Module02_Day03_PyRefresher_3

December 16, 2022

1 Python Refresher 2

1.1 Tuples

- Almost Similar to list
- Tuples are immutable
- "Read only" List, Cannot do modification

Creating Tuples

```
[ ]: t = (1,2,3,4,5)
     t
```

```
[ ]: (1, 2, 3, 4, 5)
```

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

```
[ ]: tuple
```

```
[ ]: t[0]
```

```
[ ]: 1
```

```
[ ]: t[-1]
```

```
[ ]: 5
```

```
[ ]: t[1:4]
```

```
[ ]: (2, 3, 4)
```

```
[ ]: t[0] = 100  #tuples are immutable in nature
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\816329950.py in <module>
----> 1 t[0] = 100

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

```
[ ]: t1 = (1,3,True,"String",3.14)
     t1
```

```
[ ]: (1, 3, True, 'String', 3.14)
```

```
[ ]: type(t1)
```

```
[ ]: tuple
```

```
[ ]: emp_id = (101,102,106,110)
```

```
[ ]: em_id[0] = 81 # Beneficial where you don't want to change
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\635971115.py in <module>
----> 1 em_id[0] = 81 # Beneficial where you don't want to change

NameError: name 'em_id' is not defined
```

Because they are "Read Only", they are faster than list

Tuple Properties

```
[ ]: t2 = t + t1 #Concat
     t2
```

```
[ ]: (1, 2, 3, 4, 5, 1, 3, True, 'String', 3.14)
```

```
[ ]: (1,2)*3
```

```
[ ]: (1, 2, 1, 2, 1, 2)
```

```
[ ]: for ele in t2:
         print(ele,end=" ")
```

```
1 2 3 4 5 1 3 True String 3.14
```

```
[ ]: #Chnage of refrence

     t1 = (1,2)
     t1 = (3,4)
     t1
```

```
[ ]: (3, 4)
```

```
[ ]: # Empty Tuples
     t = tuple()
```

```
type(t), t
```

```
[ ]: (tuple, ())
```

```
[ ]: [100]
```

```
[ ]: [100]
```

```
[ ]: # , is important

     t = (100)
     type(t)
```

```
[ ]: int
```

```
[ ]: t = (100,)
     type(t)
```

```
[ ]: tuple
```

```
[ ]: # For tuple important thing is ,

     t = 100,
     type(t)
```

```
[ ]: tuple
```

```
[ ]: (2+4)/6 # parenthesis are used for this, hence for tuple , are really im
```

1.1.1 Tuple packing and Unpacking

```
[ ]: a,b =1,2
     type(a)
```

```
[ ]: int
```

Packing

```
[ ]: a,b,name = 1,2,"Scaler"
     a,b,name # This created tuple, Hence TUPLE PACKING
```

```
[ ]: (1, 2, 'Scaler')
```

```
[ ]: tup = 1,2,"Scaler"
     tup
```

```
[ ]: (1, 2, 'Scaler')
```

Unpacking

```
[ ]: a,b,c = tup
     print(a)
     print(b)
     print(c)
```

```
1
2
Scaler
```

```
[ ]: def foo(a,b):
         return a+b, a-b, a*b
```

```
[ ]: foo(2,3) # This is Tuple packing
```

```
[ ]: (5, -1, 6)
```

```
[ ]: add,sub,mul = foo(2,3)
     print(add)
     print(sub)
     print(mul)
```

```
5
-1
6
```

```
[ ]: a,b = (1,2,3) # Error
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\2070136764.py in <module>
----> 1 a,b = (1,2,3) # Error

ValueError: too many values to unpack (expected 2)
```

```
[ ]: t1.count(4)
```

```
[ ]: 1
```

```
[ ]: t1.index(4)
```

```
[ ]: 1
```

```
[ ]: t1
```

```
[ ]: (3, 4)
```

```
[ ]: # Universal function works
```

```
t = 1,2,3,4,5,6,7

sorted(t,reverse = True)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\135965744.py in <module>
      4
      5 sorted(t,reverse = True)
----> 6 del t[0]

TypeError: 'tuple' object doesn't support item deletion
```

[ ]:
```
t1 = 1,2,3
t2 = 1,2,3
```

[ ]:
```
id(t1) ==  id(t2) # Interning not supported by tuples , SUPPORTED only by␣
 ↪strings & integers
```

[ ]:
```
False
```

1.2 Strings

- Sequence of Characters
- Immutable

[ ]:
```
command = "Alexa, Switch off the lights."
print(command)
```

```
Alexa, Switch off the lights.
```

[ ]:
```
type(command)
```

[ ]:
```
str
```

[ ]:
```
# Indexing

command[0]
```

[ ]:
```
'A'
```

[ ]:
```
command[-1]
```

[ ]:
```
'.'
```

[ ]:
```
command[::-1]
```

[ ]:
```
'.sthgil eht ffo hctiwS ,axelA'
```

```
HW - "Switch off" from command through idexing
```

```
[ ]: command[7:17] # Indexing
```

```
[ ]: 'Switch off'
```

```
[ ]: command[0] = "H" # Immutable
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\939510016.py in <module>
----> 1 command[0] = "H" # Immutable

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

```
[ ]: # Concat is possible

     s = "Hello"
     print(id(s))
     s += " World"
     s, print(id(s)) # String is immutable and assign value to new memory location
```

```
2188619465840
2188619465264
```

```
[ ]: ('Hello World', None)
```

```
[ ]: s1 = "hello world"
     s2 = "hello world"

     s1==s2
```

```
[ ]: True
```

1.2.1 is (Compares Memory location)

```
[ ]: s1 is s2
```

```
[ ]: False
```

```
[ ]: a = 1000
     b = 1000

     a==b, a is b
```

```
[ ]: (True, False)
```

1.2.2 Membership Operators

in

```
[ ]: "Alexa" in command
```

```
[ ]: True
```

```
[ ]: "on" in command
```

```
[ ]: False
```

```
[ ]: "lexa" in command
```

```
[ ]: True
```

```
[ ]: lst = [7,4,3,9,0,5]
```

```
[ ]: 8 in lst, 5 in lst
```

```
[ ]: (False, True)
```

Methods()

```
[ ]: s ="Hello World"
```

```
[ ]: s.index("e")
```

```
[ ]: 1
```

```
[ ]: s.index("o") # Gives the first occurence
```

```
[ ]: 4
```

```
[ ]: s.index("o",5) # Checks from index 5
```

```
[ ]: 7
```

```
[ ]: command
```

```
[ ]: 'Alexa, Switch off the lights.'
```

```
[ ]: command.index("Switch off") # can get they entire substring
```

```
[ ]: 7
```

```
[ ]: command.index("Switch on") # Error because exact string not found
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\4128256911.py in <module>
```

```
----> 1 command.index("Switch on") # Error because exact string not found

ValueError: substring not found
```

```
[ ]: command.count("Switch off")
```

```
[ ]: 1
```

```
[ ]: command.count("Switch on")
```

```
[ ]: 0
```

```
[ ]: command.lower()
```

```
[ ]: 'alexa, switch off the lights.'
```

```
[ ]: command.title()
```

```
[ ]: 'Alexa, Switch Off The Lights.'
```

```
[ ]: command.islower()
```

```
[ ]: False
```

```
[ ]: command.isnumeric()
```

```
[ ]: False
```

```
[ ]: command.startswith("A")
```

```
[ ]: True
```

```
[ ]: command.partition("Switch off")
```

```
[ ]: ('Alexa, ', 'Switch off', ' the lights.')
```

```
[ ]: command.startswith("alexa")
```

```
[ ]: False
```

```
[ ]: web = "https://www.scaler.com"
     web
```

```
[ ]: 'https://www.scaler.com'
```

```
[ ]: web.startswith("http")
```

```
[ ]: True
[ ]: web.lower().startswith("https")
```

```
[ ]: True
```

```
[ ]: _
```

```
[ ]: True
```

```
[ ]: # Multilines String supported

     email ="""Hi,
         How are you?

         Regards
         Bharat"""
```

```
[ ]: email, type(email)
```

```
[ ]: ('Hi,\n    How are you?\n\n    Regards\n    Bharat', str)
```

1.2.3 Formattng

```
[ ]: length = 2
     breadth = 5
     area = length * breadth
```

f stings

```
[ ]: print(f"The area of rectangle length:{length} & breadth:{breadth} is {area}")
```

```
The area of rectangle length:2 & breadth:5 is 10
```

format()

```
[ ]: s = "The area of rectangle length:{} & breadth:{} is {}".
       ↪format(length,breadth,area)
     print(s)
```

```
The area of rectangle length:2 & breadth:5 is 10
```

Split()

```
[ ]: # str -> list : split
     # list -> str : join

     s = "Data science is fun"
     s.split()
```

```
[ ]: ['Data', 'science', 'is', 'fun']
[ ]: lst = s.split("i") # does not count i in it
     lst
```

```
[ ]: ['Data sc', 'ence ', 's fun']
```

```
[ ]: review = "I am very happy with the movie ending"

     # Word tokenization on individual words in NLP
     review.split()
```

```
[ ]: ['I', 'am', 'very', 'happy', 'with', 'the', 'movie', 'ending']
```

```
[ ]: review = "I am happy. movie was good. best ever movie. Loved it. My email:␣
      ↪mohit@sxaler.com"
     sentences = review.split(". ")
```

```
[ ]: "#".join(sentences)
```

```
[ ]: 'I am happy#movie was good#best ever movie#Loved it#My email: mohit@sxaler.com'
```

```
[ ]: inp = input()
```

```
5 2 7 9 1
```

```
[ ]: inp.split()
```

```
[ ]: ['5', '2', '7', '9', '1']
```

```
[ ]: [int(n) for n in inp.split()] # Convert to integer
```

```
[ ]: [5, 2, 7, 9, 1]
```

```
[ ]: review = "I am happy, movie was good, best ever movie. Loved it. My email:␣
      ↪mohit@sxaler.com"
     review
```

```
[ ]: 'I am happy, movie was good, best ever movie. Loved it. My email:
     mohit@sxaler.com'
```

```
[ ]: review = review.replace(",",".")
     print(review)
     review.split(".")
```

```
I am happy. movie was good. best ever movie. Loved it. My email:
mohit@sxaler.com
```

```
[ ]: ['I am happy',
      ' movie was good',
      ' best ever movie',
      ' Loved it',
      ' My email: mohit@sxaler',
      'com']
```

1.2.4 Time

```
[ ]: %timeit (1,2,3)
```

```
7.95 ns ± 0.145 ns per loop (mean ± std. dev. of 7 runs, 100000000 loops each)
```

```
[ ]: %timeit [1,2,3]
```

```
45.2 ns ± 2.1 ns per loop (mean ± std. dev. of 7 runs, 10000000 loops each)
```

1.2.5 Dictionary

- define using {}, or dict()

```
[ ]: d = dict()
     type(dict)
```

```
[ ]: type
```

```
[ ]: p = {
         "name":"John",
         "location":"Delhi",
         "subject": ["Python", "ML"]
     }
     p,type(p)
```

```
[ ]: ({'name': 'John', 'location': 'Delhi', 'subject': ['Python', 'ML']}, dict)
```

```
[ ]: len(p)
```

```
[ ]: 3
```

Acessing the data

```
[ ]: p[0] # Gives error because indexing not supported in dict
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\943468167.py in <module>
----> 1 p[0] # Gives error because indexing not supported in dict

KeyError: 0
```

```
[ ]: # Access the values using keys
```

```
p["location"], p["name"]
```

```
[ ]: ('Delhi', 'John')
```

```
[ ]: p["name"] = "Bill"
```

```
[ ]: id(p)
```

```
[ ]: 2188624707200
```

```
[ ]: p["name"] = "John"
```

```
[ ]: id(p) # hence mutable
```

```
[ ]: 2188624707200
```

```
[ ]: p["subject"]
```

```
[ ]: ['Python', 'ML']
```

```
[ ]: p["subject"].append("Data Science")
     p["subject"]
```

```
[ ]: ['Python', 'ML', 'Data Science']
```

```
[ ]: p["subject"] = p["subject"] + ["DL"]
     p["subject"]
```

```
[ ]: ['Python', 'ML', 'Data Science', 'DL']
```

Methods

```
[ ]: p.keys(), type(p.keys())
```

```
[ ]: (dict_keys(['name', 'location', 'subject']), dict_keys)
```

```
[ ]: list(p.keys())
```

```
[ ]: ['name', 'location', 'subject']
```

```
[ ]: p.values(), type(p.values())
```

```
[ ]: (dict_values(['John', 'Delhi', ['Python', 'ML', 'Data Science', 'DL']]),
      dict_values)
```

```
[ ]: list(p.values())
```

```
[ ]: ['John', 'Delhi', ['Python', 'ML', 'Data Science', 'DL']]
```

```
[ ]: p
[ ]: {'name': 'John',
      'location': 'Delhi',
      'subject': ['Python', 'ML', 'Data Science', 'DL']}
```

```
[ ]: p.update({"age":30})
     p
```

```
[ ]: {'name': 'John',
      'location': 'Delhi',
      'subject': ['Python', 'ML', 'Data Science', 'DL'],
      'age': 30}
```

```
[ ]: p["Gender"] = "M"
     p
```

```
[ ]: {'name': 'John',
      'location': 'Delhi',
      'subject': ['Python', 'ML', 'Data Science', 'DL'],
      'age': 30,
      'Gender': 'M'}
```

```
[ ]: p["Role"] = "SDE2"
     p
```

```
[ ]: {'name': 'John',
      'location': 'Delhi',
      'subject': ['Python', 'ML', 'Data Science', 'DL'],
      'age': 30,
      'Gender': 'M',
      'Role': 'SDE2'}
```

```
[ ]: p1 = {
         "name":"Anne",
         [1,2,3]:1000
     }
     p1 # error because list is mutable and unhashable
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\1643928621.py in <module>
----> 1 p1 = {

      2     "name":"Anne",
      3     [1,2,3]:1000
      4 }
      5 p1 # error because list is mutable and unhashable
```

```
TypeError: unhashable type: 'list'
```

```
[ ]: p1 = {
         "name":"Anne",
         (1,2,3):1000
     }
     p1 # no error because tuple is immutable element, hence hashable
```

```
[ ]: {'name': 'Anne', (1, 2, 3): 1000}
```

```
[ ]: p1.keys()
```

```
[ ]: dict_keys(['name', (1, 2, 3)])
```

```
[ ]: p1["salary"]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\3382559807.py in <module>
----> 1 p1["salary"]

KeyError: 'salary'
```

```
[ ]: p1.get("Salary") # Not giving error and no value also, hence this will not stop␣
       ↪the program execution
```

1.2.6 Nested Dictionary

```
[ ]: EMP_DB = {
         "HR": {
             "967" : 51000,
             "650" : 60000
         },
         "TECH": {
             "516":95000,
             "1001" : 75000,
             "918" : 58000
         },
         "SALES": {
             "887": 45000,
             "490": 63000
         }
     }
```

```
[ ]: EMP_DB["HR"].get("650")
[ ]: 60000
```

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

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

```
[ ]: EMP_DB["TECH"]["918"]
```

```
[ ]: 58000
```

```
[ ]: # Average salary
     sum(EMP_DB["TECH"].values()) / len(EMP_DB["TECH"])
```

```
[ ]: 76000.0
```

1.2.7 Sets

- Stores unique values
- They are mutable
- Non Indexable

```
[ ]: s = {1, 4, 2,  5, 6 }
     type(s)
```

```
[ ]: set
```

```
[ ]: s1 = set()
```

```
[ ]: s.add(4) # No effect if value is present
     s
```

```
[ ]: {1, 2, 3, 4, 5, 6}
```

```
[ ]: s.add(99)
     s
```

```
[ ]: {1, 2, 3, 4, 5, 6, 99}
```

```
[ ]: s.remove(10) # error because not present
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\311776722.py in <module>
----> 1 s.remove(10) # error because not present

KeyError: 10
```

```
[ ]: s.remove(6)
```

```
[ ]: s
```

```
[ ]: {1, 2, 3, 4, 5, 99}
```

```
[ ]: s.pop() # removes first element by default and return it
```

```
[ ]: 1
```

```
[ ]: s
```

```
[ ]: {2, 3, 4, 5, 99}
```

```
[ ]: 5 in s
```

```
[ ]: True
```

```
[ ]: len(s)
```

```
[ ]: 5
```

```
[ ]: for ele in s:
         print(ele) # possible to iterate
```

```
2
3
4
5
99
```

```
[ ]: s[0] # but not index
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_21096\2245127882.py in <module>
----> 1 s[0] # but not index

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

```
[ ]: data = "be the change you wish to see in the world"
     data
```

```
[ ]: 'be the change you wish to see in the world'
```

```
[ ]: # Vocabulary - In NLP means all the unique words
```

```
[ ]: set(data) # create chars wise sepration
[ ]: {' ',
      'a',
      'b',
      'c',
      'd',
      'e',
      'g',
      'h',
      'i',
      'l',
      'n',
      'o',
      'r',
      's',
      't',
      'u',
      'w',
      'y'}
```

```
[ ]: # so better way is
     set(data.split())
```

```
[ ]: {'be', 'change', 'in', 'see', 'the', 'to', 'wish', 'world', 'you'}
```

1.2.8 Doubts

```
[ ]: p ={
         "B":15,
         "D":5,
         "A":78,
         "F":32
     }
     sorted(p)
```

```
[ ]: ['A', 'B', 'D', 'F']
```

```
[ ]: lis = [(0, 2), (1, 3), (2, 4)]
     result = [n for _, n in lis]
     print(result)
```

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

```
# Class is a type and type is a class
```

```
[ ]: type(type)
```

```
[ ]: type
```

[ ]:
```
s="hello"
s[-1:0:-1]
```

[ ]:
```
'olle'
```

[ ]:
```
def reverse(s):
    string = ""
    for i in s:
        string = i + string
    return string
reverse("hello")
```

[ ]:
```
'olleh'
```

[ ]:
```
a = "hello"
a = set(a)
"".join(a)
```

[ ]:
```
'lhoe'
```

[ ]:
```
t = (1,2,2,3,4,5)
s = {1,2,2,3,4,5}
t,s
```

[ ]:
```
((1, 2, 2, 3, 4, 5), {1, 2, 3, 4, 5})
```

[ ]:
```
d = {
    "a":1,
    "b":2
}
d.get
d1 = dict()
for ele in d:
    d1.update({d.get(ele):ele})

d1
```

[ ]:
```
{1: 'a', 2: 'b'}
```

[ ]:
```
s = "abcde"
s2 = "bcdea"

s==s2
```

[ ]:
```
False
```