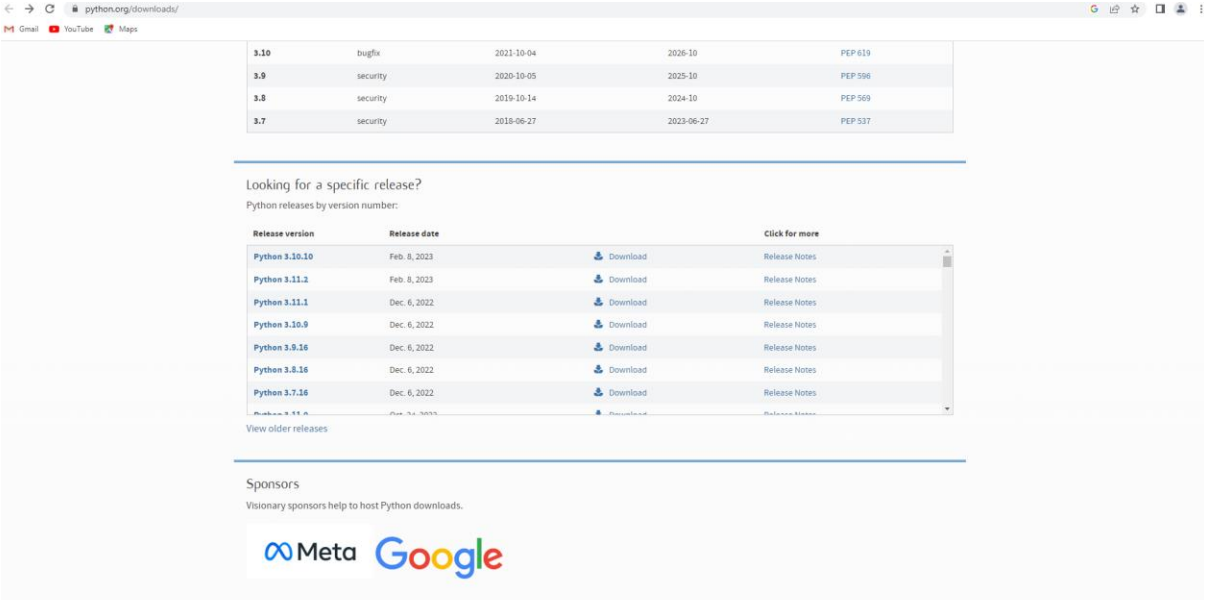
Here's a step-by-step guide on how to run the Python program in Visual Studio Code (VS Code).

- Install Python: Before running any Python program, you need to have Python installed on your computer. You can download the latest version of Python from the official Python website (https://www.python.org/downloads/). Follow the instructions provided by the installer to install Python on your computer.

1. (https://www.python.org/downloads/release/python-31010/) install this version of python so you won't see any breakout in futures.



3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569
3.7	security	2018-06-27	2023-06-27	PEP 537

Looking for a specific release?

Python releases by version number:

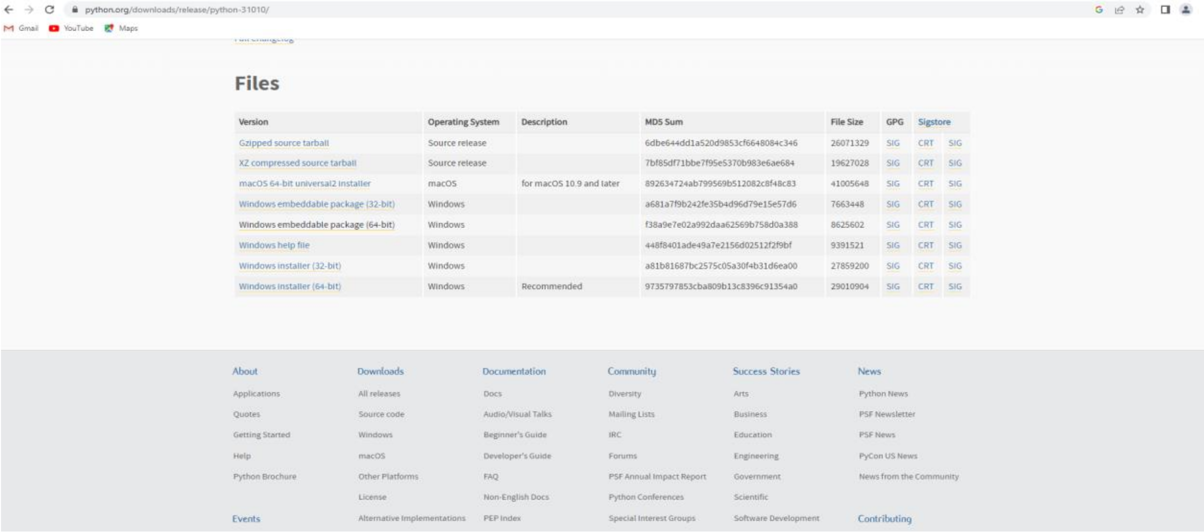
Release version	Release date		Click for more
Python 3.10.10	Feb. 8, 2023	Download	Release Notes
Python 3.11.2	Feb. 8, 2023	Download	Release Notes
Python 3.11.1	Dec. 6, 2022	Download	Release Notes
Python 3.10.9	Dec. 6, 2022	Download	Release Notes
Python 3.9.16	Dec. 6, 2022	Download	Release Notes
Python 3.8.16	Dec. 6, 2022	Download	Release Notes
Python 3.7.16	Dec. 6, 2022	Download	Release Notes

View older releases

Sponsors

Visionary sponsors help to host Python downloads.

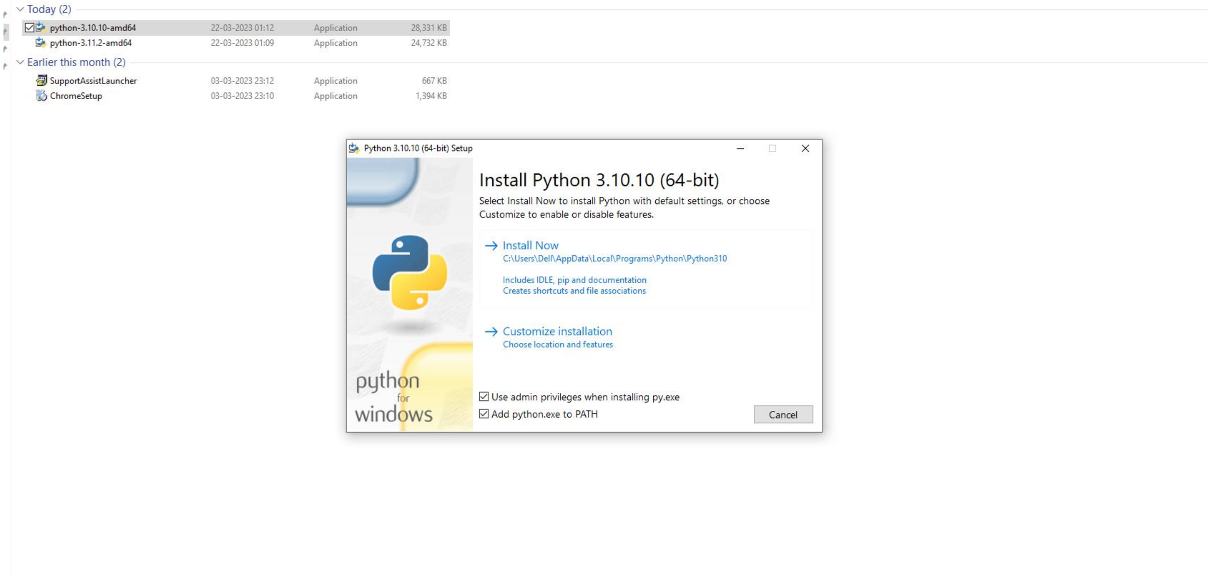
2. Go down and select windows install 64bit (For windows OS)



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG	Sigstore	
Gzipped source tarball	Source release		6dbe644dd1a520d9853cf6648084c346	26071329	SIG	CRT	SIG
XZ compressed source tarball	Source release		7bf85df71bbe7f95e5370b983e6ae684	19627028	SIG	CRT	SIG
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	892634724ab799569b512082c8f48c83	41005648	SIG	CRT	SIG
Windows embeddable package (32-bit)	Windows		a681a7f9b242fe35b4d96d79e15e57d6	7663448	SIG	CRT	SIG
Windows embeddable package (64-bit)	Windows		f38a9e7e02a992daa62569b758d0a388	8625602	SIG	CRT	SIG
Windows help file	Windows		448f8401ade49a7e2156d02512f2f9bf	9391521	SIG	CRT	SIG
Windows installer (32-bit)	Windows		a81b81687bc2575c05a30f4b31d6ea00	27859200	SIG	CRT	SIG
Windows installer (64-bit)	Windows	Recommended	9735797853cba809b13c8396c91354a0	29010904	SIG	CRT	SIG

3. Now we are going to installing the python on windows machine



Today (2)

☑ python-3.10.10-amd64	22-03-2023 01:12	Application	28,331 KB
python-3.11.2-amd64	22-03-2023 01:09	Application	24,732 KB

Earlier this month (2)

SupportAssistLauncher	03-03-2023 23:12	Application	667 KB
ChromeSetup	03-03-2023 23:10	Application	1,394 KB

Python 3.10.10 (64-bit) Setup

Install Python 3.10.10 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

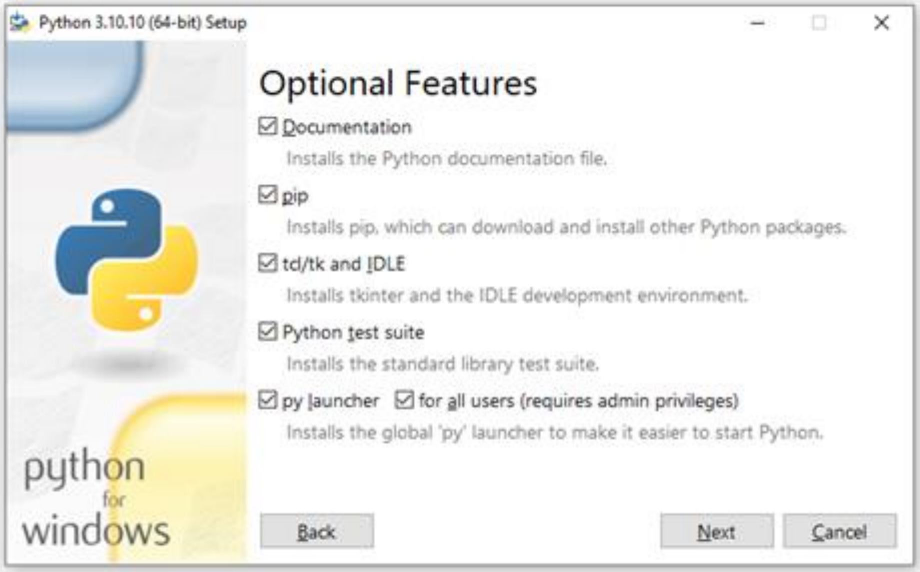
→ Install Now
C:\Users\Dell\AppData\Local\Programs\Python\Python310

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Use admin privileges when installing py.exe
☑ Add python.exe to PATH

4. select the add python.exe to PATH selectbox and click on customize installation



Python 3.10.10 (64-bit) Setup

Optional Features

☑ Documentation
Installs the Python documentation file.

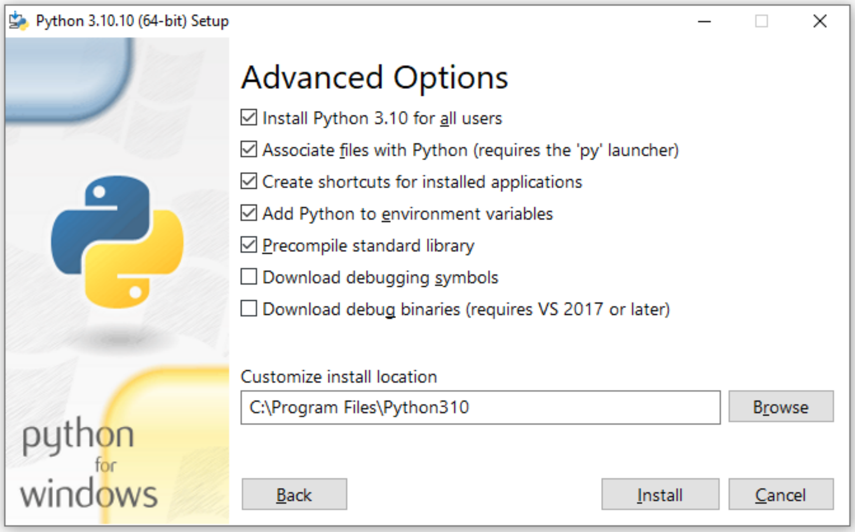
☑ pip
Installs pip, which can download and install other Python packages.

☑ tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

☑ py launcher ☑ for all users (requires admin privileges)
Installs the global 'py' launcher to make it easier to start Python.

5. Now in advance option select first checkbox, now good to go, click on install



Python 3.10.10 (64-bit) Setup

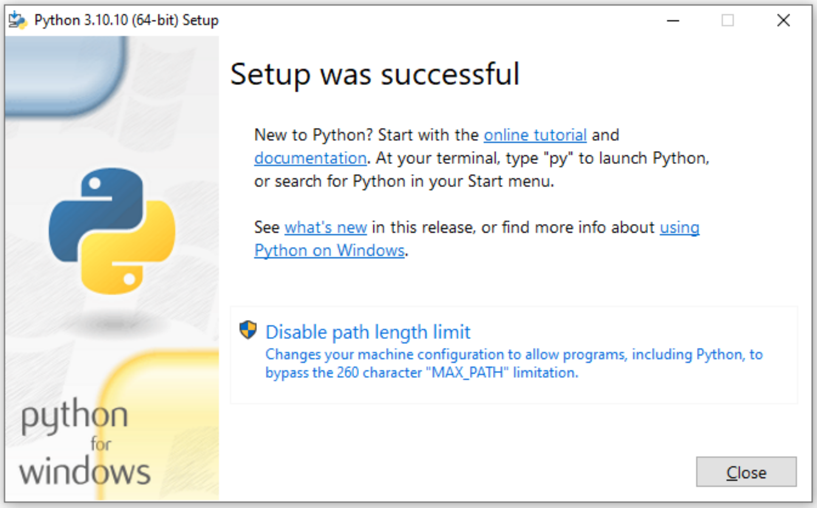
Advanced Options

☑ Install Python 3.10 for all users
☑ Associate files with Python (requires the 'py' launcher)
☑ Create shortcuts for installed applications
☑ Add Python to environment variables
☑ Precompile standard library
☐ Download debugging symbols
☐ Download debug binaries (requires VS 2017 or later)

Customize install location
C:\Program Files\Python310

it will take some time, meanwhile remember all the steps.

6. It once you see the below screen, your installation done.



Python 3.10.10 (64-bit) Setup

Setup was successful

New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

See what's new in this release, or find more info about using Python on Windows.

Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

- Install VS Code: VS Code is a free and open-source code editor developed by Microsoft. You can download and install it from the official website (https://code.visualstudio.com/download).
- Open VS Code: After installing VS Code, right click on the Folder which we shared with you and click on Open with VSCode. You should see a welcome screen with options to open a folder or file, or to create a new file.
- Open the terminal: To open the terminal in VS Code, click on the "Terminal" icon in the left-hand menu.

6. The terminal is already has your program file added to it. So you don’t need to bother how to add it.
7. With following command

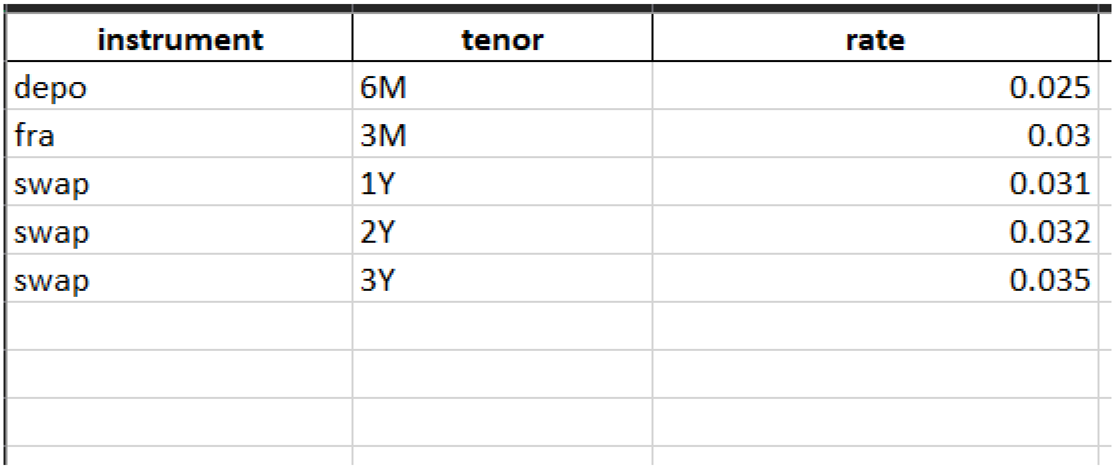
   ```
   >python installRequirePackages.py
   ```

   Run the program: Once you are in the correct folder, type "python install_packages.py" in the terminal and press "Enter". The program should run and install the required packages.
8. Now you good to go to run the main Vanilla Swapping program.

That's it! You have successfully run the Python program to install the required packages using VS Code.

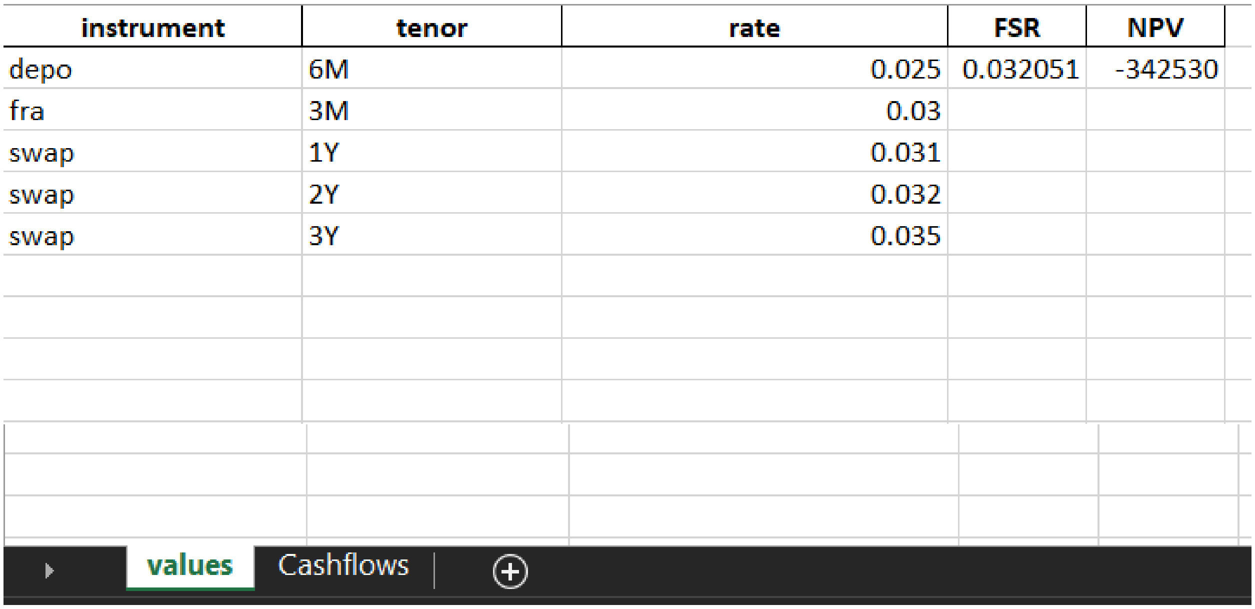
Detail and overview of Excel File:

Here, we need excel file which will have the data in following order, this file will be the input for the program



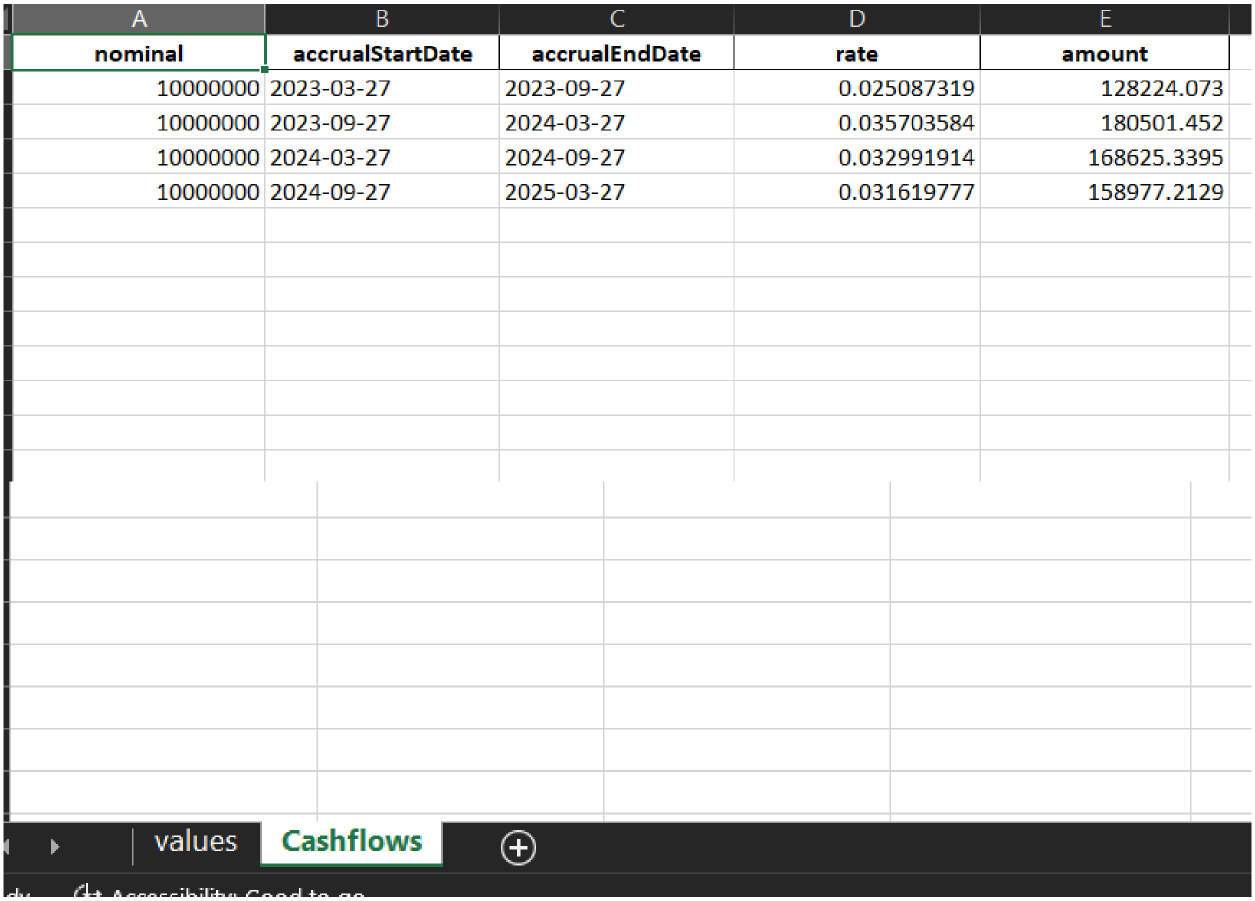
instrument	tenor	rate
depo	6M	0.025
fra	3M	0.03
swap	1Y	0.031
swap	2Y	0.032
swap	3Y	0.035

So basically, we are giving the input to the program through this excel file, here after the program will store the FSR(Fair Swap Rate) and NPV in the same sheet like follow:



instrument	tenor	rate	FSR	NPV
depo	6M	0.025	0.032051	-342530
fra	3M	0.03		
swap	1Y	0.031		
swap	2Y	0.032		
swap	3Y	0.035		

After that the program generate the Cashflow and store in the same excel file into another newly created sheet, as follow:

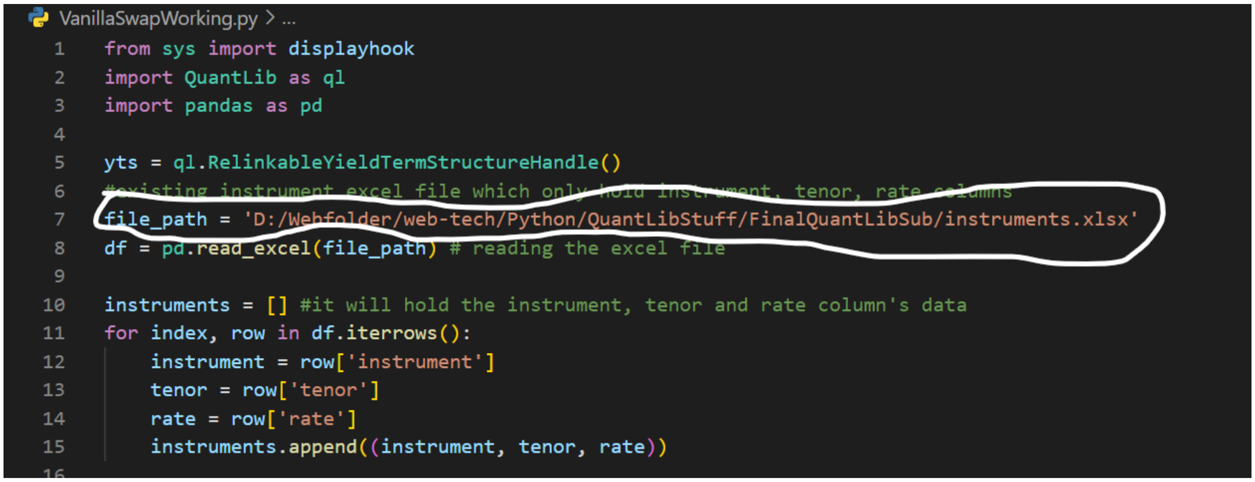


nominal	accrualStartDate	accrualEndDate	rate	amount
10000000	2023-03-27	2023-09-27	0.025087319	128224.073
10000000	2023-09-27	2024-03-27	0.035703584	180501.452
10000000	2024-03-27	2024-09-27	0.032991914	168625.3395
10000000	2024-09-27	2025-03-27	0.031619777	158977.2129

So, the value sheet will have the input of Instrument, Tenor, Rate, along with FSR and NPV column which program will generate and store the values accordingly, after then, it will create new sheet name Cashflows and stored the generated result as shown above.

How To Run The Vanilla Swap Program:

Before running the vanilla Swap Program, there you need to modify the excel file path accordingly, as shown in the image



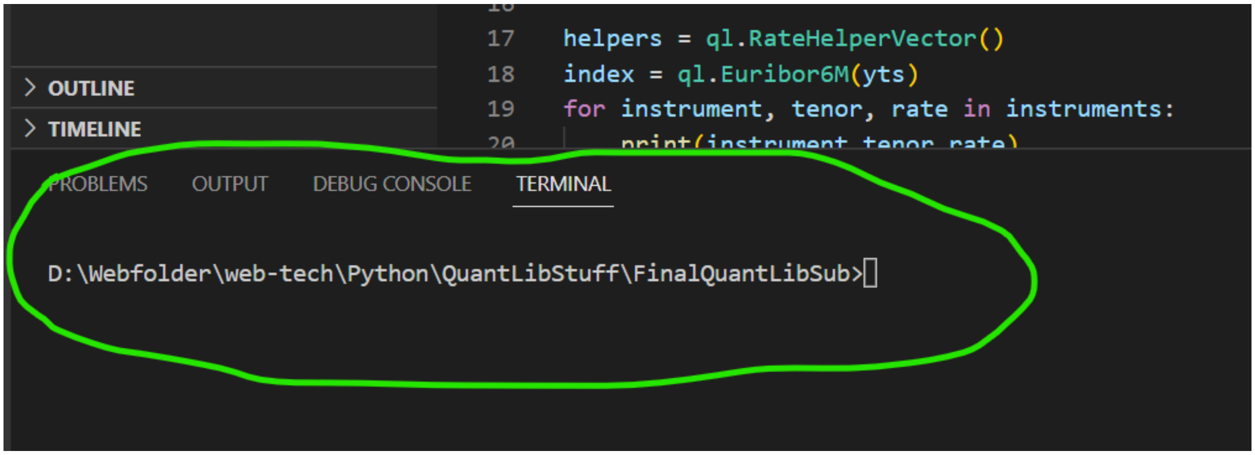
```
from sys import displayhook
import QuantLib as ql
import pandas as pd

yts = ql.RelinkableYieldTermStructureHandle()
#existing instrument excel file which only hold instrument, tenor, rate columns
file_path = 'D:/Webfolder/web-tech/Python/QuantLibStuff/FinalQuantLibSub/instruments.xlsx'
df = pd.read_excel(file_path) # reading the excel file

instruments = [] #it will hold the instrument, tenor and rate column's data
for index, row in df.iterrows():
    instrument = row['instrument']
    tenor = row['tenor']
    rate = row['rate']
    instruments.append((instrument, tenor, rate))
```

Once you modify the path inside the program, follow the next steps:

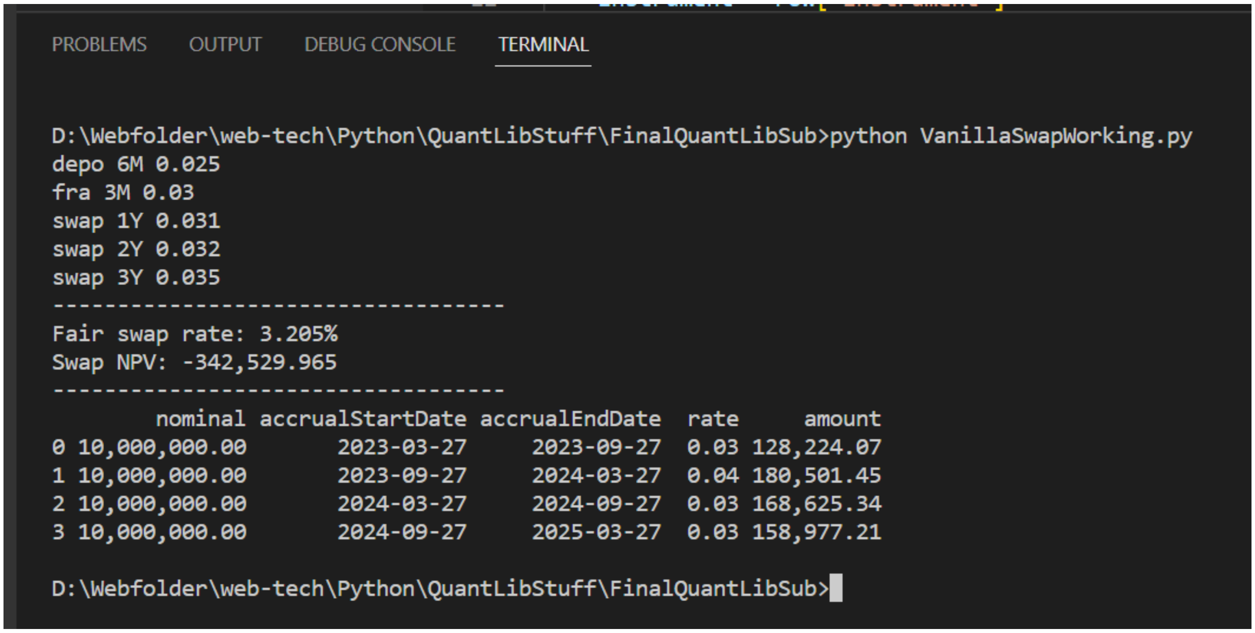
1. There you have to open the terminal as you did while running the first program



```
helpers = ql.RateHelperVector()
index = ql.Euribor6M(yts)
for instrument, tenor, rate in instruments:
    print(instrument, tenor, rate)
```

```
D:\Webfolder\web-tech\Python\QuantLibStuff\FinalQuantLibSub>
```

here you have to type the following command also shown in the image
>python VanillaSwapWorking.py
also shown in the below image.



```
D:\Webfolder\web-tech\Python\QuantLibStuff\FinalQuantLibSub>python VanillaSwapWorking.py
depo 6M 0.025
fra 3M 0.03
swap 1Y 0.031
swap 2Y 0.032
swap 3Y 0.035
-----------------------------------
Fair swap rate: 3.205%
Swap NPV: -342,529.965
-----------------------------------
        nominal accrualStartDate accrualEndDate  rate     amount
0 10,000,000.00       2023-03-27     2023-09-27  0.03 128,224.07
1 10,000,000.00       2023-09-27     2024-03-27  0.04 180,501.45
2 10,000,000.00       2024-03-27     2024-09-27  0.03 168,625.34
3 10,000,000.00       2024-09-27     2025-03-27  0.03 158,977.21

D:\Webfolder\web-tech\Python\QuantLibStuff\FinalQuantLibSub>
```

here my file name is VanillaSwapWorking.py preceding this name there is a python keyword, which mostly use for to Run the python file. Hit Enter, and you will see the result also you will see the update and generated result inside the Excel file.

Brief Documentation on Vanilla Swap

The program starts by importing required modules, QuantLib and pandas, and then reads an Excel file containing financial instrument data. The file path is provided in the 'file_path' variable. The instrument data is stored in a pandas dataframe 'df' using the 'read_excel' method.

Next, the program creates an empty list 'instruments' to hold instrument, tenor and rate column's data. A for loop iterates through each row of the dataframe 'df' using the 'iterrows' method. For each row, the instrument, tenor and rate values are extracted and appended as a tuple to the 'instruments' list.

The program then creates an empty 'RateHelperVector' object called 'helpers'. It creates an instance of the Euribor6M class called 'index'. The 'for' loop iterates through each tuple in the 'instruments' list. For each tuple, it extracts the instrument, tenor and rate values. It checks the instrument type using an 'if' statement, and if it is 'depo', it creates a DepositRateHelper object and appends it to the 'helpers' list. If it is 'fra', it creates a FraRateHelper object and appends it to the 'helpers' list. If it is 'swap', it creates a SwapRateHelper object and appends it to the 'helpers' list.

After adding all the RateHelper objects to the 'helpers' list, it creates a PiecewiseLogCubicDiscount object called 'curve'. It passes the 'helpers' list to this object, along with the 'TARGET' calendar, and a day-count convention of Actual/Actual (Actual365). This creates a yield term structure that represents the market's interest rates.

The program then links the yield term structure to the RelinkableYieldTermStructureHandle object called 'yts'. This will allow us to change the yield term structure at a later time.

Next, the program creates an instance of the DiscountingSwapEngine class called 'engine', passing in the yield term structure.

It then creates a Vanilla Swap object called 'swap', using the 'MakeVanillaSwap' method. This object represents a standard fixed-floating interest rate swap. It takes a tenor of 2 years, the Euribor6M index, a fixed rate of 5%, and a forward start date of 2 days. It also takes a notional value of 10 million euros, and the discounting swap engine 'engine' as the pricing engine.

The program calculates the fair swap rate and the NPV (Net Present Value) of the swap using the 'fairRate' and 'NPV' methods of the 'swap' object respectively. It adds two new columns 'FSR' and 'NPV' to the 'df' dataframe, and populates them with the fair swap rate and the NPV values respectively.

The program then writes the modified dataframe to the same Excel file using the 'to_excel' method of pandas.

Finally, the program creates a new sheet in the Excel file called 'Cashflows'. It creates a dataframe called 'cashflows' containing the cashflow data for the fixed leg of the swap, and writes it to the 'Cashflows' sheet using the 'to_excel' method of pandas. It also displays the 'cashflows' dataframe using the 'displayhook' method.

Overall, this program reads in instrument data from an Excel file, creates a yield term structure and pricing engine, calculates the fair swap rate and NPV of a vanilla swap, and writes the results back to the same Excel file along with cashflow data for the swap.

Submitted by:
Mr. Rudresh Sisodiya.