In [2]:
```
first = (30,"scaler",5.8)
```

In [3]:
```
type(first)
```

Out[3]:
```
tuple
```

In [4]:
```
import sys
a_list = list()
a_tuple = tuple()
a_list = [1,2,3,4,5]
a_tuple = (1,2,3,4,5)
```

In [5]:
```
print(sys.getsizeof(a_list))
print(sys.getsizeof(a_tuple))
```

```
96
80
```

In [6]:
```
from timeit import timeit

times = 1000000

t1 = timeit("list(['apple', 'orange', 'banana'])", number=times)
print(f'Time to copy a list {times} times: {t1}')

t2 = timeit("tuple(('apple', 'orange', 'banana'))", number=times)
print(f'Time to copy a tuple {times} times: {t2}')
```

```
Time to copy a list 1000000 times: 0.39720189999979993
Time to copy a tuple 1000000 times: 0.17494200000010096
```

In [7]:
```
city = ("Bangalore", 28.9949521, 72)
print(city)
```

```
('Bangalore', 28.9949521, 72)
```

In [8]:
```
new = city
print(new)
```

```
('Bangalore', 28.9949521, 72)
```

In [9]:
```
id(new)
```

Out[9]:
```
2183729662528
```

In [10]:
```
id(city)
```

Out[10]:
```
2183729662528
```

In [11]:
```
city
```

Out[11]:
```
('Bangalore', 28.9949521, 72)
```

In [12]:
```
city[0] = 'mumbai'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-12-b54fd6768773> in <module>
----> 1 city[0] = 'mumbai'

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

In [13]:
```
10
```

Out[13]:
```
10
```

In [14]:
```
(10)
```

Out[14]:
```
10
```

In [15]:
```
[10]
```

Out[15]:
```
[10]
```

In [17]:
```
type([10])
```

Out[17]:
```
list
```

In [20]:
```
t = ((10,))
```

In [21]:
```
type(t)
```

Out[21]:
```
tuple
```

In [22]:
```
t3 = 10,20,30
```

In [23]:
```
t3
```

Out[23]:
```
(10, 20, 30)
```

In [24]:
```
a,b,c = t3
```

In [25]:
```
a
```

Out[25]: 10

In [26]:
```
b
```

Out[26]: 20

In [27]:
```
c
```

Out[27]: 30

In [28]:
```
tuple([1,2,3])
```

Out[28]: (1, 2, 3)

In [29]:
```
list((10,20,30))
```

Out[29]: [10, 20, 30]

Dictionary

name:scaler age:10 phone : .....

In [30]:
```
a = {}
```

In [31]:
```
type(a)
```

Out[31]: dict

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

In [33]:
```
d["actor"]
```

Out[33]: 'amir'

In [36]:
```
d.get('list')
```

Out[36]: [23, 32, 12]

In [40]:
```
a = d.get('earth1','NA')
print(a)
```

NA

In [41]:
```
d['earth1'] = 'singer'
```

In [42]:
```
d
```

Out[42]:
```
{'actor': 'amir',
 'animal': 'cat',
 'earth': 2,
 'list': [23, 32, 12],
 'earth1': 'singer'}
```

In [43]:
```
d.keys()
```

Out[43]:
```
dict_keys(['actor', 'animal', 'earth', 'list', 'earth1'])
```

In [44]:
```
d.values()
```

Out[44]:
```
dict_values(['amir', 'cat', 2, [23, 32, 12], 'singer'])
```

In [45]:
```
new = dict(Country='Honey Singh', Songs=['Blue Eyes','night party'])
```

In [46]:
```
d
```

Out[46]:
```
{'actor': 'amir',
 'animal': 'cat',
 'earth': 2,
 'list': [23, 32, 12],
 'earth1': 'singer'}
```

In [47]:
```
new
```

Out[47]:
```
{'Country': 'Honey Singh', 'Songs': ['Blue Eyes', 'night party']}
```

In [48]:
```
d.update(new)
d
```

Out[48]:
```
{'actor': 'amir',
 'animal': 'cat',
 'earth': 2,
 'list': [23, 32, 12],
 'earth1': 'singer',
 'Country': 'Honey Singh',
 'Songs': ['Blue Eyes', 'night party']}
```

In [50]:
```
for a,b in d.items():
    print(a,"+",b)
```

```
actor + amir
animal + cat
earth + 2
list + [23, 32, 12]
earth1 + singer
Country + Honey Singh
Songs + ['Blue Eyes', 'night party']
```

In [51]:
```
d.pop('earth1')
```

Out[51]:
```
'singer'
```

In [52]:
```
d
```

Out[52]:
```
{'actor': 'amir',
 'animal': 'cat',
 'earth': 2,
 'list': [23, 32, 12],
 'Country': 'Honey Singh',
 'Songs': ['Blue Eyes', 'night party']}
```

In [53]:
```
EMP_DB = {
 'HR' : {
 101 : 45000,
 116 : 34000
 },
 'TECH' : {
 918 : 60000,
 1001 : 75000,
 815 : 65000
 },
 'SALES' : {
 887 : 45000,
 490 : 63000
 }
}
```

In [54]:
```
d.pop()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-54-663961784a31> in <module>
----> 1 d.pop()

TypeError: pop expected at least 1 argument, got 0
```

In [55]:
```
EMP_DB
```

Out[55]:
```
{'HR': {101: 45000, 116: 34000},
 'TECH': {918: 60000, 1001: 75000, 815: 65000},
 'SALES': {887: 45000, 490: 63000}}
```

In [56]:
```
EMP_DB.keys()
```

Out[56]:
```
dict_keys(['HR', 'TECH', 'SALES'])
```

In [57]:
```
EMP_DB['TECH'][918]
```

Out[57]:
```
60000
```

In [58]:
```
EMP_DB.get('TECH').get(918)
```

Out[58]: 60000

Sets

In [59]:
```
grades = ['A','B','C','A', 'C']
```

In [60]:
```
g = set(grades)
```

In [61]:
```
g
```

Out[61]: {'A', 'B', 'C'}

In [62]:
```
stud_grades = ['A','A','B','C','C','F']
```

In [63]:
```
stud_grades = set(stud_grades)
```

In [64]:
```
print(type(stud_grades))
```

```
<class 'set'>
```

In [65]:
```
stud_grades2 = ['A','N','F','N','G','A']
```

In [66]:
```
stud_grades2 = set(stud_grades2)
stud_grades2
```

Out[66]: {'A', 'F', 'G', 'N'}

In [67]:
```
print(stud_grades.intersection(stud_grades2))
```

```
{'F', 'A'}
```

In [68]:
```
print(stud_grades.union(stud_grades2))
```

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

In [69]:
```
stud_grades
```

Out[69]: {'A', 'B', 'C', 'F'}

In [70]:
```
stud_grades2
```

Out[70]: {'A', 'F', 'G', 'N'}

In [71]:
```
print(stud_grades.difference(stud_grades2))
```

```
{'B', 'C'}
```

In [72]:
```
print(stud_grades2.difference(stud_grades))
```

```
{'G', 'N'}
```

In [73]:
```
stud_grades.add('G')
print(stud_grades)
```

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

In [74]:
```
stud_grades.remove('G')
print(stud_grades)
```

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

In [75]:
```
stud_grades[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-75-f791bf172131> in <module>
----> 1 stud_grades[0]

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

In [76]:
```
stud_grades.symmetric_difference(stud_grades2)
```

Out[76]:
```
{'B', 'C', 'G', 'N'}
```

In [77]:
```
def hello_world():
    print('Hello World')
```

In [78]:
```
hello_world()
```

```
Hello World
```

In [79]:
```
def cube(num):
    out = num**3
    return(out)
```

In [80]:
```
cube(3)
```

Out[80]:
```
27
```

In [88]:
```
def submition(*args):
    print(args[2])
    return(sum(args))
```

In [89]:
```
print(submition(1,2,3,4,6))
```

```
3
16
```

In [90]:
```
squares_list= []

for x in range(1,10):
    squares_list.append(x**2)
print(squares_list)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

In [91]:
```
squares_list = [x**2 for x in range(1,10)]
print(squares_list)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

In [92]:
```
import time
iterations = 100000000
start = time.time()
mylist = []
for i in range(iterations):
    mylist.append(i+1)
end = time.time()
print(end - start)
```

```
42.77441334724426
```

In [95]:
```
students_data = {1:['Shivam Bansal', 24] , 2:['Udit Bansal',25], 3:['Sonam Gupta', 26],
students_data
```

Out[95]:
```
{1: ['Shivam Bansal', 24],
 2: ['Udit Bansal', 25],
 3: ['Sonam Gupta', 26],
 4: ['Saif Ansari', 24],
 5: ['Huzefa Calcuttawala', 27]}
```

In [93]:
```
start = time.time()
mylist = [i+1 for i in range(iterations)]
end = time.time()
print(end - start)
```

```
28.678182125091553
```

In [96]:
```
names_dict ={}

#iterate over each key, val pair
for roll_num,details in students_data.items():
    if roll_num%2==0:
        names_dict[roll_num]= details[0]
print(names_dict)
```

```
{2: 'Udit Bansal', 4: 'Saif Ansari'}
```

In [101…
```
names_comp = {details[0] for roll_num,details in students_data.items() if roll_num%2==0
type(names_comp)
```

Out[101… set

In [102…
```
names_comp = {roll_num:details[0] for roll_num,details in students_data.items() if roll
type(names_comp)
```

Out[102… dict

In [ ]:
```
Break : 10 38
```

In [103…
```
paragraph = ["There was a fox." , 'It was brown in color.', "It was seen near that farm
```

In [104…
```
vowels = ['a','e','i','o','u']
vowels_from_sentence =[]
for sentence in paragraph:
    for word in sentence.split():
        if word[0].lower() in vowels:
            vowels_from_sentence.append(word)

print(vowels_from_sentence)
```

```
['a', 'It', 'in', 'It']
```

In [105…
```
vowels_comp = [word for sentence in paragraph for word in sentence.split() if word[0].l
print(vowels_comp)
```

```
['a', 'It', 'in', 'It']
```

Map, reduce, filter

In [106…
```
# defining a function that returns the square of a number
def squared(num):
  return num**2

# original list
num_list = [1,2,3,4,5,6]

# list that will contain the squared numbers
num_list_squared = []

# using a for loop to iterate over our num_list and create a new list with the squared
for num in num_list:
  num_list_squared.append(squared(num))

print(num_list_squared)
```

```
[1, 4, 9, 16, 25, 36]
```

In [108…
```
a = [1,2,3,4,5,6]
```

```
list(map(squared, a))
```

Out[108… [1, 4, 9, 16, 25, 36]

In [109…
```
list(map(lambda pinki:pinki**2, a))
```

Out[109… [1, 4, 9, 16, 25, 36]

In [112…
```
a = [3,5,9,7]
b = [4,5,6,7]
print(list(map(lambda x,y : x+y, a,b)))
```

[7, 10, 15, 14]

In [113…
```
my_list = [3,4,5,6,7,8,9]
```

In [115…
```
list(filter(lambda x:  x % 3 == 0, my_list))
```

Out[115… [3, 6, 9]

In [116…
```
from functools import reduce
a  = reduce(lambda x, y: x+y, range(1,10))
print(a)
```

45

1,2,3,4,5 1+2 =3 3+3 = 6 6+4 = 10

venkat : 123 shivank : 34 hash(venkat) = 234567 hash(shivank) = 34567

```
ordered_menu = ['pizza','pasta','chicken','burger','salad' ,'french fry']
food_prices = [2,3,7,3,1,5]

#Filter out the veg foods
veg_foods =[]
for food in ordered_menu:
  if food == 'salad' or food == 'french fry' or food == 'corn':
    veg_foods.append(food)

#This will give us the veg foods from the menu
veg_foods = ['salad','french fry']

#Add sauce with all the ordered items
foods_with_sauce = []
for food in ordered_menu:
    foods_with_sauce.append(food + ' with sauce')

#This will give us food with sauce like this
foods_with_sauce = ['pizza with sauce','pasta with sauce''chicken with
sauce','burger with sauce','salad with sauce','french fry with sauce']

#Calclution for the total price of ordered foods
total = 0
for price in food_prices:
    total +=price

#This will give us the total price cartman spent on his food
total = 21
```

```
def filter_veg_foods(food):
    veg_items = ['salad', 'french fry','corn']

    if(food in veg_items):
        return True
    else:
        return False
ordered_menu = ['pizza','pasta','chicken','burger','salad' ,'french fry']
veg_foods = filter(filter_veg_foods,ordered_menu)
```

```
def mixing_sauce(food):
    return food + ' with sauce'

ordered_menu = ['pizza','pasta','chicken','burger','salad' ,'french fry']
foods_with_sauce = map(mixing_sauce,ordered_menu)
```

```
from functools import reduce
def sum_of_the_price(first,second):
    return first += second

food_prices = [2,3,7,3,1,5]
total = reduce(sum_of_the_price,food_prices)
```

```
from functools import reduce

ordered_menu = ['pizza','pasta','chicken','burger','salad' ,'french fry']
food_prices = [2,3,7,3,1,5]

veg_foods = filter(lambda food: food == 'salad' or food == 'french fry' or food
== 'corn',ordered_menu)

food_with_sauce = map(lambda food: food + ' with sauce',ordered_menu)
total_price = reduce(lambda first,second: first + second,food_prices)
```

In [ ]: