Python TypeCasting

In [2]:
```
int(10.3)
```

Out[2]: 10

In [3]:
```
int(True)
```

Out[3]: 1

In [4]:
```
int(False)
```

Out[4]: 0

In [5]:
```
int(2+3j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[5], line 1
----> 1 int(2+3j)

TypeError: int() argument must be a string, a bytes-like object or a real number, no
t 'complex'
```

In [6]:
```
int('10')
```

Out[6]: 10

In [7]:
```
int('ten')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[7], line 1
----> 1 int('ten')

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

In [8]:
```
float(25)
```

Out[8]: 25.0

In [9]:
```
float(25,30)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[9], line 1
----> 1 float(25,30)

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

In [10]:
```
float(True)
```

Out[10]: 1.0

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

Out[11]: 0.0

In [13]:
```
float(1+2j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[13], line 1
----> 1 float(1+2j)

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

In [14]:
```
float('9')
```

Out[14]: 9.0

In [15]:
```
float('nine')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[15], line 1
----> 1 float('nine')

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

In [16]:
```
complex(8)
```

Out[16]: (8+0j)

In [17]:
```
complex(8+10j)
```

Out[17]: (8+10j)

In [18]:
```
complex(8,10)
```

Out[18]: (8+10j)

In [19]:
```
complex(2,4,6)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[19], line 1
----> 1 complex(2,4,6)

TypeError: complex() takes at most 2 arguments (3 given)
```

In [20]:
```
complex(2.3,3.3)
```

Out[20]: (2.3+3.3j)

In [21]:
```
complex(4.5)
```

Out[21]: (4.5+0j)

In [22]:
```
complex(True,False)
```

Out[22]: (1+0j)

In [23]:
```
complex(False,True)
```

Out[23]: 1j

In [24]:
```
complex('10')
```

Out[24]: (10+0j)

In [25]:
```
complex('ten')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[25], line 1
----> 1 complex('ten')

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

In [26]:
```
complex('10','20')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[26], line 1
----> 1 complex('10','20')

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

In [27]:
```
complex('10',20)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[27], line 1
----> 1 complex('10',20)

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

In [28]:
```
complex(10,'20')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[28], line 1
----> 1 complex(10,'20')

TypeError: complex() second arg can't be a string
```

In [29]:
```
print(str(2))
print(str(2.3))
print(str(True))
print(str(1+2j))
```

```
2
2.3
True
(1+2j)
```

In [30]:
```
str(2)
```

Out[30]: '2'

In [31]:
```
str(2.3)
```

Out[31]: '2.3'

In [32]:
```
str(True)
```

Out[32]: 'True'

In [33]:
```
str(1+2j)
```

Out[33]: '(1+2j)'

In [34]:
```
bool(2)
```

Out[34]: True

In [35]:
```
bool(2.3)
```

Out[35]: True

In [36]:
```
bool(0)
```

Out[36]: False

In [37]:
```
bool()
```

Out[37]: False

In [38]:
```
bool(1+2j)
```

Out[38]: True

In [39]:
```
bool('ten')
```

Out[39]: True

In [40]:
```
print(10)
print(10,20)
print('Hi')
print(1,2,'hi')
```

```
10
10 20
Hi
1 2 hi
```

In [41]:
```
n1, n2 = 1,2
add = n1+n2
print(add)
```

```
3
```

In [42]:
```
print('i casually added', n1 , 'and' , n2, ', the result is', add)
```

```
i casually added 1 and 2 , the result is 3
```

In [43]:
```
num1=100
num2=25
num3=20
avg=(num1+num2+num3)/3
avg1=round((num1+num2+num3)/3,2)
print('the average of {},{}, and {} is = {} or {}' .format(num1,num2,num3,avg,avg1)
```

```
the average of 100,25, and 20 is = 48.333333333333336 or 48.33
```

In [45]:
```
n1,n2 = 100,200
add = n1+n2
print(f'the addition of {n1} and {n2} is {add}')
```

```
the addition of 100 and 200 is 300
```

In [46]:
```
print('Good', end = ' ')
print('Morning')
```

```
Good Morning
```

In [47]:
```
print('hi','hello','how','are','you', sep='--->')
```

```
hi--->hello--->how--->are--->you
```

In [ ]: