(3.10) Type Casting

Part - 1

We can convert one type value to another type. This conversion is called Typecasting or
Type coersion.
The following are various inbuilt functions for type casting.
1. int()
2. float()
3. complex()
4. bool()
5. str()

1.int():

We can use this function to convert values from other types to int
Eg:

In [1]:
```
print(int(123.987))

print(int(True))

print(int(False))

print(int("10"))
```

```
123
1
0
10
```

In [2]:
```
int(10+5j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/2189727595.py in <module>
----> 1 int(10+5j)

TypeError: can't convert complex to int
```

In [3]:
```
int("10.5")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/864341924.py in <module>
----> 1 int("10.5")

ValueError: invalid literal for int() with base 10: '10.5'
```

In [4]:
```
int("ten")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
```

```
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/1877291328.py in <module>
----> 1 int("ten")

ValueError: invalid literal for int() with base 10: 'ten'
```

In [5]:
```
int("0B1111")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/516961551.py in <module>
----> 1 int("0B1111")

ValueError: invalid literal for int() with base 10: '0B1111'
```

Note:

1. We can convert from any type to int except complex type.
2. If we want to convert str type to int type, compulsary str should contain only integral value and should be specified in base-10

2. float():

We can use float() function to convert other type values to float type.

In [6]:
```
print(float(10))

print(float(True))

print(float(False))

print(float("10"))

print(float("10.5"))
```

```
10.0
1.0
0.0
10.0
10.5
```

In [7]:
```
float(10+5j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/1145906427.py in <module>
----> 1 float(10+5j)

TypeError: can't convert complex to float
```

In [8]:
```
float("ten")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/2531904882.py in <module>
----> 1 float("ten")

ValueError: could not convert string to float: 'ten'
```

In [9]:

```
float("0B1111")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/1191746669.py in <module>
----> 1 float("0B1111")

ValueError: could not convert string to float: '0B1111'
```

Note:

1. We can convert any type value to float type except complex type.
2. Whenever we are trying to convert str type to float type compulsary str should be either integral or floating point literal and should be specified only in base-10.

3.complex():

We can use complex() function to convert other types to complex type.

Form-1: complex(x)

We can use this function to convert x into complex number with real part x and imaginary part 0.

In [10]:
```
print(complex(10)) #10+0j

print(complex(10.5)) #10.5+0j

print(complex(True))# 1+0j

print(complex(False)) #0j

print(complex("10")) #10+0j

print(complex("10.5")) #10.5+0j
```

```
(10+0j)
(10.5+0j)
(1+0j)
0j
(10+0j)
(10.5+0j)
```

In [11]:
```
complex("ten")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_8024/1280863479.py in <module>
----> 1 complex("ten")

ValueError: complex() arg is a malformed string
```

Form-2: complex(x,y)

We can use this method to convert x and y into complex number such that x will be real part and y will be imaginary part.

In [12]:
```
print(complex(10,-2)) #10-2j

print(complex(True,False)) #1+0j
```

```
(10-2j)
(1+0j)
```

4. bool():

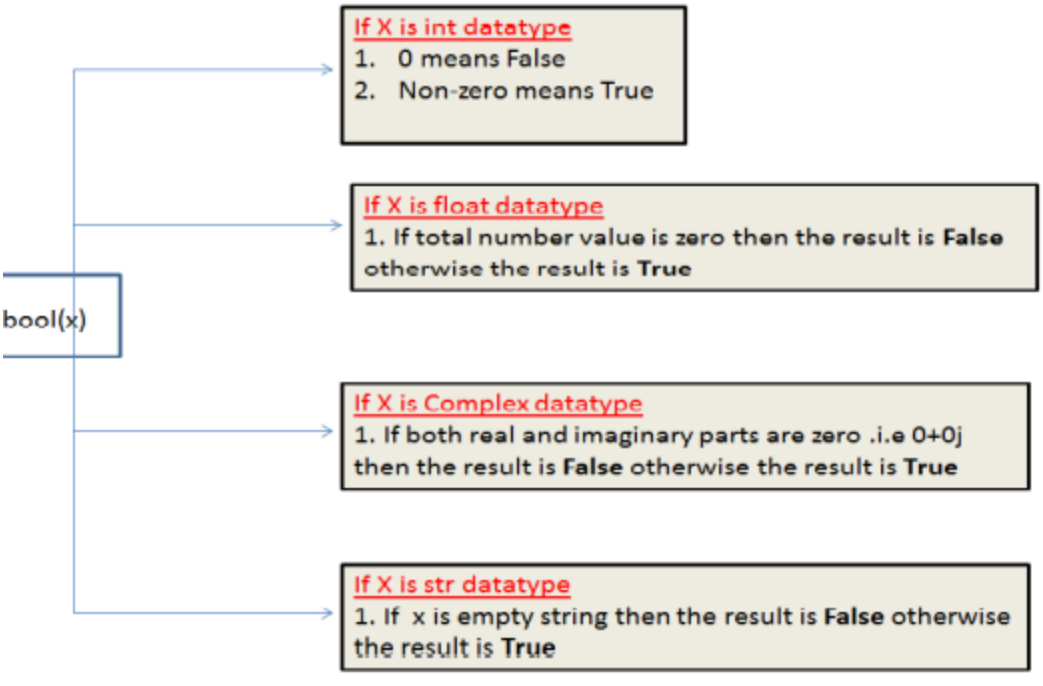
We can use this function to convert other type values to bool type.

In [14]:
```
print(bool(0)) #False

print(bool(1)) #rue

print(bool(10)) #True

print(bool(10.5)) #True

print(bool(0.178)) #True

print(bool(0.0)) #False

print(bool(10-2j)) #True

print(bool(0+1.5j)) #True

print(bool(0+0j)) #False

print(bool("True")) #True

print(bool("False")) #True

print(bool("")) #False
```

```
False
True
True
True
True
False
True
True
False
True
True
False
```

In [1]:
```
from IPython.display import Image
Image(filename='bool_type_cast.png')
```

Out[1]:



5. str():

We can use this method to convert other type values to str type
Eg:

In [15]:
```
print(str(10))

print(str(10.5))

print(str(10.5j))

print(str(True))
```

```
10
10.5
10.5j
True
```

string to decimal with base

In [4]:
```
a = int('abcd', base=16)
print(a)
```

```
43981
```

In [5]:
```
b = '1001'
a = int(b, 2)
print(a)
```

```
9
```

In [6]:
```
b = "0b1001"
a = int(b, 2)
print(a)
```

```
9
```

In [8]:
```
if 0xff == ord('q'):
    print('done')
```

Part - 2

list to set

In [2]:
```
a = set([1,2,3,4])
print(a)
```

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

unique element

In [3]:
```
a = set([1, 1, 2, 3, 2, 4])
print(a)
```

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

set to list

In [13]:
```
t = {1,2,3}
l = list(t)
print(l,type(l))
```

```
[1, 2, 3] <class 'list'>
```

In [ ]:
```

```

list to string

In [9]:
```
a = [1,2,3,4,5]

s = str(a)

print(s,type(s))

repr(s)
```

```
[1, 2, 3, 4, 5] <class 'str'>
```

Out[9]:
```
"'[1, 2, 3, 4, 5]'"
```

string to list

In [12]:
```
a = list("1234")
print(a,type(a))
```

```
['1', '2', '3', '4'] <class 'list'>
```

In [ ]:
```

```

set to string

In [10]:
```
str({1,2,3})
```

Out[10]:
```
'{1, 2, 3}'
```

string to set

In [16]:
```
s = set("1234")
print(s,type(s))
```

```
{'3', '1', '2', '4'} <class 'set'>
```

In [ ]:

dic to string

In [11]:
```
str({1:2, 2:3, 3:4})
```

Out[11]:
```
'{1: 2, 2: 3, 3: 4}'
```

In [ ]:

list to tuple

In [17]:
```
t = tuple([1,2,3,4])
print(t,type(t))
```

```
(1, 2, 3, 4) <class 'tuple'>
```

tuple to list

In [18]:
```
t = (1,2,3,4)
l = list(t)
print(l,type(l))
```

```
[1, 2, 3, 4] <class 'list'>
```

In [ ]:

dic to list

In [20]:
```
d  = {"name " : "Ayush", "roll_no " : "1212xfs"}
```

```
lst = list(d)
print(lst,type(lst))
```

```
['name ', 'roll_no '] <class 'list'>
```

d.items() give tuple of key and value

In [21]:
```
lst = list(d.items())
print(lst)
```

```
[('name ', 'Ayush'), ('roll_no ', '1212xfs')]
```

list to dic

In [22]:
```
lst = [ (1, 2), (3, [4, 5, 6]),('versino', (1, 2, 3))]
d = dict(lst)
print(d)
```

```
{1: 2, 3: [4, 5, 6], 'versino': (1, 2, 3)}
```

In [23]:
```
lst = [ 'hi', ('hello', 'bye'),
      {'bolo': 'jai', 'har har': 'mahadev'}]
d = dict(lst)
print(d)
```

```
{'h': 'i', 'hello': 'bye', 'bolo': 'har har'}
```

In [24]:
```
d = dict('hi')
print(d)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
C:\Users\PANKAJ~1\AppData\Local\Temp/ipykernel_7140/1548643142.py in <module>
----> 1 d = dict('hi')
      2 print(d)

ValueError: dictionary update sequence element #0 has length 1; 2 is required
```

Part - 3

keymap / encoding

utf-8

a -> 97 -> utf-8 -> 'a'

In [25]:
```
ord('a')
```

Out[25]:
```
97
```

In [26]:
```
ord('क')
```

Out[26]:
```
2325
```

In [29]:

```
chr(97)
```

Out[29]:
```
'a'
```

In [30]:
```
chr(2325)
```

Out[30]:
```
'क'
```

In [27]:
```
print([ chr(val) for val in range(2325, 2425)])
```

```
['क', 'ख', 'ग', 'घ', 'ङ', 'च', 'छ', 'ज', 'झ', 'ञ', 'ट', 'ठ', 'ड', 'ढ', 'ण', 'त', 'थ',
'द', 'ध', 'न', 'ऩ', 'प', 'फ', 'ब', 'भ', 'म', 'य', 'र', 'ऱ', 'ल', 'ळ', 'ऴ', 'व', 'श',
'ष', 'स', 'ह', 'ऺ', 'ऻ', '़', 'ऽ', 'ा', 'ि', 'ी', 'ु', 'ू', 'ृ', 'ॄ', 'ॅ', 'ॆ', 'े', 'ै', 'ॉ',
'ॊ', 'ो', 'ौ', '्', 'ॎ', 'ॏ', 'ॐ', '॒', '॓', '॔', 'ॕ', 'ॖ', 'ॗ', 'क़', 'ख़', 'ग़', 'ज़',
'ड़', 'ढ़', 'फ़', 'य़', 'ॠ', 'ऌ', 'ॡ', 'ॢ', '।', '॥', '०', '१', '२', '३', '४', '५', '६', '७',
'८', '९', '॰', 'ॱ', 'ॲ', 'ॳ', 'ॴ', 'ॵ', 'ॶ', 'ॷ', 'ॸ', 'ॹ', 'ॺ', 'ॻ']
```

In [ ]:
```

```