

# Meet Pandas

FinTech  
Lesson 3.2



# Class Objectives

---

By the end of today's class, you will be able to:



Describe the benefits of Pandas over spreadsheets to manipulate data for financial use cases.



Explain what a DataFrame is and how it differs from a series.



Create DataFrames from CSV files and use basic commands to manipulate them.



Clean data using built-in commands of DataFrames.



Manipulate data using DataFrame indexes.



Describe the underlying theory and calculations of returns using Pandas.



Create basic data visualizations with Pandas' built-in plotting functions.



# Hello Pandas!

# What is Pandas?

---

Pandas is one of the most powerful libraries in Python.

It was created by Wes McKinney to offer a flexible, high-performance tool for conducting quantitative analyses of financial data.



# Why Pandas?

# Spreadsheets Are AWESOME.

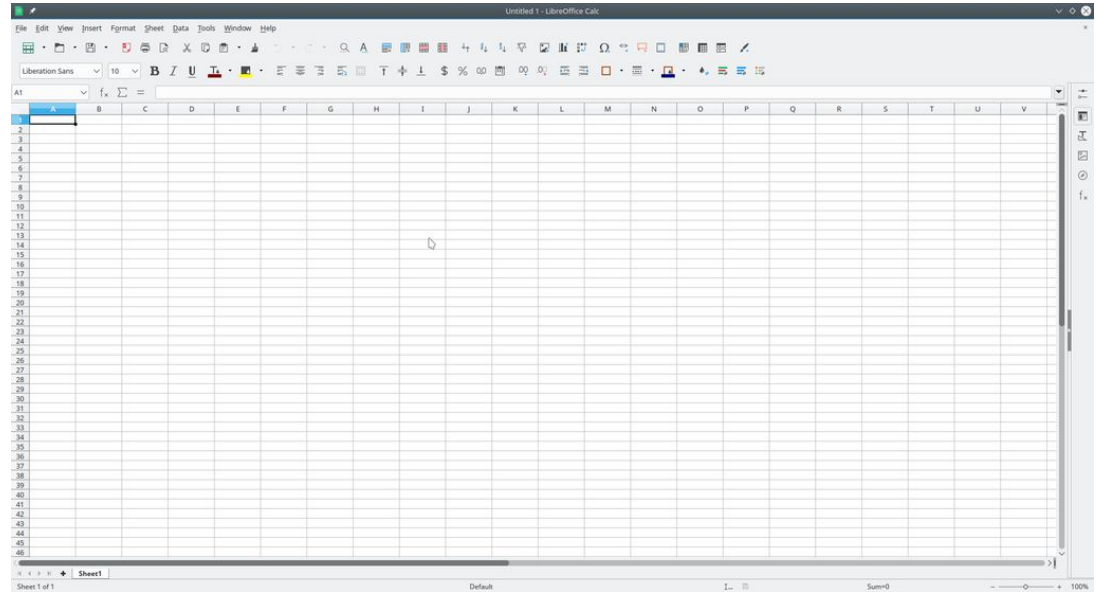
## The Rise of Spreadsheets



A screenshot of a vintage spreadsheet application. The interface is green with black text. The top bar shows 'C11 (L) TOTAL' and 'C1 25'. The spreadsheet has four columns: ITEM, NO., UNIT, and COST. The data is as follows:

ITEM	NO.	UNIT	COST
MUCK RAKE	43	12.95	556.85
BUZZ CUT	15	6.75	101.25
TOE TONER	250	49.95	12487.50
EYE SNUFF	2	4.95	9.90
SUBTOTAL			13155.50
9.75% TAX			1282.66
TOTAL			14438.16

Since they first appeared in 1969, spreadsheets have transformed finance and quants analysis.



# The Pain of Using Spreadsheets

---

Have you ever felt like this?





# The Pain of Using Spreadsheets

Spreadsheets are great, but they can become a pain when you are dealing with complex data:

- Calculations are often not reproducible.
- Data can be overwritten in the spreadsheet.
- Data cleaning may overwrite the original data.
- Sharing spreadsheets can be challenging.
- Combining data from multiple spreadsheets is difficult.
- Spreadsheets can be slow and buggy.
- Large datasets are not handled well.

# Spreadsheets

can become a pain when you are dealing with a large amount of data.

reproducible.

the spreadsheet.

the original data.

challenging.

the spreadsheets is difficult.

and buggy.

small



# Pandas to the Rescue

---

Fortunately, we have Pandas to help us manage data on Python.



# The Origins of Pandas

---

- [Pandas](#) is one of the most powerful open source libraries in Python for analyzing and manipulating data.
- The library was born on 2008 at [AQR Capital](#) when [Wes McKinney](#) was looking for a high-performing, flexible tool to perform quantitative analysis on financial data.
- Etymology: The name “Pandas” originates from “panel data structures.”

# Why Pandas is Great

---

- Pandas provides many advantages over spreadsheets due to its data structures and built-in functions for analysis.
- Python + Pandas = the perfect combination for small experiments or for implementing large-scale production systems to analyze data and make smarter decisions.
- High-performance data structures:
  - Series (1D labelled vectors)
  - DataFrame (2D structures similar to spreadsheets)
  - Panel (Collection of DataFrames as 3D labelled arrays)
- Built-in time series functionality, which is a must for financial and quants analysis.

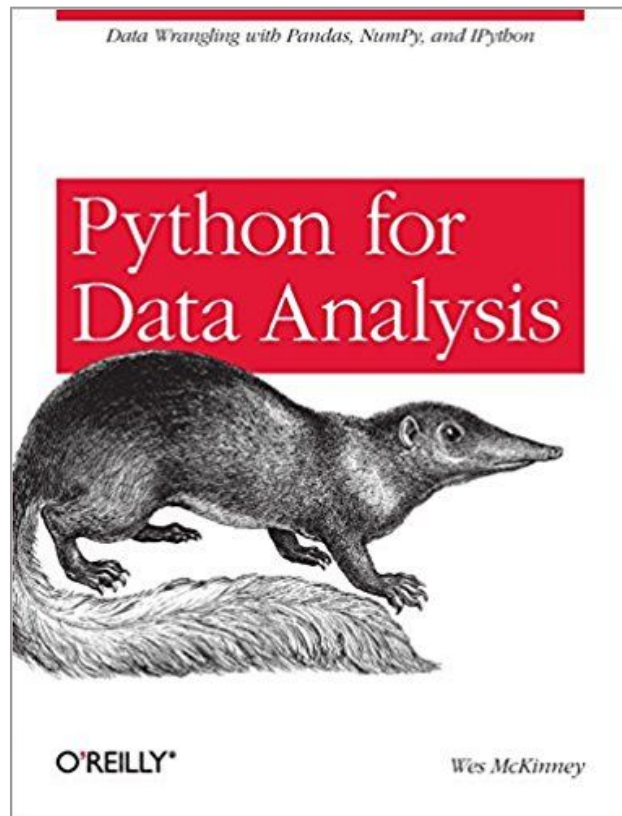


# Resources for Learning More About Pandas

---

- Official website: [pandas.pydata.org](https://pandas.pydata.org)
- Pandas on GitHub: [github.com/pydata/pandas](https://github.com/pydata/pandas)
- *Python for Data Analysis* by Wes McKinney

*Python for Data Analysis*  
by Wes McKinney  
(O'Reilly Media, 2017)





There is life beyond  
Excel to analyze data.  
Let's find the path!

# The Pandas DataFrame



# What is a Pandas DataFrame?

It is a special data structure in Pandas that is designed to work with tabular data (data that has rows and columns like a spreadsheet) and provides some useful functions to help analyze and manipulate tabular data.

A Pandas DataFrame can be created in several ways, such as:

- using a Python dictionary
- using a list of lists, or
- reading data from an external file like CSV or JSON.

The diagram illustrates a Pandas DataFrame with 5 rows and 4 columns. The columns are labeled 'Name', 'Score', 'Attempts', and 'Qualify'. The rows are indexed 0 to 4. Annotations include: 'Columns' with arrows pointing to the column headers; 'Rows' with arrows pointing to the row indices; and 'Data' with a bracket pointing to the data cells. Specific data cells are highlighted in orange: 'Dima' in row 1, 'Name' column; '16.5' in row 2, 'Score' column; '3' in row 3, 'Attempts' column; and 'no' in row 4, 'Qualify' column. Orange lines connect these highlighted cells to the 'Data' label.

	Columns			
	Name	Score	Attempts	Qualify
0	Anastasia	12.5	1	yes
1	Dima	9.0	3	no
2	Katherine	16.5	2	yes
3	James	NaN	3	no
4	Emily	9.0	2	no

Rows

Data



# Instructor Demonstration

## Reading CSV Files



## Activity: Reading Stock Data from a CSV File

In this activity, you will get hands-on experience reading CSV files into Pandas. You will use the `read_csv` function, sample data with the `head` function, and create DataFrames with specified column names.

(Instructions sent via Slack.)

**Suggested Time:**  
10 Minutes





**Time's Up!** Let's Review.



# Instructor Demonstration

## Column Manipulation

# Data Cleaning



# Data Cleaning at a Glance

---

Data cleaning is the process of getting your data ready to be used.

It consists of three parts:

01

Data exploration



02

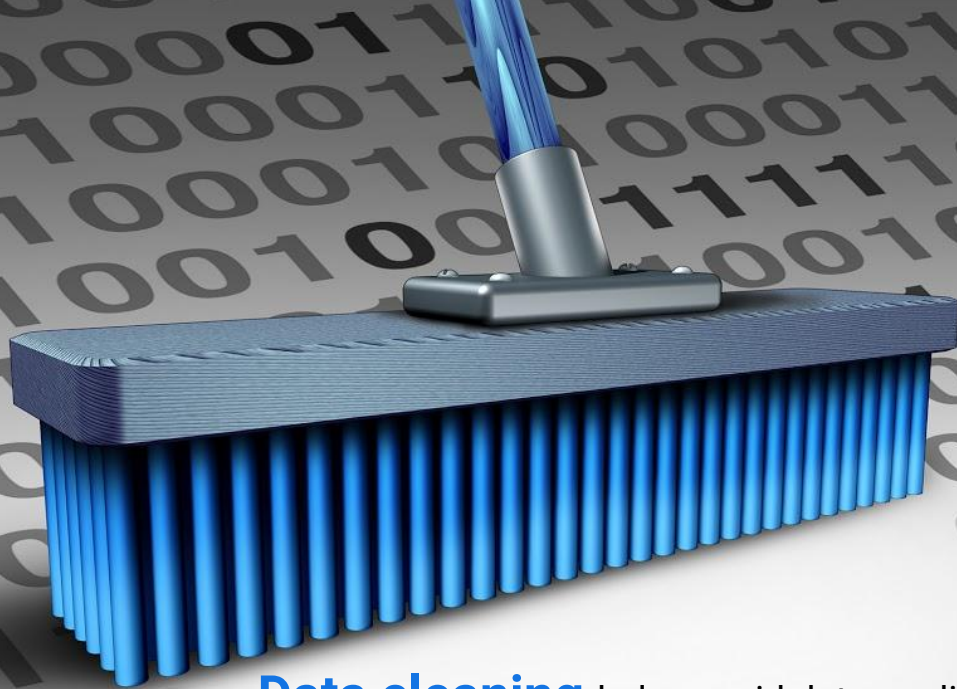
Data quality checks



03

Data cleaning strategies





**Data cleaning** helps avoid data quality issues that may compromise the integrity, or "health," of a dataset. The goal of data cleaning is to keep the "plumbing" of data pipelines clean and in working condition, so that analytics can run smoothly.

# Determining Data Quality

---

Most data quality rules are influenced by general coding etiquette.



Use correct data types and minimize use of nulls.



Numerical fields should only contain digits, not string characters.



Datasets should not contain duplicate rows.



Define a standard format for managing dates.

# Why does data become dirty?



# Common Reasons for Having Dirty Data

---

01

## **Typos**

If gone unchecked, typos can corrupt data values. There is little that can be done to get rid of typos, as it is often difficult to identify them.

02

## **Human Error**

Humans can copy and manipulate data incorrectly. For instance, someone might copy and paste data into the wrong Excel file. Or a Python function can incorrectly compute a data value.

03

## **Poor Data Management**

Data is poorly managed when it is not cleaned or stored in an effective way. Industry standards and business rules should be consistently implemented to ensure data integrity.



# Instructor Demonstration

## Data Cleaning





## Activity: Spring Cleaning

In this activity, you will perform a series of data quality checks on stock data to ensure the data is ready for analytical use. The objective is to learn how to cleanse data using Pandas native functions (`count`, `value_counts`, `isnull`, `sum`, `mean`, `contains`, and `replace`).

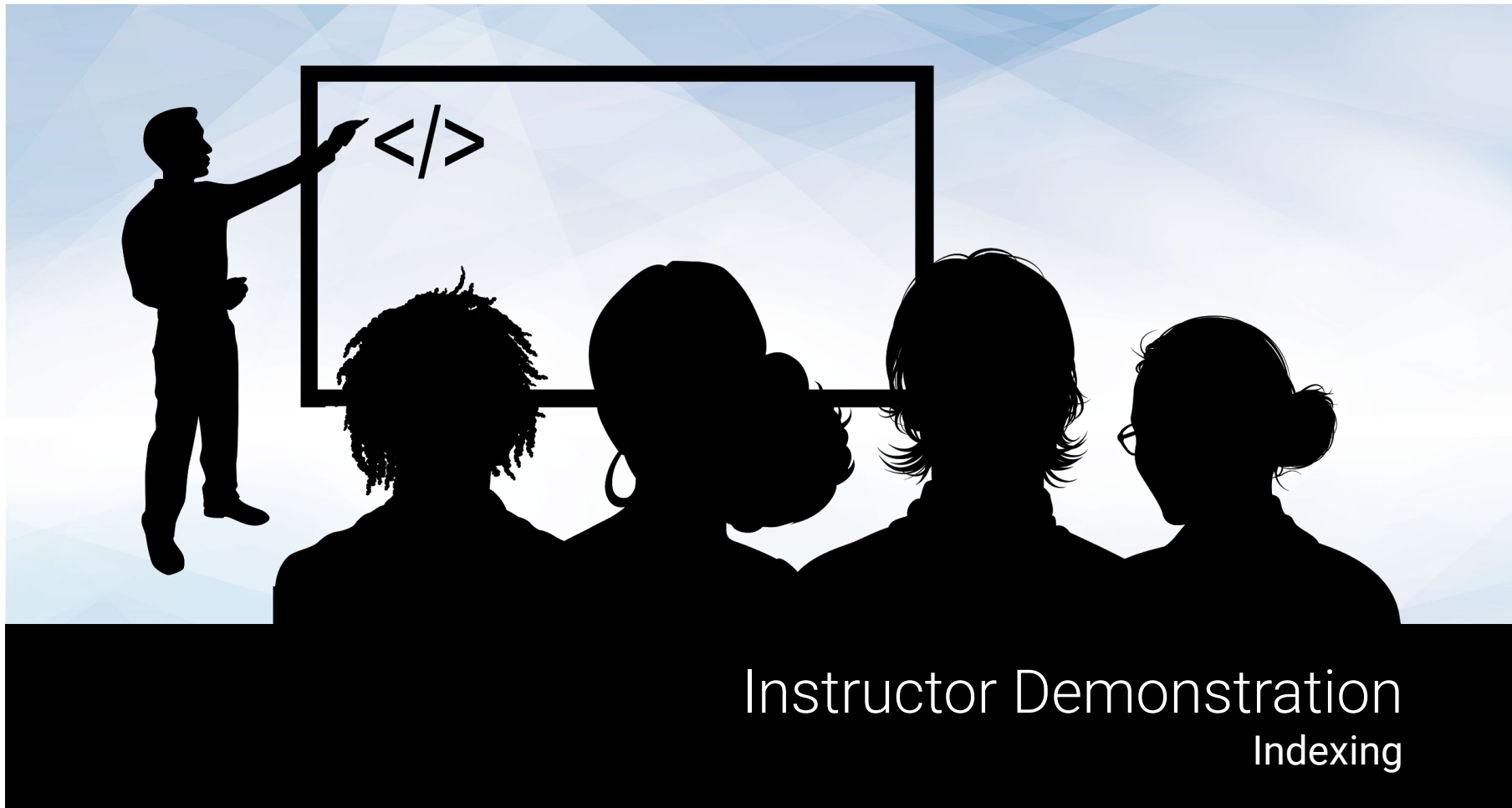
(Instructions sent via Slack.)

**Suggested Time:**  
15 Minutes





**Time's Up!** Let's Review.



# Instructor Demonstration

## Indexing



## Activity: Three-Year Loans

This activity will test your DataFrame indexing skills. You will slice and dice the `loans.csv` data to generate insights about three-year loan customers.

(Instructions sent via Slack.)

**Suggested Time:**  
15 Minutes





**Time's Up!** Let's Review.







# Instructor Demonstration

## Pandas Visualizations



## Activity: Market Analysis

In this activity, you will use Pandas to create three different charts: a pie chart, bar chart, and scatter plot.

(Instructions sent via Slack.)

**Suggested Time:**  
15 Minutes





**Time's Up!** Let's Review.

# Returns

# Return on Investment (ROI)

---

- ROI is a percentage calculation that signifies either a profit or loss relative to the initial cost of an investment.
- ROI calculations can be used to standardize and compare the investment performances of varying asset classes such as equities, bonds, real estate, etc.



# Daily Returns

---

Daily returns are a series of returns calculated over several days, with each daily return representing the relative increase or decrease in investment between days.

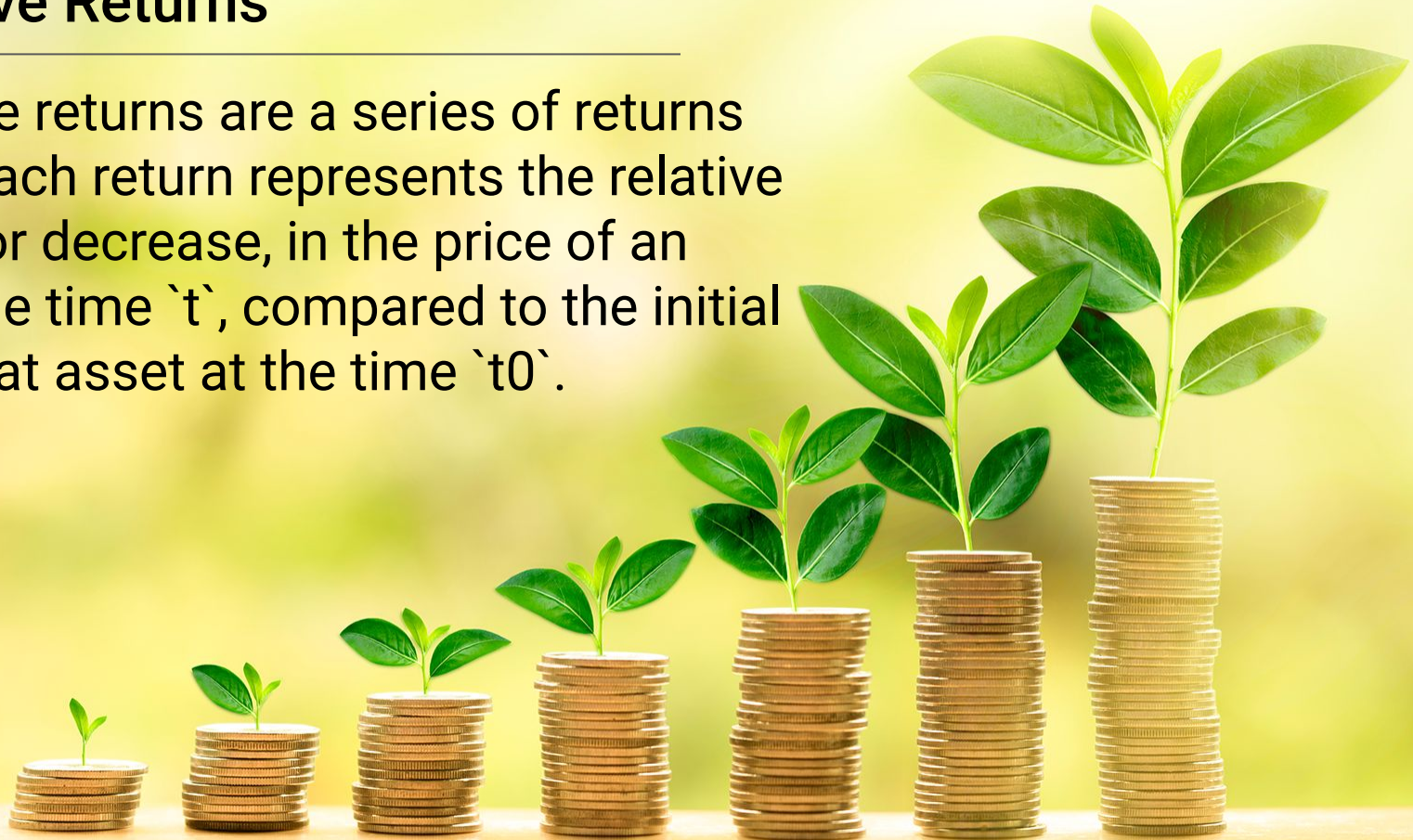




# Cumulative Returns

---

Cumulative returns are a series of returns in which each return represents the relative increase, or decrease, in the price of an asset at the time  $t$ , compared to the initial price of that asset at the time  $t_0$ .



# Calculating ROI using Python

---

```
# ROI = (Current Value of Investment - Cost of  
Investment) / Cost of Investment
```

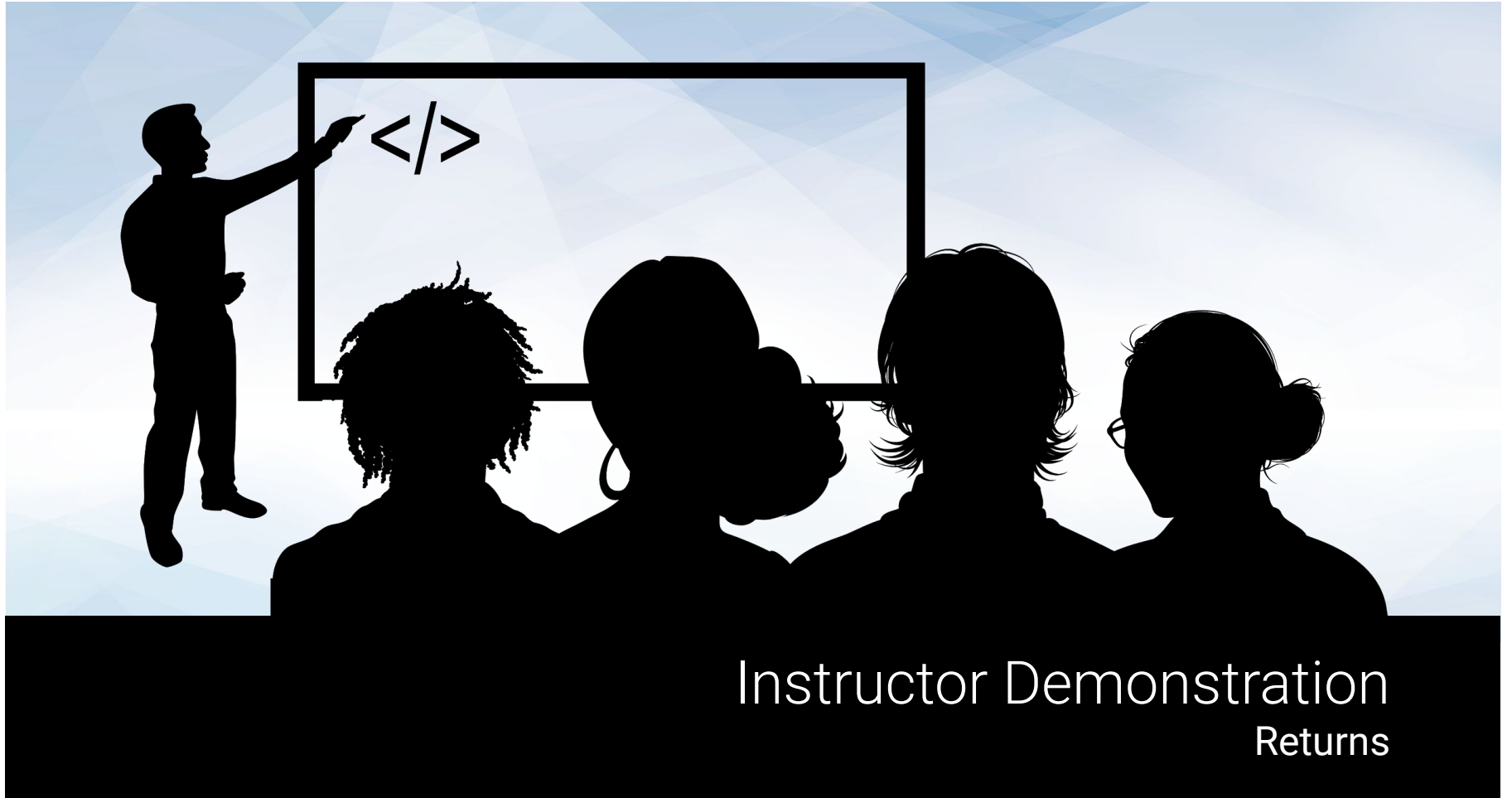
```
initial_investment = 100
```

```
current_price = 110
```

```
roi = (current_price - initial_investment) /  
initial_investment
```

```
roi_pct = roi * 100
```





# Instructor Demonstration

## Returns



## Activity:

# Returns Over Date Ranges

In this activity, you will analyze historical price data for Shopify and plot the daily returns over the previous 1-, 2-, 3-, and 4-year periods.

(Instructions sent via Slack.)

**Suggested Time:**  
15 Minutes





**Time's Up!** Let's Review.



Questions?

*The  
End*