Python Programming (Basic-Intermediate)

Module 1 - Basic Concepts

Understanding Google CoLab

```
print("Hello World!")
```

```
Hello World!
```

```
1+1
```

```
2
```

```
x = 1
y = x + 1
print(x, y)
```

```
1 2
```

```
!ls /content
```

```
sample_data
```

```
f = open('test.txt','w')
f.write('Hello World!')
f.close()
```

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

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

```python
import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/AIS_DG/Flight_flights.csv')
df.head()
```

```
   Unnamed: 0  year  month  day  dep_time  sched_dep_time  dep_delay
0           1  2013      1    1     517.0             515        2.0
1           2  2013      1    1     533.0             529        4.0
2           3  2013      1    1     542.0             540        2.0
3           4  2013      1    1     544.0             545       -1.0
4           5  2013      1    1     554.0             600       -6.0

   arr_time  sched_arr_time  arr_delay carrier  flight tailnum origi
0     830.0             819       11.0      UA    1545  N14228     EW
1     850.0             830       20.0      UA    1714  N24211     LG
2     923.0             850       33.0      AA    1141  N619AA     JF
3    1004.0            1022      -18.0      B6     725  N804JB     JF
4     812.0             837      -25.0      DL     461  N668DN     LG

   air_time  distance  hour  minute            time_hour
0     227.0      1400     5      15  2013-01-01 05:00:00
1     227.0      1416     5      29  2013-01-01 05:00:00
2     160.0      1089     5      40  2013-01-01 05:00:00
3     183.0      1576     5      45  2013-01-01 05:00:00
```

	Unnamed: 0	year	month	day	dep_time	sched_dep_time	dep_delay	arr_time	sched_arr_time	arr_delay
0	1	2013	1	1	517.0	515	2.0	830.0	819	11.0
1	2	2013	1	1	533.0	529	4.0	850.0	830	20.0
2	3	2013	1	1	542.0	540	2.0	923.0	850	33.0
3	4	2013	1	1	544.0	545	-1.0	1004.0	1022	-18.0
4	5	2013	1	1	554.0	600	-6.0	812.0	837	-25.0

```
!pip show pandas
```

```
Name: pandas
Version: 1.5.3
Summary: Powerful data structures for data analysis, time series, and
Home-page: https://pandas.pydata.org
Author: The Pandas Development Team
```

```
Author-email: pandas-dev@python.org
License: BSD-3-Clause
Location: /usr/local/lib/python3.10/dist-packages
Requires: numpy, python-dateutil, pytz
Required-by: altair, arviz, bigframes, bokeh, bqplot, cmdstanpy, cuffl
```

```
!python --version
```

```
Python 3.10.12
```

```
!ls /usr/local/lib/python3.10/dist-packages/
```

```
absl
absl_py-1.4.0.dist-info
adbc_driver_duckdb
aiohttp
aiohttp-3.9.1.dist-info
aiosignal
aiosignal-1.3.1.dist-info
alabaster
alabaster-0.7.16.dist-info
albumentations
albumentations-1.3.1.dist-info
altair
altair-4.2.2.dist-info
anyio
anyio-3.7.1.dist-info
apiclient
appdirs-1.4.4.dist-info
appdirs.py
apt
apt_inst.cpython-310-x86_64-linux-gnu.so
```

```
!pip install scikit-surprise
```

```
Collecting scikit-surprise
  Downloading scikit-surprise-1.1.3.tar.gz (771 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 772.0/772.0 kB 6.2 MB/s
  Preparing metadata (setup.py) ... done
Requirement already satisfied: joblib>=1.0.0 in /usr/local/lib/python3
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.
Building wheels for collected packages: scikit-surprise
  Building wheel for scikit-surprise (setup.py) ... done
  Created wheel for scikit-surprise: filename=scikit_surprise-1.1.3-cp
  Stored in directory: /root/.cache/pip/wheels/a5/ca/a8/4e28def53797fd
Successfully built scikit-surprise
```

```
Installing collected packages: scikit-surprise
```

```
%%writefile /content/drive/MyDrive/AIS_DG/lib/mymodule.py

def MyFunction():
  print ('My imported function')
```

```
Overwriting /content/drive/MyDrive/AIS_DG/lib/mymodule.py
```

```
!ls /content/drive/MyDrive/AIS_DG/lib
```

```
car.py  mymodule1.py  mymodule.py  __pycache__
```

```
import sys
sys.path.append('/content/drive/MyDrive/AIS_DG/lib')
```

```
import mymodule
```

```
import mymodule
mymodule.MyFunction()
```

```
My imported function
```

Basic syntax and data types

1. Markdown

Markdown is a lightweight markup language with simple formatting syntax. It can be used to annotate your notebook to explain or discuss the concepts/codes that you try to communicate.

Markdown mode can be selected from the toolbar, or use keyboard shotcut "M" when the cell is selected.

In this section, try run the markdown text below

Hello World

Section

Sub-Section

This is a normal text with underline.

1. **First** line
2. *Second* line

```
Python is fun
```

Test ***123***

2. Code

```
1+1
```

```
2
```

```
"Hello World"
```

```
'Hello World'
```

3. Assigning values to variable

```
counter = 100     # An integer assignment
miles = 1000.0   # A floating point
name = "John"    # A string

print(counter)
print(miles)
print(name)
```

```
100
1000.0
John
```

```
pprint = print
```

```
pprint(1,2,3)
```

```
1 2 3
```

```
a = b = c = 1
print(a,b,c)
```

```
1 1 1
```

```
a, b, c = 1, 2, "John"
print('Formatted text: %d, %d, %s'%(a,b,c))
```

```
Formatted text: 1, 2, John
```

```
1, 2
```

```
(1, 2)
```

```
a = 1,2,3,4,5
```

```
a
```

```
(1, 2, 3, 4, 5)
```

```
a, b = 1,2,3,4,5
a
```

```
too many values to unpack (expected 2)
```

4. Variables in cells

```
del x
```

```
x
```

```
name 'x' is not defined
```

```
x = 1
```

```
x + 1
```

```
2
```

```
x
```

```
1
```

5. Error! Why?

```
x = (1,2,3)
```

```
x[0]
```

```
1
```

```
x + 1
# Tuple cant modify
```

```
can only concatenate tuple (not "int") to tuple
```

6. Comments

A better way to explain

```
# This is a comment
x = 1 # This is a second comment
text = "Hello World! # This is not a comment"
print(text)
```

```
Hello World! # This is not a comment
```

Challenge: multiple assignment

Perform multiple assignment of a, b, and c to 'Hello', 0, 'World!', respectively. Replace in the code blank ___ with the actual code to complete the code.

```
a, b, c = 'Hello',0,'World!' # work here
a, b, c
```

```
('Hello', 0, 'World!')
```

7. Number

```
x = 1.1
y = 10
```

```
type(y)
```

```
int
```

```
dir()
```

```
['In',
 'Out',
 '_',
 '_2',
 '_22',
 '_23',
 '_29',
 '_31',
 '_36',
 '_37',
 '_39',
 '_42',
 '_44',
 '_45',
 '_8',
 '__',
 '___',
 '__builtin__',
 '__builtins__',
```

```
del a
```

```
dir()
```

```
['In',
 'Out',
 '_',
 '_2',
 '_22',
 '_23',
 '_29',
 '_31',
 '_36',
 '_37',
 '_39',
 '_42',
 '_44',
 '_45',
 '_46',
 '_8',
 '__',
 '___',
 '__builtin__',
 '__builtins__',
```

```
a
```

```
name 'a' is not defined
```

```
del b,c
```

Challenge: delete c

Delete the variable c.

```
del c
```

```
name 'c' is not defined
```

```
x = 1
```

```
type(x)
```

```
int
```

```
y = 0.0
```

```
type(y)
```

```
float
```

```
type(1+2j)
```

```
complex
```

```
2+2
```

```
4
```

```
2/2
```

```
5 % 2
```

```
1
```

```
2**3
```

```
8
```

8. String subsetting

```
s = 'Hello World!'
print(s)
```

```
Hello World!
```

```
print(s[0])       # Prints first character of the string
```

```
H
```

```
print(s[2:5])     # Prints characters starting index 2, not more than 5
```

```
llo
```

```
print(s[2:])      # Prints string starting from 3rd character
```

```
llo World!
```

```
print(s[-2:])
```

```
d!
```

```
print(s * 2)      # Prints string two times
```

```
Hello World!Hello World!
```

```
print(s + "TEST") # Prints concatenated string
```

```
Hello World!TEST
```

Challenge: subset the string

Print the string "I love Python" by subsetting str1, str2 and str3.

```
str1 = "Python Programming"
str2 = "You and I"
str3 = "What love is"
str_res = str2[-1]+ str3[4:10] + str1[:6]# work here
print(str_res)
```

```
I love Python
```

```
brand = 'Honda'
model = 'Accord'
year = '2018'
f'Car: {brand} {model} {year}'
```

```
'Car: Honda Accord 2018'
```

9. List subsetting

```
longlist = [ 'abcd', 786 , 2.23, 'john', 70.2 ]
tinylist = [123, 'john']

print(longlist)             # Prints complete list
```

```
['abcd', 786, 2.23, 'john', 70.2]
```

```
print(longlist[0])          # Prints first element of the list
```

```
abcd
```

```
print(longlist[1:3])        # Prints elements starting from 2nd till 3rd
```

```
[786, 2.23]
```

```
print(longlist[2:])        # Prints elements starting from 3rd element
```

```
[2.23, 'john', 70.2]
```

```
longlist[-1] #Last Element
```

```
70.2
```

```
print(tinylist * 2)     # Prints list two times
```

```
[123, 'john', 123, 'john']
```

```
print(longlist + tinylist)  # Prints concatenated lists
```

```
['abcd', 786, 2.23, 'john', 70.2, 123, 'john']
```

```
A = [[1,2],[3,4]]
A
```

```
[[1, 2], [3, 4]]
```

```
[0, [1, 2, 3, [4, 5]]]
```

```
[0, [1, 2, 3, [4, 5]]]
```

```
A + [5]
```

```
[[1, 2], [3, 4], 5]
```

```
B = []
```

```
B
```

```
[]
```

```
x = list()
```

```
x.append([1])
```

```
x
```

```
[[1]]
```

```
# x = x + [2]
x += [2]
```

```
x.count(2)
```

```
1
```

Challenge: list operation

Use the following variables x, y, and z to combine to get the final list.

[1, 2, 3, 4, 5, 6, 7, 8]

```
x = [1, 2, 3, 4]
y = 5
z = [6, 7, [8]]
```

```
# work here
x + [y] + z[:2] + z[-1]
```

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

10. Tuple subsetting

```
longtuple = ( 'abcd', 786 , 2.23, 'john', 70.2  )
tinytuple = (123, 'john')

print(longtuple)            # Prints complete list
```

```
('abcd', 786, 2.23, 'john', 70.2)
```

```
print(longtuple[0])         # Prints first element of the list
```

```
abcd
```

```
print(longtuple[1:3])       # Prints elements starting from 2nd till 3rd
```

```
(786, 2.23)
```

```
print(longtuple[2:])        # Prints elements starting from 3rd element
```

```
(2.23, 'john', 70.2)
```

```
print(tinytuple * 2)    # Prints list two times
```

```
(123, 'john', 123, 'john')
```

```
print(longtuple + tinytuple)# Prints concatenated lists
```

```
('abcd', 786, 2.23, 'john', 70.2, 123, 'john')
```

```
longtuple + tinylist
```

```
can only concatenate tuple (not "list") to tuple
```

11. Invalid tuple

```
x = ( 'abcd', 786 , 2.23, 'john', 70.2  )
y = [ 'abcd', 786 , 2.23, 'john', 70.2  ]
x[2] = 1000    # Invalid syntax with tuple
```

```
'tuple' object does not support item assignment
```

```
y[2] = 1000     # Valid syntax with list
print(x)
print(y)
```

```
('abcd', 786, 2.23, 'john', 70.2)
['abcd', 786, 1000, 'john', 70.2]
```

12. Dictionary operations

```
d = {} #dict
d1 = dict()
d['one'] = "This is one"
d[2]     = "This is two"

tinydict = {'name': 'john','code':6734, 'dept': 'sales'}
```

```
d
```

```
{'one': 'This is one', 2: 'This is two'}
```

```
print(d['one'])       # Prints value for 'one' key
```

```
This is one
```

```
print(d[2])           # Prints value for 2 key
```

```
This is two
```

```
print(tinydict)          # Prints complete dictionary
```

```
{'name': 'john', 'code': 6734, 'dept': 'sales'}
```

```
print(tinydict.keys())   # Prints all the keys
```

```
dict_keys(['name', 'code', 'dept'])
```

```
print(tinydict.values()) # Prints all the values
```

```
dict_values(['john', 6734, 'sales'])
```

```
x = dict()
```

13. Comparison

```
True
```

```
True
```

```
False
```

```
False
```

```
not True
```

```
False
```

```
False or True
```

```
True
```

```
False and True
```

```
False
```

```
1 > 1
```

```
False
```

```
1 > "1"
```

```
'>' not supported between instances of 'int' and 'str'
```

```
(2,2,3) > (2,1,4) # partial ordering
```

```
True
```

```
[3,3] <= [2,2]
```

```
False
```

```
1 > 2 or 3 > 2
```

```
True
```

```
1 < 2 < 3
```

```
True
```

14. Type conversion

```
x = "Hello"
y = 123
print(x+str(y))
```

```
Hello123
```

```
print(list(x))
```

```
['H', 'e', 'l', 'l', 'o']
```

```
tuple([y])
```

```
(123,)
```

Challenge: dictionary

Create a dictionary with the following information and print the key and value
THB: Thai Baht
GBP: Great British Pound
JPY: Japanese Yen

```
# Work on the challenge in this area
unit = {'THB':'Thai Baht',
        'GBP':'Great British Pound',
        'JPY':'Japanese Yen'}
unit['THB']
```

```
'Thai Baht'
```

Activity: Using web API to obtian data

In this activity, we will use the package **requests** to get the API response. The package can be used after importing it to the current kernel by using the command "**import** requests". After obtaining the response (JSON format, Python Dictionary), write your code to answer the following questions

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

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

```
import json

json.dump(cc_data, open('/content/drive/MyDrive/AIS_DG/cc1.json','w'))
```

```
cc_data[:10]
```

```
[{'id': 'bitcoin',
  'symbol': 'btc',
  'name': 'Bitcoin',
  'image': 'https://assets.coingecko.com/coins/images/1/large/bitcoir
  'current_price': 42365,
  'market_cap': 830056440968,
  'market_cap_rank': 1,
  'fully_diluted_valuation': 888834064145,
  'total_volume': 13680664375,
 'high_24h': 42743,
 'low_24h': 41750,
 'price_change_24h': 580.57,
 'price_change_percentage_24h': 1.38946,
 'market_cap_change_24h': 10178363134,
 'market_cap_change_percentage_24h': 1.24145,
 'circulating_supply': 19611293.0,
 'total_supply': 21000000.0,
 'max_supply': 21000000.0,
 'ath': 69045,
```

1. What is the latest price of BTC?

```
# Write your answer here
cc_data[0]['current_price']
```

```
42365
```

2. What date is the date of data retrieval?

```
# Write your answer here
cc_data[0]['last_updated']
```

```
'2024-01-28T15:16:31.783Z'
```

3. How do we know if our code can properly access the website or not? (require a bit of function reference searching on Internet)

```
# Write your answer here
resp.status_code
```

```
200
```

```
for i in range(len(cc_data)) :
  if cc_data[i]["name"] == "Bitcoin":
    print(cc_data)
```

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