Python Guidelines for Begineers **DAY - 2**

Day-2 Activities

- List
- Set
- Tuple
- Dictionary
- Function

LIST

List is a collection which is ordered and changeable. Allows duplicate members.

In [1]:

```
#List Declaration
a = []
a
```

Out[1]:

```
[]
```

In [2]:

```
print(type(a))
```

```
<class 'list'>
```

In [3]:

```
a = ['onion', 'potato', 'tomato']
a
```

Out[3]:

```
['onion', 'potato', 'tomato']
```

In [4]:

```
# we can store different kinds of data in one list.

a = ['onion', 'potato', 'tomato', 1, 100.2 ]
a
```

Out[4]:

```
['onion', 'potato', 'tomato', 1, 100.2]
```

In [6]:

```
# We can accees them by using index
a[2]
```

Out[6]:

```
'tomato'
```

In [7]:

```
# 0 to 5 index
a = ['onion', 'potato', 'tomato', 1, 100.2,6,50 ]
a[1:5]
```

Out[7]:

```
['potato', 'tomato', 1, 100.2]
```

In [9]:

```
#Negative Indexing
a[-2]
```

Out[9]:

```
6
```

In [11]:

```
# returns the items from index -4 (included) to index -1 (excluded)
a[-4:]
```

Out[11]:

```
[]
```

In [12]:

```
#update list
a[0]="Bangladesh"

a[3]= 50000

a[4]= "India"
a
```

Out[12]:

```
['Bangladesh', 'potato', 'tomato', 50000, 'India', 6, 50]
```

In [13]:

```
#Adding new elements

a.(append'python')
a
```

Out[13]:

```
['Bangladesh', 'potato', 'tomato', 50000, 'India', 6, 50, 'python']
```

In [14]:

```
a.insert(1, 'google')
a
```

Out[14]:

```
['Bangladesh', 'google', 'potato', 'tomato', 50000, 'India', 6, 50, 'python']
```

In [15]:

```
# Adding many value at a time
a.extend([10, 'b', 'c'])
a
```

Out[15]:

```
['Bangladesh',
 'google',
 'potato',
 'tomato',
 50000,
 'India',
 6,
 50,
 'python',
 10,
 'b',
 'c']
```

In [16]:

```
a= a + ['K', 'L', 'M']
a
```

Out[16]:

```
['Bangladesh',
 'google',
 'potato',
 'tomato',
 50000,
 'India',
 6,
 50,
 'python',
 10,
 'b',
 'c',
 'K',
 'L',
 'M']
```

In [17]:

```
#Delete item

del a[0]
a
```

Out[17]:

```
['google',
 'potato',
 'tomato',
 50000,
 'India',
 6,
 50,
 'python',
 10,
 'b',
 'c',
 'K',
 'L',
 'M']
```

In [18]:

```
a.remove("python")
a
```

Out[18]:

```
['google',
 'potato',
 'tomato',
 50000,
 'India',
 6,
 50,
```

```
 10,
 'b',
 'c',
 'K',
 'L',
 'M']
```

In [19]:

```
a.append("python")
a
```

Out[19]:

```
['google',
 'potato',
 'tomato',
 50000,
 'India',
 6,
 50,
 10,
 'b',
 'c',
 'K',
 'L',
 'M',
 'python']
```

In [20]:

```
a.append("python")
a
```

Out[20]:

```
['google',
 'potato',
 'tomato',
 50000,
 'India',
 6,
 50,
 10,
 'b',
 'c',
 'K',
 'L',
 'M',
 'python',
 'python']
```

In [24]:

```
a.pop()
a
```

Out[24]:

```
['python',
 'M',
 'L',
 'K',
 'c',
 'b',
 10,
 50,
 6,
 'India',
 50000,
 'tomato',
 'potato']
```

In [21]:

```
a.remove("python")
a
```

Out[21]:

```
['google',
 'potato',
 'tomato',
 50000,
 'India',
 6,
 50,
 10,
 'b',
 'c',
 'K',
 'L',
 'M',
 'python']
```

In [22]:

```
len(a)
```

Out[22]:

```
14
```

In [23]:

```
a.reverse()
a
```

Out[23]:

```
['python',
 'M',
 'L',
 'K',
 'c',
 'b',
 10,
 50,
 6,
 'India',
 50000,
 'tomato',
 'potato',
 'google']
```

In [25]:

```
a.sort()
a
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-25-3ce1b49a4cfa> in <module>
----> 1 a.sort()
      2 a

TypeError: '<' not supported between instances of 'int' and 'str'
```

In [26]:

```
b= ['L', 'K', 'c', 'b', 'a', 'India']
b.sort()
b
```

Out[26]:

```
['India', 'K', 'L', 'a', 'b', 'c']
```

In [27]:

```
c= [70 , 50 , 1 ,4, 10]
c.sort()
c
```

Out[27]:

```
[1, 4, 10, 50, 70]
```

Tuple

Tuple is also one kinds of list but we cannot modify tuple

Tuple is a collection which is ordered and unchangeable. Allows duplicate members.

In [28]:

```
a =()
type(a)
```

Out[28]:

```
tuple
```

In [29]:

```
a = ('onion', 'potato')
a
```

Out[29]:

```
('onion', 'potato')
```

In [30]:

```
a = ('onion', 'potato', 1 , 2)
a
```

Out[30]:

```
('onion', 'potato', 1, 2)
```

In [31]:

```
a[2]
```

Out[31]:

```
1
```

we cannot change or modify tuple

Once a tuple is created, you cannot add items to it becausetuples are unchangeable.

In [32]:

```
a[0]="Bangladesh"
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-32-1ed2d448dd8d> in <module>
----> 1 a[0]="Bangladesh"

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

In [33]:

```
a[4]=2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-33-2ab4f406c412> in <module>
----> 1 a[4]=2

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

In [34]:

```
len(a)
```

Out[34]:

```
4
```

In [35]:

```
a[-1]
```

Out[35]:

```
2
```

In [36]:

```
a[-3:-1]
```

Out[36]:

```
('potato', 1)
```

In [37]:

```
b=('a',1,'e',2,'u',5,'o',60,'p')
b[-5:-2]
```

Out[37]:

```
('u', 5, 'o')
```

In [38]:

```
#Join Two Tuples
c=a+b
c
```

Out[38]:

```
('onion', 'potato', 1, 2, 'a', 1, 'e', 2, 'u', 5, 'o', 60, 'p')
```

In [39]:

```
#Tuple count() Method
#Returns the number of times a specified value occurs in a tuple
d=c.count(1)
d
```

Out[39]:

```
2
```

Set

Set is a collection which is unordered and unindexed. No duplicate members.

- Set in Python is a data type, much like a list.
- The difference between a set and a list is that the same item can be on the list more than once but only one member or item can be on the set once.
- The most interesting thing is that in mathematics all the operations that we used to do in the set (eg: union, intersection, difference) can be done in Python.

In [40]:

```
A = {'orange', 'banana', 'pear', 'mango'}
A
```

Out[40]:

```
{'banana', 'mango', 'orange', 'pear'}
```

Find distinct elements using set

In [41]:

```
A = set('abracadabra')
A
```

Out[41]:

```
{'a', 'b', 'c', 'd', 'r'}
```

Set elements cannot be accessed with index numbers.

In [42]:

```
A[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-42-782c0e2fe61c> in <module>
----> 1 A[0]

TypeError: 'set' object is not subscriptable
```

In [43]:

```
#But you can loop through the set items using a for loop
for x in A:
  print(x)
```

```
d
c
r
b
a
```

Add new elements using add

In [44]:

```
A.add('pineapple')
A
```

Out[44]:

```
{'a', 'b', 'c', 'd', 'pineapple', 'r'}
```

Add more than one elements

In [45]:

```
A.update({'berry'}, {'grape'})
A
```

Out[45]:

```
{'a', 'b', 'berry', 'c', 'd', 'grape', 'pineapple', 'r'}
```

remove elements

In [46]:

```
A.remove("grape")
A
```

Out[46]:

```
{'a', 'b', 'berry', 'c', 'd', 'pineapple', 'r'}
```

Union , Intersection

In [47]:

```
A = {1, 2, 3, 4, 5}
B= { 5, 6, 7, 8,9}
A.union(B)
```

Out[47]:

```
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```

In [48]:

```
A = {1, 2, 3, 4, 5}
B= { 5, 6, 7, 8,9}
A.intersection(B)
```

Out[48]:

```
{5}
```

In [49]:

```
A = {1, 2, 3, 4, 5}
B= { 5, 6, 7, 8,9}
A.difference(B)
```

Out[49]:

```
{1, 2, 3, 4}
```

Dictionary

A dictionary is a collection which is unordered, changeable and indexed. In Python dictionaries are written with curly brackets, and they have keys and values.

In [50]:

```
D = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(D)
```

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

In [51]:

```
D['brand']
```

Out[51]:

```
'Ford'
```

Update

In [52]:

```
D['year']=2012
```

In [53]:

```
D
```

Out[53]:

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 2012}
```

Add new item

In [54]:

```
D['color']= "Red"
D
```

Out[54]:

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 2012, 'color': 'Red'}
```

Remove and Clear

In [55]:

```
del D['year']
D
```

Out[55]:

```
{'brand': 'Ford', 'model': 'Mustang', 'color': 'Red'}
```

In [56]:

```
D.clear()
D
```

Out[56]:

```
{}
```

get key and has key

In [57]:

```
D = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
D.get("brand")
```

Out[57]:

```
'Ford'
```

In [60]:

```
#has

"brand" in D
```

Out[60]:

```
True
```

In [61]:

```
#Find keys
D.keys()
```

Out[61]:

```
dict_keys(['brand', 'model', 'year'])
```

In [62]:

```
#find values
D.values()
```

Out[62]:

```
dict_values(['Ford', 'Mustang', 1964])
```

In [63]:

```
uname=input("Enter your username :")
passwd=input("Enter your password :")
match={
    "uname":"jaima",
    "passwd":"123"
}
s=match.get("uname")
p=match.get("passwd")
if s==uname:
    print("Congratulation")
else:
    print("Sorry!Password or username incorrect!!!!")
```

```
Enter your username :jaima
```

```
Enter your password :123
Congratulation
```

Function

- A function is a block of code which only runs when it is called.
- You can pass data, known as parameters, into a function.
- A function can return data as a result.

In [ ]:

```
def my_function():
  print("Hello from a function")

my_function()
```

In [ ]:

```
# Function with perameter:

def my_function(fname):
  print(fname + " World")

my_function("Hello")
my_function("Goodbye")
```

In [ ]:

```
def add(a, b, c):
    return a+b+c
temp = add(1, 2, 3)
print(temp)
```

In [ ]:

```
# Find Error
def add(a, b, c):
    return a+b+c
temp = add(1, 2)
print(temp)
```

Arbitrary Arguments, *args

If you do not know how many arguments that will be passed into your function, add a * before the parameter name in the function definition.

function will receive a tuple of arguments, and can access the items accordingly:

In [7]:

```
def my_function(*kids):
  print("The youngest child is " + kids[1])

my_function("A", "B","c","d")
```

```
The youngest child is B
```

In [ ]:

```
#Keyword Arguments
#send arguments with the key = value syntax
def my_function(child3, child2, child1):
```

```
  print("The youngest child is " + child3)

my_function(child1 = "A", child2 = "B", child3 = "C")
```

In [ ]:

```
#Arbitrary Keyword Arguments, **kwargs
#If the number of keyword arguments is unknown, add a double ** before the parameter name
def my_function(**kid):
  print("His last name is " + kid["lname"])

my_function(fname = "A", lname = "Z")
```

In [8]:

```
#Default Parameter Value
def my_function(country = "Norway"):
  print("I am from " + country)

my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")
```

```
I am from Sweden
I am from India
I am from Norway
I am from Brazil
```

In [ ]:

```
def factorial(number):
    if number == 0:
        return 1
    else:
        return number * factorial(number - 1)

print('Please input your number:')
number = int(input())
print(factorial(number))
```

Thank you