PYTHON ASSIGNMENT

MAKE A MOVE TO PYTHON

ASSIGNMENTS

SUBMITTED TO

YASHIKA KHATRI

By: ANKUR SINGH

TASK-3: DATA STRUCTURES

Create a list of the 10 elements of four different types of Data Type like int, string, complex and float.

```
list = []
list.append(10)
list.append("ConsultAdd")
list.append(3.14)
list.append(10+20j)
list.append(True)
list.append(False)
list.append(40)
list.append(0xface) #Hexa value
list.append(0o777) #Octal value
list.append("Training")

print(list)
```

Output:

```
[10, 'ConsultAdd', 3.14, (10+20j), True, False, 40, 64206, 511, 'Training']
```

Create a list of size 5 and execute the slicing structure.

```
list = []
list.append(10)
list.append("ConsultAdd")
list.append(3.14)
list.append(10+20j)
list.append(True)

print(list)

print(list[0:3])
print(list[::2])
```

Output:

```
[10, 'ConsultAdd', 3.14, (10+20j), True]
[10, 'ConsultAdd', 3.14]
[10, 3.14, True]
```

Write a program to get the sum and multiply of all the items in a given list.

```
list = [1, 2, 3, 4, 5, 6]
sum = 0
multiply = 1

for value in list:
   sum= sum + value
   multiply = multiply * value
print(sum)
print(multiply)
```

Output:

```
21

720
```

Find the largest and smallest number from a given list.

Method 1: Using Inbuilt function:

```
list = [7, 0, 7, 4, 1, 9, 3]
print(max(list))
print(min(list))
```

Output:

```
9
0
```

Method 2: Using Traditional Approach

```
list = [9, 0, 7, 4, 1, 5, 3]

max_value = list[0]
min_value = list[0]
```

```
for num in list:
   if num > max_value:
      max_value = num
   elif num < min_value:
      min_value = num
print("Maximum value is {0}".format(max_value))
print("Minimum value is {0}".format(min_value))
```

Output:

```
Maximum value is 9
Minimum value is 0
```

Create a new list which contains the specified numbers after removing the even numbers from a predefined list.

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

for num in list:
   if num % 2 != 0:
      new_list.append(num)
   else:
      continue
print("New List after removing even numbers is: {0}".format(new_list))
```

Output:

```
New List after removing even numbers is: [1, 3, 5, 7, 9]
```

Create a list of first and last 5 elements where the values are square of numbers between 1 and30 (both included).

```
list = []
list_1 = []
list_2 = []
for data in range(1, 31):
   list.append(data)
```

```
    if data in range(1,6):
        data = data ** 2
        list_1.append(data)
        continue
    elif data in range (25, 31):
        data = data ** 2
        list_2.append(data)
    else:
        pass
print("Original List is: {0}".format(list))
print("First five elements in list is: {0}".format(list_1))
print("Last five elements in list is: {0}".format(list_2))
print("Appended list is : {0}".format(list_1 + list_2))
```

Output:

```
Original List is: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
21, 22, 23, 24, 25, 26, 27, 28, 29, 30]

First five elements in list is: [1, 4, 9, 16, 25]

Last five elements in list is: [625, 676, 729, 784, 841, 900]

Appended list is : [1, 4, 9, 16, 25, 625, 676, 729, 784, 841, 900]
```

Write a program to replace the last element in a list with another list.
Sample data: [[1,3,5,7,9,10],[2,4,6,8]]
Expected output: [1,3,5,7,9,2,4,6,8]

```
list_1 = [1, 3, 5, 7, 9, 10]
list_2 = [2, 4, 6, 8]
list_1[-1:] =list_2
print(list_1)
```

Output:

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

Create a new dictionary by concatenating the following two dictionaries:

a={1:10,2:20}

b={3:30,4:40}

Expected Result: {1:10,2:20,3:30,4:40}

```
a = {1:10, 2:20}
b = {3:30, 4:40}
new_dict = dict()

for data in (a, b):
   new_dict.update(data)
print(new_dict)
```

Output:

```
{1: 10, 2: 20, 3: 30, 4: 40}
```

Create a dictionary that contains a number (between 1 and n) in the form(x,x*x).
Sample data (n=5)
Expected Output: {1:1,2:4,3:9,4:16,5:25}

```
dict = {}
for data in range (1, 6):
   dict.update({data: data * data})
print(dict)
```

Output:

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

Write a program which accepts a sequence of comma-separated numbers from console and generate a list and a tuple which contains every number. Suppose the following input is supplied to the program:
34,67,55,33,12,98
The output should be:
[‘34’,’67’,’55’,’33’,’12’,’98’]
(‘34’,’67’,’55’,’33’,’12’,’98’)

```
user_input = input("Please enter the values in comma seperated formats: ")
new_list = None
new_tuple = None
for data in user_input:
    new_list = user_input.split(",")
    new_tuple = tuple(new_list)

print("List is: {0}".format(new_list))
print("Tuple is: {0}".format(new_tuple))
```

Output:

```
Please enter the values in comma seperated formats: 34,67,55,33,12,98
List is: ['34', '67', '55', '33', '12', '98']
Tuple is: ('34', '67', '55', '33', '12', '98')
```

MORE QUESTIONS ON DATA STRUCTURES

Create a list of the 10 elements of four different types of Data Type like int, string, complex and float.

```
list= []
list.append(10) #Integer
list.append("ConsultADD")  #String
list.append(10 + 20j)  #Complex
list.append(99.01)  #Float
list.append(0xface) #Hexadecimal
list.append(0o777)  #Octal
list.append(0b1111) #Binary
list.append({1:"ComsultADD", 2:"Training"}) #Dictionary
list.append([1, 2, 3, 5])  #list
list.append(("Test"))

print(list)
```

Output:

```
[10, 'ConsultADD', (10+20j), 99.01, 64206, 511, 15, {1: 'ComsultADD', 2:
'Training'}, [1, 2, 3, 5], 'Test']
```

Create a list of size 5 and execute the slicing structure

```
list = []

for data in range(5):
   list.append(data)
print(list)
print(list[0:5:2])
```

Output:

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

Create a list of given structure and run
x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]
Access list [1, 2, 3, 4]
Access list [600, 700]
Access list [100, 300, 500, 600, 800]
Access list [[800, 700, 600, [1, 2, 3, 4, 5, [10, 20, 30, 40, 50], 6, 7, 8, 9], 500, 400, 300, 200, 100]]
Access list [10]
Access list [ ]

```
x=[100,200,300,400,500,[1,2,3,4,5,[10,20,30,40,50],6,7,8,9],600,700,800]

print(x[5][:4]) # Access list [1, 2, 3, 4]
print(x[6:8])   # Access list [600,  700]
print(x[::2])   # Access list [100, 300, 500, 600, 800]
print(x[::-1])  # Access list [[800, 700, 600, [1, 2, 3, 4, 5, [10, 20, 30, 40, 50], 6, 7,
8, 9], 500, 400, 300, 200, 100]]
print(x[5][5][0])   # Access list [10]
print(x[:0])    # Access list [ ]
```

Output:

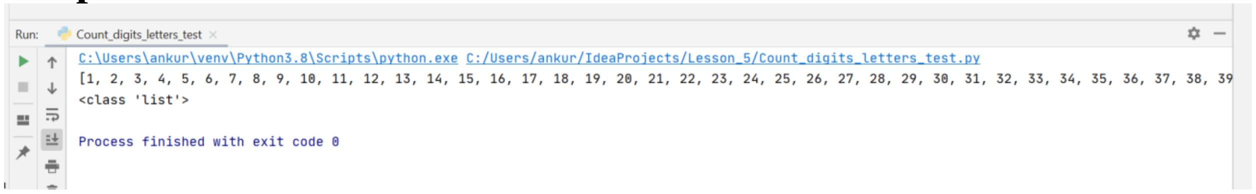
```
[1, 2, 3, 4]

[600, 700]

[100, 300, 500, 600, 800]

[800, 700, 600, [1, 2, 3, 4, 5, [10, 20, 30, 40, 50], 6, 7, 8, 9], 500, 400, 300, 200,
100]

10

[]
```

Create a list of thousand number using range and x range and see the difference between each other

Range()

```
#  Creating a list of thousand numbers using Range()
list_1 = []
for data in range(1,1001):
   list_1.append(data)
print(list_1)
print(type(list_1))
```

Output:

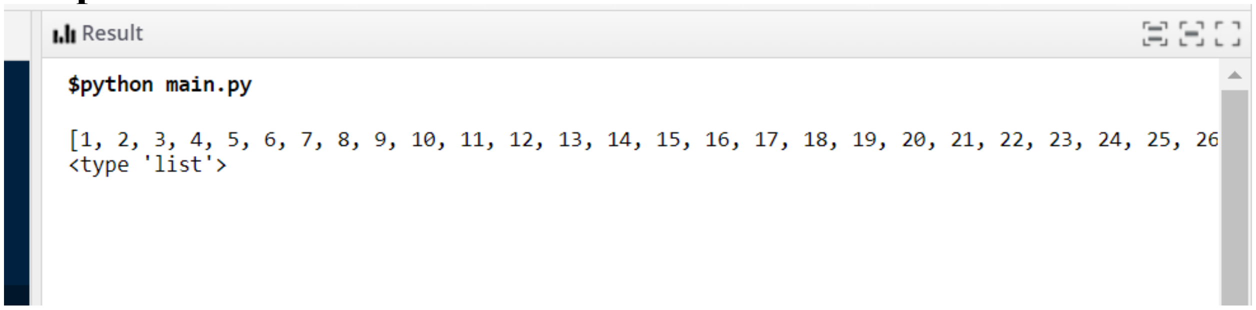


```
Run:   Count_digits_letters_test
C:\Users\ankur\venv\Python3.8\Scripts\python.exe C:/Users/ankur/IdeaProjects/Lesson_5/Count_digits_letters_test.py
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39
<class 'list'>

Process finished with exit code 0
```

Xrange()

```
# Creating a list of 1000 numbers using xrange function in Python 2.7 Online
Interpretor

list_2 = []
for data in xrange(1, 1001):
  list_2.append(data)

print(list_2)
print(type(list_2))
```

Output:



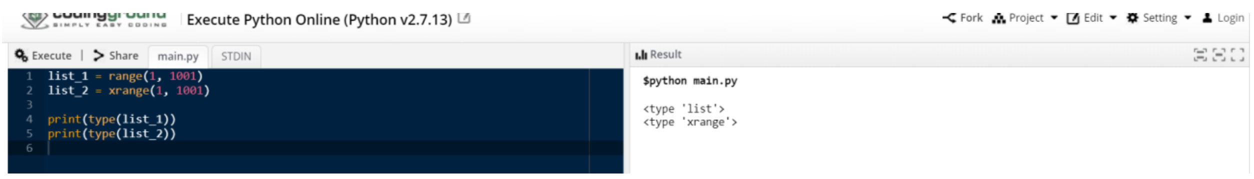
```
Result
$python main.py

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26
<type 'list'>
```

Difference between Range and XRange:

Xrange and Range are different in following ways:

1. Return Type
2. Memory
3. Operation
4. Speed

Return Type:
range() returns range object
xrange() returns xrange() object

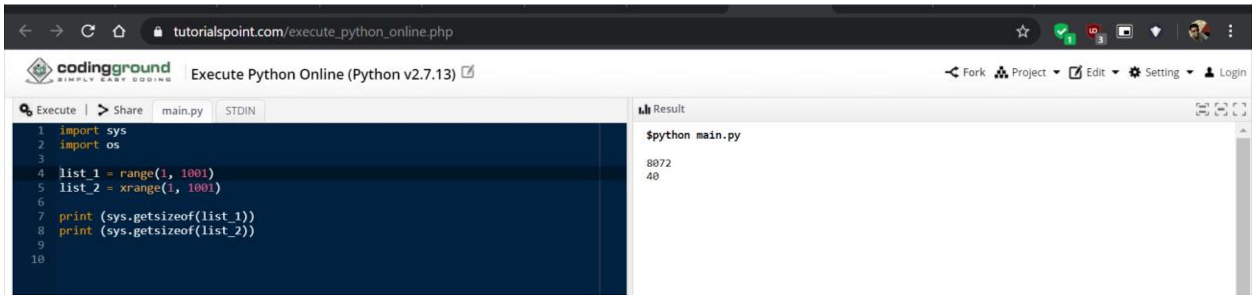


```
list_1 = range(1, 1001)
list_2 = xrange(1, 1001)

print(type(list_1))
print(type(list_2))
```

```
$python main.py

<type 'list'>
<type 'xrange'>
```

Memory:

Variable created by range() takes more memory as compared to variable created by xrange(). This is because return type of range is a list and xrange is an object.

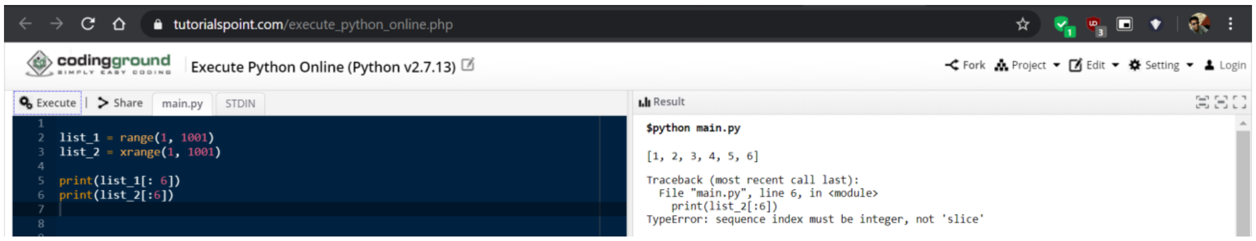

tutorialspoint.com/execute_python_online.php
codingground
Execute Python Online (Python v2.7.13)
Fork
Project
Edit
Setting
Login
Execute
Share
main.py
STDIN
Result

```
import sys
import os

list_1 = range(1, 1001)
list_2 = xrange(1, 1001)

print (sys.getsizeof(list_1))
print (sys.getsizeof(list_2))
```

```
$python main.py

8072
40
```

Operation:

In range(), all operations that are used on list can be used on range as well. However in xrange() all operations associated to a list can not be applied on xrange().


tutorialspoint.com/execute_python_online.php
codingground
Execute Python Online (Python v2.7.13)
Fork
Project
Edit
Setting
Login
Execute
Share
main.py
STDIN
Result

```
list_1 = range(1, 1001)
list_2 = xrange(1, 1001)

print(list_1[: 6])
print(list_2[:6])
```

```
$python main.py

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

Traceback (most recent call last):
  File "main.py", line 6, in <module>
    print(list_2[:6])
TypeError: sequence index must be integer, not 'slice'
```

Speed:

xrange() is faster than range(), because xrange() generates an object using process called lazy evaluation.

How Tuple is beneficial as compared to the list?

Advantages of Tuple over List is as follows:

1. Tuple being immutable requires less memory space as compared to a list
2. Tuple is considered to be faster than a List
3. A tuple can be converted to a set. However, a list can only be converted to a set if elements of set is immutable (List within a list)
4. Tuple can be used as a key in dictionary due to their hashable and immutable nature whereas list are not used as key in a dictionary because list cant handle hash functions and have mutable nature.

Write a program in Python to iterate through the list of numbers in the range of 1,100 and print the number which is divisible by 3 and a multiple of 2.

```
list = []
new_list = []

for data in range (1, 100):
  list.append(data)
  if data % 3 == 0 and data % 2 ==0:
        new_list.append(data)
    else:
        pass
print("Numbers divisible by 3 and are multiple of 2 are: {0}
".format(new_list))
```

Output:

```
Numbers divisible by 3 and are multiple of 2 are: [6, 12, 18, 24, 30, 36, 42, 48, 54,
60, 66, 72, 78, 84, 90, 96]
```

Write a program in Python to reverse a string and print only the vowel alphabet if exist in the string with their index.

```
string = "ConsultAdd is the Best!"
empty_st = ""
index = 0

for data in string:
    if data.casefold() in ("a","e","i","o","u"):
        empty_st = empty_st + data
        print("Vowel is {0} and Index value is {1}".format(data,index))
        index = index + 1
    else:
        index = index + 1
        continue
print("Original string with only vowels is: {0}".format(empty_st))
print("Reverse string with only vowels is: {0}".format(empty_st[::-1]))
```

Output:

```
Vowel is o and Index value is 1

Vowel is u and Index value is 4

Vowel is A and Index value is 7

Vowel is i and Index value is 11
Vowel is e and Index value is 16

Vowel is e and Index value is 19

Original string with only vowels is: ouAiee

Reverse string with only vowels is: eeiAuo
```

Write a program in Python to iterate through the string “hello my name is abcde” and print the string which has even length of word.

```python
string = "Hello my name is abcde"
counter = 0
empty_st = ""

for data in string:
    if data != " ":
        counter = counter + 1  #1 2 3 4 5
        empty_st = empty_st + data  #h e l l 0
    elif data == " ":
        if counter % 2 == 0:
            print("String '{}' has length {}".format(empty_st, counter))
        counter = 0
        empty_st = ""
```

Output:

```
String 'my' has length 2
String 'name' has length 4
String 'is' has length 2
```

Write a program in python to print the pair of numbers whose sum is equal to result number that is let's say 8.
x=[1,2,3,4,5,6,7,8,9,-1]

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

for num_1 in x: #2
   for num_2 in x:
      if num_1 + num_2 != 8:
         continue
      elif num_1 + num_2 == 8:
         print("{0} and {1} adds to {2}".format(num_1,num_2,(num_1 +
num_2)))
```

Output:

```
1 and 7 adds to 8

2 and 6 adds to 8

3 and 5 adds to 8

5 and 3 adds to 8

5 and 3 adds to 8

6 and 2 adds to 8

7 and 1 adds to 8

3 and 5 adds to 8
```

Write a program in Python to complete the following task:

Create two different list as in even_list and odd_list

Ask user to enter the number in the range of 1,50 and make sure if the entered number is even append it to the even_list and if the entered number is odd append it to the odd list.

Keep that in mind you can only add 5 items in each list

Make sure once you entered the total 5 element calculate the sum of the list and return the maximum out of the list.

```
def get_sum(list):
   sum = 0

   for value in list:
      sum = sum + int(value)
   return str(sum)



def geteven_odd():
   even_lst = []
   odd_lst = []

   len_even_lst = len(even_lst)
   len_odd_lst = len(odd_lst)

   print("enter any number")
   num = input()

   while (True):
      print("-------")


      if len(even_lst) >= 5 and len(odd_lst) >= 5:
         print("both even list and odd list is full , Cannot enter more
values")
```

```
            break
        else:
            if int(num) % 2 == 0:
                print("number is even")
                if len(even_lst) == 5:
                    print("Even list is full , sorry cannot enter more")
                else:
                    even_lst.append(num)
                print("even list : %s" % len(even_lst))
                print("odd list : %s" % (len(odd_lst)))
            else:
                print("number is odd")
                if len(odd_lst) == 5:
                    print("Odd list is full , sorry cannot enter more")
                else:
                    odd_lst.append(num)
                print("even list : %s" % (len(even_lst)))
                print("odd list : %s" % (len(odd_lst)))

            print("enter any number again !!")
            num = input()


    print("Even List : %s"%even_lst)
    print("max number from Even list : %s" % (max(even_lst)))
    print("Sum of all numbers in EVEN list is: %s"%(get_sum(even_lst)))

    print("Odd List : %s"%odd_lst)
    print("max number from Odd list : %s" %(max(odd_lst)))
    print("Sum of all numbers in ODD list is: %s"%(get_sum(odd_lst)))

geteven_odd()
```

Output:

```
enter any number
2
-------
number is even
even list : 1
```

```
odd list : 0
enter any number again !!
3
-------
number is odd
even list : 1
odd list : 1
enter any number again !!
4
-------
number is even
even list : 2
odd list : 1
enter any number again !!
5
-------
number is odd
even list : 2
odd list : 2
enter any number again !!
6
-------
number is even
even list : 3
odd list : 2
enter any number again !!
7
-------
number is odd
even list : 3
odd list : 3
enter any number again !!
8
-------
number is even
even list : 4
odd list : 3
enter any number again !!
9
-------
```

```
number is odd
even list : 4
odd list : 4
enter any number again !!
2
-------
number is even
even list : 5
odd list : 4
enter any number again !!
5
-------
number is odd
even list : 5
odd list : 5
enter any number again !!
13
-------
both even list and odd list is full , Cannot enter more values
Even List : ['2', '4', '6', '8', '2']
max number from Even list : 8
Sum of all numbers in EVEN list is: 22
Odd List : ['3', '5', '7', '9', '5']
max number from Odd list : 9
Sum of all numbers in ODD list is: 29
```

Write a program to find out the occurrence of a specific word from an alphanumeric statement. Example: 12abcbacbaba344ab

Output: a=5 b=5 c=2 make sure you should avoid the numbers in you logic

```
word = "12abcbacbaba344ab "

print("Word a has occured {0} times".format(word.count('a')))
print("Word b has occured {0} times".format(word.count('b')))
print("Word c has occured {0} times".format(word.count('c')))
```

Output:

```
Word a has occured 5 times
Word b has occured 5 times
Word c has occured 2 times
```

Generate and print another tuple whose values are even numbers in the given tuple (1,2,3,4,5,6,7,8,9,10)

```
tuple_given = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
list = []
print(type(tuple_given))

for data in tuple_given:
   if data % 2 == 0:
      list.append(data)
   else:
      pass
list = tuple(list)
print(list)
print(type(list))
```

Output:

```
(2, 4, 6, 8, 10)
<class 'tuple'>
```