

# Herramientas Computacionales para Economistas: Python, Datos e IA

Universidad Nacional de La Plata

Facultad de Ciencias Económicas

Maestría en Economía

**Profesora:** Belén Michel Torino

**Ayudantes:** Pilar Chaumeil y Emiliano Bohorquez

**Modalidad:** virtual, con contenidos asincrónicos y encuentros sincrónicos por Zoom

**Primer encuentro:** miércoles 23 de septiembre de 2026

**Horario de clase:** miércoles de 18:30 a 20:30

**Zoom:** <https://zoom.us/j/97661175956>

**Consultas:** el aula virtual permanecerá abierta de 18:00 a 18:30 y de 20:30 a 21:00 para consultas con el equipo docente, idealmente coordinando horarios previamente.

**Medio de comunicación principal:** [canal de Slack](#)

## Objetivo del curso

El curso tiene como objetivo introducir a los estudiantes en el uso de Python y otras herramientas computacionales aplicadas al trabajo de los economistas, con especial énfasis en la obtención, procesamiento, análisis y visualización de datos.

Python se presenta como una herramienta complementaria a otras ya utilizadas en economía, como Stata, que permite automatizar tareas, trabajar con distintas fuentes y formatos de información, acceder a APIs y bases de datos, realizar web scraping, procesar texto y datos geográficos y construir análisis reproducibles.

El curso incorpora también herramientas de inteligencia artificial generativa como apoyo para programar y resolver problemas. El objetivo no es sólo aprender a producir código, sino desarrollar la capacidad de comprenderlo, modificarlo, detectar errores, validar resultados y decidir críticamente cuándo y cómo utilizar asistencia de IA.

No se requiere experiencia previa en Python.

# Resultados de aprendizaje

Al finalizar el curso se espera que los estudiantes puedan:

- traducir un problema de investigación o análisis económico en una secuencia de tareas computacionales;
- leer, escribir, interpretar y modificar código básico en Python;
- importar, explorar, limpiar, transformar, combinar y visualizar datos;
- trabajar con datos provenientes de distintas fuentes y formatos;
- utilizar herramientas para procesamiento de texto y datos georreferenciados;
- obtener información mediante APIs, bases de datos y técnicas de scraping;
- identificar y diagnosticar errores en código y resultados;
- implementar controles básicos para verificar la calidad de los datos y la validez de las transformaciones realizadas;
- utilizar herramientas de IA como apoyo para programar, documentar, depurar y explorar soluciones;
- desarrollar flujos de trabajo claros y reproducibles utilizando Git y GitHub;
- explicar y defender las decisiones tomadas en un análisis computacional.

## Modalidad de cursada

El curso está organizado en seis módulos, cada uno compuesto por contenidos asincrónicos y un encuentro sincrónico semanal.

Las **clases asincrónicas** incluirán videos, notebooks y otros recursos que deberán revisarse antes de la clase por Zoom correspondiente. Se espera que los estudiantes acompañen los videos ejecutando y modificando el código presentado. La práctica activa es una parte central del aprendizaje de programación.

Las **clases sincrónicas** (por Zoom) estarán destinadas principalmente a resolver dudas, trabajar sobre ejercicios y problemas aplicados, programar en grupos y discutir distintas estrategias de resolución.

El principal canal de comunicación fuera de clase será Slack, donde podrán realizar consultas, compartir recursos y participar de las discusiones del curso.

## Uso de inteligencia artificial

Las herramientas de inteligencia artificial forman parte de los recursos disponibles durante el curso. Su uso podrá variar según el objetivo de cada actividad: en algunos ejercicios será importante realizar primero un intento propio; en otros se podrá utilizar IA libremente para escribir código, buscar alternativas, interpretar errores o mejorar una solución.

La regla general del curso es que **pueden utilizar IA cuando la actividad lo permita, pero cada estudiante debe poder comprender, explicar, modificar y defender el código y los resultados que entrega.**

El uso de IA no reemplaza la validación. Que un código ejecute correctamente no implica que resuelva correctamente el problema planteado. Se trabajará especialmente sobre formas de verificar resultados, detectar errores y contrastar las respuestas de estas herramientas con los datos y la documentación disponible.

## Contenidos

| Módulo                                                  | Contenidos principales                                                                                                                                                                                                      |
|---------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Clase 0:<br/>Puesta a punto</b>                      | Introducción al curso y al entorno de trabajo. Python, notebooks, Google Colab y/o VS Code. Slack. Git y GitHub. Introducción al uso de herramientas de IA para programación.                                               |
| <b>Clase 1:<br/>Fundamentos de Python</b>               | Pensamiento computacional. Variables y tipos de datos. Operadores. Strings. Listas, tuplas, sets y diccionarios. Condicionales. Ciclos <code>for</code> y <code>while</code> . Lectura e interpretación de errores.         |
| <b>Clase 2:<br/>Funciones, debugging y validación</b>   | Funciones, argumentos y valores de retorno. Métodos y módulos. Entrada y salida de archivos. Debugging. Manejo básico de errores. Controles y validaciones simples.                                                         |
| <b>Clase 3:<br/>Análisis de datos con pandas</b>        | NumPy esencial. DataFrames. Selección y filtrado. Valores faltantes. Fechas. <code>groupby</code> , agregaciones, <code>merge</code> y <code>join</code> . Reformateo de datos. Visualizaciones.                            |
| <b>Clase 4:<br/>Limpieza, texto y datos imperfectos</b> | Limpieza y estandarización de información. Manejo de strings. Expresiones regulares. Reformateo. Matching y estrategias para trabajar con datos imperfectos.                                                                |
| <b>Clase 5:<br/>Datos georreferenciados</b>             | Introducción a información geográfica. GeoPandas. Shapefiles y GeoJSON. Sistemas de coordenadas. Joins espaciales y elaboración de mapas.                                                                                   |
| <b>Clase 6:<br/>Obtención de datos</b>                  | Requests y HTTP. APIs y JSON. Web scraping con BeautifulSoup. Extracción de información de documentos y tablas. Introducción a bases de datos relacionales y SQL. Buenas prácticas para la obtención automatizada de datos. |

# Evaluación

La evaluación combinará tres componentes:

**Trabajos prácticos semanales (12% cada uno):** A lo largo del curso se propondrán ejercicios aplicados asociados con los contenidos de cada módulo. Algunos serán individuales y otros grupales.

**Proyecto integrador final (25%):** Al finalizar el curso, los estudiantes desarrollarán en grupo un proyecto que integre distintas herramientas vistas durante la cursada. El proyecto deberá incluir un problema o pregunta claramente definida, trabajo con datos, código reproducible, validación de resultados y una presentación clara de las decisiones adoptadas.

Como parte de la evaluación del proyecto habrá una **breve instancia individual** en la que cada integrante deberá poder explicar y discutir el código y las decisiones del trabajo realizado.

**Asistencia y participación (15%):** Se tendrá en cuenta la asistencia a los encuentros sincrónicos y la participación activa y relevante tanto durante las clases como en Slack.

Las consignas, fechas de entrega y criterios específicos de evaluación de cada actividad se comunicarán durante la cursada.

## Recursos y bibliografía de consulta

No se espera que los estudiantes estudien estos materiales de manera exhaustiva. Se ofrecen como recursos de referencia para profundizar los contenidos del curso.

- Downey, Allen. [\*Think Python: How to Think Like a Computer Scientist\*](#).
- McKinney, Wes. [\*Python for Data Analysis\*](#).
- Documentación oficial de Python, pandas, GeoPandas y las restantes librerías utilizadas y [\*Tutorial de Python\*](#) por el autor original de Python, Guido van Rossum, traducido al español.