Chapter-6: Python Type Casting or Type Conversion

Type Casting or Type Conversion
Type casting or type conversion is the process of converting one type of value to another type.
Python supports 2 types of type casting or type conversion

1. Implicit Type Casting
2. Explicit Type Casting

Implicit Type Casting (Type Conversion)

Implicit type casting is the automatic conversion of one data type into another by **Python itself**, without any programmer instruction.

Python does this to **prevent data loss** and to ensure that operations run smoothly.

It usually happens when **different data types are used together in an expression**.

Key Points

- Done **automatically by Python**
- No need to write conversion functions
- Conversion happens from **lower data type → higher data type**
- Safe conversion (no loss of data)

Common Implicit Conversions

- `int → float`
- `int → complex`

- `float → complex`

Examples

Example 1: int to float

```
a = 10      # int
b = 2.5     # float

c = a + b
print(c)
print(type(c))
```

Output

```
12.5
<class 'float'>
```

`int` is automatically converted to `float`

Example 2: int to complex

```
x = 5       # int
y = 2+3j    # complex

z = x + y
print(z)
print(type(z))
```

Output

```
(7+3j)
<class 'complex'>
```

`int` is automatically converted to `complex`

Example 3: float to complex

```
a = 3.5     # float
b = 1+2j    # complex

c = a + b
print(c)
print(type(c))
```

Output

```
(4.5+2j)
<class 'complex'>
```

`float` is automatically converted to `complex`

Example 4: boolean to int

```
a = True
b = 10

c = a + b
print(c)
print(type(c))
```

Output

```
11
<class 'int'>
```

`True` is treated as 1

Example Where Implicit Casting is NOT Possible

```
a = 10
b = "20"

c = a + b   # Error
```

TypeError: Python cannot automatically convert string to number.

Explicit type casting or type conversion

The type conversion done by a programmer is called explicit type casting or type conversion. This conversion is done by using inbuilt functions provided by python. These functions are type conversion functions.

1. int()
2. float()
3. complex()
4. bool()
5. str()
6. list()
7. tuple()
8. bytes()
9. bytearray()
10. set()
11. frozenset()
12. dict()

int() function

int() function is used to perform the following conversions

1. float to int
2. boolean to int
3. str to int

4. int to int

Syntax: variable-name=int(value=0)

Example:

```
>>> a=int()
>>> print(a,type(a))
0 <class 'int'>

>>> b=int(15)
>>> print(b,type(b))
15 <class 'int'>

>>> c=int(125.89)
>>> print(c)
125

>>> d=int(99.99999)
>>> print(d,type(d))
99 <class 'int'>

>>> e=int(15e-1)
>>> print(e,type(e))
1 <class 'int'>

>>> f=int(True)
>>> print(f,type(f))
1 <class 'int'>

>>> g=int(False)
>>> print(g,type(g))
0 <class 'int'>

>>> h=int("45")
```

```
>>> print(h,type(h))
45 <class 'int'>

>>> i=int("8999")
>>> print(i,type(i))
8999 <class 'int'>

>>> j=int("abc")
Traceback (most recent call last):
  File "<pyshell#18>", line 1, in <module>
    j=int("abc")
ValueError: invalid literal for int() with base 10: 'abc'

>>> h=int("12.67")
Traceback (most recent call last):
  File "<pyshell#19>", line 1, in <module>
    h=int("12.67")
ValueError: invalid literal for int() with base 10: '12.67'

>>> k=int("ab",base=16)
>>> print(k,type(k))
171 <class 'int'>

>>> l=int("101",base=2)
>>> print(l,type(l))
5 <class 'int'>

>>> m=int("0xab",base=16)
>>> n=int("0b101",base=2)
>>> print(m,n,type(m),type(n))
171 5 <class 'int'> <class 'int'>
```

Example:

```
# adding two numbers
```

```
num1=int(input("Enter First Number "))
num2=int(input("Enter Second Number "))

num3=num1+num2

print(f'Sum of {num1} and {num2} is {num3}')
```

Output

```
Enter First Number 100
Enter Second Number 200
Sum of 100 and 200 is 300
```

float() function

This function is used to perform the following conversions

1. float to float
2. int to float
3. bool to float
4. str to float

Syntax: variable-name=float(value=0.0)

```
>>> a=float()
>>> print(a,type(a))
0.0 <class 'float'>

>>> b=float(12.67)
>>> print(b,type(b))
12.67 <class 'float'>

>>> c=float(1267e-2)
>>> print(c,type(c))
12.67 <class 'float'>
```

```
>>> d=float(45)
>>> print(d,type(d))
45.0 <class 'float'>
>>> e=float(0xf)
>>> print(e,type(e))
15.0 <class 'float'>

>>> f=float(0b1010)
>>> print(f,type(f))
10.0 <class 'float'>

>>> g=float(0o12)
>>> print(g,type(g))
10.0 <class 'float'>

>>> i=float("12.45")
>>> print(i,type(i))
12.45 <class 'float'>

>>> j=float("1245e-2")
>>> print(j,type(j))
12.45 <class 'float'>

>>> k=float("12")
>>> print(k,type(k))
12.0 <class 'float'>

>>> l=float(True)
>>> print(l,type(l))
1.0 <class 'float'>

>>> m=float(False)
>>> print(m,type(m))
0.0 <class 'float'>
```

```
>>> n=float(1+2j)
Traceback (most recent call last):
  File "<pyshell#51>", line 1, in <module>
    n=float(1+2j)
TypeError: float() argument must be a string or a real number, not
'complex'
```

Example:

```
# Write a program to input rollno,name and 2 subject marks
# calculate total marks

rollno=int(input("Student Rollno "))
name=input("Student Name ")
sub1=float(input("Subject1Marks "))
sub2=float(input("Subject2Marks "))

total_marks=sub1+sub2

print(f'''StudentRollno:{rollno}
StudentName:{name}
Subject1Marks:{sub1}
Subject2Marks:{sub2}
TotalMarks:{total_marks}''')
```

Output

```
Student Rollno 2
Student Name suresh
Subject1Marks 67.89
Subject2Marks 99.99
StudentRollno:2
StudentName:suresh
Subject1Marks:67.89
Subject2Marks:99.99
```

TotalMarks:167.88

bool() function

This function is used to perform the following conversions.

1. Boolean to boolean
2. Int to boolean
3. Float to boolean
4. Complex to boolean
5. String to boolean

Syntax:

variable-name=bool(value=False)

Example:

```
>>> a=bool()
>>> print(a,type(a))
False <class 'bool'>

>>> b=bool(True)
>>> print(b,type(b))
True <class 'bool'>

>>>c=bool(False)
>>>print(c,type(c))
False <class 'bool'>

>>>d=bool(0)
>>>print(d,type(d))
False <class 'bool'>

>>>e=bool(1)
>>>print(e,type(e))
True <class 'bool'>
```

```
>>>f=bool(100)
>>>print(f,type(f))
True <class 'bool'>
>>>g=bool(-123)
>>>print(g,type(g))
True <class 'bool'>

>>>h=bool(1.5)
>>>print(h,type(h))
True <class 'bool'>

>>>i=bool(0.0)
>>>print(i,type(i))
False <class 'bool'>

>>>j=bool(1+2j)
>>>print(j,type(j))
True <class 'bool'>

>>>k=bool(0+1j)
>>>print(k,type(k))
True <class 'bool'>

>>>l=bool(0+0j)
>>>print(l,type(l))
False <class 'bool'>

>>> m=bool("")
>>> print(m,type(m))
False <class 'bool'>

>>> n=bool("abcd")
>>> print(n,type(n))
True <class 'bool'>
```

```
>>> o=bool("0")
>>> print(o,type(o))
True <class 'bool'>

>>> p=bool("True")
>>> print(p,type(p))
True <class 'bool'>

>>> q=bool("False")
>>> print(q,type(q))
True <class 'bool'>
```

str() function

This function is used to perform the following conversion.
This function is used to convert any type to string

1. Str to str
2. Int to str
3. Float to str
4. Complex to str
5. Bool to str

```
>>> s1=str()
>>> print(s1,type(s1))
 <class 'str'>

>>> s2=str("python")
>>> print(s2,type(s2))
python <class 'str'>

>>> s3=str(45)
>>> print(s3,type(s3))
45 <class 'str'>
```

```
>>> s4=str(1.5)
>>> print(s4,type(s4))
1.5 <class 'str'>

>>> s5=str(1+2j)
>>> print(s5,type(s5))
(1+2j) <class 'str'>

>>> s6=str(True)
>>> print(s6,type(s6))
True <class 'str'>

>>> s7=str(False)
>>> print(s7,type(s7))
False <class 'str'>
```

complex() function

This function is used to perform the following conversions

1. Complex to complex
2. Int to complex
3. Float to complex
4. Boolean to complex
5. String to complex

Syntax: complex(real=0.0,imag=0.0)

```
>>> c1=complex()
>>> print(c1)
0j
>>> print(c1.real,c1.imag)
0.0 0.0
>>> c2=complex(2)
```

```
>>> print(c2)
(2+0j)
>>> c3=complex(1.5)
>>> print(c3)
(1.5+0j)
>>> c4=complex(True)
>>> print(c4)
(1+0j)
c5=complex(1,2)
>>> print(c5)
(1+2j)
>>> c6=complex(1.5,2.0)
>>> print(c6)
(1.5+2j)
>>> c7=complex("1+2j")
>>> print(c7)
(1+2j)
>>> print(c7.real,c7.imag)
1.0 2.0
>>> c8=complex("1")
>>> print(c8)
(1+0j)
>>> c9=complex("2j")
>>> print(c9.real,c9.imag)
0.0 2.0
>>>
```

Example:

```
# Write a progrm to input two complex numbers and add
# (OR) print sum of two complex numbers

comp1=complex(input("Enter Complex Number1 "))
comp2=complex(input("Enter Complex Number2"))
```

```
comp3=comp1+comp2

print(f'''Complex Number1 {comp1}
Complex Number2 {comp2}
Sum is {comp3}''')
```

Output

```
Enter Complex Number1 1+2j
Enter Complex Number22+1j
Complex Number1 (1+2j)
Complex Number2 (2+1j)
Sum is (3+3j)
```