Python Programming (Basic-Intermediate)

Module 2 - Conditional and Control Flow Statements

Conditional Statements

```
import requests
resp = requests.get('https://api.coingecko.com/api/v3/coins/markets/?vs_
crypto_data = resp.json()
print(crypto_data)
```

```
[{'id': 'bitcoin', 'symbol': 'btc', 'name': 'Bitcoin', 'image': 'https
```

```
btc = crypto_data[0]
```

```
btc
```

```
{'id': 'bitcoin',
 'symbol': 'btc',
 'name': 'Bitcoin',
 'image': 'https://assets.coingecko.com/coins/images/1/large/bitcoin.
 'current_price': 42332,
 'market_cap': 830344658724,
 'market_cap_rank': 1,
 'fully_diluted_valuation': 889142101664,
 'total_volume': 14037645850,
 'high_24h': 42743,
 'low_24h': 41750,
 'price_change_24h': 473.5,
 'price_change_percentage_24h': 1.13121,
 'market_cap_change_24h': 10130947987,
 'market_cap_change_percentage_24h': 1.23516,
 'circulating_supply': 19611306.0,
 'total_supply': 21000000.0,
 'max_supply': 21000000.0,
 'ath': 69045,
```

```
btc['symbol'] == 'btc'
```

```
True
```

```
btc['id'] == 'ethereum'
```

```
False
```

```
btc['name']
```

```
'Bitcoin'
```

```
btc['name'] == 'bitcoin'
```

```
False
```

```
btc['name'].lower() == 'bitcoin'
```

```
True
```

```
btc['current_price'] < 20000
```

```
False
```

```
btc['ath'] <= 69045
```

```
True
```

```
(btc['current_price'] < 20000) and (btc['ath'] <= 69045)
```

```
False
```

```
btc['current_price'] > 20000 and btc['ath'] <= 69045
```

```
True
```

```
btc['current_price'] < 20000 or btc['ath'] <= 69045
```

```
True
```

if, else and elif

```
x = 10
y = 5

if x > y:
  print("x is greater than y")
  print(x)
```

```
x is greater than y
10
```

```
x = []
if x:
  print("x is greater than y")
```

```
y = 25
x = 10

if x > y:
    print("x is greater than y")
else:
    print("y is greater than x")
    print(x)

print(y)
```

```
y is greater than x
10
25
```

```
x = 10
y = 10

if x > y:
  print("x is greater than y")
elif x == y:
  print("x and y are equal!")
else:
  print("y is greater than x")
```

```
x and y are equal!
```

```
if btc['current_price'] < 10000:
  print('Bitcoin price is less than 10k USD')
elif btc['current_price'] < 20000:
  print('Bitcoin price is between 10k - 20k USD')
elif btc['current_price'] < 40000:
  print('Bitcoin price is between 20k - 40k USD')
else:
  print('Bitcoin price is more than 40k USD... but wen?')
```

```
Bitcoin price is more than 40k USD... but wen?
```

```
base_price = 50 # ข้าวผัดกะเพราไก่
requested_toppings = ['fried egg', 'extra rice']

total_price = base_price

if 'fried egg' in requested_toppings:
  print('Adding a fried egg')
  total_price += 10
if 'extra chicken' in requested_toppings:
  print('Adding extra chicken')
  total_price += 20
if 'extra rice' in requested_toppings:
  print('Adding extra rice')
  total_price += 10

print(f'Total price is {total_price}')
```

```
Adding a fried egg
Adding extra rice
Total price is 70
```

```
requested_toppings = []

if requested_toppings:
  if 'fried egg' in requested_toppings:
    print('Adding a fried egg')
  if 'extra chicken' in requested_toppings:
    print('Adding extra chicken')
  if 'extra rice' in requested_toppings:
    print('Adding extra rice')
else:
  print('It is a plain Kao Pad Kra Pow')
```

```
It is a plain Kao Pad Kra Pow
```

for

```
my_sequence = list(range(0,101,10))
#my_sequence = [0, 10, [20], [30, 40]]

for number in my_sequence:
  print(number)
```

```
0
10
20
30
40
50
60
70
80
90
100
```

```
my_sequence
```

```
[0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
```

```
for number in my_sequence:
  if number < 80:
    continue
  print(number)
```

```
80
90
100
```

```
for number in my_sequence:
  if number > 50:
    break
  print(number)
```

```
0
10
20
30
40
50
```

```
crypto_data[:10]
```

```
[{'id': 'bitcoin',
  'symbol': 'btc',
  'name': 'Bitcoin',
  'image': 'https://assets.coingecko.com/coins/images/1/large/bitcoir
  'current_price': 42332,
  'market_cap': 830344658724,
  'market_cap_rank': 1,
  'fully_diluted_valuation': 889142101664,
  'total_volume': 14037645850,
  'high_24h': 42743,
  'low_24h': 41750,
  'price_change_24h': 473.5,
  'price_change_percentage_24h': 1.13121,
  'market_cap_change_24h': 10130947987,
  'market_cap_change_percentage_24h': 1.23516,
  'circulating_supply': 19611306.0,
  'total_supply': 21000000.0,
  'max_supply': 21000000.0,
  'ath': 69045,
```

```
for coin in crypto_data[:10]:
  print(coin['symbol'] + "\tUSD " + str(coin['current_price']))
```

```
btc     USD 42332
eth     USD 2270.89
usdt    USD 0.99967
bnb     USD 306.03
sol     USD 97.49
xrp     USD 0.529647
```

```
usdc    USD 0.998775
steth   USD 2269.98
ada     USD 0.491594
avax    USD 35.79
```

```
btc
```

```
{'id': 'bitcoin',
 'symbol': 'btc',
 'name': 'Bitcoin',
 'image': 'https://assets.coingecko.com/coins/images/1/large/bitcoin.
 'current_price': 42332,
 'market_cap': 830344658724,
 'market_cap_rank': 1,
 'fully_diluted_valuation': 889142101664,
 'total_volume': 14037645850,
 'high_24h': 42743,
 'low_24h': 41750,
 'price_change_24h': 473.5,
 'price_change_percentage_24h': 1.13121,
 'market_cap_change_24h': 10130947987,
 'market_cap_change_percentage_24h': 1.23516,
 'circulating_supply': 19611306.0,
 'total_supply': 21000000.0,
 'max_supply': 21000000.0,
 'ath': 69045,
```

```
for t in btc:
    print(t)
```

```
id
symbol
name
image
current_price
market_cap
market_cap_rank
fully_diluted_valuation
total_volume
high_24h
low_24h
price_change_24h
price_change_percentage_24h
market_cap_change_24h
market_cap_change_percentage_24h
circulating_supply
total_supply
max_supply
ath
ath_change_percentage
```

```
for t in btc.items():
    print(t)
```

```
('id', 'bitcoin')
('symbol', 'btc')
('name', 'Bitcoin')
('image', 'https://assets.coingecko.com/coins/images/1/large/bitcoin.
('current_price', 42332)
('market_cap', 830344658724)
('market_cap_rank', 1)
('fully_diluted_valuation', 889142101664)
('total_volume', 14037645850)
('high_24h', 42743)
('low_24h', 41750)
('price_change_24h', 473.5)
('price_change_percentage_24h', 1.13121)
('market_cap_change_24h', 10130947987)
('market_cap_change_percentage_24h', 1.23516)
('circulating_supply', 19611306.0)
('total_supply', 21000000.0)
('max_supply', 21000000.0)
('ath', 69045)
```

```
for key, value in btc.items():
  print(key,'\t',value)
```

```
id       bitcoin
symbol   btc
name     Bitcoin
image    https://assets.coingecko.com/coins/images/1/large/bitcoin.pr
current_price    42332
market_cap       830344658724
market_cap_rank          1
fully_diluted_valuation          889142101664
total_volume     14037645850
high_24h         42743
low_24h          41750
price_change_24h         473.5
price_change_percentage_24h      1.13121
market_cap_change_24h    10130947987
market_cap_change_percentage_24h         1.23516
circulating_supply       19611306.0
total_supply     21000000.0
max_supply       21000000.0
ath      69045
```

```
for idx, coin in enumerate(crypto_data[:10]):
  print(idx+1, " ", coin['name'])
  for key, value in coin.items():
    print("\t", key, "\t", value)
```

```
1   Bitcoin
        id      bitcoin
        symbol          btc
        name    Bitcoin
        image   https://assets.coingecko.com/coins/images/1/large/bi
        current_price   42332
        market_cap      830344658724
        market_cap_rank         1
        fully_diluted_valuation         889142101664
        total_volume    14037645850
        high_24h        42743
        low_24h         41750
        price_change_24h        473.5
        price_change_percentage_24h     1.13121
        market_cap_change_24h   10130947987
        market_cap_change_percentage_24h        1.23516
        circulating_supply      19611306.0
        total_supply    21000000.0
        max_supply      21000000.0
```

```
from tqdm.notebook import tqdm
import time
for i in tqdm(range(10)):
    time.sleep(1) # So Cool!!!
```

```
  0%|          | 0/10 [00:00<?, ?it/s]
```

while

```
count = 0
while count < 8:
  print(crypto_data[count]['id'])
  count += 1
```

```
bitcoin
ethereum
tether
binancecoin
solana
ripple
usd-coin
staked-ether
```

```
x = 0
while True:
    x += 1
```

```
count = 0
while True:
  if (crypto_data[count]['market_cap_rank']  > 10):
    break
  else:
    print(crypto_data[count]['market_cap_rank'],
          "\t",
          crypto_data[count]['id'])
    count += 1
```

```
1       bitcoin
2       ethereum
3       tether
4       binancecoin
5       solana
6       ripple
7       usd-coin
8       staked-ether
9       cardano
10      avalanche-2
```

```
print("Top 10 Cryptocurrencies by Marketcap (excluding Stable Coins")
count = 0
while True:
  if (crypto_data[count]['market_cap_rank'] > 10):
    break
  else:
    if abs(crypto_data[count]['current_price'] - 1) < 0.1 and \
       abs(crypto_data[count]['ath'] - crypto_data[count]['atl']) < 2:
       # stable coin
       count += 1
       continue
    else:
       print("rank ",
             crypto_data[count]['market_cap_rank'],"\t",
             crypto_data[count]['symbol'],"\t",
             crypto_data[count]['current_price'])
       count += 1
```

```
Top 10 Cryptocurrencies by Marketcap (excluding Stable Coins
rank  1         btc     42332
rank  2         eth     2270.89
rank  4         bnb     306.03
rank  5         sol     97.49
rank  6         xrp     0.529647
rank  8         steth   2269.98
rank  9         ada     0.491594
rank  10        avax    35.79
```

```
r = range(10)
list(r)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
x = enumerate(crypto_data)
```

```
next(x)
```

```
(0,
 {'id': 'bitcoin',
  'symbol': 'btc',
  'name': 'Bitcoin',
  'image': 'https://assets.coingecko.com/coins/images/1/large/bitcoir
  'current_price': 42332,
  'market_cap': 830344658724,
  'market_cap_rank': 1,
  'fully_diluted_valuation': 889142101664,
  'total_volume': 14037645850,
  'high_24h': 42743,
  'low_24h': 41750,
  'price_change_24h': 473.5,
  'price_change_percentage_24h': 1.13121,
  'market_cap_change_24h': 10130947987,
  'market_cap_change_percentage_24h': 1.23516,
  'circulating_supply': 19611306.0,
  'total_supply': 21000000.0,
  'max_supply': 21000000.0,
```

```
next(x)
```

```
(1,
 {'id': 'ethereum',
  'symbol': 'eth',
  'name': 'Ethereum',
  'image': 'https://assets.coingecko.com/coins/images/279/large/ether
  'current_price': 2270.89,
  'market_cap': 272992534534,
  'market_cap_rank': 2,
  'fully_diluted_valuation': 272992534534,
  'total_volume': 6411322540,
  'high_24h': 2302.02,
  'low_24h': 2263.35,
  'price_change_24h': -1.8986871904198779,
  'price_change_percentage_24h': -0.08354,
  'market_cap_change_24h': -2728507.4877929688,
```

```
  'market_cap_change_percentage_24h': -0.001,
  'circulating_supply': 120181248.371207,
  'total_supply': 120181248.371207,
  'max_supply': None
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
f = open('/content/drive/MyDrive/AIS_DG/gpl-3.0.txt')
lines = iter(f.readlines())
print(type(iter(lines)))
f.close()
```

```
<class 'list_iterator'>
```

```
next(lines)
```

```
'                    GNU GENERAL PUBLIC LICENSE\n'
```

```
next(lines)
```

```
'                       Version 3, 29 June 2007\n'
```

```
import pandas as pd
```

```
df = pd.read_excel('/content/drive/MyDrive/AIS_DG/Superstore.xlsx')
```

```
df.groupby('Customer ID').agg({'Sales':'sum'})
```

```
                Sales
Customer ID
AA-10315     5563.560
AA-10375     1056.390
AA-10480     1790.512
AA-10645     5086.935
AB-10015      886.156
...               ...
XP-21865     2374.658
YC-21895     5454.350
```

```
YS-21880     6720.444
ZC-21910     8025.707
ZD-21925     1493.944

[793 rows x 1 columns]
```

	Sales
Customer ID	
AA-10315	5563.560
AA-10375	1056.390
AA-10480	1790.512
AA-10645	5086.935
AB-10015	886.156
...	...
XP-21865	2374.658
YC-21895	5454.350
YS-21880	6720.444
ZC-21910	8025.707
ZD-21925	1493.944

793 rows × 1 columns

```
g = df.groupby('Customer ID')
```

```
type(g)
```

```
pandas.core.groupby.generic.DataFrameGroupBy
```

```
gt = iter(g)
```

```
next(gt)
```

```
('AA-10315',
       Row ID        Order ID Order Date  Ship Date       Ship Mode
 1159    1160  CA-2014-147039 2015-06-30 2015-07-05  Standard Class
 1160    1161  CA-2014-147039 2015-06-30 2015-07-05  Standard Class
 1299    1300  CA-2012-121391 2013-10-04 2013-10-07    First Class
 2229    2230  CA-2011-128055 2012-03-31 2012-04-05  Standard Class
 2230    2231  CA-2011-128055 2012-03-31 2012-04-05  Standard Class
```

```
5198    5199  CA-2013-103982 2014-03-04 2014-03-09  Standard Class
5199    5200  CA-2013-103982 2014-03-04 2014-03-09  Standard Class
5200    5201  CA-2013-103982 2014-03-04 2014-03-09  Standard Class
5201    5202  CA-2013-103982 2014-03-04 2014-03-09  Standard Class
7468    7469  CA-2011-138100 2012-09-15 2012-09-20  Standard Class
7469    7470  CA-2011-138100 2012-09-15 2012-09-20  Standard Class

     Customer ID Customer Name   Segment        Country           Ci
1159    AA-10315    Alex Avila  Consumer  United States    Minneapol
1160    AA-10315    Alex Avila  Consumer  United States    Minneapol
1299    AA-10315    Alex Avila  Consumer  United States  San Francis
2229    AA-10315    Alex Avila  Consumer  United States  San Francis
```

```
next(gt)
```

```
('AA-10375',
       Row ID        Order ID Order Date  Ship Date       Ship Mode
535       536  CA-2013-126613 2014-07-11 2014-07-17  Standard Class
807       808  CA-2012-140921 2013-02-03 2013-02-05    First Class
808       809  CA-2012-140921 2013-02-03 2013-02-05    First Class
1172     1173  CA-2011-158064 2012-04-21 2012-04-25  Standard Class
1978     1979  CA-2012-109939 2013-05-08 2013-05-12  Standard Class
2263     2264  CA-2013-131065 2014-11-15 2014-11-17   Second Class
2264     2265  CA-2013-131065 2014-11-15 2014-11-17   Second Class
2265     2266  CA-2013-131065 2014-11-15 2014-11-17   Second Class
3007     3008  CA-2011-130729 2012-10-24 2012-10-29  Standard Class
6465     6466  CA-2012-114503 2013-11-13 2013-11-17  Standard Class
6747     6748  CA-2014-100230 2015-12-12 2015-12-16  Standard Class
6748     6749  CA-2014-100230 2015-12-12 2015-12-16  Standard Class
6749     6750  CA-2014-100230 2015-12-12 2015-12-16  Standard Class
9538     9539  US-2014-169488 2015-09-08 2015-09-10    First Class
9539     9540  US-2014-169488 2015-09-08 2015-09-10    First Class

     Customer ID Customer Name   Segment        Country
535     AA-10375    Allen Arnold  Consumer  United States
```

```
g
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7bf730adf970
```

```
for name, data_chunk in g:
    print(name, data_chunk.shape)
```

```
AA-10315 (11, 21)
AA-10375 (15, 21)
AA-10480 (12, 21)
AA-10645 (18, 21)
AB-10015 (6, 21)
```

```
AB-10060 (18, 21)
AB-10105 (20, 21)
AB-10150 (12, 21)
AB-10165 (14, 21)
AB-10255 (14, 21)
AB-10600 (8, 21)
AC-10420 (5, 21)
AC-10450 (9, 21)
AC-10615 (18, 21)
AC-10660 (6, 21)
AD-10180 (12, 21)
AF-10870 (16, 21)
AF-10885 (7, 21)
AG-10270 (14, 21)
AG-10300 (5, 21)
```

```
import h5py
```

```
coin_list = [coin['id'] for coin in crypto_data]
coin_list
```

```
['bitcoin',
 'ethereum',
 'tether',
 'binancecoin',
 'solana',
 'ripple',
 'usd-coin',
 'staked-ether',
 'cardano',
 'avalanche-2',
 'dogecoin',
 'tron',
 'polkadot',
 'chainlink',
 'matic-network',
 'the-open-network',
 'wrapped-bitcoin',
 'internet-computer',
 'shiba-inu',
 'dai',
```

```
f = open('/content/drive/MyDrive/AIS_DG/gpl-3.0.txt')
data = f.readlines()
f.close()
```

```
data_txt = " ".join(data)
```

```
word_vector = data_txt.split(" ")
len(word_vector)
```

```
6509
```

```
[t for t in word_vector if t != '']
```

```
['GNU',
 'GENERAL',
 'PUBLIC',
 'LICENSE\n',
 'Version',
 '3,',
 '29',
 'June',
 '2007\n',
 '\n',
 'Copyright',
 '(C)',
 '2007',
 'Free',
 'Software',
 'Foundation,',
 'Inc.',
 '<https://fsf.org/>\n',
 'Everyone',
 'is',
```

```
top_10_coins_list = [coin['id'] for coin in crypto_data if coin['market_
top_10_coins_list
```

```
['bitcoin',
 'ethereum',
 'tether',
 'binancecoin',
 'solana',
 'ripple',
 'usd-coin',
 'staked-ether',
 'cardano',
 'avalanche-2']
```

```
non_stable_coins = [coin['symbol']
                    if abs(coin['current_price']-1) > 0.1
                    else "stable-coin: "+coin['symbol']
                    for coin in crypto_data]
non_stable_coins[:10]
```

```
['btc',
 'eth',
 'stable-coin: usdt',
 'bnb',
 'sol',
 'xrp',
 'stable-coin: usdc',
 'steth',
 'ada',
 'avax']
```

```
import numpy as np

x = np.random.uniform(size = 100000000)
```

```
x_list = list(x)
```

```
from tqdm.notebook import tqdm
count = 0
for v in tqdm(x_list):
    if v < 0.5:
        count += 1
```

```
  0%|          | 0/100000000 [00:00<?, ?it/s]
```

```
%%time
y = [v for v in x_list if v < 0.5]
len(y)
```

```
CPU times: user 6.49 s, sys: 405 ms, total: 6.89 s
Wall time: 6.92 s


50001776
```

```
x.shape
```

```
(100000000,)
```

```
%%time
import pandas as pd
x_s = pd.Series(x)
(x_s < 0.5).sum()
```

```
CPU times: user 1.37 s, sys: 77.7 ms, total: 1.45 s
Wall time: 1.29 s


50001776
```

Activity

Write an API crawler to build a DataFrame that contains

1. timestamp
2. coin symbol
3. coin current price

Use time.sleep(5) to make Python thread sleep for 5 seconds before repeating the process for 10 times

1. API crawler to list of JSON
2. Use list comprehension to extract data

```
# work here
import time
from tqdm.notebook import tqdm
import requests

data = list()

for i in tqdm(range(5)):
    resp = requests.get('https://api.coingecko.com/api/v3/coins/markets/
    crypto_data = resp.json()
    print(resp.status_code)
    data.append(crypto_data)
    time.sleep(5)
```

```
200
200
200
200
200

  0%|          | 0/5 [00:00<?, ?it/s]
```

```
len(data)
```

```
5
```

```
coin_ts = [coin['last_updated'] for d in data for coin in d]
```

```
coin_symbol = [coin['symbol'] for d in data for coin in d]
```

```
coin_price = [coin['current_price'] for d in data for coin in d]
```

```
import pandas as pd
df = pd.DataFrame({'Time':coin_ts, 'Symbol':coin_symbol, 'Price':coin_pr
df.head(10)
```

```
                       Time Symbol         Price
0  2024-01-28T16:04:04.514Z    btc  42311.000000
1  2024-01-28T16:04:05.612Z    eth   2272.120000
2  2024-01-28T16:00:16.898Z   usdt      0.999474
3  2024-01-28T16:03:57.557Z    bnb    305.180000
4  2024-01-28T16:04:06.905Z    sol     97.250000
5  2024-01-28T16:04:05.646Z    xrp      0.529874
6  2024-01-28T16:04:07.460Z   usdc      1.001000
7  2024-01-28T16:03:40.335Z  steth   2270.300000
8  2024-01-28T16:03:56.112Z    ada      0.491464
9  2024-01-28T16:04:06.416Z   avax     35.710000
```

	Time	Symbol	Price
0	2024-01-28T16:04:04.514Z	btc	42311.000000
1	2024-01-28T16:04:05.612Z	eth	2272.120000
2	2024-01-28T16:00:16.898Z	usdt	0.999474
3	2024-01-28T16:03:57.557Z	bnb	305.180000
4	2024-01-28T16:04:06.905Z	sol	97.250000
5	2024-01-28T16:04:05.646Z	xrp	0.529874
6	2024-01-28T16:04:07.460Z	usdc	1.001000
7	2024-01-28T16:03:40.335Z	steth	2270.300000
8	2024-01-28T16:03:56.112Z	ada	0.491464
9	2024-01-28T16:04:06.416Z	avax	35.710000