T1A3: Terminal Application Presentation

By Peter Li

Synopsis

- A BTC data analyser that allows for user input to generate a variety of functions and outcomes

- Three* main features:
  1. Price checker
  2. Price comparison
  3. Volume checker

  **profit calculator currently in beta testing*

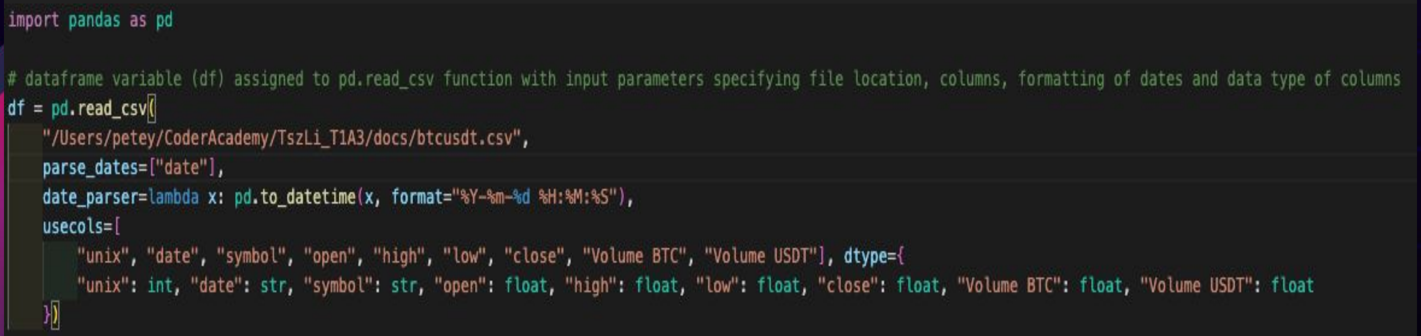
  **Multiple crypto data to be implemented*

- Data is taken from https://www.cryptodatadownload.com/data/kucoin/ in the form of a CSV file and interpreted and presented via Pandas Library

Pandas Library and Data

- Pandas library imported as pd (see Readme documentation for install instructions)

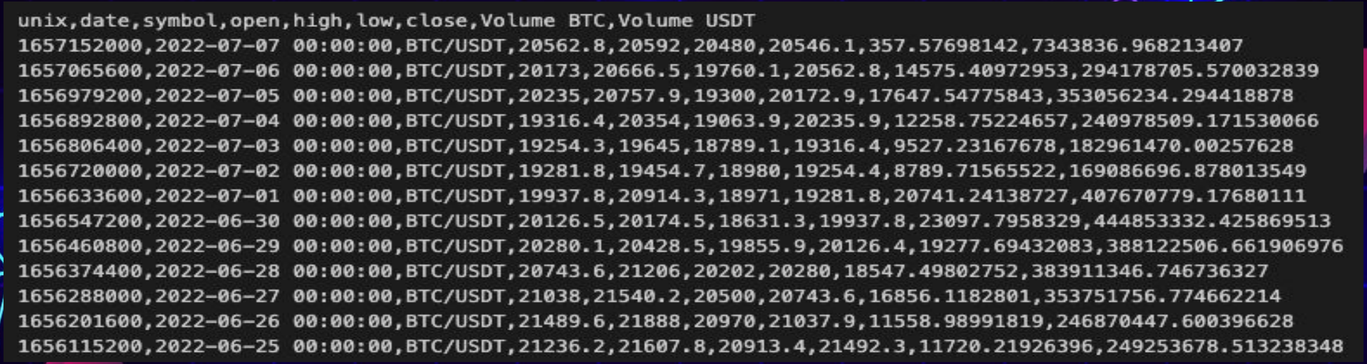
- Pd.read_csv command allows us to read and present CSV data through pandas for use in python terminal
  - Result saved to a variable called df (“dataframe”)
  - Link to file, dates and columns are all defined as parameters, in addition to the data types of the column



```
import pandas as pd

# dataframe variable (df) assigned to pd.read_csv function with input parameters specifying file location, columns, formatting of dates and data type of columns
df = pd.read_csv(
    "/Users/petey/CoderAcademy/TszLi_T1A3/docs/btcusdt.csv",
    parse_dates=["date"],
    date_parser=lambda x: pd.to_datetime(x, format="%Y-%m-%d %H:%M:%S"),
    usecols=[
        "unix", "date", "symbol", "open", "high", "low", "close", "Volume BTC", "Volume USDT"], dtype={
        "unix": int, "date": str, "symbol": str, "open": float, "high": float, "low": float, "close": float, "Volume BTC": float, "Volume USDT": float
    })
```

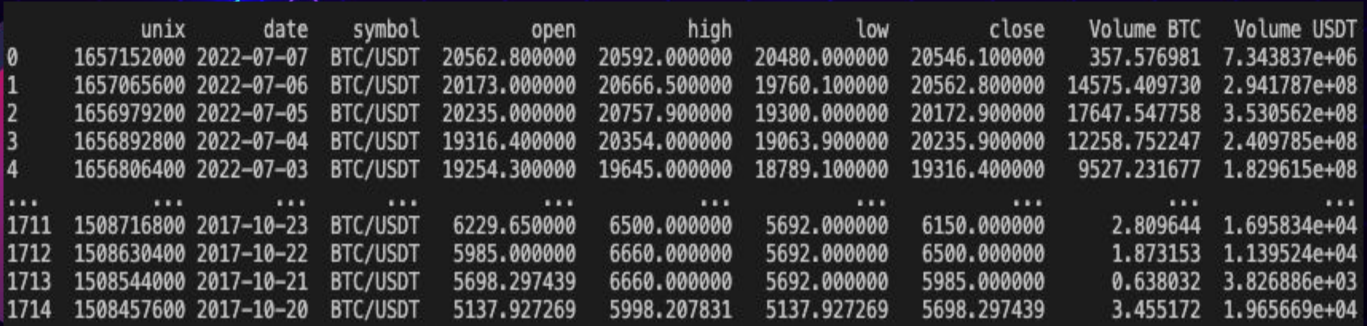
CSV data and resulting data table in Python

Before



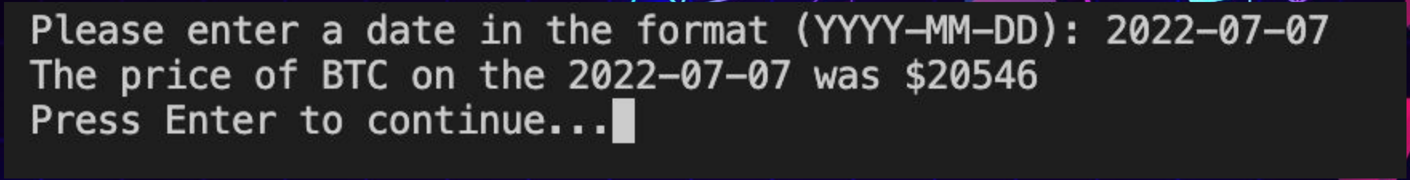
```
unix,date,symbol,open,high,low,close,Volume BTC,Volume USDT
1657152000,2022-07-07 00:00:00,BTC/USDT,20562.8,20592,20480,20546.1,357.57698142,7343836.968213407
1657065600,2022-07-06 00:00:00,BTC/USDT,20173,20666.5,19760.1,20562.8,14575.40972953,294178705.570032839
1656979200,2022-07-05 00:00:00,BTC/USDT,20235,20757.9,19300,20172.9,17647.54775843,353056234.294418878
1656892800,2022-07-04 00:00:00,BTC/USDT,19316.4,20354,19063.9,20235.9,12258.75224657,240978509.171530066
1656806400,2022-07-03 00:00:00,BTC/USDT,19254.3,19645,18789.1,19316.4,9527.23167678,182961470.00257628
1656720000,2022-07-02 00:00:00,BTC/USDT,19281.8,19454.7,18980,19254.4,8789.71565522,169086696.878013549
1656633600,2022-07-01 00:00:00,BTC/USDT,19937.8,20914.3,18971,19281.8,20741.24138727,407670779.17680111
1656547200,2022-06-30 00:00:00,BTC/USDT,20126.5,20174.5,18631.3,19937.8,23097.7958329,444853332.425869513
1656460800,2022-06-29 00:00:00,BTC/USDT,20280.1,20428.5,19855.9,20126.4,19277.69432083,388122506.661906976
1656374400,2022-06-28 00:00:00,BTC/USDT,20743.6,21206,20202,20280,18547.49802752,383911346.746736327
1656288000,2022-06-27 00:00:00,BTC/USDT,21038,21540.2,20500,20743.6,16856.1182801,353751756.774662214
1656201600,2022-06-26 00:00:00,BTC/USDT,21489.6,21888,20970,21037.9,11558.98991819,246870447.600396628
1656115200,2022-06-25 00:00:00,BTC/USDT,21236.2,21607.8,20913.4,21492.3,11720.21926396,249253678.513238348
```

After

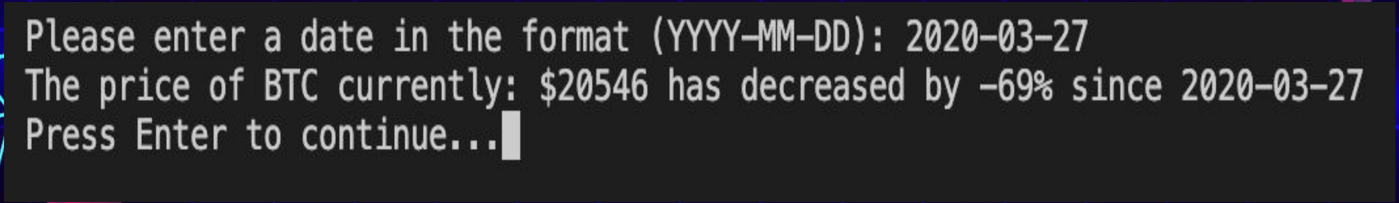


```
            unix        date    symbol          open          high           low         close    Volume BTC   Volume USDT
0     1657152000  2022-07-07  BTC/USDT  20562.800000  20592.000000  20480.000000  20546.100000    357.576981  7.343837e+06
1     1657065600  2022-07-06  BTC/USDT  20173.000000  20666.500000  19760.100000  20562.800000  14575.409730  2.941787e+08
2     1656979200  2022-07-05  BTC/USDT  20235.000000  20757.900000  19300.000000  20172.900000  17647.547758  3.530562e+08
3     1656892800  2022-07-04  BTC/USDT  19316.400000  20354.000000  19063.900000  20235.900000  12258.752247  2.409785e+08
4     1656806400  2022-07-03  BTC/USDT  19254.300000  19645.000000  18789.100000  19316.400000   9527.231677  1.829615e+08
...          ...         ...       ...           ...           ...           ...           ...           ...           ...
1711  1508716800  2017-10-23  BTC/USDT   6229.650000   6500.000000   5692.000000   6150.000000      2.809644  1.695834e+04
1712  1508630400  2017-10-22  BTC/USDT   5985.000000   6660.000000   5692.000000   6500.000000      1.873153  1.139524e+04
1713  1508544000  2017-10-21  BTC/USDT   5698.297439   6660.000000   5692.000000   5985.000000      0.638032  3.826886e+03
1714  1508457600  2017-10-20  BTC/USDT   5137.927269   5998.207831   5137.927269   5698.297439      3.455172  1.965669e+04
```

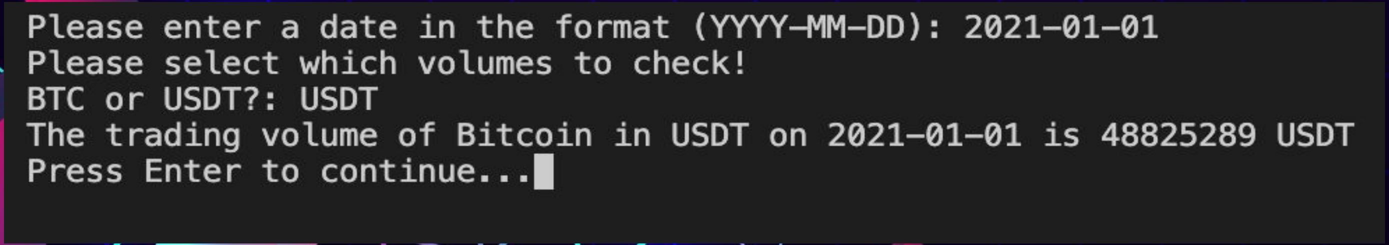
Input Example



```
Please enter a date in the format (YYYY-MM-DD): 2022-07-07
The price of BTC on the 2022-07-07 was $20546
Press Enter to continue...
```



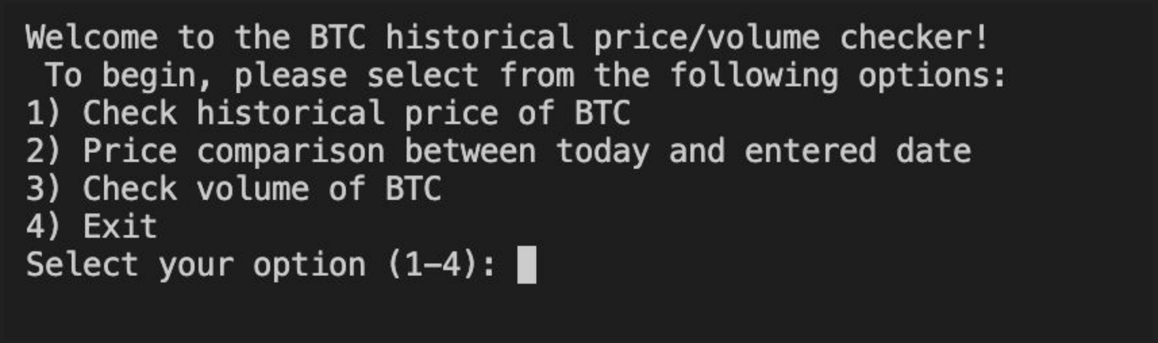
```
Please enter a date in the format (YYYY-MM-DD): 2020-03-27
The price of BTC currently: $20546 has decreased by -69% since 2020-03-27
Press Enter to continue...
```



```
Please enter a date in the format (YYYY-MM-DD): 2021-01-01
Please select which volumes to check!
BTC or USDT?: USDT
The trading volume of Bitcoin in USDT on 2021-01-01 is 48825289 USDT
Press Enter to continue...
```

Introduction and Menu

- Welcome_message and print_options function is printed when the application is run
- User selects an option by inputting 1-4 (with 4 telling the program to close)
    - Farewell message included

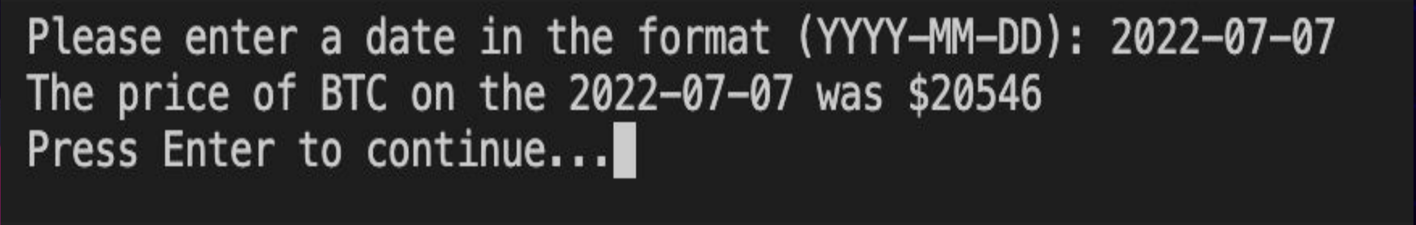


```
Welcome to the BTC historical price/volume checker!
 To begin, please select from the following options:
1) Check historical price of BTC
2) Price comparison between today and entered date
3) Check volume of BTC
4) Exit
Select your option (1-4): 
```

Price Checker

- When option 1 is selected, the user is prompted to input the specific date they want to the check for the corresponding BTC price
  - Format is specified for the user!

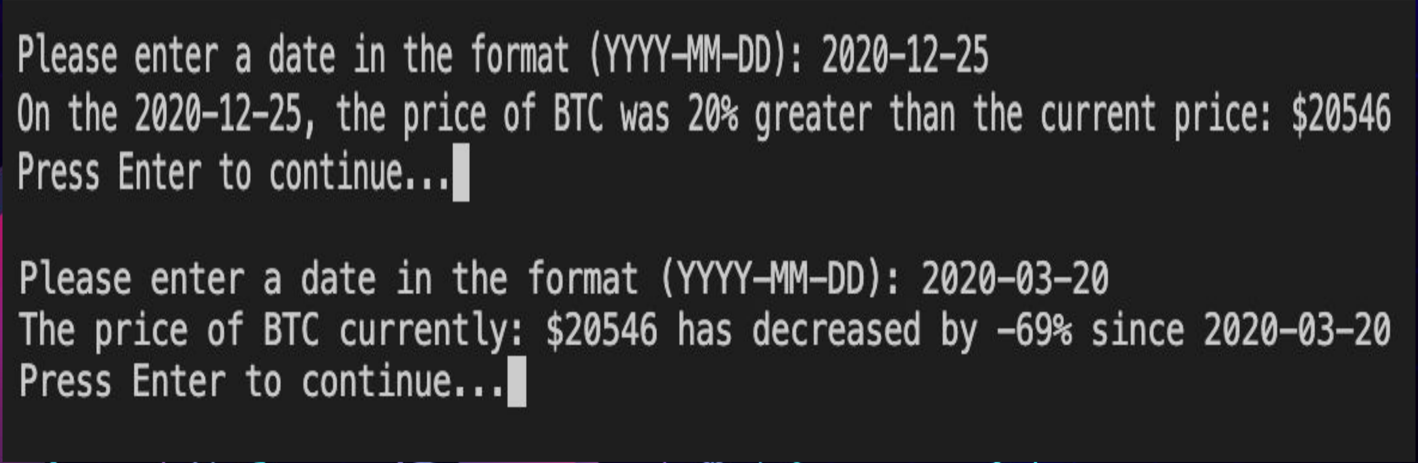
- Program will output the price of BTC on the day entered
  - Example is from July 7th 2022 (most recent data)



```
Please enter a date in the format (YYYY-MM-DD): 2022-07-07
The price of BTC on the 2022-07-07 was $20546
Press Enter to continue...
```

Price Comparison

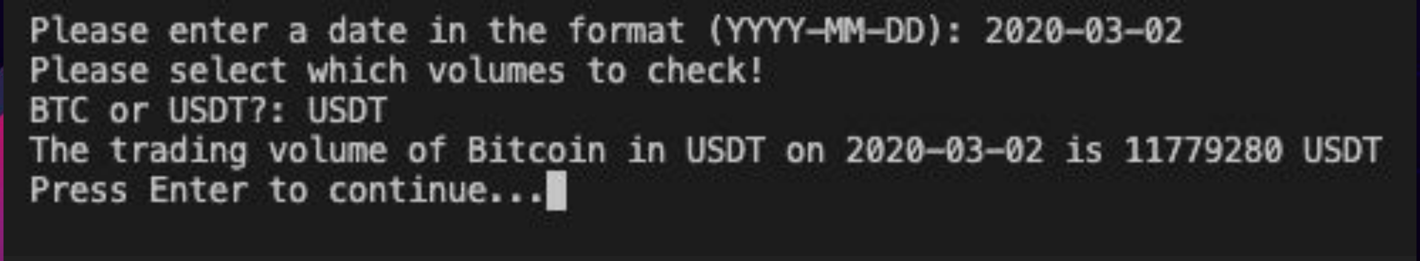
- With option 2 selected, again user inputs a date that they want to compare with the most current date (2022-07-07)
- The program will calculate the % difference between the price at the user's entered date and the current date and generate the output.
- There is a different statement generated depending on if the difference is negative or positive



```
Please enter a date in the format (YYYY-MM-DD): 2020-12-25
On the 2020-12-25, the price of BTC was 20% greater than the current price: $20546
Press Enter to continue...

Please enter a date in the format (YYYY-MM-DD): 2020-03-20
The price of BTC currently: $20546 has decreased by -69% since 2020-03-20
Press Enter to continue...
```

Volume Checker

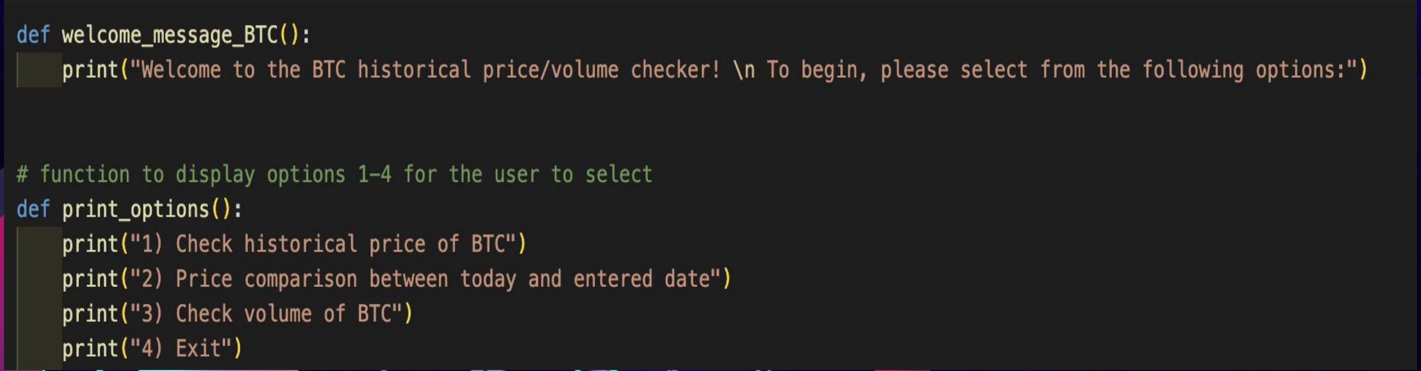
- User inputs date which then prompts a sub-menu asking for units (BTC/USDT)
    - Distinct from price checker
- Matches input date with the date in the data and outputs the trading volume in the specified unit on the input date



```
Please enter a date in the format (YYYY-MM-DD): 2020-03-02
Please select which volumes to check!
BTC or USDT?: USDT
The trading volume of Bitcoin in USDT on 2020-03-02 is 11779280 USDT
Press Enter to continue...
```

Menu Logic

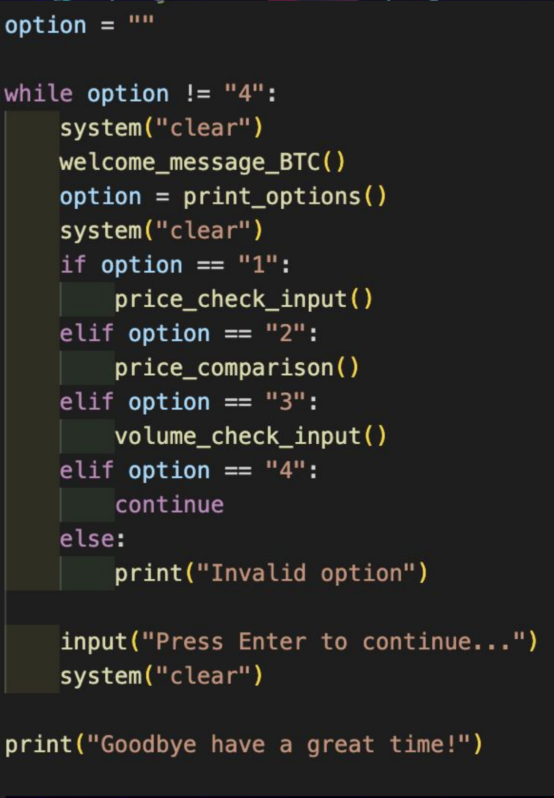
- Welcome message and menu programmed using functions and print statements.
    - First step where user interacts with the program by selecting the option they want



```python
def welcome_message_BTC():
    print("Welcome to the BTC historical price/volume checker! \n To begin, please select from the following options:")


# function to display options 1-4 for the user to select
def print_options():
    print("1) Check historical price of BTC")
    print("2) Price comparison between today and entered date")
    print("3) Check volume of BTC")
    print("4) Exit")
```

Menu Logic (cont.)

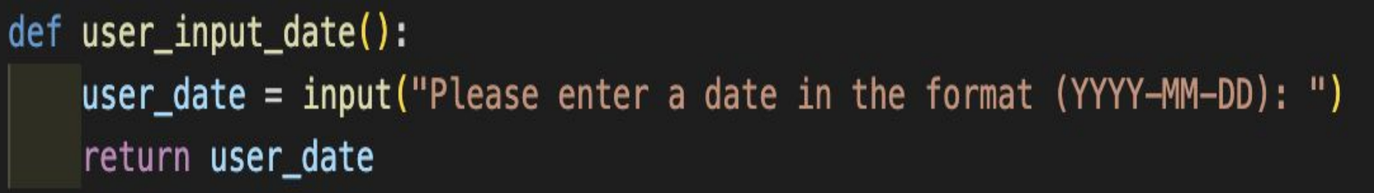
- Menu utilises while loops to control flow of program in a sequential step
- If the option selected is 1,2 or 3, the corresponding output will be to call the function to complete the task
    - E.g. Option 2 selected will result in the elif option == 2 to return True which calls the price_comparison function
- If the user selects option 4, the loop runs through to "Goodbye have a great time!" and the program ends.
- Accounted for options not (1-4), if E.g. 5 is entered, loop will recognise Else statement to be True and print to user that is not a valid option.
- "Press Enter to Continue…" is an important feature of this loop as it allows the user to return to the main menu to perform another function or exit program



```
option = ""

while option != "4":
    system("clear")
    welcome_message_BTC()
    option = print_options()
    system("clear")
    if option == "1":
        price_check_input()
    elif option == "2":
        price_comparison()
    elif option == "3":
        volume_check_input()
    elif option == "4":
        continue
    else:
        print("Invalid option")

    input("Press Enter to continue...")
    system("clear")

print("Goodbye have a great time!")
```

User Input Date Logic

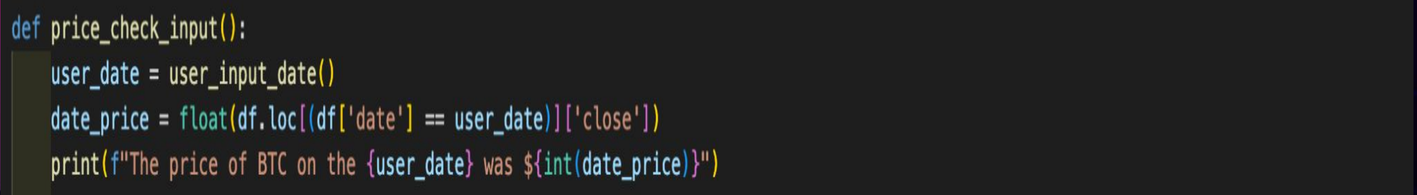
- Core part of the program is taking user input in a date format as below
    - Saved to a variable and can be called as a function in subsequent functions
    - Critical as the date comparison between user input and the data is the basis on how information is located and presented to user
    - Further implementations of code to account for the event a date is entered that does not match the format provided.



```python
def user_input_date():
    user_date = input("Please enter a date in the format (YYYY-MM-DD): ")
    return user_date
```

Price Check Logic

- Price check input function is called when user selects “1”
- User input date function is called and saved to a separate variable called user_date
- Pandas function df.loc (“dataframe.locate”) is used to check against the transformed CSV file (saved as df) and when a matching date is found, the price called by [‘close’] is then saved to a variable called Data price.
- Data price is printed to the user in a statement as an <u>integer</u>



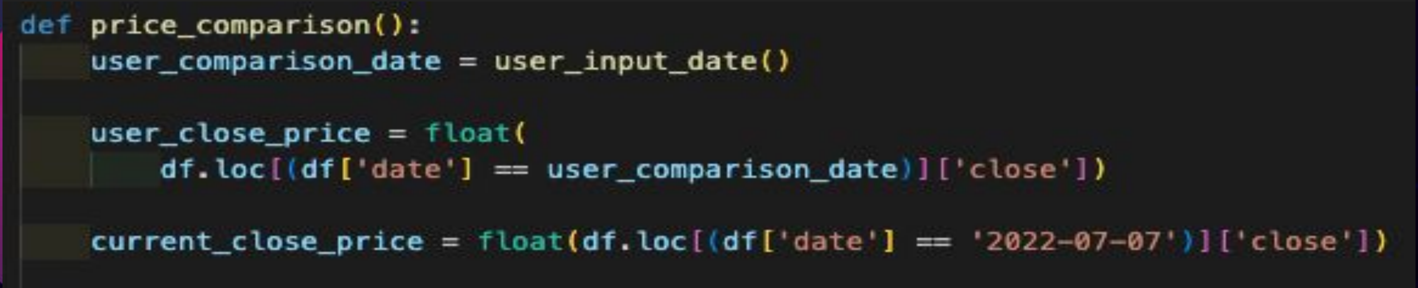
```python
def price_check_input():
    user_date = user_input_date()
    date_price = float(df.loc[(df['date'] == user_date)]['close'])
    print(f"The price of BTC on the {user_date} was ${int(date_price)}")
```

Price Comparison logic

- Again, price comparison logic calls on the user_input_date function and saved to a variable: user_comparison_date

- Df.loc matches the user_comparison_date with the exact date in the data and gets the closing price and saves the variable as a float.

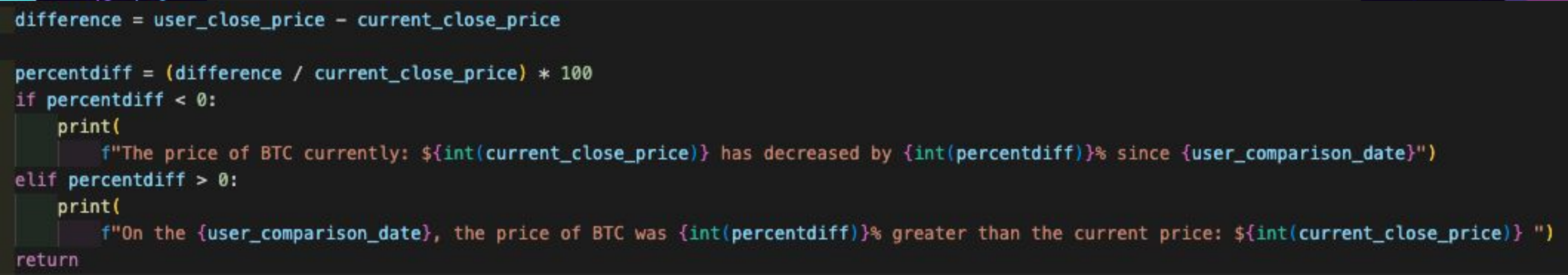
- Current close price variable is created from locating the most recent date and the closing price.



```
def price_comparison():
    user_comparison_date = user_input_date()

    user_close_price = float(
        df.loc[(df['date'] == user_comparison_date)]['close'])

    current_close_price = float(df.loc[(df['date'] == '2022-07-07')]['close'])
```

Price Comparison logic (cont.)

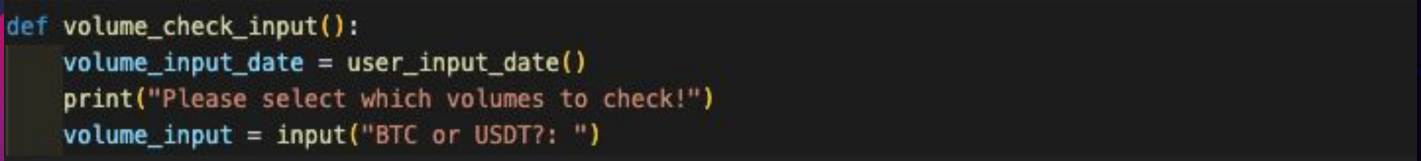
- Once the close price of the user's input date and the current date has been saved, operations can be performed on it
- To calculate the % difference, first the difference was calculated and then "percentdiff" variable was defined
- If statements to control the flow of the function where if the difference was positive or negative, this would impact the output statement shown to the user
- Function is returned at the end to be called in later functions **testing**
- Further implementations to account for percentdiff = 0



```python
difference = user_close_price - current_close_price

percentdiff = (difference / current_close_price) * 100
if percentdiff < 0:
    print(
        f"The price of BTC currently: ${int(current_close_price)} has decreased by {int(percentdiff)}% since {user_comparison_date}")
elif percentdiff > 0:
    print(
        f"On the {user_comparison_date}, the price of BTC was {int(percentdiff)}% greater than the current price: ${int(current_close_price)} ")
return
```

Volume Check Logic

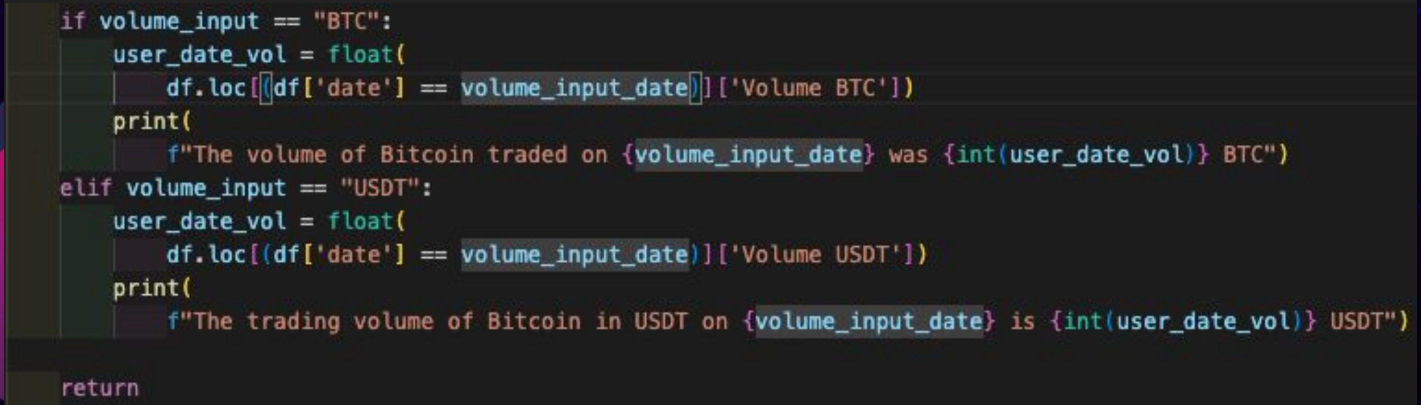
- Volume check function follows the same motif, call on user_input_date function and save to a variable
- Difference now is there is a submenu within function to ask for user input on what units to define the data provided by volume column (BTC/USDT)
    - Saved to a variable called volume_input
- Critical as CSV data provides both volume in BTC and in USDT



```
def volume_check_input():
    volume_input_date = user_input_date()
    print("Please select which volumes to check!")
    volume_input = input("BTC or USDT?: ")
```

Volume Check Logic (cont.)

- Once units are confirmed, if statements are used and when the user input matches the string, we locate the matching date and refer to the value in the columns [Volume USDT/BTC] instead of [‘close’]
- Function then prints out a statement indicating volume in the specified units.
- Function will also be built upon to account for any user inputs that do not match either of the strings



```
if volume_input == "BTC":
    user_date_vol = float(
        df.loc[(df['date'] == volume_input_date)]['Volume BTC'])
    print(
        f"The volume of Bitcoin traded on {volume_input_date} was {int(user_date_vol)} BTC")
elif volume_input == "USDT":
    user_date_vol = float(
        df.loc[(df['date'] == volume_input_date)]['Volume USDT'])
    print(
        f"The trading volume of Bitcoin in USDT on {volume_input_date} is {int(user_date_vol)} USDT")

return
```

Development Process

- Initial idea involved linking program to an API to generate real-time data
    - Final decision was made to use static data to improve speed and efficiency of locating datapoints and minimise any issues pertaining to connectivity and delay in retrieving real-time data
- Pandas library was selected to readily analyse CSV files with pd.read_csv
- Initial coding began with functions defined for the welcome message and menu

Development Process (cont.)

- Once menu was tested to be working, further implementations were taken including loops to control flow and execution of menu code
- Testing of code to display CSV into Python terminal began
- Once dataframe was formatted and presented correctly, implementation of code to specify the row and column of data began
- Eventually after many hours of testing and using df.loc , specifying data points based on user input worked correctly.

Development Process (cont.)

- Coding started on the various functions that served to match user input with corresponding date and column
- Further data manipulations were done, especially with price comparison function

Beta Testing:

- Testing profit calculator function
    - Outcome: Provide user a view of what their profit would be at the date they selected based on their overall BTC amount inputted

Development Process (cont.)

- Program will eventually include the following:
  - Error Handling tools
  - Scripts to run the program and check for installation errors
  - Implementation plan in action

Error testing

- Preliminary error testing has begun on each segment of the source code based on the philosophy:
    - What if the user input is not what is expected?
- User input and function logic are primary targets of the tests
    - Testing report to be included at a later date
- Try/Except blocks to be included

Challenges/Issues

- pd.read_csv presented a major hurdle in this program
  - Understanding how to work with that command through Pandas documentation and the output of what was generated from reading the csv was particularly difficult
- Df.loc was another challenge as specifically the understanding of the command and how to refer to the specific column
- Monitoring the code to ensure DRY principles were adhered to
  - Involved analysing the code to ensure non-repetition of code
- Connecting the program to an API so current date will be updated for most accurate data

Ethical considerations and highlights

- Data being used is from https://www.cryptodatadownload.com/data/kucoin/
- Critical milestones of the program are:
  1. Successful read of CSV data into desired format
  2. Successful location of information pertaining to the user's specified date

Submitting this assignment at 11:59pm Sunday

Credits and Attribution

- Oliver, Iryna, Jairo and other educators for their contribution and insight into the source code
- Attribution for the free background image provided below
  - `<a href='https://pngtree.com/free-backgrounds'>free background photos from pngtree.com/</a>`
- Attribution for the CSV dataset: https://www.cryptodatadownload.com/data/kucoin/
-

Thanks!!