↗ lesson 02

Python Basic Syntax: Importing Libraries

Python for Financial Analysis
Rajah Chacko

|elvtr|

Syllabus Review

1 Introduction to Python: Python in Finance

2 **Python Basic Syntax: Importing Libraries**

3 Working with Pandas

4 Pandas Underneath the Hood: Working with NumPy

5 Data Wrangling and Visualization

6 Extracting Financial Insights from Charts and Graphs

7 Financial Calculations with Python: Part 1

8 Financial Calculations with Python: Part 2

9 CAPM and Portfolio Management

10 Linear Regression

11 Time Series Analysis

12 Algorithmic Trading

+ **Bonus Class:** Cryptocurrency Beyond the Basics with a Fintech Guest Speaker

Class agenda

- Working with libraries: Pandas, NumPy, Matplotlib, and Seaborn
- Python basic syntax: Assignment statements, creating variables, indentation, conditionals, and loops
- Being Pythonic
- Pythonic: Strings and print

Working with libraries

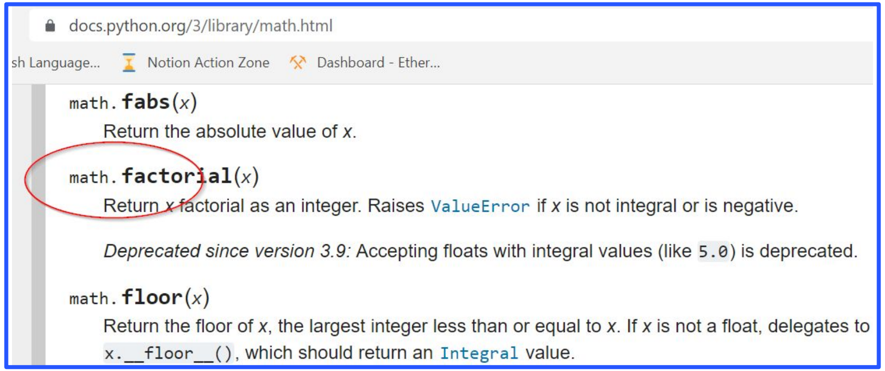
- We can do the basics (except for datetime) without libraries
- Python does a lot of specific things
  A. But you don't need square root and datetime always & everywhere
  B. We import these before we use them the first time
- Note that some libraries have no hierarchy. Some have packages within.
  A. from datetime import datetime
- We can import third-party packages
  A. Quick look at https://pypi.org/.
  B. Your employer might not allow you to access all packages
- You can create your own libraries
  A. This helps reuse (both you and others)
  B. At work, I read and write packages. A lot.

Syntax of import

- Basic form
  A. import math
  B. This imports everything
  C. Case (as always) is important
- Importing namespace packages
  A. from datetime import datetime, date
  B. This is more efficient
  C. PyCharm helps with this one Code -> Organize imports
- Importing with an alias ("as")
  A. By convention, we use this only for these major libraries.
  B. The aliases are always the same
    a. import pandas as pd
    b. import numpy as np
    c. import matplotlib.pyplot as plt
    d. import seaborn as sns

How to import libraries?

- Figure it out from the documentation
  A. For example, factorial at https://docs.python.org/3/library/math.html
- See how other people use it
  A. For example, https://stackoverflow.com/questions/415511/how-to-get-the-current-time-in-python/415519#415519



`math.fabs(x)`

Return the absolute value of *x*.

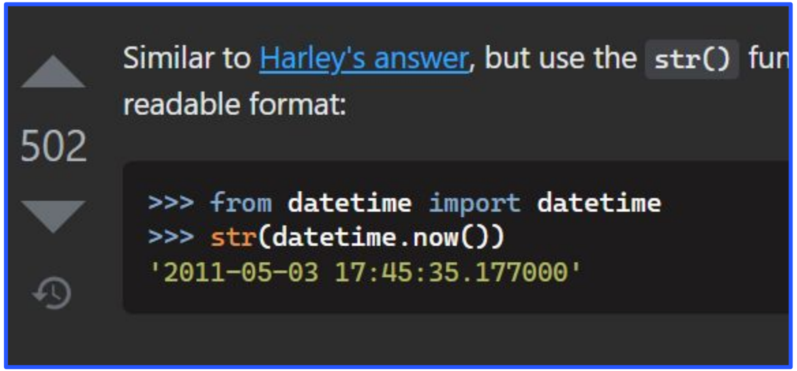
`math.factorial(x)`

Return *x* factorial as an integer. Raises `ValueError` if *x* is not integral or is negative.

Deprecated since version 3.9: Accepting floats with integral values (like `5.0`) is deprecated.

`math.floor(x)`

Return the floor of *x*, the largest integer less than or equal to *x*. If *x* is not a float, delegates to `x.__floor__()`, which should return an `Integral` value.



Similar to Harley's answer, but use the `str()` fun
readable format:

```
>>> from datetime import datetime
>>> str(datetime.now())
'2011-05-03 17:45:35.177000'
```

Python basic syntax

- Assignment statements

  A. Assignment is not equals
     a. i = i + 1

  B. You can save space with multiple assignments (at the cost of clarity)
     a. x, name, answer = 3.14, 'Bill', 42

  C. Tuple assignment
     a. pt2 = (-14.4, 3.6)
     b. x, y = pt2 # equiv to x=-14.4 and y=3.6

- Creating variables

  D. (by convention) starts with a lowercase letter

  E. (by convention) prefer default_answer to defaultAnswer

  F. Guido and PEP 8 at https://www.python.org/dev/peps/pep-0008/#prescriptive-naming-conventions

Conditionals

- Various flavors of if statements
  A. (simple) if
  B. If / else
  C. If / elif / elif … / else
  D. Rare birds
     a. Short-hand
     b. Ternary
- Indented blocks
  A. Other languages used braces
  B. Both Jupyter and PyCharm are aware of indents
     a. Tab and backspace are your friends

Loops

- For loops using integers
  A. Range
- For loops over an iterable
  A. Lists
  B. Special word about enumerate
  C. Dictionaries
- While loops
- For and while loops also have
  A. Break
  B. continue

Being Pythonic

(from the Zen of Python)

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Flat is better than nested.
- Sparse is better than dense.
- Readability counts.

Write Python so you'll want to read it. Write Python so others won't cringe when they read it. Comment your code so you can understand it next year.
Add Markdown to say what you're doing.

Making it look good

- Formatting. Can specify

  A. widths,
  B. fill characters,
  C. left-justified, right-justified, and centered
  D. Decimal places

Markup in Jupyter notebooks

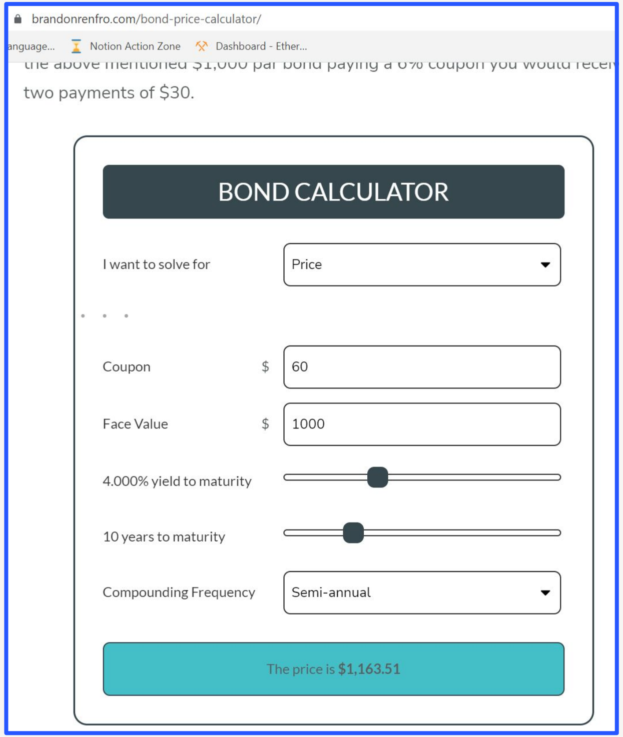
- Choose the Markdown dropdown
- See resource slide

Assignment #1

1. Datetime – You can find the Spring equinox from https://www.timeanddate.com/calendar/spring-equinox.htm
    a. Print the Spring equinox by setting it to a variable.
    b. Print “Spring is coming” if the date 2022 Spring equinox is in the future (or “Spring has sprung” if it is in the past.)
    c. Print how many days it is until (or since) the 2022 Spring equinox.
2. Create a bond calculator based on its face value, coupon payment, coupon frequency, the current interest rate, and the years to maturity
    a. What’s the price of a bond with a face value of $1000, coupon payment of $30 twice a year, with 10 years to maturity, and a prevailing market rate of 4%?
    b. See Screenshots on next page

https://www.brandonrenfro.com/bond-price-calculator/

https://dqydj.com/bond-pricing-calculator/



the above mentioned $1,000 par bond paying a 6% coupon you would recei
two payments of $30.

BOND CALCULATOR

I want to solve for: Price

Coupon $ 60

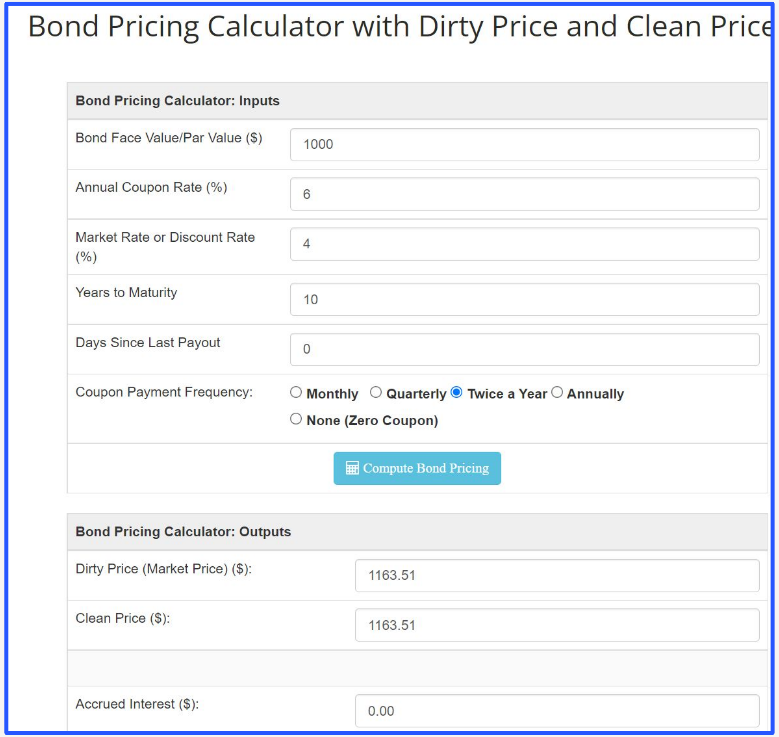
Face Value $ 1000

4.000% yield to maturity

10 years to maturity

Compounding Frequency: Semi-annual

The price is **$1,163.51**



Bond Pricing Calculator with Dirty Price and Clean Price

Bond Pricing Calculator: Inputs

Bond Face Value/Par Value ($)	1000
Annual Coupon Rate (%)	6
Market Rate or Discount Rate (%)	4
Years to Maturity	10
Days Since Last Payout	0
Coupon Payment Frequency:	○ **Monthly** ○ **Quarterly** ● **Twice a Year** ○ **Annually** ○ **None (Zero Coupon)**

Bond Pricing Calculator: Outputs

Dirty Price (Market Price) ($):	1163.51
Clean Price ($):	1163.51
Accrued Interest ($):	0.00

Take-home Assignment #2 (optional)

The Python collections and itertools API libraries can do some interesting things. Can you use either library to calculate which combinations of two six-sided dice add up to 7?

Resources

(part 1)

- Importing libraries

  https://docs.python.org/3/reference/import.html

- Style guide for Python code

  https://www.python.org/dev/peps/pep-0008/#prescriptive-naming-conventions

- Conditionals

  https://www.geeksforgeeks.org/python-if-else/

  https://www.geeksforgeeks.org/ternary-operator-in-python/

- Loops

  https://www.geeksforgeeks.org/python-range-function/

  https://docs.python.org/3/tutorial/controlflow.html

  https://www.geeksforgeeks.org/python-for-loops/

Resources

(part 2)

- The Zen of Python (or simply import this)

  https://www.techrepublic.com/article/python-programming-language-a-cheat-sheet/

- Formatting

  https://docs.python.org/3/library/string.html#format-specification-mini-language

- Jupyter markdown

  https://towardsdatascience.com/write-markdown-latex-in-the-jupyter-notebook-10985edb91fd

  https://www.ibm.com/docs/en/watson-studio-local/1.2.3?topic=notebooks-markdown-jupyter-cheatsheet

Q&A