Angéline Texier

TD9 - Automating trading

Q2 : Get a list of all available cryptocurrencies and display it

Entrée [30]:

```
import json
import requests
```

Entrée [31]:

```
link1 = requests.get('https://api.binance.com/api/v3/exchangeInfo')
list_exchangeInfo = json.loads(link1.text)

for i in range(len(list_exchangeInfo['symbols'])):
    print ("Symbol: " + list_exchangeInfo['symbols'][i]['symbol'] + " - Sign: " + list_exch
```

```
Symbol: ETHBTC - Sign: ETH
Symbol: LTCBTC - Sign: LTC
Symbol: BNBBTC - Sign: BNB
Symbol: NEOBTC - Sign: NEO
Symbol: QTUMETH - Sign: QTUM
Symbol: EOSETH - Sign: EOS
Symbol: SNTETH - Sign: SNT
Symbol: BNTETH - Sign: BNT
Symbol: BCCBTC - Sign: BCC
Symbol: GASBTC - Sign: GAS
Symbol: BNBETH - Sign: BNB
Symbol: BTCUSDT - Sign: BTC
Symbol: ETHUSDT - Sign: ETH
Symbol: HSRBTC - Sign: HSR
Symbol: OAXETH - Sign: OAX
Symbol: DNTETH - Sign: DNT
Symbol: MCOETH - Sign: MCO
Symbol: ICNETH - Sign: ICN
Symbol: MCOBTC - Sign: MCO
Symbol: WTCBTC - Sign: WTC
```

Q3 : Create a function to display the ’ask’ or ‘bid’ price of an asset.

Entrée [39]:

```
def getDepth(direction, pair):

    link2 = requests.get('https://api.binance.com/api/v3/ticker/bookTicker')
    list_bookTicker = json.loads(link2.text)

    result = ''

    if(direction == 'bid'):
        for i in range(len(list_bookTicker)):
            if (list_bookTicker[i]['symbol'] == pair):
                result = list_bookTicker[i]['bidPrice']

    elif(direction == 'ask'):
        for i in range(len(list_bookTicker)):
            if (list_bookTicker[i]['symbol'] == pair):
                result = list_bookTicker[i]['askPrice']

    else:
        result = 'no direction specified'

    return result

print(getDepth(direction = 'bid', pair = 'ETHBTC'))
print(getDepth(direction = 'ask', pair = 'ETHBTC'))
print(getDepth(direction = 'bid', pair = 'LTCBTC'))
print(getDepth(direction = 'ask', pair = 'LTCBTC'))
```

```
0.01933100
0.01933200
0.00656100
0.00656300
```

Q4 : Get order book for an asset

Entrée [43]:

```
print("Asset: ETHBTC" + " - bid: " + getDepth(direction = 'bid', pair = 'ETHBTC'))
print("Asset: ETHBTC" + " - ask: " + getDepth(direction = 'ask', pair = 'ETHBTC'))
```

```
Asset: ETHBTC - bid: 0.01933400
Asset: ETHBTC - ask: 0.01934400
```

Q5 : Create a function to read agregated trading data (candles)

Entrée [44]:

```
def refreshDataCandle(pair, duration):

    link3 = requests.get('https://api.binance.com/api/v3/klines')
    list_candles = json.loads(link3.text)

    duration = 0

    for i in range(len(list_candles)):
        duration = list_candles[i]['endTime'] + list_candles[i]['startTime']

# Quote asset volume
# Number of trades
# Taker buy base asset volume
# Taker buy quote asset volume

print(refreshDataCandle(pair = 'ETHBTC', duration = '5m’))
```

```
  File "<ipython-input-44-e0ecb8c851a4>", line 16
    print(refreshDataCandle(pair = 'ETHBTC', duration = '5m’))
                                                              ^
SyntaxError: EOL while scanning string literal
```

Entrée [ ]: