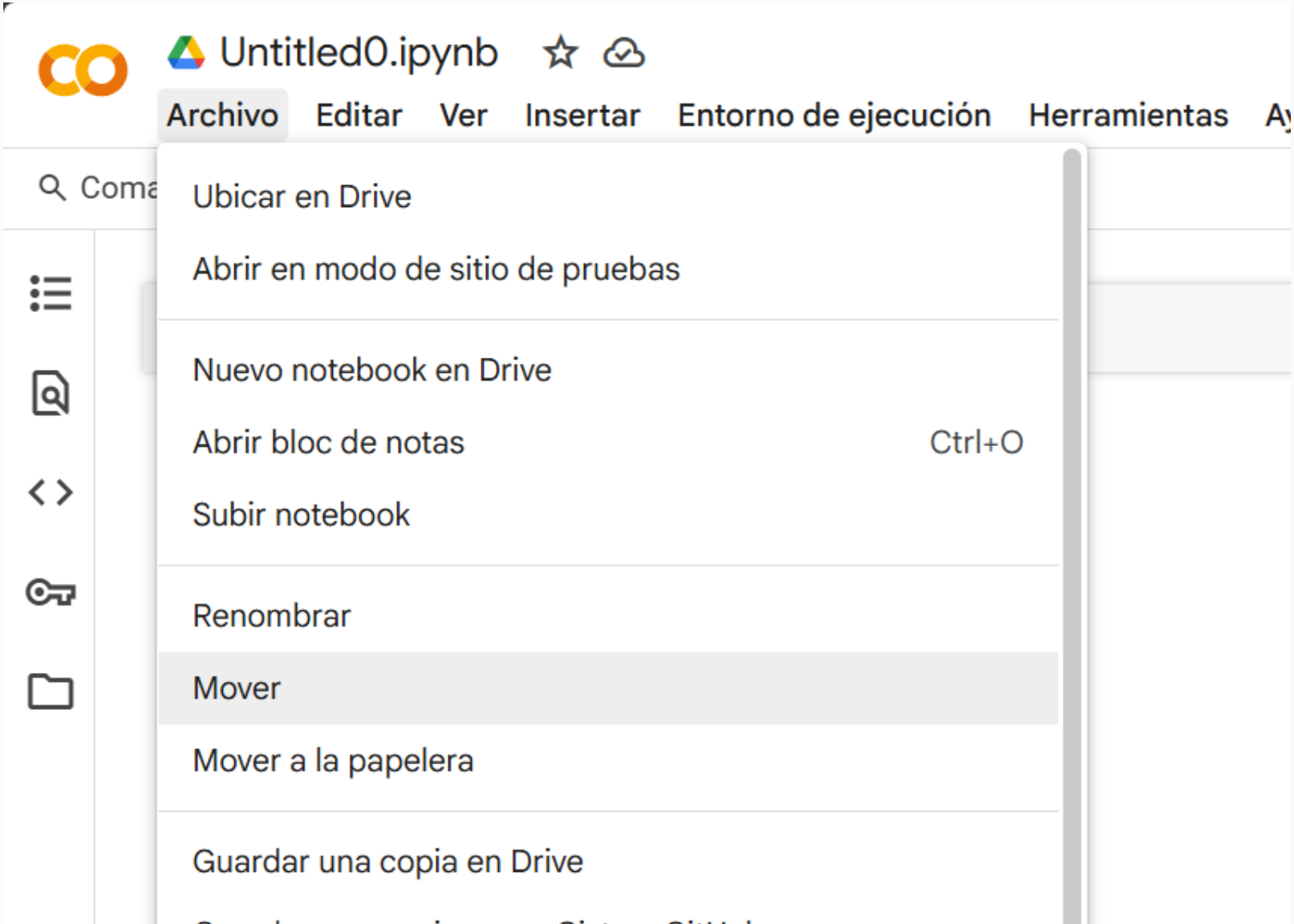
Paso 1: Acceso inicial a Google Colab



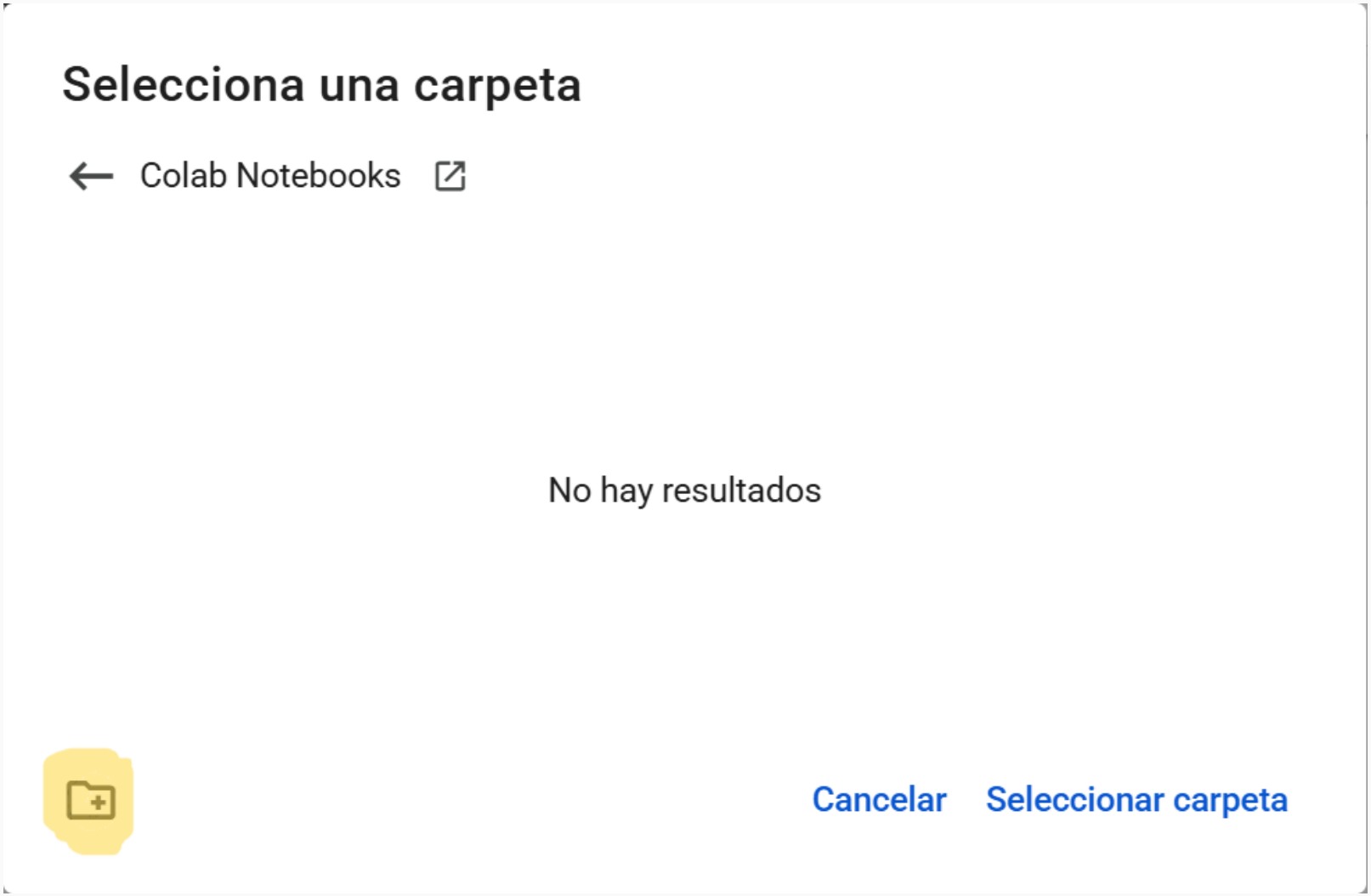
Paso 2: Creación de nuevo notebook



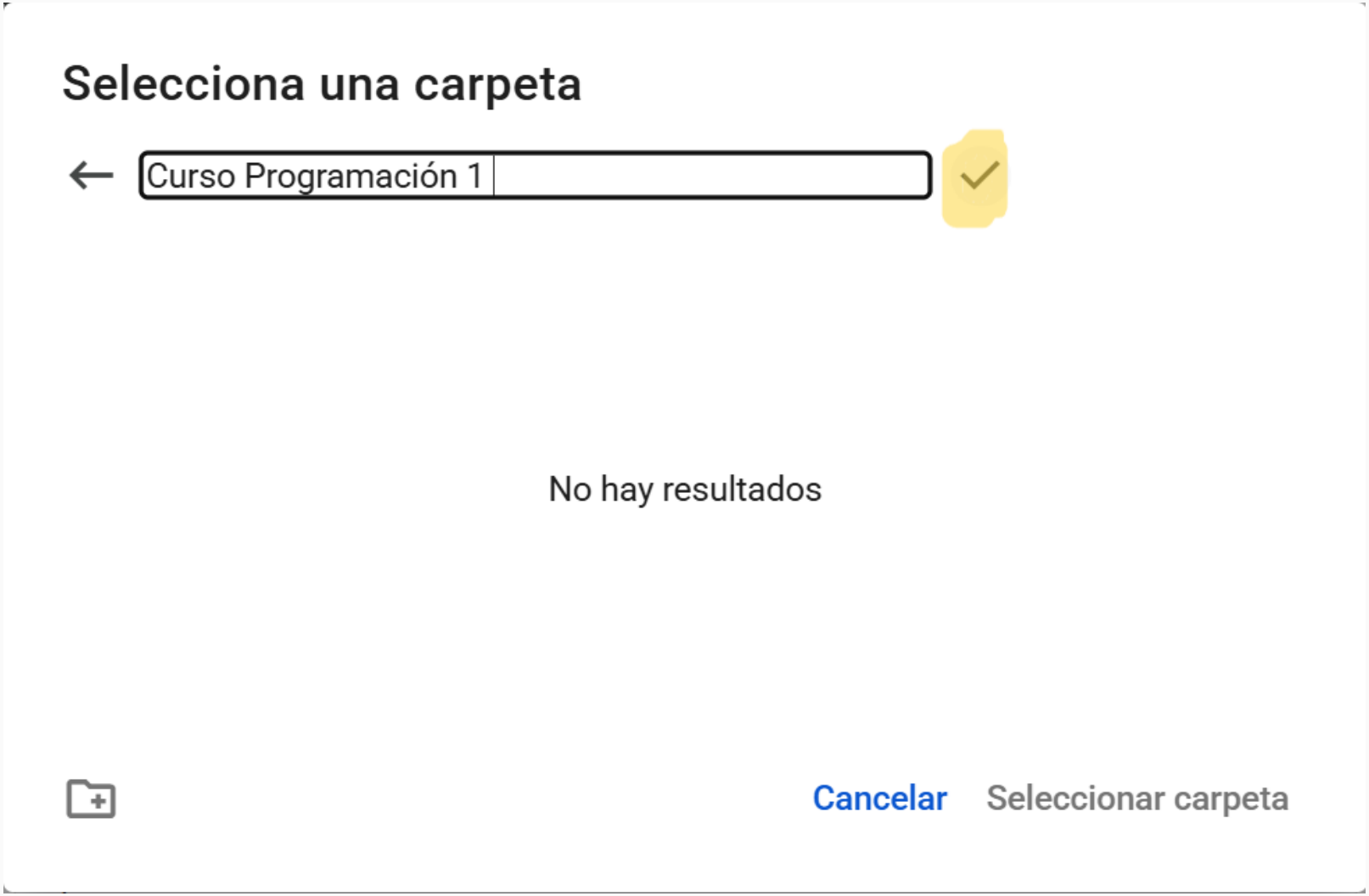
Paso 3: Donde guardar el notebook



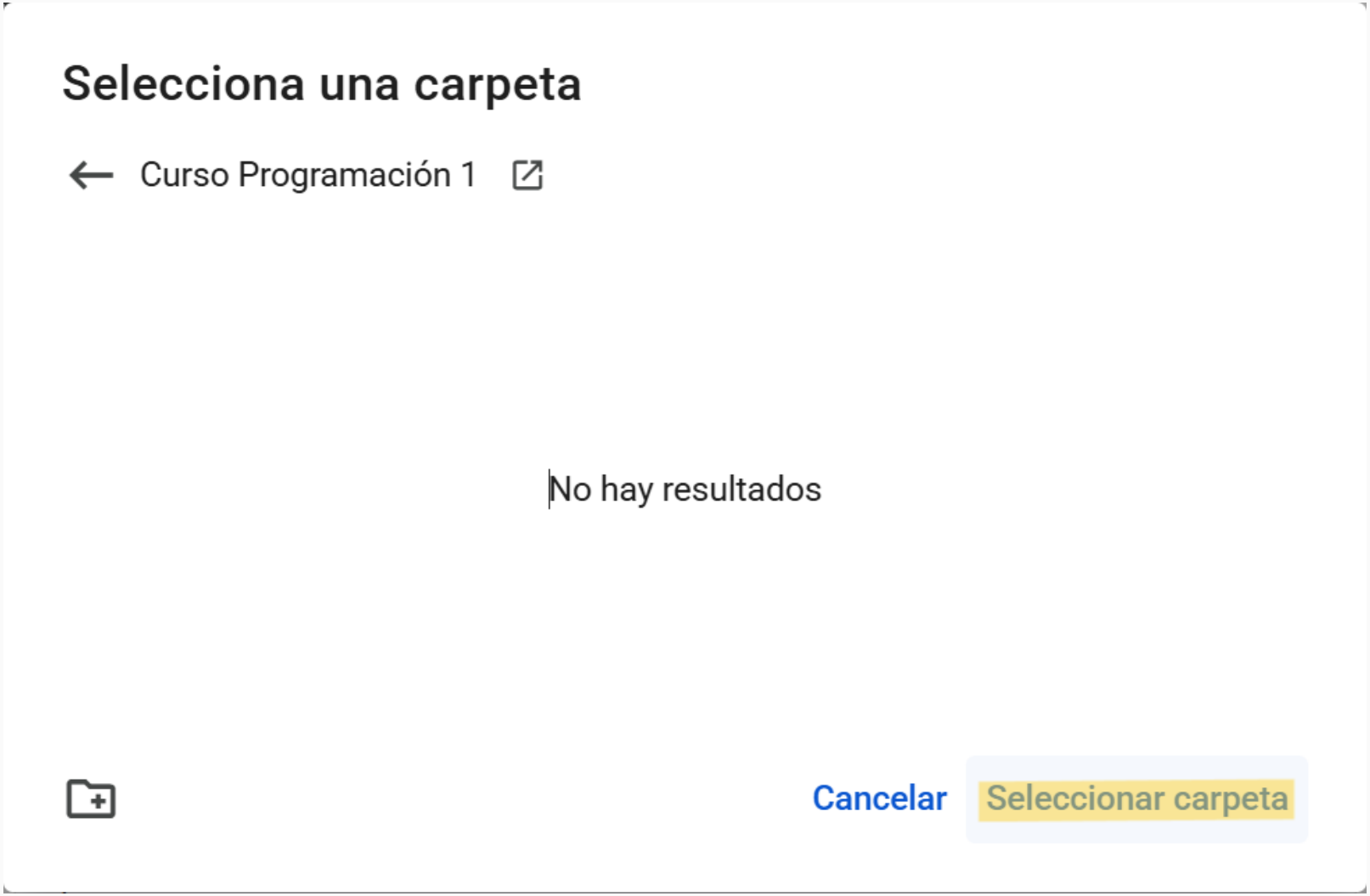
Paso 4: Empezar a crear carpetas



Paso 5: Crear carpeta con nombre del curso



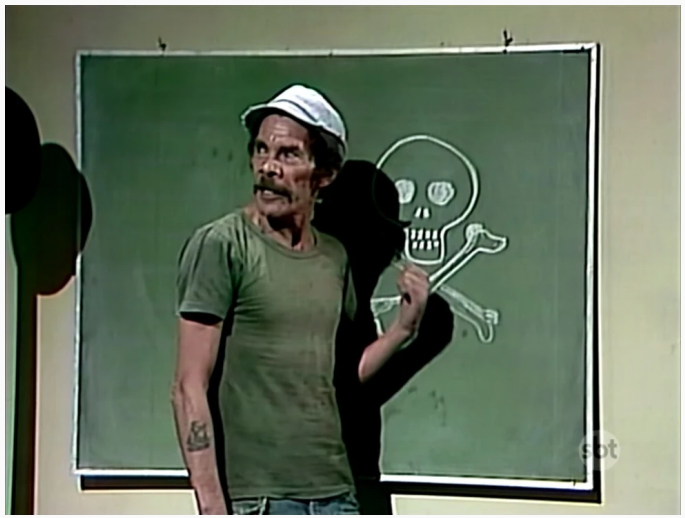
Paso 6: Selecionar la carpeta creada



Paso 7: Darle nombre al notebook

¡Cuidado con el abuso de la IA!

- La IA puede ayudarte a entender conceptos y resolver dudas.
- **PERO** el abuso puede generar problemas en tu aprendizaje.
- **Aprende haciendo**: programa tú mismo, comete errores, debuggea.
- La IA debe **complementar**, no **reemplazar** tu proceso de aprendizaje.



```
1  print("¡Hola, mundo de la Física y Astronomía!")
```

- Presiona **Shift+Enter** o haz clic en el “play” para ejecutar.
- Observa el resultado inmediatamente.

Fundamentos de Python

- **int** (enteros), **float** (reales), **str** (cadenas), **bool** (True/False).
- En Python, basta con asignar para crear una variable:

$$x = 10 \quad \text{(int)}$$

$$saludo = \text{"}Hola\text{"} \quad \text{(str)}$$

- **No se requiere declaración previa** de tipo. Otros lenguajes sí, por ejemplo C, o C++.

- Usar nombres descriptivos (`masa_objeto`, `velocidad_inicial`, etc.).
- No comenzar con dígitos (`2variable` es inválido, `variable2` es válido).
- **Distinción de mayúsculas/minúsculas**: `Radio` vs `radio`.
- Evitar palabras reservadas (`if`, `else`, `class`, etc.).

```python
a = 5
b = 2
suma          = a + b      # 7
resta         = a - b      # 3
producto      = a * b      # 10
division      = a / b      # 2.5 (float)
division_entera = a // b # 2 (int)
exponente     = a ** b     # 25 (5^2)
modulo        = a % b      # 1 (resto de la división)
```

- `/` produce un resultado float.
- `//` realiza *división entera*.

Acá si ejecutamos no veremos los resultados, ya que para mostrar estos resultados en la pantalla, debemos utilizar la función `print` de python.

- `input()` para capturar información del usuario (devuelve `str`).
- `print()` para mostrar resultados en pantalla.

```
1 nombre = input("¿Cuál es tu nombre?: ")
2 print("Hola", nombre, "bienvenido/a al curso!")
```

Esta forma de la función `print` es hoy día mayoritariamente utilizada con *f-string*, lo que sería:

```
1   nombre = input("¿Cuál es tu nombre?: ")
2   print(f'Hola {nombre} bienvenido/a al curso!')
```

Para el curso, trataremos de utilizar *f-string*, aunque otras alternativas no estan obsoletas, ni prohibídas.

Ejercicios Guiados

Ejercicios Guiados ∋ Ejercicio Guiado 1: Cálculo de Área de un Círculo

```python
import math

r_str = input("Ingresa el radio del círculo: ")
r = float(r_str)
area = math.pi * (r**2)
print(f'El área del círculo es: {area}')
```

- `import math` habilita funciones y constantes matemáticas (ej. `math.pi`).
- **Observación:** Si se introduce un valor no numérico, el programa fallará (manejo de errores).

Ejercicio

- Escribe un programa que pida la masa (kg) y la aceleración (m/s^2).
- Calcula la fuerza resultante ($F = m \times a$).
- Muestra el resultado en pantalla.

Tip: Recuerda convertir la cadena de `input()` a `float`.

Ejercicios Guiados ∋ Solución Propuesta ✓

```python
m_str = input("Ingresa la masa (kg): ")
a_str = input("Ingresa la aceleración (m/s^2): ")

m = float(m_str)
a = float(a_str)
F = m * a

print(f'La fuerza resultante es: {F} N')
```

- Añadir unidades en la salida (ej.: “N” para Newtons).
- Discutir manejo de errores, validación de datos, etc.

¡Encuentra y Corrige los Errores!

Copia este código en Colab y ejecútalo. **¡Tiene errores a propósito!**

```
# Programa para calcular el cuadrado de un número
x = input("Ingrese un número: ")
print("El número ingresado es:", x)
y = x + x
print("El doble del número es:", y)
z = x * x
print("El cuadrado del número es:" z)
```

Preguntas:

- ¿Qué errores encuentras?
- ¿Cómo los solucionarías?
- ¿El resultado es el esperado?

Errores encontrados:

1. **Error de sintaxis:** Falta comilla de cierre en `input()`
2. **Error de sintaxis:** Falta coma en el último `print()`
3. **Error lógico:** `input()` devuelve `str`, no `int`

Código corregido:

```python
# Programa para calcular el cuadrado de un número
x_str = input("Ingrese un número: ")
x = float(x_str)
print("El número ingresado es:", x)
y = x + x  # o también: y = 2 * x
print("El doble del número es:", y)
z = x * x  # o también: z = x ** 2
print("El cuadrado del número es:", z)
```

Lección: ¡Los errores son normales! Aprende a leer los mensajes de error.

Actividad Práctica

Actividad Práctica ∋ Problema 1: Conversión de Unidades de Distancia

Enunciado

- Pedir una distancia en metros al usuario.
- Convertir a kilómetros, centímetros y milímetros.
- Mostrar todos los resultados con sus unidades correspondientes.

Pistas:

- Recordar las conversiones: 1 m = 0.001 km, 1 m = 100 cm, 1 m = 1000 mm.
- Usar `f-strings` para mostrar resultados con formato claro.
- ¿Qué pasa si el usuario ingresa un número muy grande o muy pequeño?

Ejemplo esperado: Si ingresa 5 metros, debería mostrar: 5 m = 0.005 km = 500 cm = 5000 mm.

Actividad Práctica ∋ Problema 2: Calculadora de Velocidad

Enunciado

- Pedir distancia recorrida (en metros) y tiempo empleado (en segundos).
- Calcular la velocidad usando $v = \frac{d}{t}$.
- Mostrar el resultado en m/s y como **bonus** convertir a km/h.

Pistas:

- Para convertir de m/s a km/h, multiplicar por 3.6.
- Considerar qué pasa si el tiempo es cero (¡división por cero!).
- Incluir unidades en la salida para claridad.

Física relevante: Esta es una de las ecuaciones cinemáticas más fundamentales.

Actividad Práctica ∋ Problema 3: Calculadora de Índice de Masa Corporal

Enunciado

- Pedir peso (en kg) y altura (en metros) al usuario.
- Calcular el IMC usando la fórmula: $IMC = \frac{peso}{altura^2}$.
- Mostrar el resultado con interpretación básica.

Pistas:

- Usar el operador `**` para elevar al cuadrado.
- IMC < 18.5: bajo peso, 18.5-24.9: normal, 25-29.9: sobrepeso, ≥30: obesidad.
- ¿Cómo mostrar el resultado con 2 decimales?

Aplicación: Aunque no es estrictamente física, involucra cálculos matemáticos relevantes para ciencias de la salud.

Actividad Práctica ∋ Actividad Práctica

Indicaciones

- Trabajen en equipos de 2-3 personas.
- Cada equipo debe intentar resolver al menos los Problemas 1 y 2.
- El Problema 3 es opcional (para equipos que terminen rápido).
- Prueben sus programas con diferentes valores de entrada.
- Anoten cualquier error o comportamiento inesperado que encuentren.

Tiempo sugerido: 20-25 minutos para programar + 10 minutos para discusión grupal.

Importante

¡No se preocupen por hacer el código ”perfecto”! El objetivo es practicar lo aprendido y familiarizarse con la programación.

Actividad Práctica ∋ Retroalimentación

- ¿Problemas encontrados?
- ¿Qué fue lo más intuitivo / confuso?
- ¿Dudas para la próxima clase?

Mantenga estos datos en su cuaderno, le servirá para estudiar.

Curiosidades y Recursos

- Creado por Guido van Rossum a finales de los 80.
- Nombre inspirado en el grupo de comedia *Monty Python*.
- Filosofía: legibilidad, sencillez y productividad.

- **Para practicar Python:** HackerRank - Python Practice
- **python.org** - Documentación oficial.
- **Google Colab** - Entorno en la nube.
- **Real Python** (sitio con tutoriales y guías).
- **Stack Overflow** (para consultas y dudas).

- **Stack Overflow**: millones de preguntas y respuestas.
- **Reddit /r/learnpython**: foros de principiantes.
- **GitHub**: proyectos y ejemplos de ciencia y Python.

Tip: Buscar soluciones o inspiración es parte del desarrollo como programador.

Conclusiones

Conclusiones ∋ Resumen de la Sesión

- Configuramos Google Colab y ejecutamos nuestro primer código en Python.
- Conocimos los tipos de datos y operaciones aritméticas básicas.
- Hicimos ejercicios de entrada/salida y ejemplos físicos sencillos.
- Exploramos recursos para continuar aprendiendo.

Conclusiones ∋ Próximos Pasos

- **Unidad II:** Estructuras de control (`if`, `for`, `while`) y funciones en Python.
- **Tarea Sugerida**:
  - Practicar más ejercicios con `input()` y conversión de tipos.
  - Explorar `math`, `random`, etc.

¡Gracias y hasta la próxima sesión!

- Revisen la plataforma (Colab) para ejercicios adicionales.
- Traigan sus dudas la próxima clase.