datastructure

- list
- tuple
- set
- dictionary

list

In [4]:
```
l=[]
l
```

Out[4]:
```
[]
```

In [5]:
```
len(l)
```

Out[5]:
```
0
```

In [6]:
```
l.append(10) #executed two times
l
```

Out[6]:
```
[10]
```

In [7]:
```
print(l.count(10))
```

```
1
```

In [8]:
```
l.clear()
l
```

Out[8]:
```
[]
```

In [9]:
```
l.append(20,'lip',True, 10)
l
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[9], line 1
----> 1 l.append(20,'lip',True, 10)
      2 l

TypeError: list.append() takes exactly one argument (4 given)
```

In [140…
```
l.append('lip') #executed twice
l.append(10)
l
```

Out[140…
```
['lip', 10]
```

In [142…
```
l[:]
```

Out[142…
```
['lip', 10]
```

In [144…
```
l[2]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[144], line 1
----> 1 l[2]

IndexError: list index out of range
```

In [146…
```
id(l[2])
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[146], line 1
----> 1 id(l[2])

IndexError: list index out of range
```

In [148…
```
id(l)
```

Out[148… 3004745557568

In [150…
```
id(l[0])
```

Out[150… 3004784969024

In [152…
```
l2=l.copy()
l2
```

Out[152… ['lip', 10]

In [154…
```
l2.remove(10)
```

In [156…
```
l2
```

Out[156… ['lip']

In [158…
```
del l2
```

In [160…
```
l2
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[160], line 1
----> 1 l2

NameError: name 'l2' is not defined
```

In [162…
```
l
```

Out[162…
```
['lip', 10]
```

In [164…
```
x = 9
```

In [166…
```
y = 9
```

In [168…
```
x+y
```

Out[168…
```
18
```

In [170…
```
_ + 10
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[170], line 1
----> 1 _ + 10

TypeError: can only concatenate str (not "int") to str
```

In [172…
```
print(l)
l2=[1,2,3,4,5]
print(l2)
```

```
['lip', 10]
[1, 2, 3, 4, 5]
```

In [174…
```
for i in l2:
    print(i)
```

```
1
2
3
4
5
```

In [176…
```
l2.append([0,9,8,7])     #nested list
l2
```

Out[176…
```
[1, 2, 3, 4, 5, [0, 9, 8, 7]]
```

In [180…
```
l2.remove(5)                #executed twice
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[180], line 1
----> 1 l2.remove(5)

ValueError: list.remove(x): x not in list
```

In [182…
```
l2
```

Out[182…
```
[1, 2, 3, 4, [0, 9, 8, 7]]
```

In [184…
```
l2.pop()
```

Out[184…
```
[0, 9, 8, 7]
```

In [186…
```
l2
```

Out[186…
```
[1, 2, 3, 4]
```

In [188…
```
l2.pop(-2)
```

Out[188…
```
3
```

In [190…
```
l2
```

Out[190…
```
[1, 2, 4]
```

In [192…
```
l2.insert(2,'bye')
```

In [194…
```
l2
```

Out[194…
```
[1, 2, 'bye', 4]
```

In [196…
```
l2.insert(2,3+4j)
```

In [198…
```
l2
```

Out[198…
```
[1, 2, (3+4j), 'bye', 4]
```

In [200…
```
print(l)
print(l2)
```

```
['lip', 10]
[1, 2, (3+4j), 'bye', 4]
```

In [204…
```
l.extend(l2)     #l extended two times with l2 by two execution
```

In [206…
```
l
```

Out[206…
```
['lip', 10, 1, 2, (3+4j), 'bye', 4, 1, 2, (3+4j), 'bye', 4]
```

In [208…
```
l.index('bye')
```

Out[208…
```
5
```

In [210…
```
l3=[1,4,2,5,8,7]
l3
```

Out[210…
```
[1, 4, 2, 5, 8, 7]
```

In [212…
```
l3.sort()
```

In [214…
```
l3
```

Out[214…
```
[1, 2, 4, 5, 7, 8]
```

In [216…
```
l3.sort(reverse=True)
```

In [218…
```
l3
```

Out[218…
```
[8, 7, 5, 4, 2, 1]
```

In [220…
```
l4=[3.4,2,5.6,4,4+6j,'hi']
l4
```

Out[220…
```
[3.4, 2, 5.6, 4, (4+6j), 'hi']
```

In [222…
```
l4.sort()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[222], line 1
----> 1 l4.sort()

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

In [224…
```
l5=['s','d','f','r']
```

In [226…
```
l5
```

Out[226…
```
['s', 'd', 'f', 'r']
```

In [228…
```
l5.sort()
```

In [230…
```
l5
```

Out[230…
```
['d', 'f', 'r', 's']
```

In [232…
```
l5.reverse()
```

In [234…
```
l5
```

Out[234…
```
['s', 'r', 'f', 'd']
```

In [236…
```
l5[::-1]
```
Out[236…
```
['d', 'f', 'r', 's']
```
In [238…
```
l
```
Out[238…
```
['lip', 10, 1, 2, (3+4j), 'bye', 4, 1, 2, (3+4j), 'bye', 4]
```
In [240…
```
print(l)
```
```
['lip', 10, 1, 2, (3+4j), 'bye', 4, 1, 2, (3+4j), 'bye', 4]
```
In [242…
```
l[:]
```
Out[242…
```
['lip', 10, 1, 2, (3+4j), 'bye', 4, 1, 2, (3+4j), 'bye', 4]
```
In [244…
```
l[:8]
```
Out[244…
```
['lip', 10, 1, 2, (3+4j), 'bye', 4, 1]
```
In [246…
```
l[0:20:5]
```
Out[246…
```
['lip', 'bye', 'bye']
```
In [248…
```
l[::-3]
```
Out[248…
```
[4, 2, 'bye', 1]
```
In [250…
```
l[0]
```
Out[250…
```
'lip'
```
In [252…
```
l[0] = 44
```
In [254…
```
l[0]
```
Out[254…
```
44
```
In [256…
```
l
```

Out[256… [44, 10, 1, 2, (3+4j), 'bye', 4, 1, 2, (3+4j), 'bye', 4]

In [258…
```
l[-1]='jayesh'
```

In [260…
```
l
```

Out[260… [44, 10, 1, 2, (3+4j), 'bye', 4, 1, 2, (3+4j), 'bye', 'jayesh']

In [262…
```
l[-1][0]      #nested list
```

Out[262… 'j'

In [264…
```
print(l[-1][0])
print(l[-1][1])
print(l[-1][2])
print(l[-1][3])
print(l[-1][4])
print(l[-1][5])
```

```
j
a
y
e
s
h
```

In [266…
```
l2
```

Out[266… [1, 2, (3+4j), 'bye', 4]

In [268…
```
l3
```

Out[268… [8, 7, 5, 4, 2, 1]

In [270…
```
l4
```

Out[270… [2, 3.4, 4, 5.6, (4+6j), 'hi']

In [272…
```
l5
```

Out[272… ['s', 'r', 'f', 'd']

In [274…
```
l6 = l4 + l5      #adding two lists and assigning to 3rd
```

In [276…
```
l6
```

Out[276… [2, 3.4, 4, 5.6, (4+6j), 'hi', 's', 'r', 'f', 'd']

In [278…
```
l7 = l2.extend(l3)      #extending first list by second list, do not assign
```

In [280…
```
l2
```

Out[280… [1, 2, (3+4j), 'bye', 4, 8, 7, 5, 4, 2, 1]

In [282…
```
print(l7)
```

None

In [284…
```
3+4j in l2
```

Out[284… True

In [286…
```
'jayesh' in l2
```

Out[286… False

enumerate

In [288…
```
for i in l2:
    print(i)
```

```
1
2
(3+4j)
bye
4
8
7
5
4
2
1
```

In [290…
```
for i in enumerate(l2):
    print(i)
```

```
(0, 1)
(1, 2)
(2, (3+4j))
(3, 'bye')
(4, 4)
(5, 8)
(6, 7)
(7, 5)
(8, 4)
(9, 2)
(10, 1)
```

In [292…
```
l6 = [1,2,3,4]
l7 = [1,2,3,0]
l8 = [0,0,0,0]
l9 = [1,0,0,0]
```

In [294…
```
all(l6)    # return True if all elements are non zero
```

Out[294… True

In [296…
```
any(l6)    # return True if any element is non zero
```

Out[296… True

In [298…
```
all(l7)
```

Out[298… False

In [300…
```
any(l7)
```

Out[300… True

In [302…
```
all(l8)
```

Out[302… False

In [304…
```
any(l8)
```

Out[304… False

In [306…
```
l8
```

Out[306… [0, 0, 0, 0]

In [308…
```
all(l9)
```

Out[308… False

In [310…
```
any(l9)
```

Out[310… True

In [ ]:
```
l10 = [1,2,3,4,5]
for i in l10:
    l10.remove(i)
print(l10)
```

tuple

In [312…
```
t = ()
t
```

Out[312… ()

In [314…
```
print(type(t))
```
```
<class 'tuple'>
```

In [316…
```
t = (1,2,3,4,4)
t
```

Out[316… (1, 2, 3, 4, 4)

In [318…
```
print(t.count(1))
```
```
1
```

In [320…
```
t.count(4)
```

Out[320… 2

In [322…
```
t.index(3)
```

Out[322… 2

In [324…
```
l = [1,2,3,4+3J,'jan']
l
```

Out[324… [1, 2, 3, (4+3j), 'jan']

In [326…
```
l[1] = 'man'     # mutable
l
```

Out[326… [1, 'man', 3, (4+3j), 'jan']

In [328…
```
t1 =(1,2,3,4+3J,'jan')
t1
```

Out[328… (1, 2, 3, (4+3j), 'jan')

In [330…
```
t1[1] = 'man'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[330], line 1
----> 1 t1[1] = 'man'

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

In [332…
```
t1              #immutable
```

Out[332…
```
(1, 2, 3, (4+3j), 'jan')
```

In [334…
```
len(t1)
```

Out[334…
```
5
```

In [336…
```
sbi = (4567,'fff7777',98754321,'jayesh')            # applicarion of tuple in bank a/c details
sbi
```

Out[336…
```
(4567, 'fff7777', 98754321, 'jayesh')
```

In [338…
```
sbi[3] = 'vishal'                  # name unable to change
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[338], line 1
----> 1 sbi[3] = 'vishal'

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

In [340…
```
t2 = t1*3
```

In [342…
```
t2
```

Out[342…
```
(1, 2, 3, (4+3j), 'jan', 1, 2, 3, (4+3j), 'jan', 1, 2, 3, (4+3j), 'jan')
```

In [344…
```
# slicing of tuple can be done as in list and string
```

In [ ]:
```
# only two functions in tuple. index() and count()
```

In [ ]:
```
# in tuple can't add or remove elements
```

In [ ]:
```
for i in enumerate(t2):
    print(i)
```

set

In [346…
```
s = {}
s
```

Out[346…
```
{}
```

In [348…
```
type(s)                    #recognize as dictionary since empty
```

Out[348…
```
dict
```

In [350…
```
s1 = {1,2,3,4}
s1
```

Out[350…
```
{1, 2, 3, 4}
```

In [352…
```
type(s1)
```

Out[352…
```
set
```

In [354…
```
s2 = set()                 #functions to create set,dict,tuple and list from single string except dictionary
d1 = dict()
t1 = tuple()
l1 = list()
```

In [356…
```
print(type(s2))
print(type(t1))
print(type(l1))
print(type(d1))
```

```
<class 'set'>
<class 'tuple'>
<class 'list'>
<class 'dict'>
```

In [358…
```
print(t1)
print(s2)
print(l1)
print(d1)
```

```
()
set()
[]
{}
```

In [360…
```
s2 = set('jayesh')
t1 = tuple('12345')
l1 = list('jayesh')
d1 = dict(name='jayesh', age=2222, location='kerala')
```

In [362…
```
print(t1)
print(s2)
print(l1)
print(d1)
```

```
('1', '2', '3', '4', '5')
{'j', 'a', 'e', 'y', 's', 'h'}
['j', 'a', 'y', 'e', 's', 'h']
{'name': 'jayesh', 'age': 2222, 'location': 'kerala'}
```

In [364…
```
s3 = {'z', 'l', 'c', 'e', 'f'}
s3
```

Out[364… {'c', 'e', 'f', 'l', 'z'}

In [366…
```
s4 = {1, 2.3, 'nit', 1+2j, [1,2,3], (4,5,6), True}                #datastructure inside set is not possible
s4
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[366], line 1
----> 1 s4 = {1, 2.3, 'nit', 1+2j, [1,2,3], (4,5,6), True}               #datastructure inside set is not possible
      2 s4

TypeError: unhashable type: 'list'
```

In [368…
```
s4 = {1, 2.3, 'nit', 1+2j, True}                        #
s4
```

Out[368…
```
{(1+2j), 1, 2.3, 'nit'}
```

In [370…
```
print(s1)                                    #numerical data prints in ascending order
print(s2)                                     #alphebets are not in order
print(s3)
print(s4)
```

```
{1, 2, 3, 4}
{'j', 'a', 'e', 'y', 's', 'h'}
{'l', 'f', 'e', 'c', 'z'}
{1, 2.3, 'nit', (1+2j)}
```

In [372…
```
s1.add(20)
s1
```

Out[372…
```
{1, 2, 3, 4, 20}
```

In [374…
```
s1.add(15)
s1
```

Out[374…
```
{1, 2, 3, 4, 15, 20}
```

In [376…
```
s1.add(25)
s1
```

Out[376…
```
{1, 2, 3, 4, 15, 20, 25}
```

In [378…
```
s1[1:4]                                              #slicing not possible
s1
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[378], line 1
----> 1 s1[1:4]                                                 #slicing not possible
      2 s1

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

In [380…
```
s5=s4.copy()
s5
```

Out[380…
```
{(1+2j), 1, 2.3, 'nit'}
```

In [382…
```
s5.clear()
s5
```

Out[382…
```
set()
```

In [384…
```
del s5
```

In [386…
```
s5
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[386], line 1
----> 1 s5

NameError: name 's5' is not defined
```

In [388…
```
s4.remove(1+2j)
s4
```

Out[388…
```
{1, 2.3, 'nit'}
```

In [390…
```
s4.discard(1+2j)                        #discard() do not produce error
s4
```

Out[390…
```
{1, 2.3, 'nit'}
```

In [392…
```
s4.remove(1+2j)                    #remove() produce error
s4
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[392], line 1
----> 1 s4.remove(1+2j)                    #remove() produce error
      2 s4

KeyError: (1+2j)
```

In [394…
```
s4.pop()                           #pop any element and can't give index too
```

Out[394… 1

In [396…
```
s4.pop(1)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[396], line 1
----> 1 s4.pop(1)

TypeError: set.pop() takes no arguments (1 given)
```

In [398…
```
s2
```

Out[398… {'a', 'e', 'h', 'j', 's', 'y'}

In [400…
```
for i in s2:
    print(i)
```

```
j
a
e
y
s
h
```

In [402…
```
for i in enumerate(s2):
    print(i)
```

```
(0, 'j')
(1, 'a')
(2, 'e')
(3, 'y')
(4, 's')
(5, 'h')
```

In [404…
```
7 in s2
```

Out[404…
```
False
```

In [406…
```
'a' in s2
```

Out[406…
```
True
```

In [408…
```
s1
```

Out[408…
```
{1, 2, 3, 4, 15, 20, 25}
```

In [410…
```
s2.update(s1)
```

In [412…
```
s2
```

Out[412…
```
{1, 15, 2, 20, 25, 3, 4, 'a', 'e', 'h', 'j', 's', 'y'}
```

In [414…
```
s6 = {1,2,3,4,5}
s7 = {4,5,6,7,8}
s8 = {8,9,10}
```

In [416…
```
s6.union(s7)
```

Out[416…
```
{1, 2, 3, 4, 5, 6, 7, 8}
```

In [418…
```
s6.union(s7, s8)
```

Out[418…
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

In [420…
```
s6 | s7
```

Out[420… {1, 2, 3, 4, 5, 6, 7, 8}

In [422…
```
s6 | s7 | s8
```

Out[422… {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [424…
```
print(s6)
print(s7)
print(s8)
```

```
{1, 2, 3, 4, 5}
{4, 5, 6, 7, 8}
{8, 9, 10}
```

In [426…
```
s6.intersection(s7)
```

Out[426… {4, 5}

In [428…
```
s6.intersection(s7,s8)
```

Out[428… set()

In [430…
```
s6 & s7
```

Out[430… {4, 5}

In [432…
```
print(s6)
print(s7)
print(s8)
```

```
{1, 2, 3, 4, 5}
{4, 5, 6, 7, 8}
{8, 9, 10}
```

In [434…
```
s6.difference(s7)
```

Out[434… {1, 2, 3}

In [436…
```
s6 - s7
```

Out[436… {1, 2, 3}

In [438…
```
s6.symmetric_difference(s7)
```

Out[438… {1, 2, 3, 6, 7, 8}

In [440…
```
x = "Hello"
y = 10
print(type(x) == type(y))
```

```
False
```

In [442…
```
a = True
b = False
print(a + b)
```

```
1
```

In [444…
```
x = None
print(type(x))
```

```
<class 'NoneType'>
```

In [446…
```
bool("")
```

Out[446… False

In [448…
```
s1 ={1,2,3,4}
s1[0] = 5
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[448], line 2
      1 s1 ={1,2,3,4}
----> 2 s1[0] = 5

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

In [450…
```
x : int = 10
x = 10
x, y = 5, "hello"
```

In [452…
```
const PI = 3.14
```

```
  Cell In[452], line 1
    const PI = 3.14
          ^
SyntaxError: invalid syntax
```

In [454…
```
define PI 3.14
```

```
  Cell In[454], line 1
    define PI 3.14
           ^
SyntaxError: invalid syntax
```

In [456…
```
PI = 3.14
```

In [458…
```
PI = 3
```

In [460…
```
PI
```

Out[460… 3

In [462…
```
print(range(10))
```

```
range(0, 10)
```

In [464…
```
True+5
```

Out[464… 6

In [466…
```
t = (1,2,3,4)
```

In [468…
```
t[1] = 5
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[468], line 1
----> 1 t[1] = 5

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

In [470…
```
l = [1,2,3,4]
l[1] = 5
```

In [472…
```
l
```

Out[472…
```
[1, 5, 3, 4]
```

In [474…
```
s = {1,2,3,4}
```

In [476…
```
s[1] = 5
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[476], line 1
----> 1 s[1] = 5

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

In [478…
```
type(10/5)
```

Out[478…
```
float
```

In [480…
```
x = {1,2,3,3,4}
x
```

Out[480…
```
{1, 2, 3, 4}
```

In [482…
```
x = [1,2,3]
x * 2
```

Out[482…
```
[1, 2, 3, 1, 2, 3]
```

In [484…
```
_var = 5
```

In [486…
```
_var
```

Out[486…
```
5
```

var_nam = 5

In [512…
```
var_nam
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[512], line 1
----> 1 var_nam

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

var_name = 5

In [514…
```
var_name
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[514], line 1
----> 1 var_name

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

var_nom = 5

In [516…
```
var_nom
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[516], line 1
----> 1 var_nom

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

In [518…
```
var_n = 5
```

In [520…
```
var_n
```

Out[520… 5

var_now = 5

In [522…
```
var_now
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[522], line 1
----> 1 var_now

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

In [508…
```
my_name = 5
```

In [510…
```
my_name
```

Out[510…
```
5
```

In [490…
```
print(True*2)
```

```
2
```

In [492…
```
poll_data = 7
```

In [494…
```
type(poll_data)
```

Out[494…
```
int
```

In [496…
```
print(set(range(9)))
print(list(range(9)))
print(tuple(range(9)))
```

```
{0, 1, 2, 3, 4, 5, 6, 7, 8}
[0, 1, 2, 3, 4, 5, 6, 7, 8]
(0, 1, 2, 3, 4, 5, 6, 7, 8)
```

In [498…
```
a,b,c = 'pqr'
print(a)
print(b)
print(c)
```

```
p
q
r
```

In [500…
```
language = 'malayalam'
last_three = language[-3:]
```

```
print(last_three)
three_to_end = language[3:]
print(three_to_end)
```

```
lam
ayalam
```

In [524…
```
print(language.find('a'))
```

```
1
```

In [526…
```
print(language.find('z'))
```

```
-1
```

In [528…
```
num1=20
num2=30
add=num1+num2
print(f'The addition of {num1} and {num2} is= {add}') # alwase prefer this
```

```
The addition of 20 and 30 is= 50
```

In [536…
```
num1=100
num2=25
num3=333
num1=20
num2=30
avg=(num1+num2+num3)/3
avg1=round((num1+num2+num3)/3,2)
print(f'The avrage of {num1}, {num2}, and {num3} is= {avg} or {avg1}')
```

```
The avrage of 20, 30, and 333 is= 127.66666666666667 or 127.67
```

In [538…
```
# lete combine all
num1=10
num2=20
add = num1+ num2
print('The addition of',num1,'and',num2,'is=',add)
print('The addition of {} and {} is= {}'.format(num1,num2,add))
print(f'The addition of {num1} and {num2} is= {add}')
```

```
The addition of 10 and 20 is= 30
The addition of 10 and 20 is= 30
The addition of 10 and 20 is= 30
```

In [540…
```
print('hello','hai','how are you',sep='--->')
```
```
hello--->hai--->how are you
```

In [545…
```
print(1,2,end=' ')                              # print in same line
print(3)                            # will print 1 2 3.
```
```
1 2 3
```

In [549…
```
c = 3+4j
print(c.real)
print(c.imag)
```
```
3.0
4.0
```

In [563…
```
print(c.conjugate())
print(abs(c))
```
```
(3-4j)
5.0
```

In [569…
```
import cmath
print(cmath.sqrt(c))
print(cmath.phase(c))                               #phase(angle) of a complex number
print(cmath.polar(c))                                #polar form of a complex number
```
```
(2+1j)
0.9272952180016122
(5.0, 0.9272952180016122)
```

In [574…
```
challenge = 'thirty days of python'
print(challenge.swapcase())   # THIRTY DAYS OF PYTHON
challenge = 'Thirty Days Of Python'
print(challenge.swapcase())  # tHIRTY dAYS oF pYTHONchallenge = 'thirty days of python'
challenge = 'Thirty Days Of Python'
print(challenge.swapcase())  # tHIRTY dAYS oF pYTHON
```
```
THIRTY DAYS OF PYTHON
tHIRTY dAYS oF pYTHON
tHIRTY dAYS oF pYTHON
```

In [578…
```
import sys
val3 = 25 + 10j # Complex data type
```

```
print(val3)
print(type(val3)) # type of object
print(sys.getsizeof(val3)) # size of float object in bytes
print(val3, " is complex?", isinstance(val3, complex))
```

```
(25+10j)
<class 'complex'>
32
(25+10j)  is complex? True
```

In [580…

```
0x = None
print(x)        # Output: None
print(type(x))  # Output: <class 'NoneType'>
```

```
None
<class 'NoneType'>
```

In [582…

```
def greet(name=None):
    if name is None:
        print("Hello, Guest!")
    else:
        print(f"Hello, {name}!")

greet()       # Output: Hello, Guest!
greet("John") # Output: Hello, John!
```

```
Hello, Guest!
Hello, John!
```

In [584…

```
txt = " abc def ghi "
txt.lstrip()
```

Out[584… 'abc def ghi '

In [586…

```
txt = " abc def ghi "

txt.strip()
```

Out[586… 'abc def ghi'

In [596…

```
s = {1,2,3}                        # proving set is mutable
print(id(s))
s.add(4)                           # set don't support item assignment since no indexing
```

```
print(s)
print(id(s))
```

```
3004815169056
{1, 2, 3, 4}
3004815169056
```