In [1]:

```
#Lista - 1 - Create una lista e provate a usare l'operatore “del” su una slice

list =  [1, 2, 3, 4, 5]

#Slice - Affetta la sequenza, specificando anche il punto di inzio e di fine. Restituisce un oggetto slice.

sublist = list[1:5]
print(str(sublist) + "\n")

#Del - Cancella gli elementi, specificando anche il punto di inzio e di fine.

del sublist[1:3]
print(sublist)
```

```
[2, 3, 4, 5]

[2, 5]
```

In [2]:

```
#Lista - 2 - Create una lista e provate ad utilizzare i metodi: insert(), count(), extend(), index(), sort(), remove()

list = ["Nicholas", "Fabio", "Alessandro"]
print("Lista 1: " + str(list) + "\n")

#Insert - Inserisce un elemento nella lista in una specifica posizione

list.insert(1, "Nicholas")
print("Lista 1: " + str(list) + "\n")

#Count - Restituisce il numero di elementi con il valore specificato

print(str(list.count("Nicholas")) + "\n")

#Extend - Aggiunge tutti gli elementi di un iterabile(lista, tupla, stringa, ecc...) alla fine della lista

list2 = ["Federica", "Laura", "Andrea"]
print("Lista 2: " + str(list2))
list.extend(list2)
print("Lista unita: " + str(list) + "\n")

#Index - Restituisce l'indice dell'elemento dato nella lista, viene lanciato un ValueError.

print(str(list.index("Federica")) + "\n")

#Sort - Ordina la lista. Si può anche creare una funzione per decidere i criteri di ordinamento

list = sorted(list, reverse=True) #Ordine discendente
print(list)
list = sorted(list, reverse=False) #Ordine ascendente
print(str(list) + "\n")

#Remove - Prende come argomento un singolo elemento e lo toglie dalla lista

list.remove("Nicholas")
print(str(list) + "\n")
```

```
Lista 1: ['Nicholas', 'Fabio', 'Alessandro']

Lista 1: ['Nicholas', 'Nicholas', 'Fabio', 'Alessandro']

2

Lista 2: ['Federica', 'Laura', 'Andrea']
Lista unita: ['Nicholas', 'Nicholas', 'Fabio', 'Alessandro', 'Federica', 'Laura', 'Andrea']

4

['Nicholas', 'Nicholas', 'Laura', 'Federica', 'Fabio', 'Andrea', 'Alessandro']
['Alessandro', 'Andrea', 'Fabio', 'Federica', 'Laura', 'Nicholas', 'Nicholas']

['Alessandro', 'Andrea', 'Fabio', 'Federica', 'Laura', 'Nicholas']
```

In [3]:

```
#Tupla - 1 - Create una tupla e provate a fare un'operazione di slicing

my_tuple = (1,2,3,4,5,6,7)
sliced_tuple = my_tuple[1:5]
print(str(sliced_tuple) + "\n")
```

```
(2, 3, 4, 5)
```

In [4]:

```
#Tupla - 2 - Provate a chiamare la funzione type() sui seguenti due oggetti: tupla_test = (11), tupla_test = (11,)

tupla_test = (11)
print(str(type(tupla_test)) + "\n")
tupla_test = (11,)
print(type(tupla_test))
```

```
<class 'int'>

<class 'tuple'>
```

In [5]:

```
#Set - 1.1 - Create un set con un elemento di tipo set contenente un elemento di tipo str

s = {"Ciao"}
print(s)
print(type(s))
```

```
{'Ciao'}
<class 'set'>
```

In [6]:

```
#Set - 1.2 - Create un set con un elemento di tipo set contenente un elemento di tipo tuple

s = {(1, 2)}
print(s)
print(type(s))
```

```
{(1, 2)}
<class 'set'>
```

In [7]:

```
#Set - 1.3 - Create un oggetto {} e passatelo alla funzione type()

d = {}
print(type(d))
```

```
<class 'dict'>
```

In [8]:

```
#Set - 1.4 - Create un set e utilizzate i metodi pop() e clear()

s = {1, 2, 3}
print(s)

#Pop - Rimuove dalla lista l'elemento che si trova all'indice dato e restituisce l'elemento rimosso.
s.pop()
print(s)

#Clear - Rimuove tutti gli oggetti della lista
s.clear()
print(s)
```

```
{1, 2, 3}
{2, 3}
set()
```

In [9]:

```
#Set - 2 - Dati i due set: A = {1,2,3,4,5}, B = {4,5,6,7,8}
#svolgete le quattro operazioni tra set utilizzando i metodi union() intersection() difference() e symmetric_difference()

A = {1,2,3,4,5}
B = {4,5,6,7,8}

#Union - Restituisce un insieme che contiene tutti gli elementi tra due o più insiemi
print("L'unione dei due set è " + str(A.union(B)) + "\n")

# Intersection - Restituisce un insieme che contiene la somiglianza tra due o più insiemi
print("L'interesezione dei due set è " + str(A.intersection(B)) + "\n")

# Difference - Restituisce un insieme che è la differenza tra due insiemi
print("La differenza del set A con B è " + str(A.difference(B)))
print("La differenza del set B con A è " + str(B.difference(A)) + "\n")

#Symmetric Difference - Restituisce la differenza simmetrica di due insiemi
print("La differenza simmetrica dei due set è " + str(A.symmetric_difference(B)))
```

```
L'unione dei due set è {1, 2, 3, 4, 5, 6, 7, 8}

L'interesezione dei due set è {4, 5}

La differenza del set A con B è {1, 2, 3}
La differenza del set B con A è {8, 6, 7}

La differenza simmetrica dei due set è {1, 2, 3, 6, 7, 8}
```

In [10]:

```
#Dizionari - 1 - Create un dizionario e passatelo come argomento alla funzione len()

d = {'key1': 1, 'key2': 2, 'key3': 3}

#Len - Restituisce la lunghezza di un oggetto
print(len(d))
```

```
3
```

In [11]:

```
#Dizionari - 2 - Create un oggetto {} e riempitelo con due coppie chiave-valore

d = {}
d['key1'] = 1
d['key2'] = 2
print(d)
```

```
{'key1': 1, 'key2': 2}
```

In [12]:

```
#Dizionari - 3 - Create un dizionario e provate a utilizzare i metodi values(), items(), get()

d = {'key1': 1, 'key2': 2, 'key3': 3}

#Values - Restituisce un oggetto view con i valori del dizionario
print(d.values())

#Items - Restituisce un oggetto view che ontiene le coppie chiave-valore del dizionario, come tuple in un elenco
print(d.items())

#Get - Restituisce il valore dell'elemento con la chiave specificata
print(d.get('key2'))
```

```
dict_values([1, 2, 3])
dict_items([('key1', 1), ('key2', 2), ('key3', 3)])
2
```