Type Casting:

In [ ]:
```
- Change one data type to another data type

- Int ----> float
  int ----> string
  int ---- > boolean
  int ---- > complex

 -float ----> int
  float ----> string
  float ---- > boolean
  float ---- > complex
```

integer to other data types

In [1]:
```
number1=100
number1
```

Out[1]: 100

In [2]:
```
type(number1)
```

Out[2]: int

In [4]:
```
# case-1:  int ======= > float
float_number=float(number1)  # number1 to float type  100 ----> 100.0
```

In [5]:
```
float_number
```

Out[5]: 100.0

In [6]:
```
type(float_number)
```

Out[6]: float

In [7]:
```
# 100 =====string===== '100'
string_number=str(number1)
string_number
```

Out[7]: '100'

In [8]:
```
type(string_number)
```

Out[8]: str

In [9]:
```
# Case-3 : integer to boolean
boolean_number=bool(number1)  # bool(100)
boolean_number
```

Out[9]: True

In [10]:
```
type(boolean_number)
```

Out[10]:
```
bool
```

In [13]:
```
bool(0)   # When it will provide False
```

Out[13]:
```
False
```

Note

Other than zero , everything is True

bool(0) only False

In [15]:
```
# 100 ======= ?  complex(100)
complex_number= complex(number1)
complex_number
```

Out[15]:
```
(100+0j)
```

In [ ]:
```
number1=100
float_number=float(number1) # float(100)= 100.0
string_number=str(number1)  # str(100)='100'
boolean_number=bool(number1) # bool(100)=True
complex_number=complex(number1) # complex(100)=100+0j
```

In [ ]:
```
200 ==== > int ======== float ======= 200.0
200 =====> int ======== str =========='200'
200 =====> int ======== bool ======== True
200 ===== > int ====== complex ===== 200+0j
```

Convert float to all other types

In [ ]:
```
number2=100.5
int_number=int(number2) #     100
string_number=str(number2)  # '100.5'
boolean_number=bool(number2) # True
complex_number=complex(number2) # 100.5+0j
```

In [16]:
```
number2=100.5
int_number=int(number2)
int_number
```

Out[16]:
```
100
```

In [17]:
```
number2=100.5
string_number=str(number2)
string_number
```

Out[17]:
```
'100.5'
```

In [18]:
```
number2=100.5
bool_number=bool(number2)
bool_number
```

Out[18]:
```
True
```

In [19]:
```
number2=100.5
complex_number=complex(number2)
complex_number
```

Out[19]:
```
(100.5+0j)
```

In [ ]:
```
I have a doubt , why 100.5 is not 101

# we are not rounding off
# why not 100?
```

In [21]:
```
round(100.6)
```

Out[21]:
```
101
```

In [ ]:
```
100
100.5
'100'
True
100+5j
```

string to other data types

In [ ]:
```
# case-1
string_value1='apple'
int(string_value1)    #
float(string_value1)
bool(string_value1)
complex(string_value1)
```

In [22]:
```
string_value1='apple'  # English
int(string_value1)     # int maths

# you cant convert english to maths
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[22], line 2
      1 string_value1='apple'
----> 2 int(string_value1)

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

In [23]:
```
string_value1='apple'
float(string_value1)
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[23], line 2
      1 string_value1='apple'
----> 2 float(string_value1)

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

In [24]:
```
string_value1='apple'
bool(string_value1)
```

Out[24]:
```
True
```

In [25]:
```
string_value1=''  # empty string
bool(string_value1)
```

Out[25]:
```
False
```

Note:

    Empty string boolean conversion given as False

In [26]:
```
string_value1='apple'
complex(string_value1)
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[26], line 2
      1 string_value1='apple'
----> 2 complex(string_value1)

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

In [ ]:
```
string_value1='apple'
int(string_value1)    #  error
float(string_value1)  # error
bool(string_value1)   # True
complex(string_value1) # Error
```

In [ ]:
```
string_value2='10.5' # base type float === represents in quotes === finally
int(string_value2)   # error
float(string_value2) #  10.5
bool(string_value2)  # True
complex(string_value2)  #
```

In [27]:
```
string_value2='10.5' # base type float === represents in quotes
int(string_value2)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[27], line 2
      1 string_value2='10.5' # base type float === represents in quotes
----> 2 int(string_value2)

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

In [28]:
```
string_value2='10.5' # base type float === represents in quotes
float(string_value2)
```

Out[28]: 10.5

In [30]:
```
int('10.5')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[30], line 1
----> 1 int('10.5')

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

In [31]:
```
string_value2='10.5' # base type float === represents in quotes === finally
complex(string_value2)
```

Out[31]: (10.5+0j)

In [33]:
```
complex(10.5)
complex('10.5')
```

Out[33]: (10.5+0j)

In [34]:
```
float(10.5)
```

Out[34]: 10.5

In [35]:
```
float('10.5')
```

Out[35]: 10.5

In [36]:
```
int(10.5)
```

Out[36]: 10

In [37]:
```
int('10.5')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[37], line 1
----> 1 int('10.5')

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

In [ ]:
```
string_value2='10' #
int(string_value2)   # 10
float(string_value2) #  10.0
bool(string_value2)  #   True
complex(string_value2) # 10+0j
```

In [38]:
```
float('10')# works
int('10.5')  # fail
```

Out[38]:
```
10.0
```

In [ ]:
```
sir how to convert string to integer without error
```

In [39]:
```
string1='10.5'   # 10
n1=float(string1) # '10.5' ==== 10.5
n2=int(n1)        # 10.5  ===== 10
n2
```

Out[39]:
```
10
```

In [ ]:
```
# Boolean conversion
val1=True
int(val1)
float(val1)
str(val1)
complex(val1)
```

In [47]:
```
int(False)
```

Out[47]:
```
0
```

In [ ]:
```
only string conversion  not working in Boolean. and complex. why?
bool('apple')
```

In [ ]:
```
# Complex to other
val2=100+200j
int(val2)  # error  complex(100) = 100+0j
float(val2) # error
str(val2)  # '100+200j'
bool(val2) #  True
```

In [ ]:
```
val2=100+0j
int(val2)  # error  complex(100) = 100+0j
float(val2) # error
str(val2)  # '100+200j'
bool(val2) #  True
```

In [48]:
```
int(100+200j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[48], line 1
----> 1 int(100+200j)

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

In [49]:
```
int(100+0j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[49], line 1
----> 1 int(100+0j)

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

In [50]:
```
bool(0j)
```

Out[50]:
```
False
```

In [ ]:
```
can u plz run the boolean to string conversion again?
```

In [ ]:
```
bool('apple')
```

In [51]:
```
str(True)
```

Out[51]:
```
'True'
```

In [ ]:

In [ ]:

In [ ]: