Python Data Structures

Types of Data Structures:

- List | Tuple
- Set | Dictionary

Data Structures type	Mutable	Comments	Indexing	Ordered	Duplicacy
tuple ()	immutable	immutable version of list	possible	yes	allowed
list []	mutable	-	possible	yes	allowed
set {}	mutable	-	not	no	not
dict {key:value}	mutable	-	possible	no	not

💫 immutable => can't be changed
💫 mutable => can be changed

In [ ]:

```
```

Lists

Create an empty list with the name 'a', print the value of a and type(a).

In [1]:

```
# create empty list, name it 'a'
a=[]
```

In [2]:

```
# print the value of a

a
```

Out[2]:

```
[]
```

In [3]:

```
# print the type of a
type(a)
```

Out[3]:

```
list
```

Create a list , languages = ['R','Python', 'SAS', 'Scala', 42],

In [4]:

```
languages = ['R','Python', 'SAS', 'Scala', 42]
```

Print the number of elements in the list

In [5]:

```
type(languages)
```

Out[5]:

```
list
```

Using for loop iterate and print all the elements in the list

In [6]:

```
for i in languages:
    print(i)
```

```
R
Python
SAS
Scala
42
```

Select the second item, 'Python' and store it in a new variable named 'temp'

In [7]:

```
temp = languages[1]
```

Print the value of temp and type(temp)

In [8]:

```
print(temp)
type(temp)
```

```
Python
```

Out[8]:

```
str
```

Append the element 'Java' in the list

In [12]:

```
languages.append("java")
```

Remove the element 42 from the list and print the list

In [13]:

```
languages.remove(42)

print(languages)
```

```
['R', 'Python', 'SAS', 'Scala', 'java']
```

Create a list, colors = [‘Red’, ‘Blue’, ‘White’]

In [15]:

```
colors = ['Red', 'Blue', 'White']
```

Append the element 'Black' to colors

In [18]:

```
colors.append('Black')
colors
```

Out[18]:

```
['Red', 'Orange', 'Blue', 'White', 'Black']
```

Append the color 'Orange' to second position (index=1) and print the list

In [19]:

```
colors.insert(1,'Orange')
colors
```

Out[19]:

```
['Red', 'Orange', 'Orange', 'Blue', 'White', 'Black']
```

Print the list

In [20]:

```
colors
```

Out[20]:

```
['Red', 'Orange', 'Orange', 'Blue', 'White', 'Black']
```

Create another list, colors2 = [‘Grey’, ‘Sky Blue’]

In [21]:

```
colors2 = ['Grey', 'Sky', 'Blue']
```

Add the elements of colors2 to colors using extend function in the list

In [22]:

```
colors.extend(colors2)
colors
```

Out[22]:

```
['Red', 'Orange', 'Orange', 'Blue', 'White', 'Black', 'Grey', 'Sky', 'Blu
e']
```

Print len of colors and its elements

In [23]:

```
len(colors)
```

Out[23]:

```
9
```

Sort the list and print it.

In [24]:

```
colors.sort()
colors
```

Out[24]:

```
['Black', 'Blue', 'Blue', 'Grey', 'Orange', 'Orange', 'Red', 'Sky', 'Whit
e']
```

Create a string, sent = ‘Coronavirus Caused Lockdowns Around The World.”

In [25]:

```
sent = "Coronavirus Caused Lockdowns Around The World."
```

Use split function to convert the string into a list of words and save it in variable words and print the same

In [26]:

```
words = sent.split(', ')
words
```

Out[26]:

```
['Coronavirus Caused Lockdowns Around The World.']
```

Convert each word in the list to lower case and store it in variable words_lower. Print words_lower

In [27]:

```
words_lower = [i.lower() for a,i in enumerate(words)]
print(words_lower)
```

```
['coronavirus caused lockdowns around the world.']
```

Check whether ‘country’ is in the list

In [28]:

```
if words == 'country':
    print('country is present in list')
else:
    print('county is not present in list')
```

```
county is not present in list
```

Remove the element ‘the’ from the list and print the list.

In [29]:

```
words.pop(0)
```

Out[29]:

```
'Coronavirus Caused Lockdowns Around The World.'
```

Select the first 4 words from the list words_lower using slicing and store them in a new variable x4

In [30]:

```
x4 = words_lower[0:5]
```

In [31]:

```
# print x4

x4
```

Out[31]:

```
['coronavirus caused lockdowns around the world.']
```

Convert the list of elements to single string using join function and print it

In [ ]:

```

```

Sets

Create stud_grades = ['A','A','B','C','C','F']

In [32]:

```
stud_grades = ['A','A','B','C','C','F']
```

Print the len of stud_grades

In [33]:

```

stud_grades
```

Out[33]:

```
['A', 'A', 'B', 'C', 'C', 'F']
```

Create a new variable, stud_grades_set = set(stud_grades)

In [34]:

```
stud_grades_set = set(stud_grades)
```

Print stud_grades_set.

In [35]:

```
stud_grades_set
```

Out[35]:

```
{'A', 'B', 'C', 'F'}
```

print the type of stud_grades and stud_grades_set and print their corresponding elements. Try to understand the difference between them.

In [36]:

```
type(stud_grades_set)
```

Out[36]:

```
set
```

Add a new element ‘G’ to stud_grades_set

In [37]:

```
stud_grades_set.add("G")
```

Add element 'F' to stud_grades_set. and print it.

In [38]:

```
stud_grades_set.add("F")

stud_grades_set
```

Out[38]:

```
{'A', 'B', 'C', 'F', 'G'}
```

!!Did you notice? set doesn't add an element if it's already present in it, unlike lists.

Remove ‘F’ from stud_grades_set

In [39]:

```
stud_grades_set.remove("F")
```

Print the elements and the length of stud_grades_set

In [40]:

```
len( stud_grades_set)
```

Out[40]:

```
4
```

Create colors = [‘red’,’blue’,’orange’], and fruits = [‘orange’,’grapes’,’apples’]

In [41]:

```
colors = ['red','blue','orange']

fruits = ['orange','grapes','apples']
```

Print color and fruits

In [42]:

```
print(colors)
print(fruits)
```

```
['red', 'blue', 'orange']
['orange', 'grapes', 'apples']
```

Create colors_set, and fruits_set. (using set()) and print them

In [43]:

```
colors_set = set(colors)
fruits_set = set(fruits)
print(colors_set)
print(fruits_set)
```

```
{'blue', 'orange', 'red'}
{'grapes', 'orange', 'apples'}
```

Find the union of both the sets.

In [44]:

```
colors_set.union(fruits_set)
```

Out[44]:

```
{'apples', 'blue', 'grapes', 'orange', 'red'}
```

Find the intersection of both the sets

In [45]:

```
colors_set.intersection(fruits_set)
```

Out[45]:

```
{'orange'}
```

Find the elements which are Fruits but not colors (using set.difference())

In [46]:

```
fruits_set.difference(colors_set)
```

Out[46]:

```
{'apples', 'grapes'}
```

In [ ]:

```

```

TUPLES

Create temp = [17, 'Virat', 50.0]

In [50]:

```
temp = [17, 'Virat', 50.0]
```

Iterate through temp and print all the items in temp

In [51]:

```
temp
```

Out[51]:

```
[17, 'Virat', 50.0]
```

replace first element with 11 in temp

In [52]:

```
temp[0] = 11
```

Set temp1 = tuple(temp)

In [53]:

```
temp1 = tuple(temp)
```

Iterate through temp1 and print all the items in temp1.

In [54]:

```
print(temp1)
```

```
(11, 'Virat', 50.0)
```

replace first element with 17 in temp1

In [ ]:

```
temp1[0] = 17
```

Oops!! You got an error. Hey Don't worry! Its because Once a tuple is created, you cannot change its values unlike list.

Create city = ("Bangalore", 28.9949521, 72)

In [56]:

```
city = ("Bandlore", 28.9949521, 72)
```

Print first element of city

In [57]:

```
print(city[0])
```

```
Bandlore
```

Create city2 = ('Chennai', 30.01, 74)

In [58]:

```
city2 = ('Chennai', 30.01,74)
```

Create cities which consist of city and city2

In [59]:

```
cities = city,city2
```

Print cities

In [60]:

```
print(cities)
```

```
(('Bandlore', 28.9949521, 72), ('Chennai', 30.01, 74))
```

Print type of first element in cities

In [61]:

```
print(type(cities[0]))
```

```
<class 'tuple'>
```

print the type of cities

In [62]:

```
print(type(cities))
```

```
<class 'tuple'>
```

Hey that implies you made a nested tuples!!

DICTIONARY

Create a dictionary d = {"actor":"amir","animal":"cat","earth":2,"list": [23,32,12]}

In [65]:

```
d = {"actor":"amir","animal":"cat","earth":2,"list":[23,32,12]}
```

Print the value of d[0]

In [ ]:

```
print(d[0])
```

Oops!! again an error. again a fun fact. Dictionary return the value for key if key is in the dictionary, else throws KeyError and we don't have key 0 here :(

Store the value of d['actor'] to a new variable actor.

In [69]:

```
d['actor'] = 'actor'
```

Print the type of actor

In [70]:

```
print(type('actor'))
```

```
<class 'str'>
```

Store the value of d['list'] in new variable List.

In [71]:

```
List = d['list']
```

Print the type of List.

In [72]:

```
print(List)
```

```
[23, 32, 12]
```

Create d1 = { 'singer' : 'Kr$na' , 'album': 'Still here', 'genre' : 'hip-hop'}

In [73]:

```
d1 = {'singer':'Kr$na','album':'Still here','genre':'hip-hop'}
```

Merge d1 into d.

In [74]:

```
def Merge(d1,d):
    return(d1.update(d))
d = {"actor":"amir","animal":"cat","earth":2,"list":[23,32,12]}
d1 = {'singer':'Kr$na','album':'Still here','genre':'hip-hop'}

print(Merge(d,d1))

print(d1)
```

```
None
{'singer': 'Kr$na', 'album': 'Still here', 'genre': 'hip-hop'}
```

print d

In [75]:

```
d
```

Out[75]:

```
{'actor': 'amir',
 'animal': 'cat',
 'earth': 2,
 'list': [23, 32, 12],
 'singer': 'Kr$na',
 'album': 'Still here',
 'genre': 'hip-hop'}
```

Print all the keys in d

In [76]:

```
print(d.keys)
```

```
<built-in method keys of dict object at 0x000002344D533C80>
```

Print all the values in d

In [77]:

```
print(d.values)
```

```
<built-in method values of dict object at 0x000002344D533C80>
```

Iterate over d, and print each key, value pair as this - (actor ----> amir)

In [78]:

```
d
```

Out[78]:

```
{'actor': 'amir',
 'animal': 'cat',
 'earth': 2,
 'list': [23, 32, 12],
 'singer': 'Kr$na',
 'album': 'Still here',
 'genre': 'hip-hop'}
```

count the number of occurences of charachters in string named "sent" using dictionary and print the same.

In [79]:

```
sent
```

Out[79]:

```
'Coronavirus Caused Lockdowns Around The World.'
```