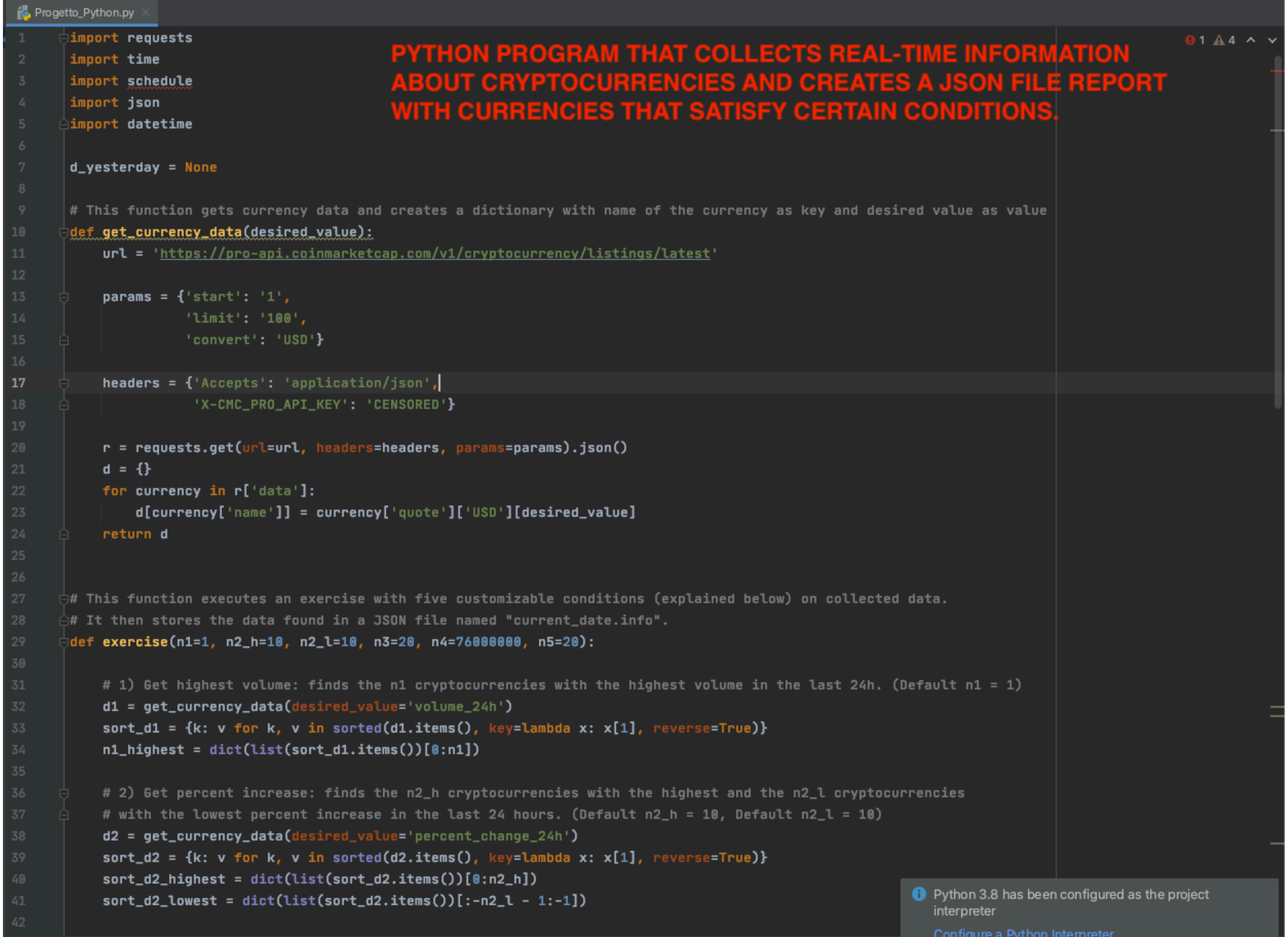
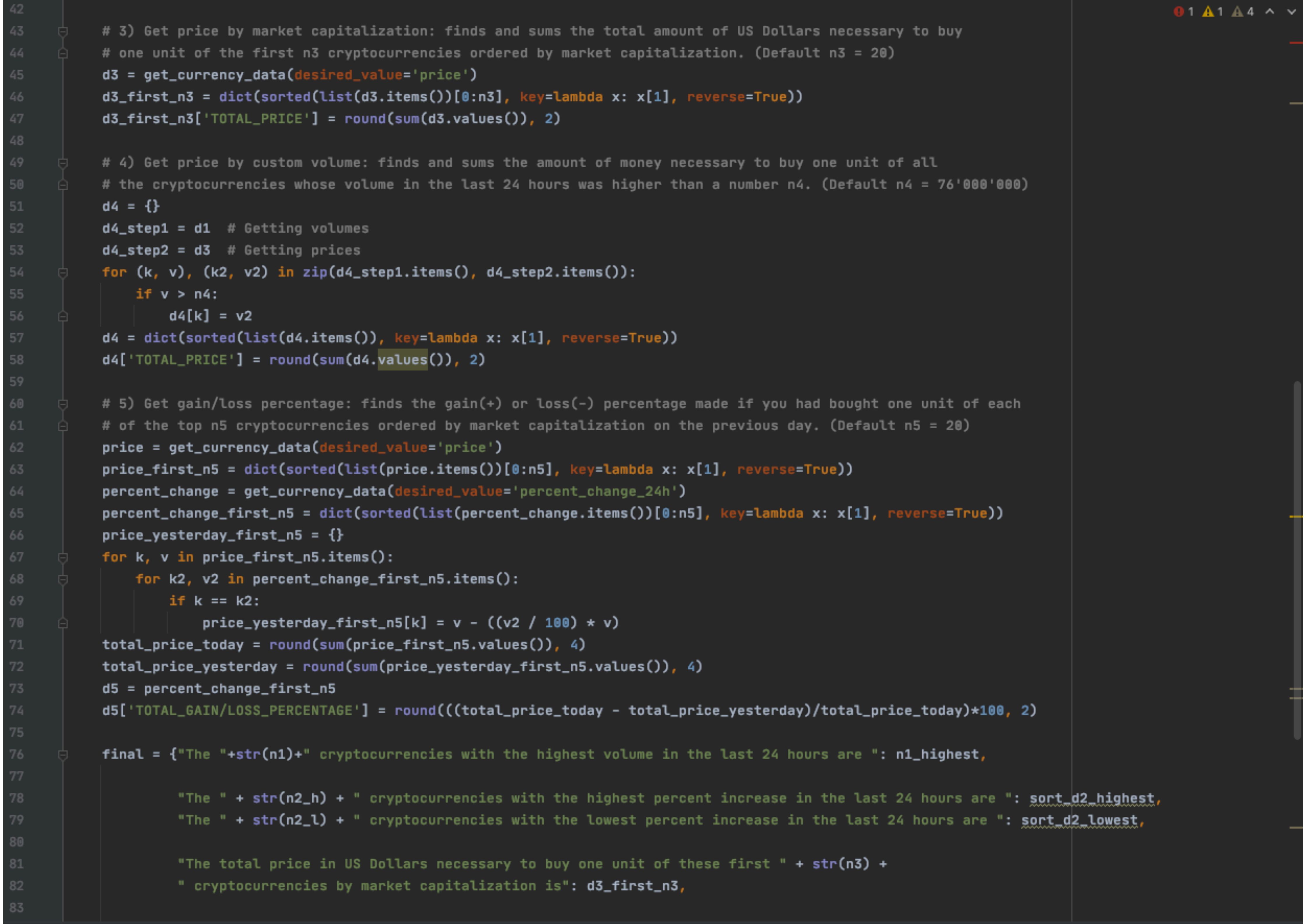


PYTHON PROGRAM THAT COLLECTS REAL-TIME INFORMATION ABOUT CRYPTOCURRENCIES AND CREATES A JSON FILE REPORT WITH CURRENCIES THAT SATISFY CERTAIN CONDITIONS.

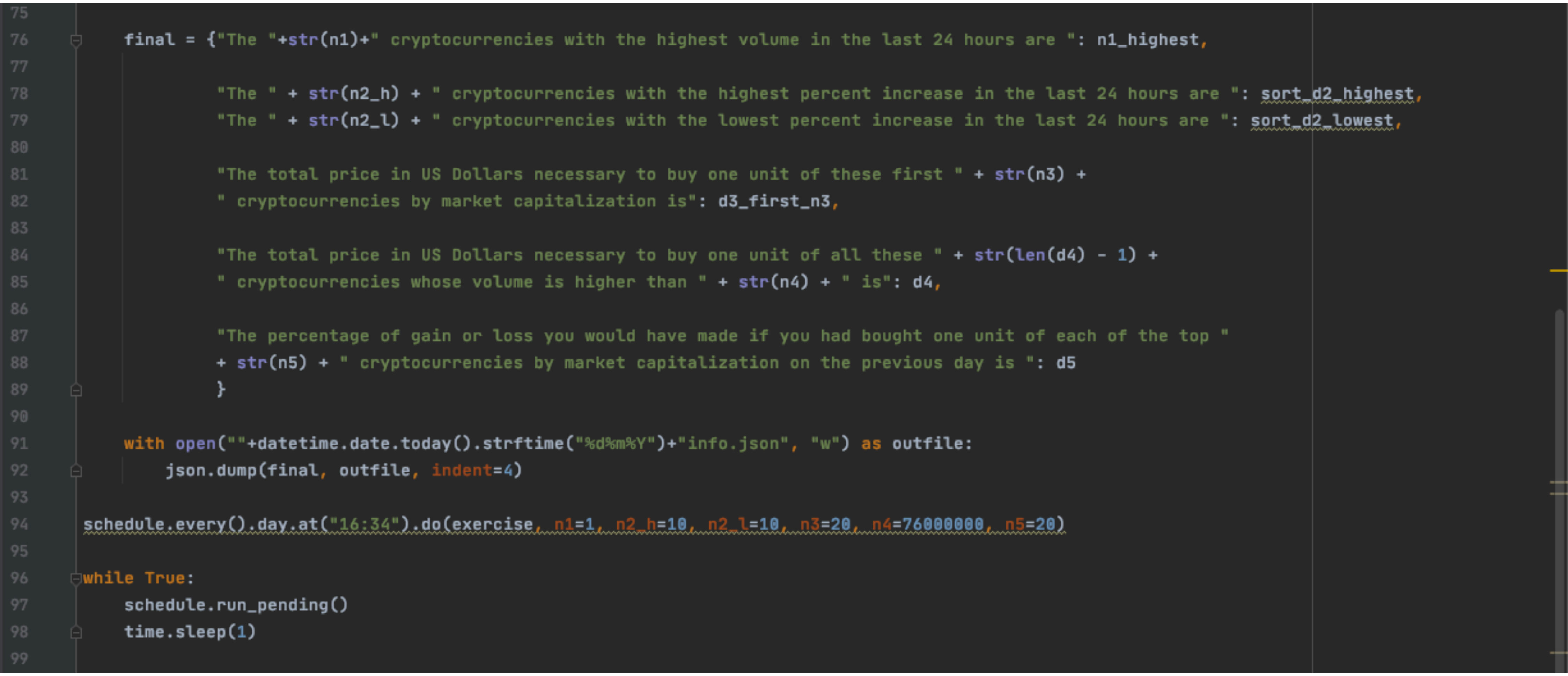
```python
import requests
import time
import schedule
import json
import datetime

d_yesterday = None

# This function gets currency data and creates a dictionary with name of the currency as key and desired value as value
def get_currency_data(desired_value):
    url = 'https://pro-api.coinmarketcap.com/v1/cryptocurrency/listings/latest'

    params = {'start': '1',
              'limit': '100',
              'convert': 'USD'}

    headers = {'Accepts': 'application/json',
               'X-CMC_PRO_API_KEY': 'CENSORED'}

    r = requests.get(url=url, headers=headers, params=params).json()
    d = {}
    for currency in r['data']:
        d[currency['name']] = currency['quote']['USD'][desired_value]
    return d


# This function executes an exercise with five customizable conditions (explained below) on collected data.
# It then stores the data found in a JSON file named "current_date.info".
def exercise(n1=1, n2_h=10, n2_l=10, n3=20, n4=76000000, n5=20):

    # 1) Get highest volume: finds the n1 cryptocurrencies with the highest volume in the last 24h. (Default n1 = 1)
    d1 = get_currency_data(desired_value='volume_24h')
    sort_d1 = {k: v for k, v in sorted(d1.items(), key=lambda x: x[1], reverse=True)}
    n1_highest = dict(list(sort_d1.items())[0:n1])

    # 2) Get percent increase: finds the n2_h cryptocurrencies with the highest and the n2_l cryptocurrencies
    # with the lowest percent increase in the last 24 hours. (Default n2_h = 10, Default n2_l = 10)
    d2 = get_currency_data(desired_value='percent_change_24h')
    sort_d2 = {k: v for k, v in sorted(d2.items(), key=lambda x: x[1], reverse=True)}
    sort_d2_highest = dict(list(sort_d2.items())[0:n2_h])
    sort_d2_lowest = dict(list(sort_d2.items())[:-n2_l - 1:-1])

```

Python 3.8 has been configured as the project interpreter

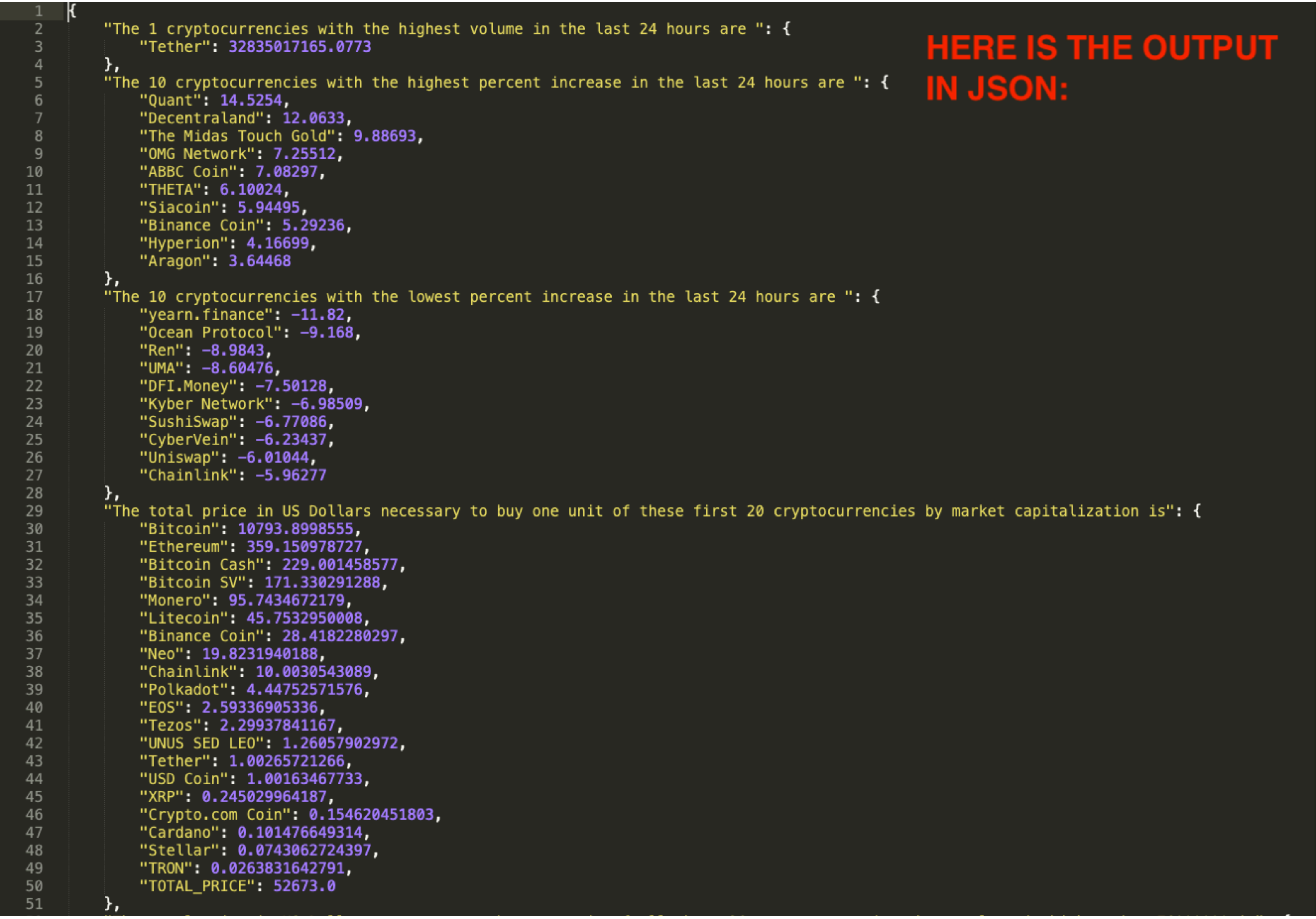
Configure a Python Interpreter



```python
    # 3) Get price by market capitalization: finds and sums the total amount of US Dollars necessary to buy
    # one unit of the first n3 cryptocurrencies ordered by market capitalization. (Default n3 = 20)
    d3 = get_currency_data(desired_value='price')
    d3_first_n3 = dict(sorted(list(d3.items())[0:n3], key=lambda x: x[1], reverse=True))
    d3_first_n3['TOTAL_PRICE'] = round(sum(d3.values()), 2)

    # 4) Get price by custom volume: finds and sums the amount of money necessary to buy one unit of all
    # the cryptocurrencies whose volume in the last 24 hours was higher than a number n4. (Default n4 = 76'000'000)
    d4 = {}
    d4_step1 = d1  # Getting volumes
    d4_step2 = d3  # Getting prices
    for (k, v), (k2, v2) in zip(d4_step1.items(), d4_step2.items()):
        if v > n4:
            d4[k] = v2
    d4 = dict(sorted(list(d4.items()), key=lambda x: x[1], reverse=True))
    d4['TOTAL_PRICE'] = round(sum(d4.values()), 2)

    # 5) Get gain/loss percentage: finds the gain(+) or loss(-) percentage made if you had bought one unit of each
    # of the top n5 cryptocurrencies ordered by market capitalization on the previous day. (Default n5 = 20)
    price = get_currency_data(desired_value='price')
    price_first_n5 = dict(sorted(list(price.items())[0:n5], key=lambda x: x[1], reverse=True))
    percent_change = get_currency_data(desired_value='percent_change_24h')
    percent_change_first_n5 = dict(sorted(list(percent_change.items())[0:n5], key=lambda x: x[1], reverse=True))
    price_yesterday_first_n5 = {}
    for k, v in price_first_n5.items():
        for k2, v2 in percent_change_first_n5.items():
            if k == k2:
                price_yesterday_first_n5[k] = v - ((v2 / 100) * v)
    total_price_today = round(sum(price_first_n5.values()), 4)
    total_price_yesterday = round(sum(price_yesterday_first_n5.values()), 4)
    d5 = percent_change_first_n5
    d5['TOTAL_GAIN/LOSS_PERCENTAGE'] = round(((total_price_today - total_price_yesterday)/total_price_today)*100, 2)

    final = {"The "+str(n1)+" cryptocurrencies with the highest volume in the last 24 hours are ": n1_highest,

             "The " + str(n2_h) + " cryptocurrencies with the highest percent increase in the last 24 hours are ": sort_d2_highest,
             "The " + str(n2_l) + " cryptocurrencies with the lowest percent increase in the last 24 hours are ": sort_d2_lowest,

             "The total price in US Dollars necessary to buy one unit of these first " + str(n3) +
             " cryptocurrencies by market capitalization is": d3_first_n3,

```

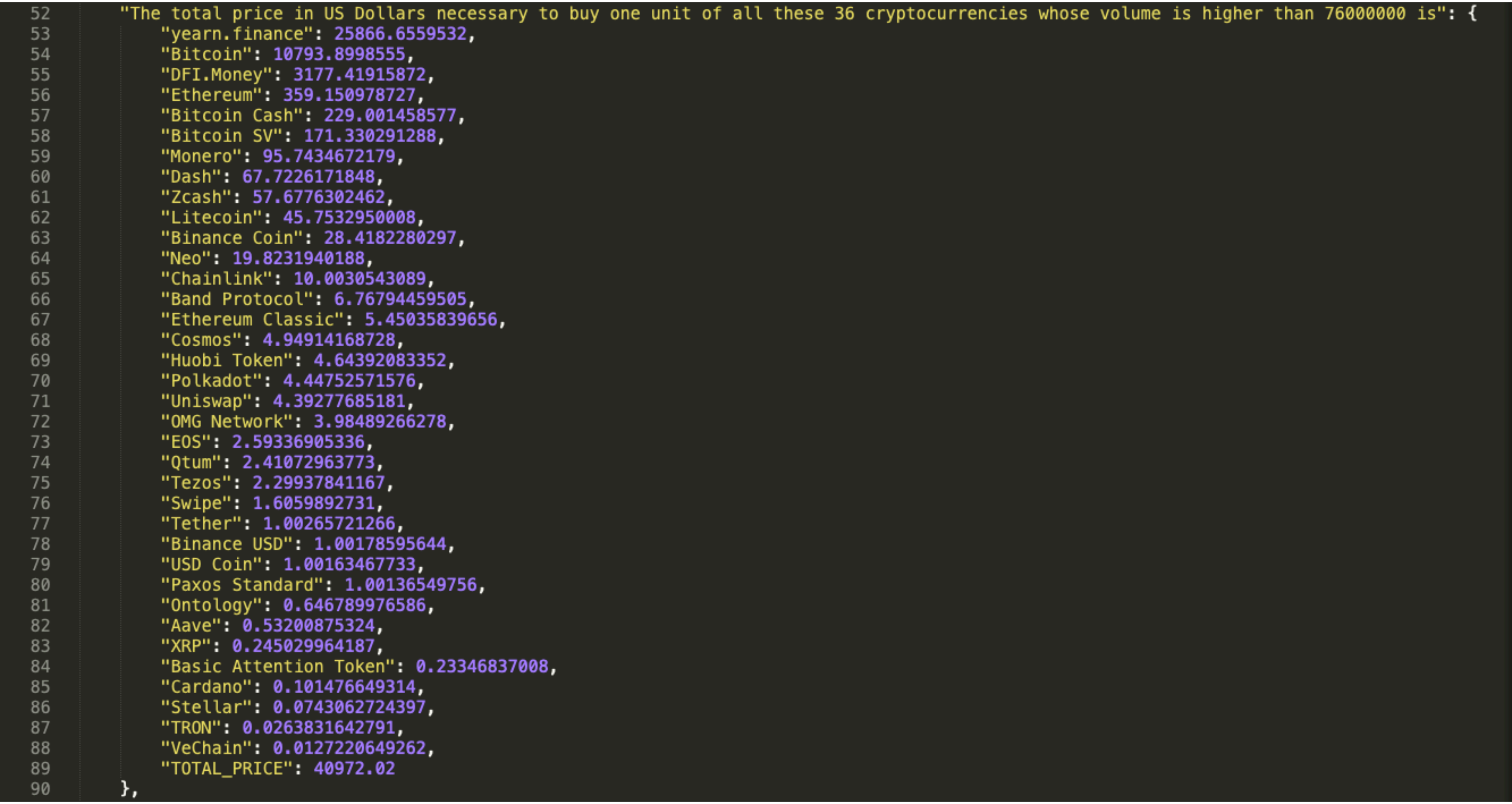


```python

    final = {"The "+str(n1)+" cryptocurrencies with the highest volume in the last 24 hours are ": n1_highest,

             "The " + str(n2_h) + " cryptocurrencies with the highest percent increase in the last 24 hours are ": sort_d2_highest,
             "The " + str(n2_l) + " cryptocurrencies with the lowest percent increase in the last 24 hours are ": sort_d2_lowest,

             "The total price in US Dollars necessary to buy one unit of these first " + str(n3) +
             " cryptocurrencies by market capitalization is": d3_first_n3,

             "The total price in US Dollars necessary to buy one unit of all these " + str(len(d4) - 1) +
             " cryptocurrencies whose volume is higher than " + str(n4) + " is": d4,

             "The percentage of gain or loss you would have made if you had bought one unit of each of the top "
             + str(n5) + " cryptocurrencies by market capitalization on the previous day is ": d5
             }

    with open(""+datetime.date.today().strftime("%d%m%Y")+"info.json", "w") as outfile:
        json.dump(final, outfile, indent=4)

schedule.every().day.at("16:34").do(exercise, n1=1, n2_h=10, n2_l=10, n3=20, n4=76000000, n5=20)

while True:
    schedule.run_pending()
    time.sleep(1)

```

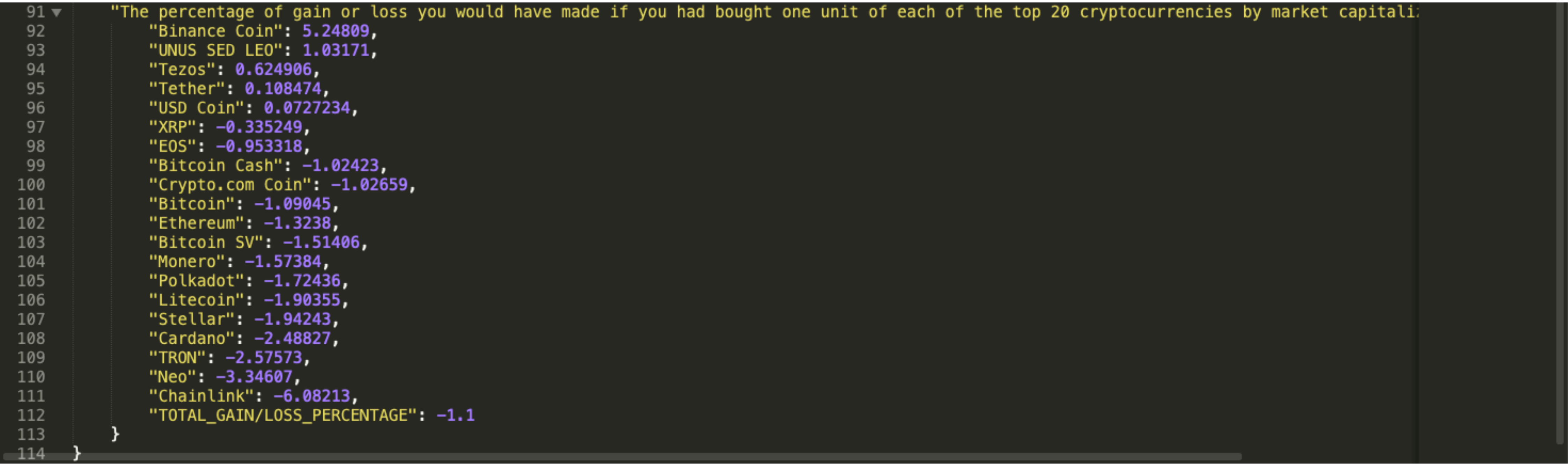


HERE IS THE OUTPUT IN JSON:

```
{
    "The 1 cryptocurrencies with the highest volume in the last 24 hours are ": {
        "Tether": 32835017165.0773
    },
    "The 10 cryptocurrencies with the highest percent increase in the last 24 hours are ": {
        "Quant": 14.5254,
        "Decentraland": 12.0633,
        "The Midas Touch Gold": 9.88693,
        "OMG Network": 7.25512,
        "ABBC Coin": 7.08297,
        "THETA": 6.10024,
        "Siacoin": 5.94495,
        "Binance Coin": 5.29236,
        "Hyperion": 4.16699,
        "Aragon": 3.64468
    },
    "The 10 cryptocurrencies with the lowest percent increase in the last 24 hours are ": {
        "yearn.finance": -11.82,
        "Ocean Protocol": -9.168,
        "Ren": -8.9843,
        "UMA": -8.60476,
        "DFI.Money": -7.50128,
        "Kyber Network": -6.98509,
        "SushiSwap": -6.77086,
        "CyberVein": -6.23437,
        "Uniswap": -6.01044,
        "Chainlink": -5.96277
    },
    "The total price in US Dollars necessary to buy one unit of these first 20 cryptocurrencies by market capitalization is": {
        "Bitcoin": 10793.8998555,
        "Ethereum": 359.150978727,
        "Bitcoin Cash": 229.001458577,
        "Bitcoin SV": 171.330291288,
        "Monero": 95.7434672179,
        "Litecoin": 45.7532950008,
        "Binance Coin": 28.4182280297,
        "Neo": 19.8231940188,
        "Chainlink": 10.0030543089,
        "Polkadot": 4.44752571576,
        "EOS": 2.59336905336,
        "Tezos": 2.29937841167,
        "UNUS SED LEO": 1.26057902972,
        "Tether": 1.00265721266,
        "USD Coin": 1.00163467733,
        "XRP": 0.245029964187,
        "Crypto.com Coin": 0.154620451803,
        "Cardano": 0.101476649314,
        "Stellar": 0.0743062724397,
        "TRON": 0.0263831642791,
        "TOTAL_PRICE": 52673.0
    },
```



```
    "The total price in US Dollars necessary to buy one unit of all these 36 cryptocurrencies whose volume is higher than 76000000 is": {
        "yearn.finance": 25866.6559532,
        "Bitcoin": 10793.8998555,
        "DFI.Money": 3177.41915872,
        "Ethereum": 359.150978727,
        "Bitcoin Cash": 229.001458577,
        "Bitcoin SV": 171.330291288,
        "Monero": 95.7434672179,
        "Dash": 67.7226171848,
        "Zcash": 57.6776302462,
        "Litecoin": 45.7532950008,
        "Binance Coin": 28.4182280297,
        "Neo": 19.8231940188,
        "Chainlink": 10.0030543089,
        "Band Protocol": 6.76794459505,
        "Ethereum Classic": 5.45035839656,
        "Cosmos": 4.94914168728,
        "Huobi Token": 4.64392083352,
        "Polkadot": 4.44752571576,
        "Uniswap": 4.39277685181,
        "OMG Network": 3.98489266278,
        "EOS": 2.59336905336,
        "Qtum": 2.41072963773,
        "Tezos": 2.29937841167,
        "Swipe": 1.6059892731,
        "Tether": 1.00265721266,
        "Binance USD": 1.00178595644,
        "USD Coin": 1.00163467733,
        "Paxos Standard": 1.00136549756,
        "Ontology": 0.646789976586,
        "Aave": 0.53200875324,
        "XRP": 0.245029964187,
        "Basic Attention Token": 0.23346837008,
        "Cardano": 0.101476649314,
        "Stellar": 0.0743062724397,
        "TRON": 0.0263831642791,
        "VeChain": 0.0127220649262,
        "TOTAL_PRICE": 40972.02
    },
```



```
    "The percentage of gain or loss you would have made if you had bought one unit of each of the top 20 cryptocurrencies by market capitali
        "Binance Coin": 5.24809,
        "UNUS SED LEO": 1.03171,
        "Tezos": 0.624906,
        "Tether": 0.108474,
        "USD Coin": 0.0727234,
        "XRP": -0.335249,
        "EOS": -0.953318,
        "Bitcoin Cash": -1.02423,
        "Crypto.com Coin": -1.02659,
        "Bitcoin": -1.09045,
        "Ethereum": -1.3238,
        "Bitcoin SV": -1.51406,
        "Monero": -1.57384,
        "Polkadot": -1.72436,
        "Litecoin": -1.90355,
        "Stellar": -1.94243,
        "Cardano": -2.48827,
        "TRON": -2.57573,
        "Neo": -3.34607,
        "Chainlink": -6.08213,
        "TOTAL_GAIN/LOSS_PERCENTAGE": -1.1
    }
}
```