Lists

In [1]:

```
#creating list
my_shopping_list = ['apple', 'banana', 'mango']
print(my_shopping_list)
```

```
['apple', 'banana', 'mango']
```

In [2]:

```
#adding an element at end
my_shopping_list.append('pineapple')
print(my_shopping_list)
```

```
['apple', 'banana', 'mango', 'pineapple']
```

In [3]:

```
#adding multiple element
my_shopping_list.extend(['table', 'chair', 'fan'])
print(my_shopping_list)
```

```
['apple', 'banana', 'mango', 'pineapple', 'table', 'chair', 'fan']
```

In [4]:

```
#adding element at perticular location
my_shopping_list.insert(1, 'cake')
print(my_shopping_list)
```

```
['apple', 'cake', 'banana', 'mango', 'pineapple', 'table', 'chair', 'fan']
```

In [5]:

```
#remove an elemenet
my_shopping_list.remove('chair')
print(my_shopping_list)
```

```
['apple', 'cake', 'banana', 'mango', 'pineapple', 'table', 'fan']
```

In [6]:

```
#remove an element and return it
my_fourth_element = my_shopping_list.pop(4)
print(my_shopping_list)
print(my_fourth_element)
```

```
['apple', 'cake', 'banana', 'mango', 'table', 'fan']
pineapple
```

In [7]:

```
#make the list empty
my_shopping_list.clear()
print(my_shopping_list)
```

```
[]
```

In [8]:

```
#deducing index of an element
my_shopping_list = ['apple', 'cake', 'banana', 'mango', 'pineapple', 'table', 'chair', '
fan', 'cake']
index_of_mango = my_shopping_list.index('mango')
print(index_of_mango)
```

```
3
```

In [9]:

```
#counting an element
num_cake = my_shopping_list.count('cake')
print(num_cake)
```

```
2
```

In [10]:

```
#sorting the list
my_shopping_list.sort()
print(my_shopping_list)
```

```
['apple', 'banana', 'cake', 'cake', 'chair', 'fan', 'mango', 'pineapple', 'table']
```

In [11]:

```
#reversing the list
my_shopping_list.reverse()
print(my_shopping_list)
```

```
['table', 'pineapple', 'mango', 'fan', 'chair', 'cake', 'cake', 'banana', 'apple']
```

In [12]:

```
#boolean operation
my_bollean_list = [True, False, True]
all(my_bollean_list)
```

Out[12]:

```
False
```

In [13]:

```
#boolean operation
any(my_bollean_list)
```

Out[13]:

```
True
```

In [14]:

```
#enumerating the list
for index, name in enumerate(my_shopping_list):
    print(f'element at {index}: {name}')
```

```
element at 0: table
element at 1: pineapple
element at 2: mango
element at 3: fan
element at 4: chair
element at 5: cake
element at 6: cake
element at 7: banana
element at 8: apple
```

In [15]:

```
#length of the list
len_of_shopping = len(my_shopping_list)
print(f'number of elemenets in shopping cart is {len_of_shopping}')
```

```
number of elemenets in shopping cart is 9
```

In [16]:

```
#converting to the list
```

```
my_tuple = ('Gaurav', 'Kumar')
my_list = list(my_tuple)
print(my_list)
print(type(my_list))
```

```
['Gaurav', 'Kumar']
<class 'list'>
```

In [17]:

```
#max, min, sort, sum a list
my_numerical_list = [9,7,4,2,3,4,1,2,0,1]
print(f'numerical list: {my_numerical_list}')
print(f'max number: {max(my_numerical_list)}')
print(f'min number: {min(my_numerical_list)}')
print(f'sorted numbers: {sorted(my_numerical_list)}')
print(f'sum of numbers: {sum(my_numerical_list)}')
```

```
numerical list: [9, 7, 4, 2, 3, 4, 1, 2, 0, 1]
max number: 9
min number: 0
sorted numbers: [0, 1, 1, 2, 2, 3, 4, 4, 7, 9]
sum of numbers: 33
```

In [18]:

```
#change an element
my_shopping_list = ['apple', 'banana', 'mango']
my_shopping_list[-1] = 'mangoes'
my_shopping_list
```

Out[18]:

```
['apple', 'banana', 'mangoes']
```

In [19]:

```
#delete element in list
del my_shopping_list[-1]
my_shopping_list
```

Out[19]:

```
['apple', 'banana']
```

Tuple

In [20]:

```
# empty tuple
my_tuple = ()
```

In [21]:

```
#creating a single element tuple
my_tuple = (1,)
my_tuple
```

Out[21]:

```
(1,)
```

In [22]:

```
# creating a tuple from a set
my_tuple = tuple({1, 2,3, 2})
my_tuple
```

Out[22]:

```
(1, 2, 3)
```

In [23]:

```
# accessing first element
my_tuple[0]
```

Out[23]:

```
1
```

In [24]:

```
#Checks whether an element exists in the specified tuple.
print(4 in my_tuple)
```

```
False
```

In [25]:

```
#Checks whether an element does not exist in the specified tuple.
print(4 not in my_tuple)
```

```
True
```

In [26]:

```
#counting number of a perticular element
my_tuple.count(2)
```

Out[26]:

```
1
```

In [27]:

```
#index of an element
my_tuple.index(3)
```

Out[27]:

```
2
```

In [28]:

```
#selecting last two elements
my_tuple = (1, 2, 3, 4, 1+3j)
my_tuple[-2:]
```

Out[28]:

```
(4, (1+3j))
```

In [29]:

```
#selecting everything except the first and last element
my_tuple[1:-1]
```

Out[29]:

```
(2, 3, 4)
```

In [30]:

```
#converting list into tuple
tuple(my_shopping_list)
```

Out[30]:

```
('apple', 'banana')
```

Dictionary

In [31]:

```
#Creating empty dictionary
my_dict = {}
my_dict
```

Out[31]:

```
{}
```

In [32]:

```
#creating dict
my_dict = {'name':'Gaurav', 'age':29,  'gender': 'male'}
my_dict
```

Out[32]:

```
{'name': 'Gaurav', 'age': 29, 'gender': 'male'}
```

In [33]:

```
#get an element
print(my_dict['name'])
print(my_dict.get('name'))
```

```
Gaurav
Gaurav
```

In [34]:

```
#change an element
my_dict['age'] = 28
my_dict
```

Out[34]:

```
{'name': 'Gaurav', 'age': 28, 'gender': 'male'}
```

In [35]:

```
#get all keys and values
print(my_dict.keys())
print(my_dict.values())
print(my_dict.items())
```

```
dict_keys(['name', 'age', 'gender'])
dict_values(['Gaurav', 28, 'male'])
dict_items([('name', 'Gaurav'), ('age', 28), ('gender', 'male')])
```

In [36]:

```
# remove all elemeent
my_dict.clear()
my_dict
```

Out[36]:

```
{}
```

In [37]:

```
# Returns a copy of the dictionary
my_dict = {'name':'Gaurav', 'age':29,  'gender': 'male'}
my_new_dict = my_dict.copy()
my_new_dict
```

Out[37]:

```
{'name': 'Gaurav', 'age': 29, 'gender': 'male'}
```

In [38]:

```
# Returns a dictionary with the specified keys and value
```

```
my_dict.fromkeys(['age', 'name'])
```

Out[38]:

```
{'age': None, 'name': None}
```

In [39]:

```
# Remove an element by key
my_dict.pop('age')
my_dict
```

Out[39]:

```
{'name': 'Gaurav', 'gender': 'male'}
```

In [40]:

```
#remove last inserted key - value pair
my_dict.popitem()
my_dict
```

Out[40]:

```
{'name': 'Gaurav'}
```

In [41]:

```
# Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
my_dict = {'name':'Gaurav', 'age':29,  'gender': 'male'}
my_dict.setdefault('height', 5.5)
my_dict
```

Out[41]:

```
{'name': 'Gaurav', 'age': 29, 'gender': 'male', 'height': 5.5}
```

In [42]:

```
# update a dictionary
my_dict.update({'age': 28, 'sex': 'male'})
my_dict
```

Out[42]:

```
{'name': 'Gaurav', 'age': 28, 'gender': 'male', 'height': 5.5, 'sex': 'male'}
```

In [43]:

```
#del by a key
del my_dict['age']
my_dict
```

Out[43]:

```
{'name': 'Gaurav', 'gender': 'male', 'height': 5.5, 'sex': 'male'}
```

Set

In [44]:

```
#create empty set
my_set = {}
```

In [45]:

```
#create a set
set_1 = {1, 2, 3, 4}
set_2 = {3, 4, 5, 6}
print(set_1)
print(set_2)
```

```
{1, 2, 3, 4}
{3, 4, 5, 6}
```

In [46]:

```
# add an element
my_set = {'apple', 'banana'}
my_set.add('mango')
my_set
```

Out[46]:

```
{'apple', 'banana', 'mango'}
```

In [47]:

```
# add multiple elements
my_set.update(['cold-drink', 'tea'])
my_set
```

Out[47]:

```
{'apple', 'banana', 'cold-drink', 'mango', 'tea'}
```

In [48]:

```
# clear all element
my_set.clear()
my_set
```

Out[48]:

```
set()
```

In [49]:

```
# Removes the specified element
my_set = {'apple', 'banana'}
my_set.remove('apple')
my_set
```

Out[49]:

```
{'banana'}
```

In [50]:

```
# Remove the specified item
my_set.remove('banana')
my_set
```

Out[50]:

```
set()
```

In [51]:

```
# Removes and return an element from the set
my_set = {'apple', 'banana'}
pop = my_set.pop()
print(pop)
print(my_set)
```

```
banana
{'apple'}
```

In [52]:

```
# Returns a set containing the difference between two or more sets
set_1 = {1, 2, 3, 4}
set_2 = {3, 4, 5, 6}
print(set_1)
```

```
print(set_2)
print(set_1 - set_2)
```

```
{1, 2, 3, 4}
{3, 4, 5, 6}
{1, 2}
```

In [53]:

```
# Returns a copy of the set
my_new_set = my_set.copy()
my_new_set
```

Out[53]:

```
{'apple'}
```

In [54]:

```
# Return a set containing the union of sets
print(set_1 | set_2)
```

```
{1, 2, 3, 4, 5, 6}
```

In [55]:

```
# Returns a set, that is the intersection of two other sets
print(set_1 & set_2)
```

```
{3, 4}
```

In [56]:

```
# Returns a set with the symmetric differences of two sets
print(set_1 ^ set_2)
```

```
{1, 2, 5, 6}
```