```
In [1]: import sys
        sys.version

Out[1]: '3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit (AMD64)]'

In [ ]: x = 3
        x

Out[ ]: 3

In [3]: x = 4
        x

Out[3]: 4

In [4]: y = 3
        y

Out[4]: 3

In [5]: x, y

Out[5]: (4, 3)

In [6]: x, y = 3

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[6], line 1
----> 1 x, y = 3

TypeError: cannot unpack non-iterable int object

In [ ]: type(x)

In [7]: y

Out[7]: 3

In [8]: x1 = 4
        x1
        type(x1)

Out[8]: int

In [9]: x,x1

Out[9]: (4, 4)

In [10]: y = 3
         id(y)

Out[10]: 140733999903592

In [11]:
```

```
x1 = 4
id(x1)
```

```
Out[11]:  140733999903624
```

```
In [12]: y = False
         type(y)
```

```
Out[12]:  bool
```

```
In [13]: x, y
```

```
Out[13]:  (4, False)
```

```
In [14]: import sys
         sys.version
```

```
Out[14]:  '3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit (AMD64)]'
```

```
In [15]: import sys
         sys.version
```

```
Out[15]:  '3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit (AMD64)]'
```

```
In [16]: a = 5
         print(a)

         type(a)
```

```
5
```

```
Out[16]:  int
```

```
In [17]: a@ = 6
         a@
```

```
  Cell In[17], line 1
    a@ = 6
       ^
SyntaxError: invalid syntax
```

```
In [18]: 6 = b
```

```
  Cell In[18], line 1
    6 = b
    ^
SyntaxError: cannot assign to literal here. Maybe you meant '==' instead of '='?
```

```
In [ ]: import sys
        sys.version
```

```
Out[ ]:  '3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit (AMD64)]'
```

```
In [20]: a = 5
         a
```

```
Out[20]:  5
```

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

Out[21]: int

In [22]: b = 5.5
         b

Out[22]: 5.5

In [23]: type(b)

Out[23]: float

In [24]: c = 'hi'
         c

Out[24]: 'hi'

In [25]: type(c)

Out[25]: str

In [ ]: x = 2
        x

Out[ ]: 2

In [27]: x@ = 3
         x@

  Cell In[27], line 1
    x@ = 3
       ^
SyntaxError: invalid syntax

In [ ]: a = 2
        type(a)

Out[ ]: int

In [ ]: b = 9223372036
        print(b)

9223372036

In [ ]: pi = 3.17
        print(pi)

3.17

In [31]: type(pi)

Out[31]: float

In [ ]: c = 'A'
        print(c)

A
```

```
In [33]: type(c)

Out[33]: str

In [ ]: name = 'John Doe'
        print(name)
        type(name)

        John Doe

Out[ ]: str

In [ ]: q = True
        print(q)

        True

In [ ]: x = None
        print(x)

        None

In [ ]: 5 = x

          Cell In[37], line 2
            5 = x
            ^
        SyntaxError: cannot assign to literal here. Maybe you meant '==' instead of '='?

In [38]: x = 5
         x

Out[38]: 5

In [40]: ABC = 50
         ABC

Out[40]: 50

In [41]: abc= 60
         abc

Out[41]: 60

In [42]: Abc = 70
         ABCD

        ---------------------------------------------------------------------------
        NameError                                 Traceback (most recent call last)
        Cell In[42], line 2
              1 Abc = 70
        ----> 2 ABCD

        NameError: name 'ABCD' is not defined

In [43]: xyz = 20000
         xyz

Out[43]: 20000
```

```
In [44]: ABCD = 70
         ABCD

Out[44]: 70

In [45]: NIT = 15000
         nit1

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[45], line 2
      1 NIT = 15000
----> 2 nit1

NameError: name 'nit1' is not defined

In [46]: nIT = 20
         nIT

Out[46]: 20

In [47]: montycorps = 78
         montcorP

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[47], line 2
      1 montycorps = 78
----> 2 montcorP

NameError: name 'montcorP' is not defined

In [48]: cash123 = 10
         cash123

Out[48]: 10

In [49]: 123cash = 20
         123cash

  Cell In[49], line 1
    123cash = 20
      ^
SyntaxError: invalid decimal literal

In [50]: 1A = 5

  Cell In[50], line 1
    1A = 5
    ^
SyntaxError: invalid decimal literal

In [51]: A1 = 5
         A1

Out[51]: 5

In [ ]:
```

```
cash = 10
cash
```

```
Out[ ]:  10
```

```
In [53]: ca$h = 20
         ca$h
```

```
  Cell In[53], line 1
    ca$h = 20
      ^
SyntaxError: invalid syntax
```

```
In [54]: ca*h = 20
         ca*h
```

```
  Cell In[54], line 1
    ca*h = 20
    ^
SyntaxError: cannot assign to expression here. Maybe you meant '==' instead of '='?
```

```
In [ ]: CASH = 20
        CASH
```

```
Out[ ]:  20
```

```
In [ ]: CASH1 = 30
        CASH1
```

```
Out[ ]:  30
```

```
In [ ]: 123total = 30
        123total
```

```
  Cell In[57], line 3
    123total = 30
      ^
SyntaxError: invalid decimal literal
```

```
In [ ]: Abcde = 20
        type(Abcde)
```

```
Out[ ]:  int
```

```
In [59]: new = 30
         NEW
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[59], line 2
      1 new = 30
----> 2 NEW

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

```
In [ ]: Total5 = 30
        Total5
```

```
Out[ ]:  30
```

```
In [61]:  def = 4.6
          def
```

```
  Cell In[61], line 1
    def = 4.6
        ^
SyntaxError: invalid syntax
```

```
In [62]:  del = 9
```

```
  Cell In[62], line 1
    del = 9
        ^
SyntaxError: invalid syntax
```

```
In [63]:  import keyword
          keyword.kwlist
```

```
Out[63]:  ['False',
           'None',
           'True',
           'and',
           'as',
           'assert',
           'async',
           'await',
           'break',
           'class',
           'continue',
           'def',
           'del',
           'elif',
           'else',
           'except',
           'finally',
           'for',
           'from',
           'global',
           'if',
           'import',
           'in',
           'is',
           'lambda',
           'nonlocal',
           'not',
           'or',
           'pass',
           'raise',
           'return',
           'try',
           'while',
           'with',
           'yield']
```

```
In [64]:  len(keyword.kwlist)
```

```
Out[64]:  35
```

```
In [65]: DEF = 4
         DEF

Out[65]: 4

In [66]: if = 780
         if

  Cell In[66], line 1
    if = 780
       ^
SyntaxError: invalid syntax

In [67]: IF = 780
         IF

Out[67]: 780

In [ ]: DEF = 5.6
        DEF

Out[ ]: 5.6

In [69]: def = 7
         def

  Cell In[69], line 1
    def = 7
        ^
SyntaxError: invalid syntax

In [70]: import keyword
         keyword.kwlist
```

```
Out[70]: ['False',
          'None',
          'True',
          'and',
          'as',
          'assert',
          'async',
          'await',
          'break',
          'class',
          'continue',
          'def',
          'del',
          'elif',
          'else',
          'except',
          'finally',
          'for',
          'from',
          'global',
          'if',
          'import',
          'in',
          'is',
          'lambda',
          'nonlocal',
          'not',
          'or',
          'pass',
          'raise',
          'return',
          'try',
          'while',
          'with',
          'yield']
```

```
In [71]: len(keyword.kwlist)

Out[71]: 35
```

```
In [ ]: for = 50
        for
```

```
  Cell In[72], line 4
    for = 50
        ^
SyntaxError: invalid syntax
```

```
In [73]: FOR = 58
         FOR

Out[73]: 58
```

```
In [74]: def = 30
         def
```

```
  Cell In[74], line 1
    def = 30
        ^
SyntaxError: invalid syntax
```

In [75]:
```
if = 30
if
```

```
  Cell In[75], line 1
    if = 30
       ^
SyntaxError: invalid syntax
```

In [ ]:
```
IFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFdfgsdfgsfgsdfgsdfgfdgsfghh
IFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFdfgsdfgsfgsdfgsdfgfdgsfghh
```

Out[ ]:
```
56
```

In [ ]:
```
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwxx
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwxx
```

Out[ ]:
```
10
```

In [78]:
```
_abc_def_gef  = 20
_abc_def_gef
```

Out[78]:
```
20
```

In [79]:
```
x_train = 80
x_train
```

Out[79]:
```
80
```

In [80]:
```
print('hello')
```

```
hello
```

In [ ]:
```
'hello'
```

Out[ ]:
```
'hello'
```

In [82]:
```
import keyword
len(keyword.kwlist)
keyword.kwlist
```

```
Out[82]:  ['False',
           'None',
           'True',
           'and',
           'as',
           'assert',
           'async',
           'await',
           'break',
           'class',
           'continue',
           'def',
           'del',
           'elif',
           'else',
           'except',
           'finally',
           'for',
           'from',
           'global',
           'if',
           'import',
           'in',
           'is',
           'lambda',
           'nonlocal',
           'not',
           'or',
           'pass',
           'raise',
           'return',
           'try',
           'while',
           'with',
           'yield']
```

```
In [ ]:  a = True
          a
```

```
Out[ ]:  True
```

```
In [84]:  a1 = true
          a1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[84], line 1
----> 1 a1 = true
      2 a1

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

```
In [85]:  True = a
```

```
  Cell In[85], line 1
    True = a
    ^
SyntaxError: cannot assign to True
```

```
In [ ]:  b = None
         b
```

```
In [87]: b = none
         b
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[87], line 1
----> 1 b = none
      2 b

NameError: name 'none' is not defined
```

```
In [ ]:  c = False
         c

Out[ ]:  False
```

```
In [89]: true + true
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[89], line 1
----> 1 true + true

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

```
In [90]: True + True

Out[90]: 2
```

```
In [91]: True*True

Out[91]: 1
```

```
In [ ]:  True / True

Out[ ]:  1.0
```

```
In [93]: True // True

Out[93]: 1
```

```
In [94]: False/True

Out[94]: 0.0
```

```
In [95]: True/False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[95], line 1
----> 1 True/False

ZeroDivisionError: division by zero
```

```
In [96]: pi = 3.14
         pi

Out[96]: 3.14

In [97]: pi = 3.17
         pi

Out[97]: 3.17

In [ ]: import keyword
        keyword.kwlist

Out[ ]: ['False',
         'None',
         'True',
         'and',
         'as',
         'assert',
         'async',
         'await',
         'break',
         'class',
         'continue',
         'def',
         'del',
         'elif',
         'else',
         'except',
         'finally',
         'for',
         'from',
         'global',
         'if',
         'import',
         'in',
         'is',
         'lambda',
         'nonlocal',
         'not',
         'or',
         'pass',
         'raise',
         'return',
         'try',
         'while',
         'with',
         'yield']

In [99]: kw = keyword.kwlist
         kw
```

```
Out[99]: ['False',
          'None',
          'True',
          'and',
          'as',
          'assert',
          'async',
          'await',
          'break',
          'class',
          'continue',
          'def',
          'del',
          'elif',
          'else',
          'except',
          'finally',
          'for',
          'from',
          'global',
          'if',
          'import',
          'in',
          'is',
          'lambda',
          'nonlocal',
          'not',
          'or',
          'pass',
          'raise',
          'return',
          'try',
          'while',
          'with',
          'yield']
```

```
In [ ]: import pandas as pd
        df = pd.DataFrame(keyword.kwlist)
        df
```

Out[ ]:

	0
0	False
1	None
2	True
3	and
4	as
5	assert
6	async
7	await
8	break
9	class
10	continue
11	def
12	del
13	elif
14	else
15	except
16	finally
17	for
18	from
19	global
20	if
21	import
22	in
23	is
24	lambda
25	nonlocal
26	not
27	or
28	pass
29	raise
30	return
31	try
32	while
33	with
34	yield

In [101…

```
a = 10
id(a)
```

Out[101…

```
140733999903816
```

```
In [102…  b=10
          id(b)

Out[102…  140733999903816

In [ ]:   a = 10
          a

Out[ ]:   10

In [104…  id(a)

Out[104…  140733999903816

In [105…  b = 10
          id(b)

Out[105…  140733999903816

In [106…  c = 20
          id(c)

Out[106…  140733999904136

In [107…  a = 10
          b = 10
          id(a)

Out[107…  140733999903816

In [108…  a = 10
          a
          id(a)

Out[108…  140733999903816

In [109…  a = 1111
          a

Out[109…  1111

In [110…  type(a)

Out[110…  int

In [111…  id(a)

Out[111…  1601666068336

In [ ]:   b = 0b1111
          b

Out[ ]:   15

In [113…  bin(15)
```

```
Out[113…  '0b1111'
```

```
In [114…  b_1 = 0b11
          b_1
```

```
Out[114…  3
```

```
In [115…  bin(b_1)
```

```
Out[115…  '0b11'
```

```
In [116…  b_ = 0b1
          b_
```

```
Out[116…  1
```

```
In [117…  b2 = 0b22
          b2
```

```
  Cell In[117], line 1
    b2 = 0b22
           ^
SyntaxError: invalid digit '2' in binary literal
```

```
In [118…  b1 = 111
          b1
```

```
Out[118…  111
```

```
In [119…  c = 0b111
          c
```

```
Out[119…  7
```

```
In [120…  b3 = 0b2
          b3
```

```
  Cell In[120], line 1
    b3 = 0b2
           ^
SyntaxError: invalid digit '2' in binary literal
```

```
In [121…  True/False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[121], line 1
----> 1 True/False

ZeroDivisionError: division by zero
```

```
In [ ]:  b = 0b10
         b
```

```
Out[ ]:  2
```

```
In [123…  c = 0b100
          c

Out[123…  4

In [ ]:   b1 = 0o11
          b1

Out[ ]:   9

In [125…  i = 0b22

  Cell In[125], line 1
    i = 0b22
         ^
SyntaxError: invalid digit '2' in binary literal

In [126…  i1 = 0o22
          i1

Out[126…  18

In [127…  b2 = 0o27
          b2

Out[127…  23

In [ ]:   a = 10
          b = 0b10
          c = 0o100
          a
          b
          c

Out[ ]:   64

In [129…  c1 = 0O33
          c1

Out[129…  27

In [130…  b

Out[130…  2

In [131…  c

Out[131…  64

In [132…  A = 78
          type(A)

Out[132…  int

In [133…  b = 67.9
          print(b)
```

```
67.9
```

```
In [134…  type(b)
```

```
Out[134…  float
```

```
In [135…  b1 = 0b1.1
          b1
```

```
  Cell In[135], line 1
    b1 = 0b1.1
            ^
SyntaxError: invalid syntax
```

```
In [136…  c = 0o11.6
          c
```

```
  Cell In[136], line 1
    c = 0o11.6
            ^
SyntaxError: invalid syntax
```

```
In [ ]:  d = 0o4567.67
         d
```

```
  Cell In[137], line 7
    d = 0o4567.67 # This is octal
              ^
SyntaxError: invalid syntax
```

```
In [138…  f1 = 1e4
          f1
          type(f1)
```

```
Out[138…  float
```

```
In [139…  f = 1e3
          f
```

```
Out[139…  1000.0
```

```
In [ ]:  g = 2.4E3
         g
```

```
Out[ ]:  2400.0
```

```
In [141…  g1 = 23e3
          g1
```

```
Out[141…  23000.0
```

```
In [ ]:  e = 5.e3
         e
```

```
Out[ ]:  5000.0
```

complex datatypes -

- Complex datatype format are:-(a+bj) (a--Real part/b--Imaginary part/j^2=-1)

- j is the compulsory value & there is no other value accepted in complex type
- j^2 = -1
- Value of j is (j square is equal to -1) (j =(square root of -1) is equal to (j^2 = -1) pure mathe
  if you want to develop mathmetic application or scientific application then python is the bes
- Real type any type base can be accepted but imaginary part allow only integer

```
In [ ]: x = 30+40j
        x
```

```
Out[ ]: (30+40j)
```

```
In [144… type(x)
```

```
Out[144… complex
```

```
In [ ]: y = 1+2j
        z = 3+2j
        y + z
        y-z
        y*z
        y/z
```

```
Out[ ]: (0.5384615384615384+0.30769230769230776j)
```

```
In [146… y-z
```

```
Out[146… (-2+0j)
```

```
In [147… y*z
```

```
Out[147… (-1+8j)
```

```
In [148… y/z
```

```
Out[148… (0.5384615384615384+0.30769230769230776j)
```

```
In [ ]: c = 15+0b111j
        c
```

```
  Cell In[149], line 1
    c = 15+0b111j # Imaginary part cannot be binary,octal
               ^
SyntaxError: invalid binary literal
```

```
In [150… c = 1 + 0b10j
         c
```

```
  Cell In[150], line 1
    c = 1 + 0b10j
               ^
SyntaxError: invalid binary literal
```

```
In [151… d2 = 0b111+15j
         d2
```

```
Out[151… (7+15j)
```

```
In [152…  e1 = 4 + 15j
          e1

Out[152…  (4+15j)

In [153…  a1 = 20+30j
          b1 = 40+50j
          a1+b1
          a1-b1
          a1*b1
          a1/b1

Out[153…  (0.5609756097560976+0.04878048780487805j)

In [154…  a1 * b1

Out[154…  (-700+2200j)

In [155…  20*40

Out[155…  800

In [156…  a = 2+3j
          type(a)

Out[156…  complex

  In [ ]:  a1 = 10+20j
           a1.real
           a1.imag

 Out[ ]:  20.0

In [158…  a1.real

Out[158…  10.0

In [159…  help()
```

```
Welcome to Python 3.11's help utility! If this is your first time using
Python, you should definitely check out the tutorial at
https://docs.python.org/3.11/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To get a list of available
modules, keywords, symbols, or topics, enter "modules", "keywords",
"symbols", or "topics".

Each module also comes with a one-line summary of what it does; to list
the modules whose name or summary contain a given string such as "spam",
enter "modules spam".

To quit this help utility and return to the interpreter,
enter "q" or "quit".


You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)".  Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
```

In [160…
```
com = 10 + 15j
type(com)
```

Out[160…
```
complex
```

In [161…
```
com.real
```

Out[161…
```
10.0
```

In [162…
```
com.imag
```

Out[162…
```
15.0
```

In [163…
```
a = 10
b = 20
c = a>b
c
```

Out[163…
```
False
```

In [164…
```
type(c)
#type(a)
```

Out[164…
```
bool
```

In [165…
```
True+True
True*True
True-True
True/True
True//True
False+False
False+True
True/False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[165], line 8
      6 False+False
      7 False+True
----> 8 True/False

ZeroDivisionError: division by zero
```

```
In [166…  True/True
```

```
Out[166…  1.0
```

```
In [ ]:  True/False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[167], line 1
----> 1 True/False # error

ZeroDivisionError: division by zero
```

```
In [168…  pi = 3.14
          pi
          type(pi)
```

```
Out[168…  float
```

```
In [169…  pi = 3.17
          pi
```

```
Out[169…  3.17
```

```
In [170…  ABC = '''good for datascience'''
          ABC
```

```
Out[170…  'good for datascience'
```

```
In [171…  type(ABC)
```

```
Out[171…  str
```

```
In [ ]:  DEF = '''good for datascience'''
         DEF
         type(DEF)
```

```
Out[ ]:  str
```

```
In [ ]:  w = '''good
                     for datascience'''
         w
```

```
Out[ ]:  'good\n             for datascience'
```

```
In [174…  ts = '''The most common cause of the Python SyntaxError:
              EOL while scanning string literal is due to missing quotes at the end of a s
```

```
    This refers to a string being opened by using either
    ' , " , or """ and not closing the string properly.'''
```

In [175…
```
ts
```

Out[175…
```
'The most common cause of the Python SyntaxError: \n    EOL while scanning string l
is due to missing quotes at the end of a string. \n    This refers to a string bein
by using either \n    \' , " , or """ and not closing the string properly.'
```

In [ ]:
```
y = '''good for datascience'''
y
```

Out[ ]:
```
'good for datascience'
```

In [ ]:
```
y = '''good for
 datascience'''
y
```

Out[ ]:
```
'good for \n datascience'
```

In [ ]:
```
a = '''hallo
how
are
you'''

a
```

Out[ ]:
```
'hallo\nhow \nare \nyou'
```

In [179…
```
b = '''hello
    hi'''
b
```

Out[179…
```
'hello \n    hi'
```

In [ ]:
```
b = '''('hallo'
  'how'
    'are you')'''
b
```

Out[ ]:
```
"('hallo' \n  'how'\n    'are you')"
```

In [182…
```
x, y, z, m, n = 10, True, 10.9, 1 + 10j, 'hi'
```

In [183…
```
type(n)
```

Out[183…
```
str
```

In [184…
```
type(m)
```

Out[184…
```
complex
```

In [185…
```
y
```

Out[185…
```
True
```

In [186…
```
z
```

Out[186… 10.9

Type casting or Type conversion -

int() -- float() -- complex() -- bool() -- str()

In [ ]:
```
int(10.123)
int(True)
int(False)
int('10')
```

Out[ ]:
```
10
```

In [ ]:
```
float(10)
float(False)
float('11')
```

Out[ ]:
```
11.0
```

In [189…
```
float(10,20)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[189], line 1
----> 1 float(10,20)

TypeError: float expected at most 1 argument, got 2
```

In [ ]:
```
complex(10)
complex(10.5)
complex(True)
complex(False)
complex('10')
complex(10,20)
complex(10,20.5)
complex('10')
```

Out[ ]:
```
(10+0j)
```

In [ ]:
```
bool(0)
bool(-10)
bool(0.0)
bool(0.01)
bool(10+20j)
bool(0+1j)
bool(" ")
bool('abc')
bool(' ')
```

Out[ ]:
```
True
```

In [192…
```
bool(-10)
```

Out[192…
```
True
```

In [193…
```
bool(0+1j)
```

```
Out[193…  True

In [ ]:   str(10)
          str(10.50)
          str(True)
          str(10+20j)

Out[ ]:   '(10+20j)'

In [195…  x2 = 10
          y2 = 10
          z2 = 20
          print(id(x2))
          print(id(y2))
          print(id(z2))

        140733999903816
        140733999903816
        140733999904136

In [196…  x = 10 #id - adddres of the memory location
          y = 10
          print(id(x))
          print(id(y))

        140733999903816
        140733999903816

In [197…  id(y)

Out[197…  140733999903816

In [198…  # is operator
          x = 20 # x,y = 20
          y = 20
          x is y
          y is x

Out[198…  True

In [199…  x = True
          y = True
          z = False
          x is y
          y is z
          z is x
          z is y

Out[199…  False

In [200…  l = []
          type(l)
          #print(type(l))

Out[200…  list

In [201…  l
```

```
Out[201…  []

In [202…  #Now i want to add an object
          l.append(10)
          l.append(20)
          l.append(30)
          l.append(10)

In [203…  l

Out[203…  [10, 20, 30, 10]

In [204…  l.add(50)

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[204], line 1
----> 1 l.add(50)

AttributeError: 'list' object has no attribute 'add'

In [205…  l

Out[205…  [10, 20, 30, 10]

In [206…  print(l)

[10, 20, 30, 10]

In [ ]:  l.append('amx')
         l.append(8.0)
         l.append(None)
         l.append(1+2j)
         l.append(True)

In [208…  l

Out[208…  [10, 20, 30, 10, 'amx', 8.0, None, (1+2j), True]

In [ ]:  l.remove('amx')

In [210…  l

Out[210…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [211…  l[-4]

Out[211…  8.0

In [212…  l[-7]

Out[212…  20

In [213…  l

Out[213…  [10, 20, 30, 10, 8.0, None, (1+2j), True]
```

```
In [214…  l[4]

Out[214…  8.0

In [215…  l

Out[215…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [216…  l[6]

Out[216…  (1+2j)

In [217…  l[7]

Out[217…  True

In [218…  l

Out[218…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [219…  l[12]

---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[219], line 1
----> 1 l[12]

IndexError: list index out of range

In [224…  l

Out[224…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [225…  l[5]

In [226…  l[3]

Out[226…  10

In [227…  l

Out[227…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [228…  l[8]

---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[228], line 1
----> 1 l[8]

IndexError: list index out of range

In [229…  l

Out[229…  [10, 20, 30, 10, 8.0, None, (1+2j), True]
```

```
In [230…  l[-1]

Out[230…  True

In [231…  l[-2]

Out[231…  (1+2j)

In [232…  l

Out[232…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

  In [ ]: l[:]

  Out[ ]: [10, 20, 30, 10, 8.0, None, (1+2j), True]

  In [ ]: l[2:4]

  Out[ ]: [30, 10]

In [235…  l

Out[235…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

  In [ ]: l[0:8]

  Out[ ]: [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [237…  l[14]

---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[237], line 1
----> 1 l[14]

IndexError: list index out of range

In [238…  l

Out[238…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [239…  l[3:5]

Out[239…  [10, 8.0]

In [240…  l

Out[240…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [241…  l[2:4]

Out[241…  [30, 10]

In [242…  l
```

```
Out[242…  [10, 20, 30, 10, 8.0, None, (1+2j), True]

In [243…  l[0]

Out[243…  10

In [244…  l[1]

Out[244…  20

In [245…  l[0] = 100

In [246…  l

Out[246…  [100, 20, 30, 10, 8.0, None, (1+2j), True]

In [247…  l[1] = 'NIT'

In [248…  l[:]

Out[248…  [100, 'NIT', 30, 10, 8.0, None, (1+2j), True]

In [249…  l

Out[249…  [100, 'NIT', 30, 10, 8.0, None, (1+2j), True]

In [250…  l[0]

Out[250…  100

In [251…  l[0] = 1000

In [252…  l

Out[252…  [1000, 'NIT', 30, 10, 8.0, None, (1+2j), True]

In [253…  l[1] = 'AMX'

In [254…  l[-2] = 'monty'

In [255…  l

Out[255…  [1000, 'AMX', 30, 10, 8.0, None, 'monty', True]

In [256…  l[:]

Out[256…  [1000, 'AMX', 30, 10, 8.0, None, 'monty', True]

In [257…  l[2]

Out[257…  30

  In [ ]:  l[2:5]
  Out[ ]: [30, 10, 8.0]

In [259…  l

Out[259… [1000, 'AMX', 30, 10, 8.0, None, 'monty', True]

In [260…  l[4]

Out[260… 8.0

In [261…  l

Out[261… [1000, 'AMX', 30, 10, 8.0, None, 'monty', True]

   In [ ]: l[4:]

  Out[ ]: [8.0, None, 'monty', True]

In [263…  l

Out[263… [1000, 'AMX', 30, 10, 8.0, None, 'monty', True]

In [264…  l[:4]

Out[264… [1000, 'AMX', 30, 10]

In [265…  l

Out[265… [1000, 'AMX', 30, 10, 8.0, None, 'monty', True]

In [266…  l[2]

Out[266… 30

In [267…  l[2] = 'hi'
          l

Out[267… [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

   In [ ]: l

  Out[ ]: [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

In [269…  l[2:-2]

Out[269… ['hi', 10, 8.0, None]

In [270…  l

Out[270… [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

In [271…  l[0:-1]

Out[271… [1000, 'AMX', 'hi', 10, 8.0, None, 'monty']
```

```
In [272…  l[:]

Out[272…  [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

In [273…  l

Out[273…  [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

  In [ ]:  l[1]

  Out[ ]:  'AMX'

  In [ ]:  l[:]

  Out[ ]:  [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

  In [ ]:  l[-1]

  Out[ ]:  True

In [277…  int(True)

Out[277…  1

In [278…  l[-2]

Out[278…  'monty'

In [279…  l

Out[279…  [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

  In [ ]:  l[:]

  Out[ ]:  [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

In [281…  l

Out[281…  [1000, 'AMX', 'hi', 10, 8.0, None, 'monty', True]

In [282…  l.pop(6)

Out[282…  'monty'

In [283…  l

Out[283…  [1000, 'AMX', 'hi', 10, 8.0, None, True]

In [284…  del l[-2]

In [285…  l

Out[285…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [286…  l[:-1]
```

```
Out[286…  [1000, 'AMX', 'hi', 10, 8.0]

In [287…  l

Out[287…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [288…  l[:-2]

Out[288…  [1000, 'AMX', 'hi', 10]

In [289…  a, b, c = [1, 'nit', [1,2,3]]

In [290…  a

Out[290…  1

In [291…  b

Out[291…  'nit'

In [292…  c

Out[292…  [1, 2, 3]

In [293…  a1, b1, c1 = [1, 'nit', [1,2,3]]

In [294…  a1

Out[294…  1

In [295…  b1

Out[295…  'nit'

In [296…  l2 = [a1,b1,c1]

In [297…  l2

Out[297…  [1, 'nit', [1, 2, 3]]

In [298…  l

Out[298…  [1000, 'AMX', 'hi', 10, 8.0, True]

  In [ ]: l[::-1]

  Out[ ]: [True, 8.0, 10, 'hi', 'AMX', 1000]

In [300…  l

Out[300…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [301…  l[::-2]
```

```
Out[301…  [True, 10, 'AMX']

In [302…  l

Out[302…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [303…  l[::-3]

Out[303…  [True, 'hi']

In [304…  l

Out[304…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [305…  l[::-1]

Out[305…  [True, 8.0, 10, 'hi', 'AMX', 1000]

In [306…  l

Out[306…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [307…  l[:-2]

Out[307…  [1000, 'AMX', 'hi', 10]

In [308…  l

Out[308…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [309…  l[-4:]

Out[309…  ['hi', 10, 8.0, True]

In [310…  l

Out[310…  [1000, 'AMX', 'hi', 10, 8.0, True]

In [311…  l.remove(8.0)
          l

Out[311…  [1000, 'AMX', 'hi', 10, True]

In [313…  l

Out[313…  [1000, 'AMX', 'hi', 10, True]

In [314…  l.remove('7')
          l
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[314], line 1
----> 1 l.remove(    )
      2 l

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

```
In [315…  l[-1]

Out[315…  True

In [316…  l

Out[316…  [1000, 'AMX', 'hi', 10, True]

In [317…  l[:-3]

Out[317…  [1000, 'AMX']

In [318…  l

Out[318…  [1000, 'AMX', 'hi', 10, True]

In [319…  l[2]

Out[319…  'hi'

In [320…  l

Out[320…  [1000, 'AMX', 'hi', 10, True]

In [321…  l[:-1]

Out[321…  [1000, 'AMX', 'hi', 10]

  In [ ]: l = [10,20,30,40]
          t = (10,20,30,40)

In [323…  type(l)

Out[323…  list

In [324…  type(t)

Out[324…  tuple

In [325…  l

Out[325…  [10, 20, 30, 40]

  In [ ]: t

 Out[ ]:  (10, 20, 30, 40)

In [327…  l[:]
```

```
Out[327…   [10, 20, 30, 40]

In [328…   t[:]

Out[328…   (10, 20, 30, 40)

In [329…   t1 = (10,'amx',True, 5.8, 10)
           t1

Out[329…   (10, 'amx', True, 5.8, 10)

In [330…   t1[0]

Out[330…   10

In [331…   t1[0] = 100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[331], line 1
----> 1 t1[0] = 100

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

```
In [332…   icici = (1234, 'abc34r',37, 200000)
           type(icici)

Out[332…   tuple

In [333…   icici[1234] = 2345
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[333], line 1
----> 1 icici[1234] = 2345

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

```
In [334…   icici.append(45)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[334], line 1
----> 1 icici.append(45)

AttributeError: 'tuple' object has no attribute 'append'
```

```
In [335…   icici

Out[335…   (1234, 'abc34r', 37, 200000)

In [336…   icici.remove(1234)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[336], line 1
----> 1 icici.remove(1234)

AttributeError: 'tuple' object has no attribute 'remove'
```

```
In [337…
len(icici)
```

```
Out[337… 4
```

```
In [338…
len(t1)
```

```
Out[338… 5
```

```
In [339…
t1
```

```
Out[339… (10, 'amx', True, 5.8, 10)
```

```
In [340…
t1[0]
```

```
Out[340… 10
```

```
In [341…
type(t1)
```

```
Out[341… tuple
```

```
In [342…
t1
```

```
Out[342… (10, 'amx', True, 5.8, 10)
```

```
In [ ]:
t1[0] = 20
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[343], line 1
----> 1 t1[0] = 20 # tuple is immutable

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

```
In [344…
t1
```

```
Out[344… (10, 'amx', True, 5.8, 10)
```

```
In [ ]:
t1[1] = 20
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[345], line 1
----> 1 t1[1] = 20 # tuple immutable ( not changable) e.g - kyc / adhar

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

```
In [346…
t1
```

```
Out[346… (10, 'amx', True, 5.8, 10)
```

```
In [ ]: t1[0:3]

Out[ ]: (10, 'amx', True)

In [348… t1

Out[348… (10, 'amx', True, 5.8, 10)

In [349… t1[0:4]

Out[349… (10, 'amx', True, 5.8)

In [350… t1

Out[350… (10, 'amx', True, 5.8, 10)

In [351… t

Out[351… (10, 20, 30, 40)

In [ ]: t[0]

Out[ ]: 10

In [ ]: l

Out[ ]: [10, 20, 30, 40]

In [354… l.append(50)

In [355… l

Out[355… [10, 20, 30, 40, 50]

In [356… l[0]

Out[356… 10

In [ ]: l[0] = 30

In [358… l

Out[358… [30, 20, 30, 40, 50]

In [359… l.append(60)

In [360… l

Out[360… [30, 20, 30, 40, 50, 60]

In [361… l[:10]

Out[361… [30, 20, 30, 40, 50, 60]

         l[10:]
```

```
In [362…
Out[362… []
In [363… l
         type(l)
Out[363… list
In [364… l
Out[364… [30, 20, 30, 40, 50, 60]
In [365… l[:-3]
Out[365… [30, 20, 30]
In [366… l
Out[366… [30, 20, 30, 40, 50, 60]
In [367… l[2:]
Out[367… [30, 40, 50, 60]
In [368… l
Out[368… [30, 20, 30, 40, 50, 60]
In [369… l[:2]
Out[369… [30, 20]
In [370… t
Out[370… (10, 20, 30, 40)
In [371… t[0]
Out[371… 10
  In [ ]: t[0]= 20
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[372], line 1
----> 1 t[0]= 20
      2 # cannot change any value once you decleare cuz tuple is immutable

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

```
In [373… t1
Out[373… (10, 'amx', True, 5.8, 10)
In [374… t
```

```
Out[374…  (10, 20, 30, 40)

In [375…  t.append(50)

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[375], line 1
----> 1 t.append(50)

AttributeError: 'tuple' object has no attribute 'append'

In [376…  t.add(50)

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[376], line 1
----> 1 t.add(50)

AttributeError: 'tuple' object has no attribute 'add'

In [377…  t

Out[377…  (10, 20, 30, 40)

In [378…  t.remove(30)

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[378], line 1
----> 1 t.remove(30)

AttributeError: 'tuple' object has no attribute 'remove'

In [379…  t

Out[379…  (10, 20, 30, 40)

In [380…  t = t*3
          t

Out[380…  (10, 20, 30, 40, 10, 20, 30, 40, 10, 20, 30, 40)

In [381…  t[0] = 20

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[381], line 1
----> 1 t[0] = 20

TypeError: 'tuple' object does not support item assignment

In [382…  t1

Out[382…  (10, 'amx', True, 5.8, 10)

In [ ]:   t2 = t1 * 2

In [384…  t2
```

```
Out[384…  (10, 'amx', True, 5.8, 10, 10, 'amx', True, 5.8, 10)
```

```
In [ ]:  t3 = (10,20,(2,6))
```

```
In [386…  t3
          type(t3)

Out[386…  tuple
```

```
In [387…  colors = "red", "green", "blue"
          colors

Out[387…  ('red', 'green', 'blue')
```

```
In [388…  colors = "red", "green", "blue"
          colors
          rev = colors[::-1]
          rev

Out[388…  ('blue', 'green', 'red')
```

```
In [389…  rev = colors[:-1]
          rev

Out[389…  ('red', 'green')
```

```
In [390…  rev

Out[390…  ('red', 'green')
```

```
In [391…  colors

Out[391…  ('red', 'green', 'blue')
```

```
In [392…  rev1 = colors[::-2]
          rev1

Out[392…  ('blue', 'red')
```

```
In [393…  colors = "red", "green", "blue"
          colors
          rev = colors[:-1]
          rev

Out[393…  ('red', 'green')
```

```
In [394…  colors = "red", "green", "blue"
          colors
          rev = colors[:-2]
          rev

Out[394…  ('red',)
```

```
In [395…  colors

Out[395…  ('red', 'green', 'blue')
```

```
In [396…  colors[-1]
Out[396…  'blue'

In [397…  colors
Out[397…  ('red', 'green', 'blue')

In [ ]:   rev = colors[::-1]
          rev
Out[ ]:   ('blue', 'green', 'red')

In [399…  colors
Out[399…  ('red', 'green', 'blue')

In [400…  rev = colors[::-1]
          rev
Out[400…  ('blue', 'green', 'red')

In [401…  type(colors)
Out[401…  tuple

In [402…  rev1 = colors[:-1]
          rev1
Out[402…  ('red', 'green')

In [ ]:   rev = colors[::-1]
          rev
Out[ ]:   ('blue', 'green', 'red')

In [404…  colors
Out[404…  ('red', 'green', 'blue')

In [ ]:   rev2 = colors[::-2]
          rev2
Out[ ]:   ('blue', 'red')

In [ ]:   r = range(5)
          r
Out[ ]:   range(0, 5)

In [407…  r1 = range(10)
          r1
Out[407…  range(0, 10)
```

```
In [408…  range(10,20)

Out[408…  range(10, 20)

In [ ]:   r2 = list(range(10))
          r2
          r_ = list(range(3))
          r_

Out[ ]:   [0, 1, 2]

In [410…  list(range(5,20))

Out[410…  [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

In [411…  list(range(10,100,10))

Out[411…  [10, 20, 30, 40, 50, 60, 70, 80, 90]

In [412…  list(range(10,100,5))

Out[412…  [10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95]

In [413…  list(range(5,20,5,2))

          ---------------------------------------------------------------------------
          TypeError                                 Traceback (most recent call last)
          Cell In[413], line 1
          ----> 1 list(range(5,20,5,2))

          TypeError: range expected at most 3 arguments, got 4

In [ ]:   r = range(10)
          r

Out[ ]:   range(0, 10)

In [415…  list(r)

Out[415…  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

In [416…  for i in r:
              print('yes')
              print(i)
```

```
yes
0
yes
1
yes
2
yes
3
yes
4
yes
5
yes
6
yes
7
yes
8
yes
9
```

In [417…

```
r
```

Out[417…

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

In [ ]:

```
range(10.0, 11.5)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[418], line 1
----> 1 range(10.0, 11.5) # you cannot declare float argument

TypeError: 'float' object cannot be interpreted as an integer
```

In [419…

```
w1 = range(10,20)
w1
```

Out[419…

```
range(10, 20)
```

In [420…

```
for i in w1:
    print(i)
```

```
10
11
12
13
14
15
16
17
18
19
```

In [421…

```
r
```

Out[421…

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

In [422…

```
r[4]
```

```
Out[422…  4
```

```
In [423…  r[0]
```

```
Out[423…  0
```

```
In [424…  r[5]
```

```
Out[424…  5
```

```
In [425…  r[0:3]
```

```
Out[425…  range(0, 3)
```

```
In [ ]:  range(100)
```

```
Out[ ]:  range(0, 100)
```

```
In [ ]:  range(10,30)
```

```
Out[ ]:  range(10, 30)
```

**Form:3 (if we passed 3 arguments)

```
In [429…  range(50)
```

```
Out[429…  range(0, 50)
```

```
In [ ]:  range(10,50)
```

```
Out[ ]:  range(10, 50)
```

```
In [ ]:  range(10,50,5)
```

```
Out[ ]:  range(10, 50, 5)
```

```
In [432…  range(10,50,5,6)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[432], line 1
----> 1 range(10,50,5,6)

TypeError: range expected at most 3 arguments, got 4
```

```
In [433…  range(10,100,10.56)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[433], line 1
----> 1 range(10,100,10.56)

TypeError: 'float' object cannot be interpreted as an integer
```

```
In [434…  for i in range(10):
              print(i)
```

```
0
1
2
3
4
5
6
7
8
9
```

In [435…

```
for i in range(10,20):
    print(i)
```

```
10
11
12
13
14
15
16
17
18
19
```

In [ ]:

```
for i in range(10,100,10):
    print(i)
```

```
10
20
30
40
50
60
70
80
90
```

In [ ]:

```
range(10,20,5,6)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[437], line 1
----> 1 range(10,20,5,6) #you cannot declare 4 aruguments once becusae max you can as
for 3 arguments or 3 parameter

TypeError: range expected at most 3 arguments, got 4
```

In [ ]:

```
s = {10,20,30,10,20,30}
```

In [ ]:

```
s
```

Out[ ]:

```
{10, 20, 30}
```

In [440…

```
type(s)
```

Out[440…

```
set
```

In [441…

```
s_ = {56,30,75,109 }
s_
```

```
Out[441…  {30, 56, 75, 109}

In [442…  s1 = {30,10,20,10,'abc',5.0,True}
          s1

Out[442…  {10, 20, 30, 5.0, True, 'abc'}

In [443…  s1[0]

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[443], line 1
----> 1 s1[0]

TypeError: 'set' object is not subscriptable

In [444…  s1[1:4]

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[444], line 1
----> 1 s1[1:4]

TypeError: 'set' object is not subscriptable

In [445…  s

Out[445…  {10, 20, 30}

In [ ]:   s[:]

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[446], line 1
----> 1 s[:]    #set object does not support indexing or slicing or subscirptive

TypeError: 'set' object is not subscriptable

In [447…  s

Out[447…  {10, 20, 30}

In [448…  s[1:]

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[448], line 1
----> 1 s[1:]

TypeError: 'set' object is not subscriptable

In [449…  s

Out[449…  {10, 20, 30}

In [ ]:   s.append(True)
          s
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[450], line 1
----> 1 s.append(True)  #mutable
      2 s

AttributeError: 'set' object has no attribute 'append'
```

In [ ]:
```python
s.add(True)
s
```

Out[ ]:
```
{True, 10, 20, 30}
```

In [452…
```python
s.add(300)
s
```

Out[452…
```
{True, 10, 20, 30, 300}
```

In [453…
```python
s.add('b')
s
```

Out[453…
```
{10, 20, 30, 300, True, 'b'}
```

In [454…
```python
s.add('c')
```

In [455…
```python
s.add('c','d','d')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[455], line 1
----> 1 s.add(   ,   ,   )

TypeError: set.add() takes exactly one argument (3 given)
```

In [456…
```python
s
```

Out[456…
```
{10, 20, 30, 300, True, 'b', 'c'}
```

In [457…
```python
s[1]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[457], line 1
----> 1 s[1]

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

In [ ]:
```python
s.remove(300)
```

In [459…
```python
s
```

Out[459…
```
{10, 20, 30, True, 'b', 'c'}
```

In [460…
```python
s.add('d')
s
```

Out[460…
```
{10, 20, 30, True, 'b', 'c', 'd'}
```

```
In [461…  s3 = {[10,20,30], 40, True}
          s3
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[461], line 1
----> 1 s3 = {[10,20,30], 40, True}
      2 s3

TypeError: unhashable type: 'list'
```

```
In [462…  s1
```

```
Out[462…  {10, 20, 30, 5.0, True, 'abc'}
```

```
In [463…  s[0] = 50
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[463], line 1
----> 1 s[0] = 50

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

```
In [464…  myset = {'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight'}

          for i in myset:
              print(i)
```

```
five
six
two
one
eight
three
seven
four
```

```
In [465…  myset.add('nine')
```

```
In [466…  myset
```

```
Out[466…  {'eight', 'five', 'four', 'nine', 'one', 'seven', 'six', 'three', 'two'}
```

```
In [467…  for i in enumerate(myset):
              print(i)
```

```
(0, 'five')
(1, 'six')
(2, 'two')
(3, 'one')
(4, 'eight')
(5, 'three')
(6, 'seven')
(7, 'nine')
(8, 'four')
```

```
In [468…  'nine' in myset
```

```
Out[468…  True
```

```
In [469…  myset.add([10,20])
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[469], line 1
----> 1 myset.add([10,20])

TypeError: unhashable type: 'list'
```

```
In [470…  printmyset.update([10,20])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[470], line 1
----> 1 printmyset.update([10,20])

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

```
In [471…  myset.add((30,40,50,50))
          myset.update(('ab', 56,[1,2,3]))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[471], line 2
      1 myset.add((30,40,50,50))
----> 2 myset.update((    , 56,[1,2,3]))

TypeError: unhashable type: 'list'
```

```
In [472…  myset
```

```
Out[472…  {(30, 40, 50, 50),
           56,
           'ab',
           'eight',
           'five',
           'four',
           'nine',
           'one',
           'seven',
           'six',
           'three',
           'two'}
```

```
In [473…  myset.discard('nine')
```

```
In [474…  A = {1,2,3,4,5}
          B = {4,5,6,7,8}
          C = {8,9,10}
```

```
In [475…  A | B
```

```
Out[475…  {1, 2, 3, 4, 5, 6, 7, 8}
```

```
In [476…  A.union(B)
```

```
Out[476… {1, 2, 3, 4, 5, 6, 7, 8}

In [477… A.union(C)

Out[477… {1, 2, 3, 4, 5, 8, 9, 10}

In [478… A.union(C)

Out[478… {1, 2, 3, 4, 5, 8, 9, 10}

In [479… A | C | B

Out[479… {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In [480… A, B, C

Out[480… ({1, 2, 3, 4, 5}, {4, 5, 6, 7, 8}, {8, 9, 10})

In [481… A & B

Out[481… {4, 5}

In [482… A & C

Out[482… set()

In [483… B & C

Out[483… {8}

In [484… A.intersection(B)

Out[484… {4, 5}

In [485… A.intersection_updaten(B)

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[485], line 1
----> 1 A.intersection_updaten(B)

AttributeError: 'set' object has no attribute 'intersection_updaten'

In [486… A

Out[486… {1, 2, 3, 4, 5}

In [487… B

Out[487… {4, 5, 6, 7, 8}

In [488… D = {10, 11, 12, 13, 14, 15}
         len(D)

Out[488… 6
```

```
In [489…  A - B

Out[489…  {1, 2, 3}

In [490…  len(D)

Out[490…  6

In [491…  A | D

Out[491…  {1, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15}

In [492…  A & D

Out[492…  set()

In [493…  A - D

Out[493…  {1, 2, 3, 4, 5}

In [494…  C | A

Out[494…  {1, 2, 3, 4, 5, 8, 9, 10}

In [ ]:

In [495…  myset

Out[495…  {(30, 40, 50, 50),
           56,
           'ab',
           'eight',
           'five',
           'four',
           'one',
           'seven',
           'six',
           'three',
           'two'}

In [496…  myset.discard('eleven')

In [497…  myset

Out[497…  {(30, 40, 50, 50),
           56,
           'ab',
           'eight',
           'five',
           'four',
           'one',
           'seven',
           'six',
           'three',
           'two'}

In [498…  myset.remove('eleven')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[498], line 1
----> 1 myset.remove(        )

KeyError: 'eleven'
```

In [499…
```
myset1 = myset.copy()
```

In [500…
```
myset1
```

Out[500…
```
{(30, 40, 50, 50),
 56,
 'ab',
 'eight',
 'five',
 'four',
 'one',
 'seven',
 'six',
 'three',
 'two'}
```

In [501…
```
myset
```

Out[501…
```
{(30, 40, 50, 50),
 56,
 'ab',
 'eight',
 'five',
 'four',
 'one',
 'seven',
 'six',
 'three',
 'two'}
```

In [502…
```
myset.clear()
```

In [503…
```
myset
```

Out[503…
```
set()
```

In [504…
```
myset1
```

Out[504…
```
{(30, 40, 50, 50),
 56,
 'ab',
 'eight',
 'five',
 'four',
 'one',
 'seven',
 'six',
 'three',
 'two'}
```

In [505…
```
del myset1
```

```
In [506…  myset1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[506], line 1
----> 1 myset1

NameError: name 'myset1' is not defined
```

```
In [507…  s

Out[507…  {10, 20, 30, True, 'b', 'c', 'd'}

In [508…  s.insert(10,20)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[508], line 1
----> 1 s.insert(10,20)

AttributeError: 'set' object has no attribute 'insert'
```

```
In [509…  a = {1,2,3}
          b = {4,5,6}
          c = { 7,8,9}

  In [ ]: a | b

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

In [511…  b | c

Out[511…  {4, 5, 6, 7, 8, 9}

In [512…  a | b | c

Out[512…  {1, 2, 3, 4, 5, 6, 7, 8, 9}

In [513…  a | c

Out[513…  {1, 2, 3, 7, 8, 9}

In [514…  d = {'nit', 'hi', 2.3}

In [515…  d | a

Out[515…  {1, 2, 2.3, 3, 'hi', 'nit'}

In [516…  a | b

Out[516…  {1, 2, 3, 4, 5, 6}

In [517…  a.union(b)

Out[517…  {1, 2, 3, 4, 5, 6}
```

```
In [518…  b.union(d)

Out[518…  {2.3, 4, 5, 6, 'hi', 'nit'}

In [519…  a , b, c, d

Out[519…  ({1, 2, 3}, {4, 5, 6}, {7, 8, 9}, {2.3, 'hi', 'nit'})

In [520…  a.union(b,c,d)

Out[520…  {1, 2, 2.3, 3, 4, 5, 6, 7, 8, 9, 'hi', 'nit'}

In [521…  b.union(a,d)

Out[521…  {1, 2, 2.3, 3, 4, 5, 6, 'hi', 'nit'}

In [522…  a.update(b,c)
          a

Out[522…  {1, 2, 3, 4, 5, 6, 7, 8, 9}

In [523…  a

Out[523…  {1, 2, 3, 4, 5, 6, 7, 8, 9}

In [524…  b

Out[524…  {4, 5, 6}

In [525…  c

Out[525…  {7, 8, 9}

In [526…  b.update(c)

In [527…  a, b, c, d

Out[527…  ({1, 2, 3, 4, 5, 6, 7, 8, 9},
           {4, 5, 6, 7, 8, 9},
           {7, 8, 9},
           {2.3, 'hi', 'nit'})

In [528…  a & b

Out[528…  {4, 5, 6, 7, 8, 9}

In [529…  a & b & c

Out[529…  {7, 8, 9}

In [530…  b & c & d

Out[530…  set()

In [531…  a - b
```

```
Out[531… {1, 2, 3}

In [532… b

Out[532… {4, 5, 6, 7, 8, 9}

In [533… id(b)

Out[533… 1601671837312

In [534… id(a)

Out[534… 1601671838208

In [535… type(s)

Out[535… set

In [537… s3 = {10,20,30,40}
         s3.add(70)
         s3

Out[537… {10, 20, 30, 40, 70}

In [538… s3.add('hallo')
         s3

Out[538… {10, 20, 30, 40, 70, 'hallo'}

In [539… fs = frozenset(s3)
         fs

Out[539… frozenset({10, 20, 30, 40, 70, 'hallo'})

In [540… type(fs)

Out[540… frozenset

In [ ]: fs.add(50)

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[541], line 1
----> 1 fs.add(50) #add, remove such type of concept are not applicable in frozenset
      2 #fs.remove(10)

AttributeError: 'frozenset' object has no attribute 'add'

In [542… fs.remove(40)

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[542], line 1
----> 1 fs.remove(40)

AttributeError: 'frozenset' object has no attribute 'remove'
```

```
In [543…  fs[1]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[543], line 1
----> 1 fs[1]

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

```
In [544…  s5 = {}
          s5
```

```
Out[544…  {}
```

```
In [545…  type(s5)
```

```
Out[545…  dict
```

```
In [546…  s6 = set()
          type(s6)
```

```
Out[546…  set
```

```
In [547…  myset = {'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight'}
          for a in myset:
              print(a)
```

```
five
six
two
one
eight
three
seven
four
```

```
In [548…  for i in enumerate(myset):
              print(i)
```

```
(0, 'five')
(1, 'six')
(2, 'two')
(3, 'one')
(4, 'eight')
(5, 'three')
(6, 'seven')
(7, 'four')
```

```
In [549…  myset
```

```
Out[549…  {'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

```
In [ ]:  if 'three' in myset:
             print('Three is present in the set')
         else:
             print('Three is not present in the set')
```

```
Three is present in the set
```

```
In [551…  if 'fifty' in myset:
              print('fift is present')
          else:
              print('fifty is absent')

          fifty is absent
```

```
In [552…  myset.add('Nine')
```

```
In [553…  myset

Out[553…   {'Nine', 'eight', 'five', 'four', 'one', 'seven', 'six', 'three', 'two'}
```

```
In [554…  myset.update(['TEN' , 'ELEVEN' , 'TWELVE', 'TWENTY', 'THIRTY'])
          myset

Out[554…   {'ELEVEN',
            'Nine',
            'TEN',
            'THIRTY',
            'TWELVE',
            'TWENTY',
            'eight',
            'five',
            'four',
            'one',
            'seven',
            'six',
            'three',
            'two'}
```

```
In [555…  myset.discard('THIRTY')
```

```
In [556…  l = [10,20,30]
          t = (10,20,30)
```

```
In [557…  #l.discard(10)
          t.discard(10)

        ---------------------------------------------------------------------------
        AttributeError                            Traceback (most recent call last)
        Cell In[557], line 2
              1 #l.discard(10)
        ----> 2 t.discard(10)

        AttributeError: 'tuple' object has no attribute 'discard'
```

```
    In [ ]:  myset.discard('THIRTY','TWENTY','ELEVEN')

        ---------------------------------------------------------------------------
        TypeError                                 Traceback (most recent call last)
        Cell In[558], line 1
        ----> 1 myset.discard(          ,          ,          )
        # YOU CAN NOT DECLEAR MORE THEN 3 ARGUMET IN SET

        TypeError: set.discard() takes exactly one argument (3 given)
```

```
In [559…  myset
```

```
Out[559… {'ELEVEN',
          'Nine',
          'TEN',
          'TWELVE',
          'TWENTY',
          'eight',
          'five',
          'four',
          'one',
          'seven',
          'six',
          'three',
          'two'}
```

```
In [560… A = {1,2,3,4,5}
         B = {4,5,6,7,8}
         C = {8,9,10}
```

```
In [561… A | B
```

```
Out[561… {1, 2, 3, 4, 5, 6, 7, 8}
```

```
In [562… A.union(B)
```

```
Out[562… {1, 2, 3, 4, 5, 6, 7, 8}
```

```
In [563… A.union(C)
```

```
Out[563… {1, 2, 3, 4, 5, 8, 9, 10}
```

```
In [564… A.update(B,C)
```

```
In [565… A
```

```
Out[565… {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

**DICTIONARY DATATYPES (dict)

- oxford dictionary -- in oxford dictionary words along with meaning is there
- i can say that key:values
- in the case of list,tuple,set,range,set,frozenset we represent individually & all objects are indi object right guys.
- i want to represent as group of object as pair example -- (rollno:name, fruits:price, mobileno
- dict is very important, very special category compare to all category
- duplicate keys are not allowed but values can be duplicate
- dict are represent as {}// : you can assined with given operator
- keys & values both can be hetrogeneous
- No such type of rule that all keys are integer types & values are string type
- keys & values can any type of object

```
In [566… d = {100:'amx', 200:'shiv', 300:'nar'}
         d
```

```
Out[566… {100: 'amx', 200: 'shiv', 300: 'nar'}
```

```
In [567…  type(d)

Out[567…  dict

In [ ]:   d1 = {}
          print(d1)
          type(d1)

         {}
Out[ ]:   dict

In [569…  print(type(d1))

         <class 'dict'>

In [ ]:   s = set()
          type(s)

Out[ ]:   set

In [571…  d_ = dict()
          type(d_)

Out[571…  dict

In [572…  print(type(s))

         <class 'set'>

In [ ]:   d2 = {}

In [574…  d2

Out[574…  {}

In [575…  d2.100 = 'hi'

           Cell In[575], line 1
             d2.100 = 'hi'
               ^
         SyntaxError: invalid syntax

In [576…  d2

Out[576…  {}

In [ ]:   d2[100] = 'hi'
          d2

Out[ ]:   {100: 'hi'}

In [ ]:   d2['hi'] = 100
          d2

Out[ ]:   {100: 'hi', 'hi': 100}

In [579…  d2[200 + 10j] = 'amx'
In [580…  d2[300] = 'ABC'
```

```
In [581…  d2
```

```
Out[581…  {100: 'hi', 'hi': 100, (200+10j): 'amx', 300: 'ABC'}
```

```
In [582…  d2[250, 350] = 7, 'hi'
          d2
```

```
Out[582…  {100: 'hi', 'hi': 100, (200+10j): 'amx', 300: 'ABC', (250, 350): (7, 'hi')}
```

```
In [583…  d2[[250, 350]] = 7, 'hi'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[583], line 1
----> 1 d2[[250, 350]] = 7, 'hi'

TypeError: unhashable type: 'list'
```

```
In [584…  d2[True] = 'yyy'
          d2
```

```
Out[584…  {100: 'hi',
           'hi': 100,
           (200+10j): 'amx',
           300: 'ABC',
           (250, 350): (7, 'hi'),
           True: 'yyy'}
```

```
In [585…  d2[8.0] = 56
          d2
```

```
Out[585…  {100: 'hi',
           'hi': 100,
           (200+10j): 'amx',
           300: 'ABC',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56}
```

```
In [ ]:  d2
```

```
Out[ ]:  {100: 'hi',
          'hi': 100,
          (200+10j): 'amx',
          300: 'ABC',
          (250, 350): (7, 'hi'),
          True: 'yyy',
          8.0: 56}
```

```
In [587…  d2['l'] = [10,20], 34
```

```
In [588…  d2
```

```
Out[588…  {100: 'hi',
           'hi': 100,
           (200+10j): 'amx',
           300: 'ABC',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34)}
```

```
In [589…  d2[100]
```

```
Out[589…  'hi'
```

```
In [590…  del d2[100] # how to remove key & values in dict
```

```
In [591…  del d2[300]
```

```
In [592…  d2
```

```
Out[592…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34)}
```

```
In [593…  a1 = {}
          type(a1)
```

```
Out[593…  dict
```

```
In [594…  math.sqrt(25)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[594], line 1
----> 1 math.sqrt(25)

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

```
In [595…  a2 = dict()
          type(a2)
```

```
Out[595…  dict
```

```
In [596…  d2
```

```
Out[596…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34)}
```

```
In [597…  d2.keys()
```

```
Out[597…  dict_keys(['hi', (200+10j), (250, 350), True, 8.0, 'l'])
```

In [598…
```
d2.values()
```
Out[598…
```
dict_values([100, 'amx', (7, 'hi'), 'yyy', 56, ([10, 20], 34)])
```
In [599…
```
d2
```
Out[599…
```
{'hi': 100,
 (200+10j): 'amx',
 (250, 350): (7, 'hi'),
 True: 'yyy',
 8.0: 56,
 'l': ([10, 20], 34)}
```
In [600…
```
d2.items()
```
Out[600…
```
dict_items([('hi', 100), ((200+10j), 'amx'), ((250, 350), (7, 'hi')), (True, 'yyy')
56), ('l', ([10, 20], 34))])
```
In [601…
```
len(d2.items())
```
Out[601…
```
6
```
In [602…
```
d2['criket'] = (23, 45, 56, 67, 78, 89)
```
In [603…
```
d2
```
Out[603…
```
{'hi': 100,
 (200+10j): 'amx',
 (250, 350): (7, 'hi'),
 True: 'yyy',
 8.0: 56,
 'l': ([10, 20], 34),
 'criket': (23, 45, 56, 67, 78, 89)}
```
In [604…
```
type(d2['criket'])
```
Out[604…
```
tuple
```
In [605…
```
d2.get(True)
```
Out[605…
```
'yyy'
```
In [606…
```
d2.keys(True)
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[606], line 1
----> 1 d2.keys(True)

TypeError: dict.keys() takes no arguments (1 given)
```
In [607…
```
d2
```

```
Out[607…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34),
           'criket': (23, 45, 56, 67, 78, 89)}
```

```
In [608…  d2['criket'] = 'ipl'
```

```
In [609…  d2
```

```
Out[609…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34),
           'criket': 'ipl'}
```

```
In [610…  d2.pop('criket')
```

```
Out[610…  'ipl'
```

```
In [611…  d2
```

```
Out[611…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34)}
```

```
In [612…  d4 = d2.copy()
```

```
In [613…  d4
```

```
Out[613…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34)}
```

```
In [614…  d2
```

```
Out[614…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34)}
```

```
In [615…  d4
```

```
Out[615…  {'hi': 100,
           (200+10j): 'amx',
           (250, 350): (7, 'hi'),
           True: 'yyy',
           8.0: 56,
           'l': ([10, 20], 34)}
```

```
In [616…  d2 == d4
```

```
Out[616…  True
```

```
In [617…  print(id(d2))
          print(id(d4))
```

```
1601667730816
1601674006272
```

```
In [618…  id(d2), id(d4)
```

```
Out[618…  (1601667730816, 1601674006272)
```

```
In [619…  a = b = c = 30
```

```
In [620…  a is b
```

```
Out[620…  True
```

```
In [621…  id(a) is id(b)
```

```
Out[621…  False
```

```
In [622…  a = b = c = d = 10
          print(id(a))
          print(id(b))
          print(id(c))
          print(id(d))
```

```
140733999903816
140733999903816
140733999903816
140733999903816
```

```
In [623…  id(c) is id(d)
```

```
Out[623…  False
```

```
In [624…  a is b
```

```
Out[624…  True
```

```
In [625…  c is d
```

```
Out[625…  True
```