9/10

Group_04_Exercice_00

November 24, 2020

1 Exercice 00

1.1 1 Numbers

1.1.1 1.a. What is the *type* of the result of the expression 3 + 1.5 + 4? (without typing code)

type

-0.25 pt

[1]:
```
result= 3 + 1.5 +4
type(result)
```

[1]:
```
float
```

1.1.2 1.b. How do you get it with code? (method?)

[2]:
```
# get the type of the result from 1.a
type(result)
```

[2]:
```
float
```

1.1.3 1.c. Ask the user for an iput and then save to input to an integer called "user_in" and then print the value multiplied by 5.

[3]:
```
# value multiplied by 5
user_in=int(input())
mul_user_in=user_in*5
mul_user_in
```

```
6
```

[3]:
```
30
```

1.1.4 1.d. Ask the user for an iput and then save to input to an integer called "square_root_value" and calculate the square_root of the number from the user

-0.25 pt

[9]:
```
# Square root
number=int(input())
from math import sqrt
square_root_value=sqrt(number)
square_root_value
```

```
9
```

[9]: 3.0

1.1.5 1.e. Ask the user for an iput and then save to input to an integer called "square_value" and calculate the square of the number from the user

[12]:
```
# Square
number=int(input())
square_value=(number**2)
square_value
```

```
4
```

[12]: 16

1.2 2 Strings

1.2.1 2.a. Given the string 'hello' give an index command that returns 'e'. Enter your code in the cell below:

[16]:
```
greeting = 'hello'
# Print out 'e' using indexing
print(greeting[1])
```

```
e
```

1.2.2 2.b. Given the string 'hello' give an index command that returns 'hell'. Enter your code in the cell below:

[17]:
```
greeting = 'hello'
# Print out 'hell' using indexing
greeting[:-1]
```

[17]: 'hell'

1.2.3 2.c Given the string 'hello', create a new string variable called 'greeting_rest' from it to and save 'llo' in the new variable

[19]:
```
greeting = 'hello'
# Save the part 'llo' in a new variable called 'greeting_rest' using indexing
greeting_rest = greeting[2:]
greeting_rest
```

[19]:
```
'llo'
```

1.2.4 2.d. Ask the user for his or her name and then save the input to a variable named "user_name". Then print "Hello, *user_name* !"

[5]:
```
Name=input('user_name:')
print('Hello {} !'.format(Name))
```

```
user_name:Yang
Hello Yang !
```

1.2.5 2.e. Ask the user for his or her 'first_name', 'last_name' and 'age' and print the reust in a multi-line string like:

'Hello, first_name last_name.

You are age years old. '

[1]:
```
# hint: 3 inputs => 3 variables
first_name = input ('Your first name please: ')
last_name = input ('Your last name please: ')
age = input ('Your age is: ')
print ('Hello, {}, {}!\nYou are {} years old'.format (first_name,last_name,age))
```

```
Your first name please: Yang
Your last name please: Bai
Your age is: 32
Hello, Yang, Bai!
You are 32 years old
```

1.3 3. List

1.3.1 3.a Create a list with 4 elements "45,25,56" in two differents way and save it to a variable called 'my_list'

[15]:
```
# my_list =
my_list = [45,25,56,65]
print(my_list)
my_list2 = [45,25,56]
my_list2.append(65)
print(my_list2)
```

```
[45, 25, 56, 65]
[45, 25, 56, 65]
```

1.3.2 3.b. From 'my_list' change the first value (index 0) to 0.

[16]:
```
# index 0 must be 0
my_list[0]=0
my_list
```

[16]:
```
[0, 25, 56, 65]
```

1.3.3 3.c. Save the sum of all number in the list to a variable called 'sum_of_my_list'

[18]:
```
# sum of 0,25,56
sum_of_my_list = sum(my_list[0:])
sum_of_my_list
```

[18]:
```
146
```

[3]:
```
list = [4,5,6,3,6,7,10,2,9]
list.sort()
print(list)
```

```
[2, 3, 4, 5, 6, 6, 7, 9, 10]
```

1.3.4 3.d. sort the list bellow:

- 0.5 pt

1.3.5 3.e. Get the last 3 elements of the list using indexing and save it to a variable called 'list2'. Then make again the sum of 'list2' and insert the result to 'list2'

[10]:
```
# hint: you might use 3 differents variables
list[6]=23
list[7]=98
list[8]=32
list2=list
sum_list2=sum(list2[0:])
print(list2)
print(sum_list2)
list2.append(sum_list2)
print(list2)
```

```
[2, 3, 4, 5, 6, 6, 23, 98, 32, 179]
358
[2, 3, 4, 5, 6, 6, 23, 98, 32, 179, 358]
```

1.3.6 3.f. swap list elements

Swap the first and last elements from the list `one_to_five`

[17]:
```
# create list
one_to_five = [5,2,3,4,1]
one_to_five[0]=one_to_five[len(one_to_five)-1]
one_to_five1[len(one_to_five)-1]=5
one_to_five1
```

[17]:
```
[1, 2, 3, 4, 5]
```

1.4 4. Dictionaries

Using keys and indexing, grap the word *Bremerhaven* from the following dictionaries:

[20]:
```
name = {'university':'Bremerhaven'}
# Get 'Bremerhaven'
name['university']
```

[20]:
```
'Bremerhaven'
```

[21]:
```
name = {'institution':{'name':'Bremerhaven'}}
# Get 'Bremerhaven'
name['institution']['name']
```

[21]:
```
'Bremerhaven'
```

[24]:
```
name = {'region':[{'University':'Oldenburg','Hochschule':'Bremerhaven'}]}
# Get Bremerhaven
name['region'][0]['Hochschule']
```

[24]:
```
'Bremerhaven'
```

1.5 5. What is the major difference between tuples and lists?

tuples can not be changed.

1.6 6. Sets

1.6.1 6.a. What is unique about a set?‘

every element in the set appears once.

1.6.2 6.b. Use a set to find the unique values of the list below:

[26]:
```
# create the list
unsorted_list = [1,2,2,1,3,5,4,8,7,74,8,8,9,9,5,4,45,12,4,2]
set (unsorted_list)
```

[26]:
```
{1, 2, 3, 4, 5, 7, 8, 9, 12, 45, 74}
```

1.7 6. Boolean

What will be the value of the following boolean?

[27]:
```
4**0.5 != 2
```

[27]: False

[30]:
```
4**0.5 == 2
```

[30]: True

[29]:
```
a = 1 < 4
a
```

[29]: True

[32]:
```
b = 'b' < 'c'
b
```

[32]: True

[42]:
```
c = (a == b)
a = 6
b = 'g'
c
```

[42]: False

[37]:
```
d = (c or False)
d = c
d
```

[37]: True

[43]:
```
e = (c and False) # equivalent to 'e=((a==b) and False)' <=>␣
↪'e=(((1<4)==('b'<'c')) and False'
e
```

[43]: False

[ ]:
```

```