# Programación para Física y Astronomía

## Departamento de Física.

Corodinadora: C Loyola
Profesoras/es C Loyola / C Femenías / Y Navarrete / C Ruiz / F Bugini

Primer Semestre 2025

Universidad Andrés Bello
Departamento de Física y Astronomía

Universidad Andrés Bello®

python powered

# Resumen - Semana 4, Sesión 2 (Sesión 8)

Introducción y Repaso

Módulos y Paquetes (Repaso)

Librerías Externas

Tarea Semanal

Conclusiones y Próximos Pasos

# Introducción y Repaso

# Recapitulación de la Sesión Anterior (Sesión 7)

- **Semana 4, Sesión 1 (Sesión 7)** se centró en:
  - **Funciones**: sintaxis (`def`), parámetros, valores por defecto, alcance de variables.
  - **Módulos y Paquetes**: cómo organizar el código en archivos `.py` y carpetas.
  - Ejemplos de proyectos pequeños con `import` y definición de funciones útiles.
- **Objetivo de hoy**: Ampliar la práctica con funciones y módulos, e **introducir el uso de librerías externas** (vía `pip` o Colab).

# Objetivos de la Sesión 8

- **Profundizar** en el flujo de trabajo al crear y reutilizar módulos en Python.
- **Explorar** la instalación de librerías externas (`pip`, Google Colab).
- **Diseñar** una actividad grupal donde se combine la creación de funciones propias con el uso de librerías de terceros.
- **Fomentar** la colaboración y la discusión sobre buenas prácticas de organización.

# Módulos y Paquetes (Repaso)

# Estructura Básica de un Proyecto

- **Carpetas** y `.py` para agrupar funcionalidades.
- Ejemplo:
- `main.py` orquesta la lógica usando `import mis_modulos.fisica` y así sucesivamente.
- Facilita la mantenibilidad y escalabilidad.

# Formas de Importar

- Import completo:

```
1 import mis_modulos.matematicas
2 res = mis_modulos.matematicas.sumar(2, 3)
```

- From / Import:

```
1 from mis_modulos.matematicas import sumar
2 res = sumar(2, 3)
```

- Import renombrado:

```
1 import mis_modulos.matematicas as mm
2 res = mm.sumar(2, 3)
```

# Librerías Externas

# ¿Por qué Librerías Externas?

- **Ahorra tiempo**: aprovechas código ya probado por la comunidad.
- **Funcionalidades avanzadas**: Desde manejo de redes hasta machine learning.
- **Ejemplos**: `requests` para peticiones web, `numpy` para cálculo numérico, `pandas` para data frames, etc.
- **Comunidad activa**: librerías mantenidas, actualizaciones frecuentes.

# Instalación con `pip`

- **pip**: el gestor de paquetes oficial de Python.
- Comando general en terminal:

  `pip install nombre_paquete`
- Si usas **Google Colab**, puedes instalar temporalmente en una celda:

```
1  !pip install nombre_paquete
```

- La librería quedará disponible para importarse en el resto del entorno (hasta reiniciar).

# Ejemplo: `requests` en Colab

```
# En una celda de Colab:
!pip install requests

import requests

resp = requests.get("https://api.github.com")
print(resp.status_code)
print(resp.json())
```

- **Uso real**: Conectarse a APIs, descargar datos, etc.
- **Sugerencia**: Manejar casos de error (`resp.status_code != 200`).

# Ejemplo: numpy Básico

```
# Desde terminal local
pip install numpy
```

```
# Uso en el código
import numpy as np

arr = np.array([1, 2, 3, 4])
print(arr * 2)  # [2 4 6 8]
```

- **NumPy** es la base de muchas librerías científicas en Python.
- **Operaciones vectorizadas**: eficiencia y simplicidad.

# Tarea Semanal

# Actividad General

## Objetivo

- Desarrollar un **módulo propio** con funciones matemáticas o de análisis.
- Integrar **una librería externa** (p.e. **numpy**) para realizar operaciones.
- Probar el resultado en un **notebook** o script principal.

# Lineamientos de la Actividad

- Crear en Colab o localmente la siguiente estructura:
  `taller4/`
  `  mi_modulo.py`
  `  main.ipynb` (o main.py)
- **mi_modulo.py**:
  - Define algunas funciones (e.g., `multiplicar_vector(vec, escalar)`) que internamente use **numpy**.
  - Podrías incluir una función que calcule la **media** y **desviación estándar** de un arreglo con **numpy**.
- **main.ipynb**:
  - Instala (si es necesario) la librería **numpy**.
  - Importa `mi_modulo` y ejecuta las funciones, imprimiendo resultados.

# Conclusiones y Próximos Pasos

# Análisis General de la Actividad

- Ventajas de **combinar módulos propios con librerías externas**:
  - Reutilización de código (módulos).
  - Potencia y robustez (librerías de terceros).
- Importancia de la **organización de archivos** y de un **flujo de trabajo** claro.
- Manejar **entornos virtuales** (en local) es otra buena práctica (tema futuro).

# Recomendaciones de Estudio

- Revisar la **documentación oficial** de `pip` y `virtualenv`.
- Explorar **PyPI** (`https://pypi.org/`) para descubrir librerías útiles.
- Practicar la creación de **módulos** en proyectos pequeños.
- Investigar qué librerías podrían ser útiles para futuros trabajos de Física/Astronomía (p.e. `astropy`, `scipy`).

# Próxima Sesión

- **Sesion 9 (Semana 5)**: Repaso integral de Unidades I y II, y **Solemne I**.
- **Recomendación**:
  - Revisa todos los conceptos vistos: Sintaxis, Estructuras de Control, Funciones, Módulos.
  - Práctica con ejercicios y ejemplos de exámenes pasados (si los hubiera).

**¡Prepárate para la evaluación!**

# Recursos Adicionales

- Official Python Packaging Tutorial
- Documentación de la Biblioteca Estándar de Python
- **PyPI** - Python Package Index
- Numpy Docs
- Matplotlib Docs

# ¡Muchas gracias y éxito en su práctica!

- Recuerden subir su trabajo a Google Drive o repositorio compartido.
- Próxima sesión: **Solemne I** y repaso integral.
- ¡Sigan explorando librerías externas y creando módulos propios!