Type Conversion or Type Casting

Type Conversion is a process converting one type of value to another type (OR) converting one type of object to another type This type conversion is done using type conversion functions.

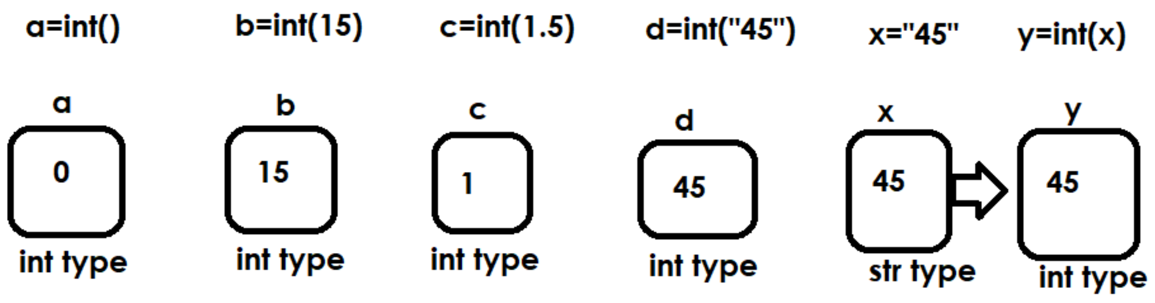
1. int()
2. float()
3. complex()
4. bool()
5. str()

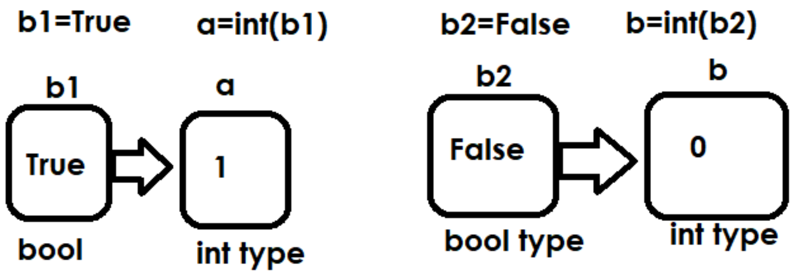
int() function

It is a predefined function in python.
This function is used to perform the following conversions.

1. Int to int
2. Float to int
3. Bool to int
4. String to int

Syntax: int([value])





```
# Write a program to print sum of two integers
# input two integers from keyboard (during runtime)

n1=input("Enter First Integer Value :")
n2=input("Enter Second Integer Value :")
n3=int(n1)+int(n2)
print(n1,n2,n3)
```

Output

```
Enter First Integer Value :15
Enter Second Integer Value :21
15 21 36
```

Note: whenever string is converted into integer type, the string must contain integer value. If string contains any other type of value, int function raises ValueError

```
>>> a=int("1.5")
```

```
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    a=int("1.5")
ValueError: invalid literal for int() with base 10: '1.5'
>>> b=int("abcd")
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    b=int("abcd")
ValueError: invalid literal for int() with base 10: 'abcd'
>>> c=int("65")
>>> c
65
>>> d=int(1.65)
>>> print(d)
1
>>> e=int(True)
>>> e
1
>>> f=int(False)
>>> f
0
>>> g=int(1+2j)
Traceback (most recent call last):
  File "<pyshell#10>", line 1, in <module>
    g=int(1+2j)
TypeError: int() argument must be a string, a bytes-like object or a
real number, not 'complex'
```

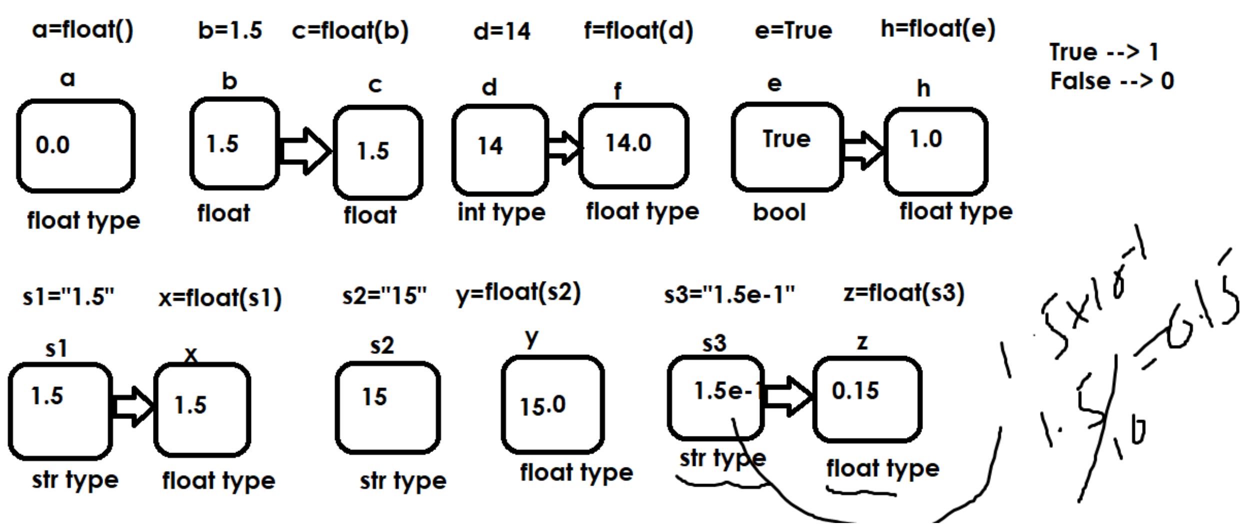
float() function

It is a predefined function in python

This function is used to perform the following conversions

1. float to float
2. int to float
3. boolean to float
4. string to float

Syntax: float([value])



Example:
# Write a program to find area of circle
# input radious of circle from keyboard

```
r=float(input("Enter Radious of Circle "))
area=3.147*r*r
print(area)
```

Output
```
Enter Radious of Circle 1.2
4.53168
```

Example:
```
a=float()
```

```
print(a)
b=float(15)
print(b)
c=float("45")
print(c,type(c))
d=float("1.5")
print(d,type(d))
e=float("1.5e-1")
print(e,type(e))
f=float(True)
print(f,type(f))
g=float(False)
print(g,type(g))
#h=float(1+2j)
#print(h)
#i=float("abc")
#print(i)
```

Output

```
0.0
15.0
45.0 <class 'float'>
1.5 <class 'float'>
0.15 <class 'float'>
1.0 <class 'float'>
0.0 <class 'float'>
```

complex() function

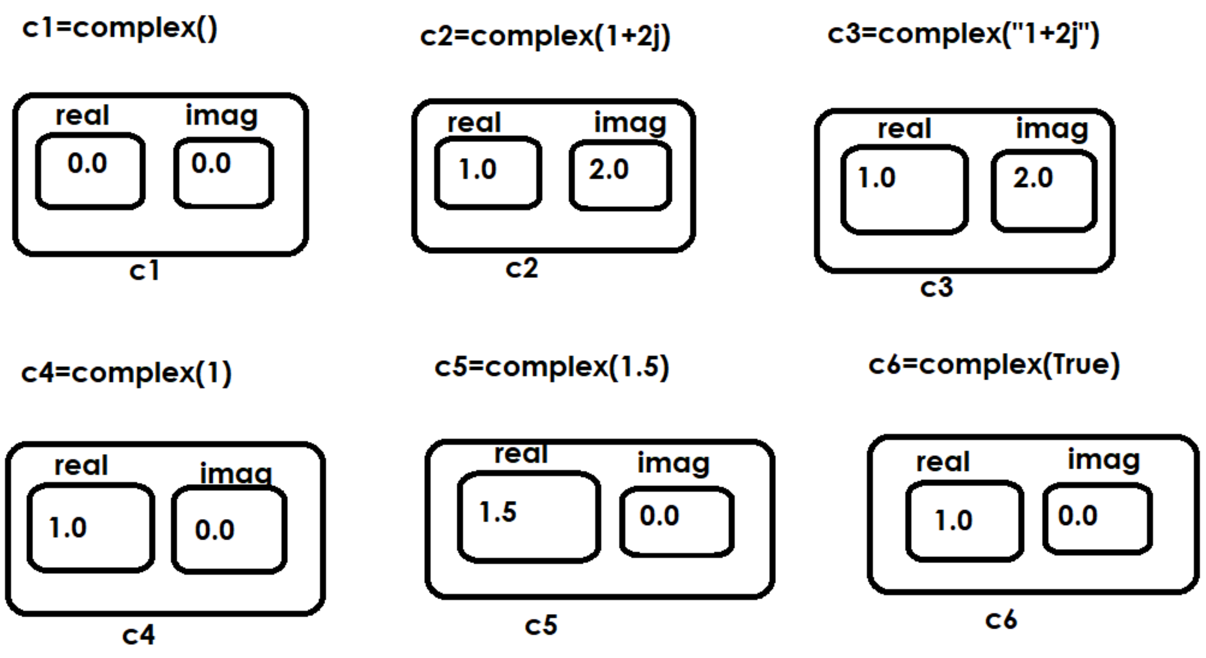
It is a predefined function in python

This function is used to perform the following conversions

1. complex to complex

2. int to complex
3. float to complex
4. string to complex

Syntax: complex([value])



Example:

```
# Write a program to add two complex numbers
# input complex numbers from keyboard

comp1=complex(input("Input Complex Number1 :"))
comp2=complex(input("Input Complex NUmber2 :"))
comp3=comp1+comp2

print(type(comp1),type(comp2),type(comp3))
print(comp1,comp2,comp3)
```

Output

```
Input Complex Number1 :1+2j
Input Complex NUmber2 :1+3j
<class 'complex'> <class 'complex'> <class 'complex'>
(1+2j) (1+3j) (2+5j)
```

Example:

```
c1=complex()
print(c1.real,c1.imag)
c2=complex(1+2j)
print(c2,c2.real,c2.imag)
c3=complex("2+4j")
print(c3,c3.real,c3.imag)
c4=complex("2")
print(c4,c4.real,c4.imag)
c5=complex("2j")
print(c5,c5.real,c5.imag)
c6=complex(True)
print(c6,c6.real,c6.imag)
```

Output

```
0.0 0.0
(1+2j) 1.0 2.0
(2+4j) 2.0 4.0
(2+0j) 2.0 0.0
2j 0.0 2.0
(1+0j) 1.0 0.0
```

Note: Any python automatically import on default library called __builtins__

bool() function

It is a predefined function in python

This function 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: bool([value])

Example:

```
b1=bool()
print(b1)
b2=bool(True)
print(b2)
b3=bool(1)
print(b3,type(b3))
b4=bool(0)
print(b4,type(b4))
b5=bool(200)
print(b5,type(b5))
b6=bool(-1)
print(b6,type(b6))
b7=bool(1+2j)
print(b7,type(b7))
b8=bool(0+0j)
print(b8,type(b8))
b9=bool(1+0j)
print(b9,type(b9))
b10=bool(0+1j)
print(b10,type(b10))
b11=bool("A")
```

```
print(b11,type(b11))
b12=bool("ABCD")
print(b12,type(b12))
b13=bool("False")
print(b13,type(b13))
b14=bool("")
print(b14)
b15=bool("   ")
print(b15)
```

Output

```
False
True
True <class 'bool'>
False <class 'bool'>
True <class 'bool'>
True <class 'bool'>
True <class 'bool'>
False <class 'bool'>
True <class 'bool'>
True <class 'bool'>
True <class 'bool'>
True <class 'bool'>
True <class 'bool'>
False
True
```

str()

str() is predefined function in python
This function performs the following conversions

1. str to str
2. int to string

3. float to string
4. complex to string
5. bool to string

Example:

```
a=45
b=str(a)
print(a,b,type(a),type(b))
c=1.5
d=str(c)
print(c,d,type(c),type(d))
d=1+2j
e=str(d)
print(d,e,type(d),type(e))
f=True
g=str(f)
print(f,g,type(f),type(g))
s1="PYTHON"
s2=str(s1)
print(s1,s2,type(s1),type(s2))
```

Output

```
45 45 <class 'int'> <class 'str'>
1.5 1.5 <class 'float'> <class 'str'>
(1+2j) (1+2j) <class 'complex'> <class 'str'>
True True <class 'bool'> <class 'str'>
PYTHON PYTHON <class 'str'> <class 'str'>
```

Operators

What is operator?

Operator is a special symbol, which is used to perform operations. Based on the operands on which it performs operations, the operators are classified into 3 categories

1. Binary Operators
2. Unary Operators
3. Ternary Operators

Binary Operator: An operator required 2 operands to perform operation is called binary operator

Unary Operator: an operator required 1 operand to perform operation is called unary operator

Ternary Operator: an operator required 3 operands to perform operation is called ternary operator.

Types of operators

1. Arithmetic Operators
2. Relational Operators
3. Logical Operators
4. Assignment Operators
5. Membership Operators
6. Identity Operators
7. Bitwise Operators
8. Conditional Operators
9. Walrus Operator