Practical 0

Goals

- Understanding fundamentals of Python programming:
- Variables, data types, lists, sets and tuples.
- Conditional expressions and loops
- Sort
- Dictionary
- Files and user interaction

Exercise 1

1. Comments

In [ ]:
```
# This is a comment
print("Bonjour")
```

```
Bonjour
```

2. Variables

In [ ]:
```
# a variable
message = "le monde!"
print(message)
```

```
le monde!
```

In [ ]:
```
a = 10
b = 20
c = a + b
print(c)
```

```
30
```

In [ ]:
```
# floating point numbers
pi = 3.14
print(pi)
```

```
3.14
```

In [ ]:
```
# data types
message1 = "Bonjour"
a = 12
pi = 3.14
print(type(message1))
print(type(a))
print(type(pi))
```

```
<class 'str'>
<class 'int'>
<class 'float'>
```

3. Concatenation of two strings

In [ ]:
```
# concatenation of two strings
message = "le monde!"
print("Bonjour" + message)
```

```
Bonjourle monde!
```

In [ ]:
```
# concatenation of two strings
message1 = "Bonjour "
message2 = "le monde!"
print(message1 + message2)
```

```
Bonjour le monde!
```

In [ ]:
```
# concatenation involving two variables of different data types
# operation + on two different data types

# Uncomment the print statement and run the code
message1 = "Bonjour en Python"
a = 3
# print(message1 + a)
```

Why did you get this error? In the following code, we correct this error.

In [ ]:
```
# concatenation solution involving two variables of different data types
message1 = "Bonjour en Python "
a = 3
print(message1 + str(a))
```

```
Bonjour en Python 3
```

4. Lists

In [ ]:
```
a = [10, 20, 30, 40, 50]
print(a)
```

```
[10, 20, 30, 40, 50]
```

In [ ]:
```
a = [10, 20, 30, 40, 50]
print(a[0])
print(a[1])
print(a[2])
print(a[3])
print(a[4])
```

```
10
20
30
40
50
```

In [ ]:
```
# Uncomment the print statement and run the code
a = [10, 20, 30, 40, 50]
# print(a[8])
```

Why did you get this error? We are trying to access a element at an index that does not exist.

In [ ]:
```
message1 = "Bonjour en Python "
print(message1[0])
print(message1[1])
print(message1[2])
print(message1[3])
print(message1[4])
print(message1[5])
print(message1[6])
print(message1[7])
```

```
B
o
n
j
o
u
r
```

The above code displayed the individual characters in the string (or list of characters). We will now get the length of this string.

In [ ]:
```
message1 = "Bonjour en Python "
print(len(message1))
```

```
18
```

Nous allons maintenant créer une liste d'entiers.

In [ ]:
```
a = [10, 20, 30, 40, 50]
print(len(a))
```

```
5
```

In [ ]:
```
a = [10, 20, 30, 40, 50]

# add a new number at the end of the list
a.append(60)
print(a)
```

```
[10, 20, 30, 40, 50, 60]
```

In [ ]:
```
a = [10, 20, 30, 40, 50]

# modify a number at a particular index
a[0] = 0
print(a)
```

```
[0, 20, 30, 40, 50]
```

Why did we get this error? We are modifying an element at a non-existing index.

In [ ]:
```
# Uncomment the assignment statement and run the code
a = [10, 20, 30, 40, 50]
# a[6] = 20
print(a)
```

```
[10, 20, 30, 40, 50]
```

In [ ]:
```
a = [10, 20, 30, 40, 50]

# inserting an element at a particular index will modify the list
a.insert(0, 0)
print(a)
print(len(a))
```

```
[0, 10, 20, 30, 40, 50]
6
```

In [ ]:
```
a = [10, 20, 30, 40, 50]
a.insert(6, 60)
print(a)
print(len(a))
```

```
[10, 20, 30, 40, 50, 60]
6
```

In [ ]:
```
a = [10, 20, 30, 40, 50]

# We will now try to insert a number at an index greater than the length
# of the list. We will see that we do not get any error and the new number
# is added at the end of the list
a.insert(10, 60)
print(a)
print(len(a))
```

```
[10, 20, 30, 40, 50, 60]
6
```

5. Tuples (non-modifiable lists)

In [ ]:
```
a = (10, 20, 30, 40, 50)
print(a)
```

```
(10, 20, 30, 40, 50)
```

In [ ]:
```
a = (10, 20, 30, 40, 50)
print(a[0])
```

```
10
```

In [ ]:
```
a = (10, 20, 30, 40, 50)

# We now try to modify a tuple
# Uncomment the code below and run the code
# A tupe is a non-modifiable list

# a[0] = 0
print(a)
```

```
(10, 20, 30, 40, 50)
```

6. Sets

In [ ]:
```
# A set is a collection of distinct elements
a = {10, 20, 30, 40, 50, 10, 20, 30, 40, 50}
print(a)
```

```
{50, 20, 40, 10, 30}
```

In [ ]:
```
a = {10, 20, 30, 40, 50, 10, 20, 30, 40, 50}
a.add(10)
print(a)
```

```
{50, 20, 40, 10, 30}
```

In [ ]:
```
a = {10, 20, 30, 40, 50, 10, 20, 30, 40, 50}
a.add(60)
print(a)
```

```
{50, 20, 40, 10, 60, 30}
```

In [ ]:
```
a = {10, 20, 30, 40, 50, 10, 20, 30, 40, 50}
a.remove(40)
print(a)
```

```
{50, 20, 10, 30}
```

We will now try different data types with the numbers and print the result

In [ ]:
```
# set
a = {10, 20, 30, 40, 50, 10, 20, 30, 40, 50}
print(a)
print(type(a))

# tuple
b = (10, 20, 30, 40, 50, 10, 20, 30, 40, 50)
print(b)
print(type(b))

# list
c = [10, 20, 30, 40, 50, 10, 20, 30, 40, 50]
print(c)
print(type(c))
```

```
{50, 20, 40, 10, 30}
<class 'set'>
(10, 20, 30, 40, 50, 10, 20, 30, 40, 50)
<class 'tuple'>
[10, 20, 30, 40, 50, 10, 20, 30, 40, 50]
<class 'list'>
```

Exercise 2

1. Conditional Expressions

In [ ]:
```
a = 12
if a % 2 == 0:
    print(a, " is divisible by 2")
else:
    print(a, " is not divisible by 2")
```

```
12  is divisible by 2
```

In [ ]:
```
lang = "Français"
if lang == "Français":
    print("Bonjour le monde!")
else:
    print("Hello World!")
```

```
Bonjour le monde!
```

2. Loops

Loops can also be used to access the elements at different indices.

In [ ]:
```
for i in [10, 20, 30, 40, 50, 10, 20, 30, 40, 50]:
    print(i)
```

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

In [ ]:
```
for i in (10, 20, 30, 40, 50, 10, 20, 30, 40, 50):
    print(i)
```

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

In [ ]:
```
for i in {10, 20, 30, 40, 50, 10, 20, 30, 40, 50}:
    print(i)
```

```
40
10
50
20
30
```

2. Range

In [ ]:
```
for i in range(0, 10):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
```

In [ ]:
```
for i in range(0, 10, 2):
    print(i)
```

```
0
2
4
6
8
```

In [ ]:
```
# print() by default displays the message followed by a new line
# But you can change its behaviour
for i in range(0, 10, 2):
    print(i, end=" ")
```

```
0 2 4 6 8
```

In [ ]:
```
for i in range(10, 0, -2):
    print(i)
```

```
10
8
6
4
2
```

In [ ]:
```
for i in range(10, 0):
    print(i)
```

split(): the function can be used to separate a string using a specified delimited. By default, the delimiter is a white space.

In [ ]:
```
for i in "Bonjour,le,monde!".split():
    print(i)
```

```
Bonjour,le,monde!
```

In [ ]:
```
for i in "Bonjour,le,monde!".split(","):
    print(i)
```

```
Bonjour
le
monde!
```

Write a program in Python to display the following output

1

12

123

1234

12345

123456

1234567

12345678

In [ ]:
```
s = ""
for i in range (1,9):
    s = f"{s}{str(i)}"
    print(s, end="\n\n")
```

```
1

12

123

1234

12345

123456

1234567

12345678
```

Exercise 3

1. Sort

In [ ]:
```
num = [10, 20, 30, 40, 50, 10, 20, 30, 40, 50]
num.sort()
print(num)
```

```
[10, 10, 20, 20, 30, 30, 40, 40, 50, 50]
```

2. Sort (decreasing order)

In [ ]:
```
num = [10, 20, 30, 40, 50, 10, 20, 30, 40, 50]
num.sort(reverse=True)
print(num)
```

```
[50, 50, 40, 40, 30, 30, 20, 20, 10, 10]
```

3. minimum

In [ ]:
```
num = [10, 20, 30, 40, 50, 10, 20, 30, 40, 50]
print(min(num))
```

```
10
```

4. maximum

In [ ]:
```
num = [10, 20, 30, 40, 50, 10, 20, 30, 40, 50]
print(max(num))
```

```
50
```

5. sorted()

You can use the function if you do not wish to modify your initial list by sorting.

In [ ]:
```
num = [70, 20, 30, 10, 50, 60, 20, 80, 70, 50]
sortednum = sorted(num, reverse=True)
print(num)
print(sortednum)
```

```
[70, 20, 30, 10, 50, 60, 20, 80, 70, 50]
[80, 70, 70, 60, 50, 50, 30, 20, 20, 10]
```

In [ ]:
```
num = [70, 20, 30, 10, 50, 60, 20, 80, 70, 50]

# select first five numbers
sortednum = sorted(num, reverse=True)[:5]
print(sortednum)
```

```
[80, 70, 70, 60, 50]
```

Modify the code given below to display the five greatest unique numbers.

In [ ]:
```
print(sorted("Bonjour le monde!".split(), key=str.lower, reverse=True))
```

```
['monde!', 'le', 'Bonjour']
```

Exercise 4

1. Dictionary

In [ ]:
```
a = {"contente": 12, "content": 12, "triste": 2}
print(a)
print(type(a))
```

```
{'contente': 12, 'content': 12, 'triste': 2}
<class 'dict'>
```

In [ ]:
```
a = {"contente": 12, "content": 12, "triste": 2}
for key in a:
    print("la phrase ", key, " apparait ", a[key], " fois")
```

```
la phrase  contente  apparait  12  fois
la phrase  content  apparait  12  fois
la phrase  triste  apparait  2  fois
```

In [ ]:
```
a = {"contente": 12, "content": 12, "triste": 2}
for key, value in a.items():
    print("la phrase ", key, " apparait ", value, " fois")
```

```
la phrase  contente  apparait  12  fois
la phrase  content  apparait  12  fois
la phrase  triste  apparait  2  fois
```

In [ ]:
```
a = {"contente": 12, "content": 12, "triste": 2}
a["joie"] = 10
print(a)
```

```
{'contente': 12, 'content': 12, 'triste': 2, 'joie': 10}
```

In [ ]:
```
a = {"contente": 12, "content": 12, "triste": 2}
del a["triste"]
print(a)
```

```
{'contente': 12, 'content': 12}
```

In [ ]:
```
mots = {"contente": 12, "content": 12, "triste": 2, "joie": 10}
print(sorted(mots))
```

```
['content', 'contente', 'joie', 'triste']
```

In [ ]:
```
mots = {"contente": 12, "content": 12, "triste": 2, "joie": 10}
mots_tuple = [(key, value) for key, value in mots.items()]
print(mots_tuple)
```

```
[('contente', 12), ('content', 12), ('triste', 2), ('joie', 10)]
```

2. itemgetter

In [ ]:
```
from operator import itemgetter

mots = {"contente": 12, "content": 12, "triste": 2, "joie": 10}
mots_tuple = [(key, value) for key, value in mots.items()]
print(sorted(mots_tuple, key=itemgetter(1)))
```

```
[('triste', 2), ('joie', 10), ('contente', 12), ('content', 12)]
```

3. Interaction with user

In [ ]:
```
# nom = input("Quel est votre nom?")
# print(nom)
```

In [ ]:
```
# age = input("Quel est votre âge? ")
# print(age)
# print(type(age))
```

In [ ]:
```
# age = input("Quel est votre âge? ")
# age = int(age)
# print(age)
# print(type(age))
```

Question: Write a program in Python that interacts with the user to obtain the following information of 5 students:

- Name of student
- Age
- Grades in 5 modules

Once the information for all the five students is obtained, calculate and display the following values for every module:

- average grade
- maximum grade
- minimum grade

Exercise 5

1. Files

In [ ]:
```
message = "Bonjour le monde"
with open("bonjour.txt", "w") as file:
    file.write(message)
file.close()
```

In [ ]:
```
with open("bonjour.txt", "r") as file:
    text = file.read()
    print(text)
file.close()
```

```
Bonjour le monde
```

In [ ]:
```
message1 = "Bonjour le monde"
message2 = "Programmation en Python"
with open("bonjour.txt", "w") as file:
    file.write(message1)
    file.write(message2)
file.close()
```

In [ ]:
```
with open("bonjour.txt", "r") as file:
    text = file.read()
    print(text)
file.close()
```

```
Bonjour le mondeProgrammation en Python
```

In [ ]:
```
message1 = "Bonjour le monde\n"
message2 = "Programmation en Python"
with open("bonjour.txt", "w") as file:
    file.write(message1)
    file.write(message2)
file.close()

with open("bonjour.txt", "r") as file:
    text = file.read()
    print(text)
file.close()
```

```
Bonjour le monde
Programmation en Python
```

2. readline()

This function can be used to read a file line by line and not the complete content in a single call like read()

In [ ]:
```
message1 = "Bonjour le monde\n"
message2 = "Programmation en Python"
with open("bonjour.txt", "w") as file:
    file.write(message1)
    file.write(message2)
file.close()
```

In [ ]:
```
with open("bonjour.txt", "r") as file:
    text = file.readline()
    print(text)
file.close()
```

```
Bonjour le monde
```

In [ ]:
```
message1 = "Bonjour le monde\n"
message2 = "Programmation en Python\n"
with open("bonjour.txt", "w") as file:
    file.write(message1)
    file.write(message2)
file.close()
```

In [ ]:
```
with open("bonjour.txt", "r") as file:
    for line in file:
        print(line)
file.close()
```

```
Bonjour le monde

Programmation en Python
```

Question: Copy any HTML file in your home directory. Write a program in Python to get the following values:

- number of characters in the HTML file
- number of lines in the HTML file
- number of words in the HTML file
- first twenty words in the HTML file
- distinct words in the HTML file

Question: Copy the CSV file population.csv from the **data** folder. The file contains the population values between 1901 and 2016. Write a program in Python to get the maximum value.

- the maximum value of population
- the minimum value of population

In [ ]:
```

```

Numpy

In [ ]:
```
import numpy as np
```

Matrix operations

In [ ]:
```
a1 = [1, 2, 3, 4]
b1 = [5, 6, 7, 8]
```

In [ ]:
```
c1 = np.add(a1, b1)
print(c1)
```

```
[ 6  8 10 12]
```

In [ ]:
```
c1 = np.subtract(a1, b1)
print(c1)
```

```
[-4 -4 -4 -4]
```

In [ ]:
```
c1 = np.multiply(a1, b1)
print(c1)
```

```
[ 5 12 21 32]
```

In [ ]:
```
c1 = np.multiply(4, b1)
print(c1)
```

```
[20 24 28 32]
```

In [ ]:
```
c1 = np.dot(a1, b1)
print(c1)
```

```
70
```

In [ ]:
```
c1 = np.dot(5, b1)
print(c1)
```

```
[25 30 35 40]
```

In [ ]:
```
b1 = [5, 6, 7, 8]
c1 = np.dot(a1, b1)
print(c)
```

```
[10, 20, 30, 40, 50, 10, 20, 30, 40, 50]
```

In [ ]:
```
a1 = [1, 2]
moyen = np.average(a1)
print(moyen)
```

```
1.5
```

Matrix Transposition

In [ ]:
```
b1 = [[5, 6, 7, 8, 9], [1, 2, 3, 4, 5]]
print(b1)
b1 = np.array(b1)
print(b1.T)
```

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

Split a matrix to two, horizontally and vertically

In [ ]:
```
a = [[0, 0, 0], [0, 1, 0], [1, 0, 0], [1, 1, 1]]
a = np.array(a)

a = np.hsplit(a, [2])

a1 = a[0]
a2 = a[1]
print(a1)
print(a2)
```

```
[[0 0]
 [0 1]
 [1 0]
 [1 1]]
[[0]
 [0]
 [0]
 [1]]
```

In [ ]:
```
a = [[0, 0, 0], [0, 1, 0], [1, 0, 0], [1, 1, 1]]
a = np.array(a)

a = np.vsplit(a, [2])

a1 = a[0]
a2 = a[1]
print(a1)
print(a2)
```

```
[[0 0 0]
 [0 1 0]]
[[1 0 0]
 [1 1 1]]
```

Operations on two-dimensional matrices

In [ ]:
```
a2 = [[1, 2, 3, 4], [1, 2, 3, 4]]
b2 = [[5, 6, 7, 8], [5, 6, 7, 8]]
```

In [ ]:
```
c2 = np.add(a2, b2)
print(c2)
```

```
[[ 6  8 10 12]
 [ 6  8 10 12]]
```

In [ ]:
```
c2 = np.subtract(a2, b2)
print(c2)
```

```
[[-4 -4 -4 -4]
 [-4 -4 -4 -4]]
```

In [ ]:
```
c2 = np.multiply(a2, b2)
print(c2)
```

```
[[ 5 12 21 32]
 [ 5 12 21 32]]
```

In [ ]:
```
a2 = [[1, 2, 3, 4], [1, 2, 3, 4]]
a2 = np.array(a2)
b2 = [[5, 6, 7, 8], [5, 6, 7, 8]]
b2 = np.array(b2)
c2 = np.dot(a2, b2.T)
print(c2)
```

```
[[70 70]
 [70 70]]
```

Change the shape of a matrix

In [ ]:
```
b2.reshape(4, 2)
print(b2)
```

```
[[5 6 7 8]
 [5 6 7 8]]
```

Creation of a matrix with random numbers

In [ ]:
```
a = np.random.rand()
print(a)
```

```
0.2116019380716021
```

In [ ]:
```
a = np.random.rand(1)
print(a)
```

```
[0.7760056]
```

In [ ]:
```
a = np.random.rand(2, 5)
print(a)
```

```
[[0.60270168 0.51871624 0.20191849 0.30506221 0.25734665]
 [0.70538326 0.61819284 0.76842407 0.77868538 0.76078   ]]
```

In [ ]:
```
a = np.random.rand(2, 3, 3)
print(a)
```

```
[[[0.43797447 0.83255925 0.9964601 ]
  [0.74075691 0.2263524  0.0016106 ]
  [0.4931213  0.08453818 0.52184633]]

 [[0.80781199 0.40305347 0.71927369]
  [0.39679995 0.81791481 0.88003347]
  [0.95890496 0.62249067 0.07393747]]]
```

In [ ]:
```
a = np.zeros(1)
print(a)
```

```
[0.]
```

In [ ]:
```
a = np.zeros(1, dtype=int)
print(a)
```

```
[0]
```

In [ ]:
```
a = np.zeros((2, 5), dtype=int)
print(a)
```

```
[[0 0 0 0 0]
 [0 0 0 0 0]]
```

Some functions to work with lists

In [ ]:
```
a = [12, 13, 14, 15]
a_reverse = reversed(a)

for num in a_reverse:
    print(num)
```

```
15
14
13
12
```

In [ ]:
```
days = ["dimanche", "lundi", "mardi", "mercredi", "jeudi", "vendredi"]
days_num = list(enumerate(days))
print(days_num)

days_num = list(enumerate(days, start=1))
print(days_num)
```

```
[(0, 'dimanche'), (1, 'lundi'), (2, 'mardi'), (3, 'mercredi'), (4, 'jeudi'), (5,
'vendredi')]
[(1, 'dimanche'), (2, 'lundi'), (3, 'mardi'), (4, 'mercredi'), (5, 'jeudi'), (6,
'vendredi')]
```

In [ ]:
```
a1 = [1, 2, 3, 4]
b1 = [5, 6, 7, 8]

zipped = list(zip(a1, b1))
print(zipped)
```

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

In [ ]:
```
a1 = [1, 2, 3, 4]
b1 = [5, 6, 7, 8]

ezipped = list(enumerate(zip(a1, b1)))
print(ezipped)
```

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