



InverseNeural Lab

Guía de Inicio

👋 Bienvenido

Esta guía te llevará paso a paso desde tener una computadora limpia hasta estar listo para empezar con trading algorítmico. **No necesitas conocimientos previos de programación.**



Tabla de Contenidos

01

Instalar Visual Studio Code

Editor de código gratuito y profesional

02

Verificar Python

Lenguaje de programación para trading

03

Crear tu Proyecto

Estructura base del proyecto

04

Configurar el Entorno Virtual

Ambiente aislado para dependencias

05

Crear la Estructura de Carpetas

Organización profesional del código

06

Instalar Dependencias

Librerías necesarias para el proyecto

07

Verificar Instalación

Prueba que todo funcione correctamente

08

Próximos Pasos

Continuar con el curso de trading

1. Instalar Visual Studio Code

¿Qué es Visual Studio Code?

VS Code es un **editor de código** gratuito creado por Microsoft. Es donde escribirás y ejecutarás tus programas de trading. Piénsalo como Microsoft Word, pero para código.

Descargar e Instalar

Para Mac

1. Ve a: <https://code.visualstudio.com/>
2. Haz clic en "Download for Mac"
3. Abre el archivo .zip descargado
4. Arrastra **Visual Studio Code.app** a tu carpeta **Aplicaciones**
5. Abre VS Code desde Aplicaciones

Para Windows

1. Ve a: <https://code.visualstudio.com/>
2. Haz clic en "Download for Windows"
3. Ejecuta el instalador descargado (.exe)
4. Sigue las instrucciones (deja las opciones predeterminadas)
5. Abre VS Code desde el Menú Inicio

Para Linux (Ubuntu/Debian)

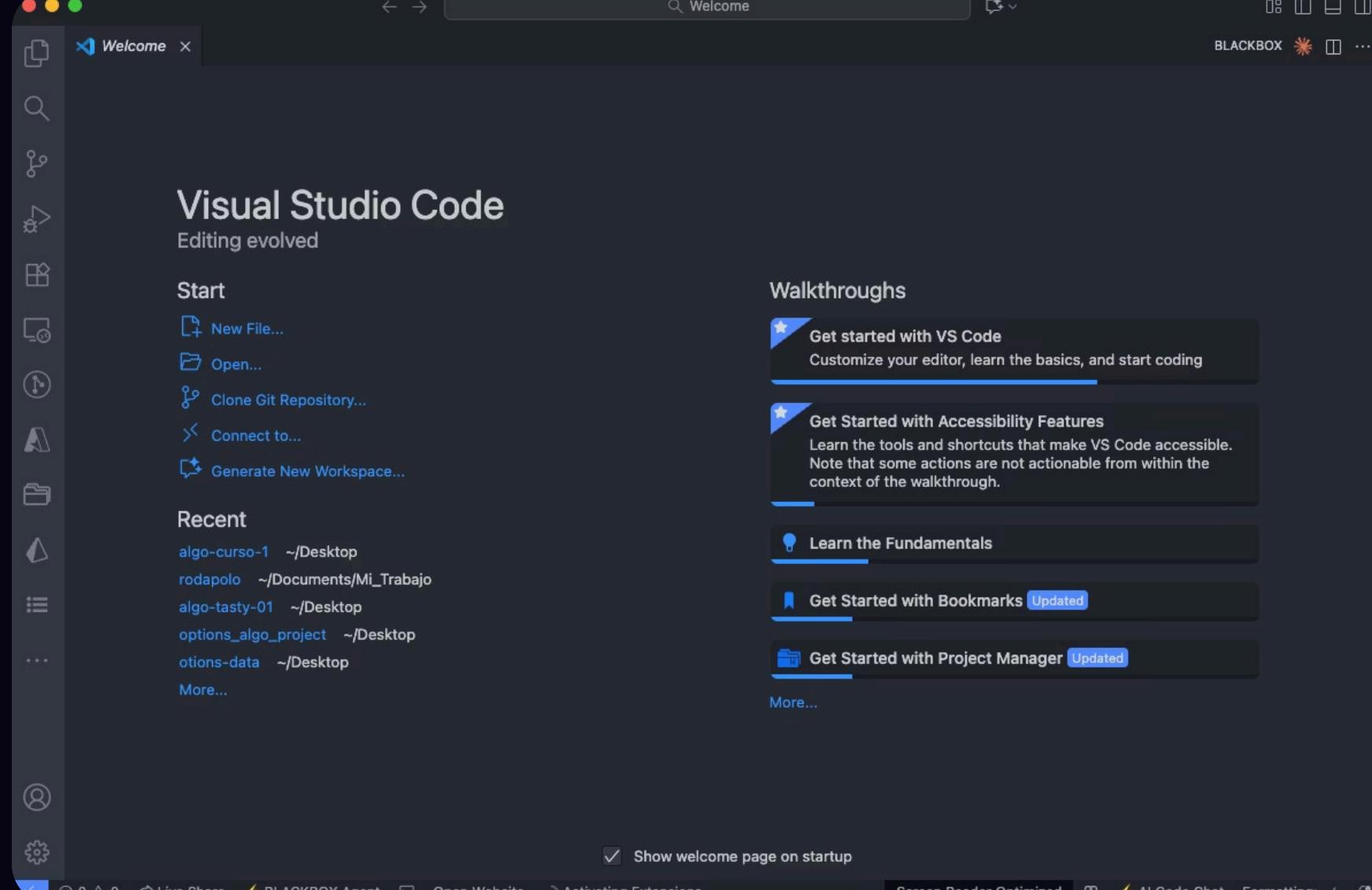
1. Ve a: <https://code.visualstudio.com/>
2. Descarga el paquete .deb
3. Abre Terminal y ejecuta:

```
cd ~/Downloads  
sudo dpkg -i code_*.deb  
sudo apt-get install -f
```

4. Abre VS Code desde el menú de aplicaciones

✓ Verificación

Deberías ver una ventana como esta:



2. Verificar Python

¿Qué es Python?

Python es el **lenguaje de programación** que usaremos. Es como el idioma en el que le hablarás a la computadora para hacer trading algorítmico.

Verificar si ya tienes Python

En Mac/Linux

1. Abre VS Code
2. Abre la **Terminal** dentro de VS Code:
 - o Menú: Terminal → New Terminal
 - o O usa el atajo: Ctrl + ñ (Mac: Cmd + ñ)
3. Verás una ventana en la parte inferior con algo como:

```
user@computer ~ %
```

4. Escribe este comando y presiona **Enter**:

```
python3 --version
```

En Windows

1. Abre VS Code
2. Abre la **Terminal** dentro de VS Code:
 - o Menú: Terminal → New Terminal
3. Escribe este comando y presiona **Enter**:

```
python --version
```

Interpretar el Resultado

✓ Si ves algo como:

```
Python 3.9.7
```

```
o
```

```
Python 3.10.5
```

iPerfecto! Ya tienes Python instalado (versión 3.9 o superior es ideal).

✗ Si ves:

```
command not found: python
```

```
o
```

```
'python' is not recognized
```

Necesitas instalar Python.

Instalar Python (si no lo tienes)

Para Mac

Opción 1: Usando Homebrew (recomendado)

1. Primero instala Homebrew. En la Terminal, pega:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

1. Presiona **Enter** y sigue las instrucciones
2. Luego instala Python:

```
brew install python@3.11
```

Opción 2: Descarga directa

1. Ve a:
<https://www.python.org/downloads/>
2. Descarga Python 3.11 para macOS

3. Ejecuta el instalador .pkg

4. Sigue las instrucciones

Para Windows

1. Ve a:
<https://www.python.org/downloads/>
2. Descarga Python 3.11 para Windows
3. **IMPORTANTE:** En el instalador, marca la casilla "**Add Python to PATH**"
4. Haz clic en "**Install Now**"
5. Espera a que termine
6. Cierra y vuelve a abrir VS Code

Para Linux (Ubuntu/Debian)

```
sudo apt update  
sudo apt install  
python3.11 python3-pip  
python3-venv
```

✓ Verificación Final

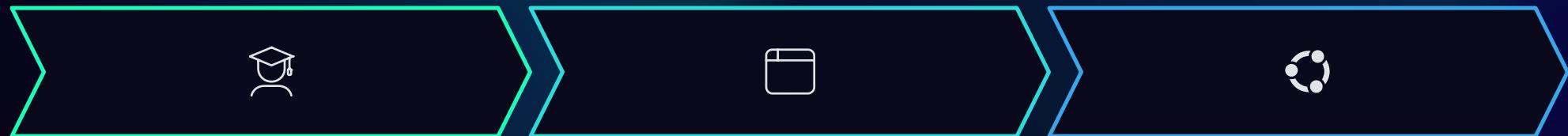
Cierra y vuelve a abrir VS Code, abre la Terminal y ejecuta:

```
python3 --version
```

Deberías ver: Python 3.11.x o similar.

3. Crear tu Proyecto

Paso 1: Crear la Carpeta del Proyecto



En Mac

1. Abre **Finder**
2. Ve a tu **Escritorio**
3. Haz clic derecho → **Nueva Carpeta**
4. Nómbrala: algo-options

En Windows

1. Abre **Explorador de Archivos**
2. Ve a tu **Escritorio**
3. Clic derecho → **Nuevo** → **Carpeta**
4. Nómbrala: algo-options

En Linux

```
cd ~/Desktop  
mkdir algo-options
```

Paso 2: Abrir la Carpeta en VS Code

Método 1: Arrastrar y Soltar

1. Abre VS Code
2. Arrastra la carpeta algo-options desde tu Escritorio
3. Suéltala en la ventana de VS Code

Método 2: Desde el Menú

1. En VS Code: File → Open Folder...
2. Navega a tu Escritorio
3. Selecciona la carpeta algo-options
4. Haz clic en "**Open**" o "**Abrir**"

✓ Verificación

En el lado izquierdo de VS Code deberías ver: algo-options

4. Configurar el Entorno Virtual

¿Qué es un Entorno Virtual?

Un **entorno virtual** es como una caja aislada donde instalamos todas las herramientas (librerías) que necesitamos para nuestro proyecto. Esto evita conflictos con otros proyectos.

Piénsalo así:

- **Sin entorno virtual:** Todas las herramientas se mezclan en un solo cajón (desorden)
- **Con entorno virtual:** Cada proyecto tiene su propia caja de herramientas (organizado)

Crear el Entorno Virtual

1. Abre la **Terminal** en VS Code (Terminal → New Terminal)
2. Asegúrate de estar en la carpeta correcta. Deberías ver algo como:

```
~/Desktop/algo-options %
```

3. Ejecuta este comando:

Para Mac/Linux:

```
python3 -m venv venv
```

Para Windows:

```
python -m venv venv
```

4. Espera 10-30 segundos. Esto crea una carpeta llamada `venv` con todas las herramientas base.

Activar el Entorno Virtual

Para Mac/Linux:

```
source venv/bin/activate
```

Para Windows (PowerShell):

```
venv\Scripts\Activate.ps1
```

Para Windows (CMD):

```
venv\Scripts\activate.bat
```

✓ Verificación

Tu Terminal debería cambiar y mostrar `(venv)` al inicio:

```
(venv) ~/Desktop/algo-options %
```

 **Importante:** Cada vez que cierres VS Code, tendrás que **activar nuevamente** el entorno virtual con el comando `source venv/bin/activate` (Mac/Linux) o `venv\Scripts\Activate.ps1` (Windows).

💡 Tip: Activación Automática

En VS Code, puedes configurar para que se active automáticamente:

1. Cmd + Shift + P (Mac) o Ctrl + Shift + P (Windows)
2. Escribe: Python: Select Interpreter
3. Selecciona el que dice `venv` o `./venv/bin/python`

5. Crear la Estructura de Carpetas

¿Por qué necesitamos estructura?

Organizar el código en carpetas es como organizar documentos en un archivador. Cada cosa tiene su lugar y es más fácil encontrar lo que necesitas.

Estructura del Proyecto

Vamos a crear esta estructura:

```
algo-options/
    ├── venv/          # Entorno virtual (ya lo creamos)
    ├── data/          # Datos históricos y análisis
    |   ├── historical/ # Datos crudos de opciones
    |   └── analysis/  # Resultados de análisis
    ├── scripts/       # Todos nuestros programas
    |   ├── data_pipeline/ # Scripts para obtener datos
    |   ├── strategies/  # Estrategias de trading
    |   ├── quantitative/ # Análisis cuantitativo
    |   ├── backtest/     # Backtesting y análisis
    |   ├── dashboard/    # Dashboard interactivo
    |   |   └── components/ # Componentes del dashboard
    |   └── utils/        # Utilidades generales
    ├── logs/          # Registros de ejecución
    ├── documents/     # Documentación
    └── README.md      # Descripción del proyecto
```

Crear la Estructura Automáticamente

En Mac/Linux

Copia y pega este comando completo en la Terminal (con venv activado):

```
mkdir -p data/historical data/analysis
scripts/data_pipeline scripts/strategies
scripts/quantitative scripts/backtest
scripts/dashboard/components scripts/utils
logs documents
```

En Windows (PowerShell)

```
mkdir data, data\historical, data\analysis,
scripts, scripts\data_pipeline,
scripts\strategies, scripts\quantitative,
scripts\backtest, scripts\dashboard,
scripts\dashboard\components, scripts\utils,
logs, documents
```

Crear Archivos Iniciales

Ahora vamos a crear algunos archivos básicos:

- 1 **README.md**
(descripción del proyecto)

En la Terminal:

```
touch README.md
```

- 2 **.gitignore (archivos a ignorar en Git)**

```
touch .gitignore
```

- 3 **requirements.txt**
(dependencias del proyecto)

```
touch
requirements.txt
```

✓ Verificación

En el explorador de VS Code (lado izquierdo) deberías ver: Las carpetas creadas

6. Instalar Dependencias

¿Qué son las Dependencias?

Las **dependencias** son librerías (paquetes de código) creadas por otros programadores que nos facilitan la vida.

Por ejemplo:



pandas

Para trabajar con datos (como Excel en código)



numpy

Para cálculos matemáticos



matplotlib

Para crear gráficos



streamlit

Para crear dashboards interactivos

Paso 1: Actualizar pip

pip es el instalador de paquetes de Python. Primero lo actualizamos:

Mac/Linux:

```
pip install --upgrade pip
```

Windows:

```
python -m pip install --upgrade pip
```

Paso 2: Instalar Paquetes Básicos

Copia y pega este comando (puede tardar 2-5 minutos):

```
pip install pandas numpy scipy matplotlib seaborn yfinance requests python-dotenv
```

Explicación de Cada Paquete

- **pandas**: Manejo de datos tabulares (como Excel)
- **numpy**: Cálculos matemáticos y arrays
- **scipy**: Funciones científicas (estadística, probabilidad)
- **matplotlib**: Gráficos básicos
- **seaborn**: Gráficos bonitos y profesionales
- **yfinance**: Obtener datos de Yahoo Finance
- **requests**: Hacer llamadas a APIs (web)
- **python-dotenv**: Manejar variables de entorno (seguridad)

Paso 3: Instalar Paquetes para Dashboard

```
pip install streamlit plotly
```

- **streamlit**: Crear dashboards web interactivos sin HTML/CSS
- **plotly**: Gráficos interactivos (zoom, hover, etc.)

Paso 4: Guardar las Dependencias

Este comando guarda la lista de todo lo instalado:

```
pip freeze > requirements.txt
```

✓ Verificación

Ejecuta:

```
pip list
```

Deberías ver una lista larga con todos los paquetes, incluyendo:

```
pandas 2.x.x  
numpy 1.x.x  
scipy 1.x.x  
matplotlib 3.x.x  
seaborn 0.x.x  
streamlit 1.x.x  
plotly 5.x.x
```

...

7. Verificar Instalación

Prueba Python

Vamos a crear un programa simple para verificar que todo funciona.

1. En VS Code, haz clic en File → New File
2. Pega este código:

```
# test_installation.py
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

print("
```

8. Próximos Pasos

✓ Checklist de Completitud

Antes de continuar, asegúrate de tener:

- VS Code instalado y funcionando
- Python 3.9+ instalado
- Proyecto algo-options creado en el Escritorio
- Entorno virtual venv creado y activado
- Estructura de carpetas completa
- Dependencias instaladas (pandas, numpy, etc.)
- Test de instalación ejecutado exitosamente

🎓 Continuar con el Curso

Si todo está ✓, estás listo para:



Opción 1: Empezar con Datos

- Ir a: documents/DATA_PIPELINE.md
- Aprender a obtener datos históricos de opciones
- Conectar con Polygon.io API



Opción 2: Empezar con Estrategias

- Ir a: documents/strategies_README.md
- Aprender sobre Covered Calls e Iron Condors
- Entender lógica de estrategias



Opción 3: Empezar con Análisis Cuantitativo

- Ir a: documents/quantitative_README.md
- Aprender Black-Scholes, Greeks, Probabilidades
- Fundamentos matemáticos



Opción 4: Ver el Dashboard

Si solo quieres ver el dashboard interactivo:

```
cd scripts/dashboard  
streamlit run app.py
```

Se abrirá automáticamente en tu navegador:
<http://localhost:8501>

Recursos Adicionales

Atajos de Teclado Útiles en VS Code

General:

- Cmd + Shift + P / Ctrl + Shift + P: Paleta de comandos
- Cmd + B / Ctrl + B: Mostrar/ocultar barra lateral
- Cmd + Shift + E / Ctrl + Shift + E: Explorador de archivos
- Cmd + Ñ / Ctrl + Ñ: Abrir/cerrar Terminal

Edición:

- Cmd + S / Ctrl + S: Guardar archivo
- Cmd + Z / Ctrl + Z: Deshacer
- Cmd + Shift + Z / Ctrl + Shift + Z: Rehacer
- Cmd + C / Ctrl + C: Copiar
- Cmd + V / Ctrl + V: Pegar

Terminal:

- Ctrl + C: Detener programa en ejecución
- clear: Limpiar terminal
- pwd: Mostrar carpeta actual
- ls (Mac/Linux) / dir (Windows): Listar archivos

Comandos de Terminal Útiles

Navegación:

```
cd nombre_carpeta # Entrar a una carpeta  
cd ..      # Subir un nivel  
pwd       # Ver ruta actual  
ls        # Listar archivos (Mac/Linux)  
dir       # Listar archivos (Windows)
```

Archivos:

```
touch archivo.py    # Crear archivo (Mac/Linux)  
echo. > archivo.py # Crear archivo (Windows)  
rm archivo.py     # Eliminar archivo (Mac/Linux)  
del archivo.py    # Eliminar archivo (Windows)
```

Git (veremos más adelante):

```
git status          # Ver estado del repositorio  
git add .          # Agregar todos los cambios  
git commit -m "mensaje" # Guardar cambios  
git push           # Subir a GitHub
```

Extensiones Recomendadas para VS Code

1 Python (Microsoft)

- Sintaxis highlighting, debugging, IntelliSense
- Instalar: Cmd/Ctrl + Shift + X → buscar "Python"

2 Jupyter (Microsoft)

- Para notebooks interactivos
- Instalar: buscar "Jupyter" en extensiones

3 GitLens (GitKraken)

- Visualizar historial de Git
- Instalar: buscar "GitLens"

4 Path Intellisense

- Autocompletar rutas de archivos
- Instalar: buscar "Path Intellisense"

Solución de Problemas Comunes

Problema 1: "python: command not found"

Solución:

- Mac/Linux: Usa python3 en lugar de python
- Windows: Reinstala Python y marca "Add to PATH"

Problema 2: "Permission denied" al crear entorno virtual

Solución (Mac/Linux):

```
sudo chown -R $USER:$USER ~/Desktop/algo-options
```

Problema 3: PowerShell no ejecuta scripts (Windows)

Solución:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```

Problema 4: Entorno virtual no se activa

Solución:

1. Elimina la carpeta venv
2. Vuelve a crear: python -m venv venv
3. Activa de nuevo

Problema 5: "Module not found" al ejecutar programa

Solución:

1. Verifica que (venv) esté visible en la Terminal
2. Si no: source venv/bin/activate (Mac/Linux) o venv\Scripts\Activate.ps1 (Windows)
3. Reinstala: pip install nombre_paquete



¿Necesitas Ayuda?

Antes de preguntar, intenta:

1. Leer el mensaje de error completo
2. Buscar el error en Google
3. Revisar esta guía nuevamente
4. Cerrar y volver a abrir VS Code

Si aún no funciona:

1. Toma una captura de pantalla del error
2. Copia el comando que ejecutaste
3. Copia el mensaje de error completo
4. Publica en el foro de la comunidad con toda esta información

Formato de Pregunta Ideal:

Descripción del problema
Estoy intentando [acción] pero recibo [error]

Qué he intentado

1. [Paso 1]
2. [Paso 2]
3. [Paso 3]

Mi entorno

- OS: macOS 13.5 / Windows 11 / Ubuntu 22.04
- Python: 3.11.4
- VS Code: 1.80.0

Captura de pantalla

[Imagen del error]

Código/Comando ejecutado

```
```bash
comando que ejecuté
````
```

Error completo

```

mensaje de error completo

```

¡Felicitaciones!

Si llegaste hasta aquí y todo funciona, ¡estás oficialmente listo para empezar con trading algorítmico!

Lo que has logrado:

→ Configurar un entorno de desarrollo profesional

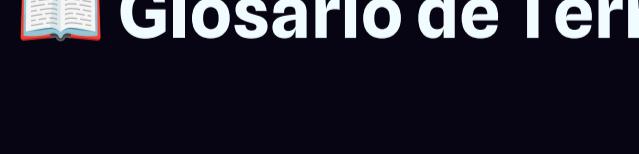
→ Instalar y configurar Python

→ Crear un proyecto organizado

→ Instalar librerías de data science

→ Ejecutar tu primer programa

Próximo nivel:



Curso 1: Backtesting de Estrategias

Aprenderás a analizar datos históricos y desarrollar estrategias ganadoras

Curso 2: Paper Trading en Tiempo Real

Implementarás tu sistema con datos en vivo sin arriesgar dinero

Curso 3: Machine Learning para Trading

Optimizarás tus estrategias con inteligencia artificial

Versión: 1.0

Fecha: Octubre 21, 2025

Autor: Pablo Felipe

Proyecto: algo-options

Comunidad: Skool.com - Trading Algorítmico Cuantitativo

Glosario de Términos

• **API:** Application Programming Interface - forma de comunicarse con servicios externos

• **CLI:** Command Line Interface - terminal de comandos

• **CSV:** Comma-Separated Values - formato de archivo para datos tabulares

• **DataFrame:** Estructura de datos de pandas (como tabla de Excel)

• **Entorno Virtual:** Carpeta aislada con dependencias específicas del proyecto

• **Git:** Sistema de control de versiones (historial de cambios)

• **IDE:** Integrated Development Environment - editor de código avanzado

• **Librería/Paquete:** Código pre-escrito que podemos reusar

• **pip:** Instalador de paquetes de Python

• **Python:** Lenguaje de programación que usaremos

• **Script:** Archivo con código Python (.py)

• **Terminal:** Interfaz de texto para ejecutar comandos

• **VS Code:** Editor de código que usaremos

• **venv:** Módulo de Python para crear entornos virtuales