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

Out[1]:  '3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]'

In [6]:  x = 3
         x

Out[6]:  3

In [4]:  x = 4
         x

Out[4]:  4

In [7]:  y = 3
         y

Out[7]:  3

In [9]:   x , y

Out[9]:  (3, 3)

In [10]: type(x)

Out[10]: int

In [11]: x1 = 4
         x1
         type(x1)

Out[11]: int

In [12]: y = 3
         id(y)

Out[12]: 140729291450872

In [13]: x1 = 4
         id(x1)

Out[13]: 140729291450904

In [14]: y = False
         type(y)

Out[14]: bool

In [15]: x, y

Out[15]: (3, False)

In [16]: import sys
         sys.version

Out[16]: '3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]'

In [18]: a = 5
         print(a)

         type(a)

       5
Out[18]: int

In [19]: a@ = 5
         a@

          Cell In[19], line 1
            a@ = 5
              ^
        SyntaxError: invalid syntax

In [20]: 6=b
```

```
  Cell In[20], line 1
    6=b
    ^
SyntaxError: cannot assign to literal here. Maybe you meant '==' instead of '='?
```

In [21]:
```
## verify if python is installed
import sys
sys.version
```

Out[21]:
```
'3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]'
```

In [22]:
```
a = 5
a
```

Out[22]:
```
5
```

In [23]:
```
type(a)
```

Out[23]:
```
int
```

In [26]:
```
b = 5.6
b
```

Out[26]:
```
5.6
```

In [27]:
```
type(b)
```

Out[27]:
```
float
```

In [28]:
```
c = 'Hello'
c
```

Out[28]:
```
'Hello'
```

In [29]:
```
type(c)
```

Out[29]:
```
str
```

In [30]:
```
x = 2
x
```

Out[30]:
```
2
```

In [32]:
```
type(x)
```

Out[32]:
```
int
```

In [33]:
```
x@ = 5
x@
```

```
  Cell In[33], line 1
    x@ = 5
       ^
SyntaxError: invalid syntax
```

In [34]:
```
## Integer
a = 2
type(a)
```

Out[34]:
```
int
```

In [35]:
```
b = 597936735065
print(b)
```

```
597936735065
```

In [36]:
```
#floating point
pi = 3.15
print(pi)
```

```
3.15
```

In [37]:
```
type(pi)
```

Out[37]:
```
float
```

In [38]:
```
## String
c = 'S'
print(c)
```

```
S
```

```
In [39]: type(c)

Out[39]: str
```

```
In [40]: # string

         name = 'sow'
         print(name)
         type(name)

         sow
Out[40]: str
```

```
In [41]: ##Boolean

         q = True
         print(q)

         True
```

```
In [42]: #Empty value or null data type
         x = None
         print(x)

         None
```

```
In [43]: ## Variable assignment works from left to right. so the following will give you an Syntax error

         5= x

           Cell In[43], line 3
             5= x
             ^
         SyntaxError: cannot assign to literal here. Maybe you meant '==' instead of '='?
```

```
In [45]: x = 5
         x

Out[45]: 5
```

```
In [46]: ABC =100
         ABC

Out[46]: 100
```

```
In [47]: xyz = 80
         xyz

Out[47]: 80
```

```
In [48]: AbC = 50
         AbcD

         ---------------------------------------------------------------------------
         NameError                                 Traceback (most recent call last)
         Cell In[48], line 2
               1 AbC = 50
         ----> 2 AbcD

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

```
In [49]: NIT = 15000
         nit1

         ---------------------------------------------------------------------------
         NameError                                 Traceback (most recent call last)
         Cell In[49], line 2
               1 NIT = 15000
         ----> 2 nit1

         NameError: name 'nit1' is not defined
```

```
In [50]: Nit = 30
         Nit

Out[50]: 30
```

```
In [52]: montycorps = 76
         montycorp
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[52], line 2
      1 montycorps = 76
----> 2 montycorp

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

In [53]:
```
123cash = 45
123cash
```

```
  Cell In[53], line 1
    123cash = 45
      ^
SyntaxError: invalid decimal literal
```

In [54]:
```
A1 = 5
A1
```

Out[54]:
```
5
```

In [55]:
```
# x = 10 ## x is the variabel & 10 is the value
cash = 10 # Identefeier rules alphabet
#ca$h = 10 ## $-symbol is not allowed in python identifier but in java is allowed
#ca$h
cash
```

Out[55]:
```
10
```

In [56]:
```
ca$h = 20
ca$h
```

```
  Cell In[56], line 1
    ca$h = 20
      ^
SyntaxError: invalid syntax
```

In [57]:
```
ca*h = 20
ca*h
```

```
  Cell In[57], line 1
    ca*h = 20
    ^
SyntaxError: cannot assign to expression here. Maybe you meant '==' instead of '='?
```

In [58]:
```
CASH = 15
# cahs1
CASH1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[58], line 3
      1 CASH = 15
      2 # cahs1
----> 3 CASH1

NameError: name 'CASH1' is not defined
```

In [60]:
```
# 2. Identifiers should not starts with digit ===
#sum123 = 20  ##zdigit rules identifier
123total = 30
123total
#sum123
```

```
  Cell In[60], line 3
    123total = 30
      ^
SyntaxError: invalid decimal literal
```

In [61]:
```
# 3. Identifiers are case sensitive
#total =10  #python are case sensitive

Abcde = 20
#total
type(Abcde)
```

Out[61]:
```
int
```

In [62]:
```
new = 30
NEW
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[62], line 2
      1 new = 30
----> 2 NEW

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

```
In [64]: Total4 = 30
         #TOTAL
         Total4
```

```
Out[64]: 30
```

```
In [65]: def = 4.5
         def
```

```
  Cell In[65], line 1
    def = 4.5
        ^
SyntaxError: invalid syntax
```

```
In [66]: del = 9
```

```
  Cell In[66], line 1
    del = 9
        ^
SyntaxError: invalid syntax
```

```
In [67]: import keyword
         keyword.kwlist

         len(keyword.kwlist)
```

```
Out[67]: 35
```

```
In [68]: DEF = 5
         DEF
```

```
Out[68]: 5
```

```
In [69]: if =650
         if
```

```
  Cell In[69], line 1
    if =650
       ^
SyntaxError: invalid syntax
```

```
In [70]: IF = 650
         IF
```

```
Out[70]: 650
```

```
In [71]: DEF = 4.5
         DEF
```

```
Out[71]: 4.5
```

```
In [72]: def = 8
         def
```

```
  Cell In[72], line 1
    def = 8
        ^
SyntaxError: invalid syntax
```

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

        len(keyword.kwlist)
```

```
In [74]: # 4. Keywords cannot be assigned as Identifier
         #if = 10   #if is keyword
         #DEF = 20  # def is keyword

         for = 50
         #DEF
         #if
         for
```

```
  Cell In[74], line 5
    for = 50
        ^
SyntaxError: invalid syntax
```

In [75]:
```
FOR = 50
FOR
```

Out[75]:
```
50
```

In [76]:
```
def = 20
def
```

```
  Cell In[76], line 1
    def = 20
        ^
SyntaxError: invalid syntax
```

In [77]:
```
if = 30
if
```

```
  Cell In[77], line 1
    if = 30
       ^
SyntaxError: invalid syntax
```

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

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

In [79]:
```
x_train = 35
x_train
```

Out[79]:
```
35
```

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

```
Hello
```

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

In [82]:
```
# a =True
a = True
a
```

Out[82]:
```
True
```

In [83]:
```
a1 = true
a1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[83], line 1
----> 1 a1 = true
      2 a1

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

In [84]:
```
True = a
```

```
  Cell In[84], line 1
    True = a
    ^
SyntaxError: cannot assign to True
```

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

In [88]:
```
b = none
b
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[88], line 1
----> 1 b = none
      2 b

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

```
In [89]: c = False
         c
```

```
Out[89]: False
```

```
In [90]: true + true
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[90], line 1
----> 1 true + true

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

```
In [91]: True + True
```

```
Out[91]: 2
```

```
In [92]: True + true
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[92], line 1
----> 1 True + true

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

```
In [93]: True * True
```

```
Out[93]: 1
```

```
In [94]: True / True
```

```
Out[94]: 1.0
```

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

```
Out[95]: 1
```

```
In [96]: True / False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[96], line 1
----> 1 True / False

ZeroDivisionError: division by zero
```

```
In [97]: False / True
```

```
Out[97]: 0.0
```

```
In [98]: pi = 3.12
         pi
```

```
Out[98]: 3.12
```

```
In [99]: pi = 3.18
         pi
```

```
Out[99]: 3.18
```

```
In [100…: import pandas as pd
          df = pd.DataFrame(keyword.kwlist)
          df
```

Out[100…

	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…

```
140729291451096
```

In [102…

```
b = 10
id(b)
```

Out[102…

```
140729291451096
```

In [106…

```
a = 10
a
print(a)
type(a)
id(a)
```

```
10
```

Out[106…

```
140729291451096
```

In [107…

```
a = 10
b = 10
```

```
id(a)
```

Out[107… 140729291451096

int datatypes

In [108…

```
a = 1111
a
```

Out[108… 1111

In [109…

```
type(a)
```

Out[109… int

In [110…

```
id(a)
```

Out[110… 1624490876592

In [113…

```
# 2. Binary form(Base 2)
#a = 1111 (Value is declared)
# Now above value should be convert as binary value

b = 0b1111
b
```

Out[113… 15

In [112…

```
bin(15)
```

Out[112… '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 [120…

```
b2 = 0b222
b2
```

```
  Cell In[120], line 1
    b2 = 0b222
           ^
SyntaxError: invalid digit '2' in binary literal
```

In [118…

```
b1 = 111
b1
```

Out[118… 111

In [119…

```
c = 0b111
c
```

Out[119… 7

In [121…

```
b3 = 0b2
b3
```

```
  Cell In[121], line 1
    b3 = 0b2
           ^
SyntaxError: invalid digit '2' in binary literal
```

In [122…

```
b = 0b10
b
```

Out[122… 2

In [123…

```
c = 0b100
```

```
c
```

Out[123… 4

In [130…
```
# 3. Octal form(Base 8)
#a = 111 (value is declared)
b1 = 0o11 #(Now python virtual mechaine (pvm) converts value to octal value "o is using as octal")
b1
```

Out[130… 9

In [131…
```
i = 0b22
```

```
  Cell In[131], line 1
    i = 0b22
          ^
SyntaxError: invalid digit '2' in binary literal
```

In [132…
```
i1 = 0o22
i1
```

Out[132… 18

In [133…
```
b2 = 0o27
b2
```

Out[133… 23

In [134…
```
# final summary of INTEGRAL DATATYPES

a = 10
b = 0b10
c = 0o100
a
b
c
```

Out[134… 64

In [135…
```
c1 = 0o33
c1
```

Out[135… 27

In [136…
```
b
```

Out[136… 2

In [137…
```
c
```

Out[137… 64

In [138…
```
A = 80
type(A)
```

Out[138… int

Float datatypes::

In [139…
```
b = 70.1
print(b)
```

```
70.1
```

In [140…
```
type(b)
```

Out[140… float

In [141…
```
b1 = 0b1.1
b1
```

```
  Cell In[141], line 1
    b1 = 0b1.1
            ^
SyntaxError: invalid syntax
```

In [142…
```
c = 0o11.8
c
```

```
  Cell In[142], line 1
    c = 0o11.8
        ^
SyntaxError: invalid syntax
```

In [146…

```
b2 = 0b1111.22
b2
print(type(b))
type(b2)
```

```
  Cell In[146], line 1
    b2 = 0b1111.22
               ^
SyntaxError: invalid syntax
```

In [147…

```
d = 0o4567.67
d
```

```
  Cell In[147], line 1
    d = 0o4567.67
              ^
SyntaxError: invalid syntax
```

In [149…

```
f1 = 1e4
f1
type(f1)
```

Out[149…

```
float
```

In [150…

```
f = 1e3 (# Here, only 'e' letter is allowed )
f
```

Out[150…

```
1000.0
```

In [154…

```
g = 2.4E4 # except 'E' you can't execute any programe
g
```

Out[154…

```
24000.0
```

g1 = 23e3 g1

In [155…

```
e = 5.e3
e
```

Out[155…

```
5000.0
```

In [156…

```
type(e)
```

Out[156…

```
float
```

Complex Datatypes:

In [157…

```
x = 30+40j
x
```

Out[157…

```
(30+40j)
```

In [158…

```
type(x)
```

Out[158…

```
complex
```

In [163…

```
y = 1+2j
z = 3+2j
y+z
```

Out[163…

```
(4+4j)
```

In [160…

```
y-z
```

Out[160…

```
(-2+0j)
```

In [161…

```
y*z
```

Out[161…

```
(-1+8j)
```

In [162…

```
y/z
```

Out[162…

```
(0.5384615384615384+0.30769230769230776j)
```

```
In [164…  c = 15 + 0b111j
          c
```

```
  Cell In[164], line 1
    c = 15 + 0b111j
                 ^
SyntaxError: invalid binary literal
```

```
In [165…  c = 1+0b10j
          c
```

```
  Cell In[165], line 1
    c = 1+0b10j
             ^
SyntaxError: invalid binary literal
```

```
In [166…  d2 = 0b111+15j
          d2

Out[166…  (7+15j)
```

```
In [167…  e1 = 4+15j
          e1

Out[167…  (4+15j)
```

```
In [168…  a1 = 20+30j
          b1 = 40+50j
          a1+b1

Out[168…  (60+80j)
```

```
In [169…  a1-b1

Out[169…  (-20-20j)
```

```
In [170…  a1*b1

Out[170…  (-700+2200j)
```

```
In [171…  a1/b1

Out[171…  (0.5609756097560976+0.04878048780487805j)
```

```
In [172…  a1*b1
          20*30

Out[172…  600
```

```
In [173…  a = 2+3j
          type(a)

Out[173…  complex
```

```
In [180…  a1 = 10 + 20j
          a1.real
          a1.imag

Out[180…  20.0
```

```
In [181…  com=10 + 16j
          type(com)

Out[181…  complex
```

```
In [184…  com.real

Out[184…  10.0
```

```
In [183…  com.imag

Out[183…  16.0
```

Boolean Datatypes::

```
In [185…  a = 10
          b = 20
```

```
c = a>b
c
```

Out[185… False

In [186…
```
type(c)
```

Out[186… bool

In [187…
```
true +true
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[187], line 1
----> 1 true +true

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

In [188…
```
True + True
```

Out[188… 2

In [189…
```
True - True
```

Out[189… 0

In [190…
```
True * True
```

Out[190… 1

In [191…
```
True / True
```

Out[191… 1.0

In [192…
```
False + False
```

Out[192… 0

In [193…
```
False * True
```

Out[193… 0

In [194…
```
True/False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[194], line 1
----> 1 True/False

ZeroDivisionError: division by zero
```

In [195…
```
pi = 3.15
pi
```

Out[195… 3.15

String Datatypes:

In [199…
```
ABC = '''Datascience is good to study'''
ABC
```

Out[199… 'Datascience is good to study'

In [197…
```
type(ABC)
```

Out[197… str

In [200…
```
w = '''Datascience
                  is good to study'''
w
```

Out[200…
```
'Datascience\n                  is good to study'
```

In [201…
```
ts = '''The most common cause of the Python SyntaxError:
     EOL while scanning string literal is due to missing quotes at the end of a string.
     This refers to a string being opened by using either
     ',", or """ and not closing the string properly.'''
ts
```

```
Out[201… 'The most common cause of the Python SyntaxError:\n     EOL while scanning string literal is due to missing quo
         tes at the end of a string.\n     This refers to a string being opened by using either\n     \',", or """ and n
         ot closing the string properly.'
```

```
In [206… a = '''hello
         how
         are
         you'''
         a
```

```
Out[206… 'hello\nhow\nare \nyou'
```

```
In [207… b = '''('hello
           'how'
              'are you')'''
         b
```

```
Out[207… "('hello\n  'how'\n     'are you')"
```

```
In [210… x,y,z,m,n = 10 True, 10.9, 1+10j,'hello'

         y
```

```
  Cell In[210], line 1
    x,y,z,m,n = 10 True, 10.9, 1+10j,'hello'
                   ^
SyntaxError: invalid syntax
```

Type Casting or Type Conversion::

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

```
In [215… # int(): we can convert from other type to int type except complex
         int(10.25) ##using float to int
```

```
Out[215… 10
```

```
In [216… int(10+20j) ## we cannot convert from complex to int
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[216], line 1
----> 1 int(10+20j)

TypeError: int() argument must be a string, a bytes-like object or a real number, not 'complex'
```

```
In [218… int(True) ##Using bool to int
         int(False)
```

```
Out[218… 0
```

```
In [219… int('10') ##Using string to int
```

```
Out[219… 10
```

```
In [220… ## FLOAT():: Convert from any type to float except complex
         float(10)  ##Using int to float
```

```
Out[220… 10.0
```

```
In [221… float(10+20j)  ## we cannot convert complex to float
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[221], line 1
----> 1 float(10+20j)

TypeError: float() argument must be a string or a real number, not 'complex'
```

```
In [222… float(Flase)  ##Boolean to float
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[222], line 1
----> 1 float(Flase)

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

```
In [223… float('15') ## using string to float
```

```
Out[223… 15.0
```

```
In [224… float(10,20)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[224], line 1
----> 1 float(10,20)

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

```
In [228… ## Complex():  convert any other type to complex type
         ## --this is only for 1 argument
         complex(10)  ##using int to complex
```

```
Out[228… (10+0j)
```

```
In [229… complex(10.5) ##using float to complex
```

```
Out[229… (10.5+0j)
```

```
In [230… complex(True)  ##using bool to complex
```

```
Out[230… (1+0j)
```

```
In [231… complex('10')  ##using string to complex
```

```
Out[231… (10+0j)
```

```
In [232… #----Now checking with 2 arguments----

         complex(10,20)  ##using int to complex
```

```
Out[232… (10+20j)
```

```
In [233… complex(10,20.5)  ##using float to complex
```

```
Out[233… (10+20.5j)
```

```
In [234… complex('10','2')  ##using string to complex, we cannot assign 2 arguments
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[234], line 1
----> 1 complex('10','2')

TypeError: complex() can't take second arg if first is a string
```

```
In [235… ## Bool() --(0 means false // 1 means non zero)

         bool(0)  ##Using int to bool
```

```
Out[235… False
```

```
In [236… bool(-10)  ##Using int to bool
```

```
Out[236… True
```

```
In [237… bool(0.0)  ##Using float to  bool
```

```
Out[237… False
```

```
In [238… bool(0.05)  ##Using float to  bool
```

```
Out[238… True
```

```
In [239… bool(10+20j)  ##Using complex to  bool
```

```
Out[239… True
```

```
In [240… bool(0+20j)  ##Using complex to  bool
```

```
Out[240… True
```

```
In [243… bool("")  ##Using string to  bool (if argument is empty then it could be false)
```

```
Out[243… False
```

```
In [242… bool('abc')  ##Using string to  bool (if argument is empty then it could be true)
```

```
Out[242… True
```

```
In [244… bool(' ')  ##Using string to bool (space is also treated as character so non empty string)
```

```
Out[244… True
```

```
In [245… bool(-10)
```

```
Out[245… True
```

```
In [246… bool(0+1j)
```

```
Out[246… True
```

```
In [247… ## str(): --any type is possible in string

         str(10)  ##using int to string
```

```
Out[247… '10'
```

```
In [248… str(10.6)  ##using float to string
```

```
Out[248… '10.6'
```

```
In [ ]: str(True)  ##using bool to string
```

```
In [249… str(10+15j)  ##using complex to string
```

```
Out[249… '(10+15j)'
```

Fundamental datatypes vs Immutability::

```
In [250… x2 = 10
         y2 = 15
         z2 = 20
         print(id(x2))
         print(id(y2))
         print(id(z2))
```

```
140729291451096
140729291451256
140729291451416
```

```
In [251… x = 10
         y = 10
         print(id(x))
         print(id(y))
```

```
140729291451096
140729291451096
```

```
In [253… # is operator
         x = 20
         y = 10
         x,y
```

```
Out[253… (20, 10)
```

```
In [254… x is y
         y is x
```

```
Out[254… False
```

```
In [255… x = True
         y = True
         z = False

         x is y
```

```
Out[255… True
```

```
In [256… y is z
```

```
Out[256… False
```

```
In [257… z is x
```

```
Out[257… False
```

In [258…

```
z is y
```

Out[258…

```
False
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js