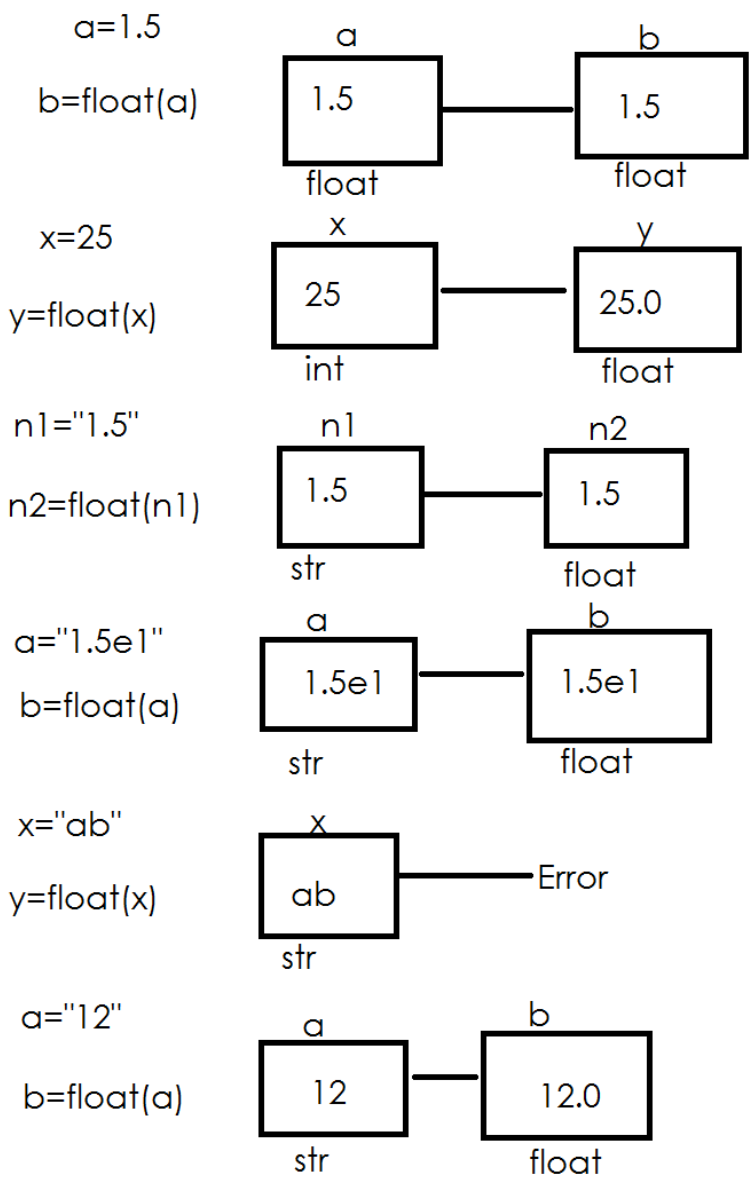
float() function
This function is used to perform the following conversions.

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

Syntax: float(value)




```
>>> a=float(1.5)
```

```
>>> b=float(12)
>>> c=float("1.5")
>>> d=float("1.5e1")
>>> f=float(True)
>>> g=float(False)
>>> print(a,b,c,d,f,g,sep="\n")
1.5
12.0
1.5
15.0
1.0
0.0
>>> h=float("ab")
Traceback (most recent call last):
  File "<pyshell#8>", line 1, in <module>
    h=float("ab")
ValueError: could not convert string to float: 'ab'
>>>
```

Example:

```
# Write a program to input two float numbers and add

n1=input("Enter first float number")
n2=input("Enter second float number")
n3=float(n1)+float(n2)
print(n1,n2,n3)
```

Output:

```
Enter first float number1.5
Enter second float number2.5
1.5 2.5 4.0
```

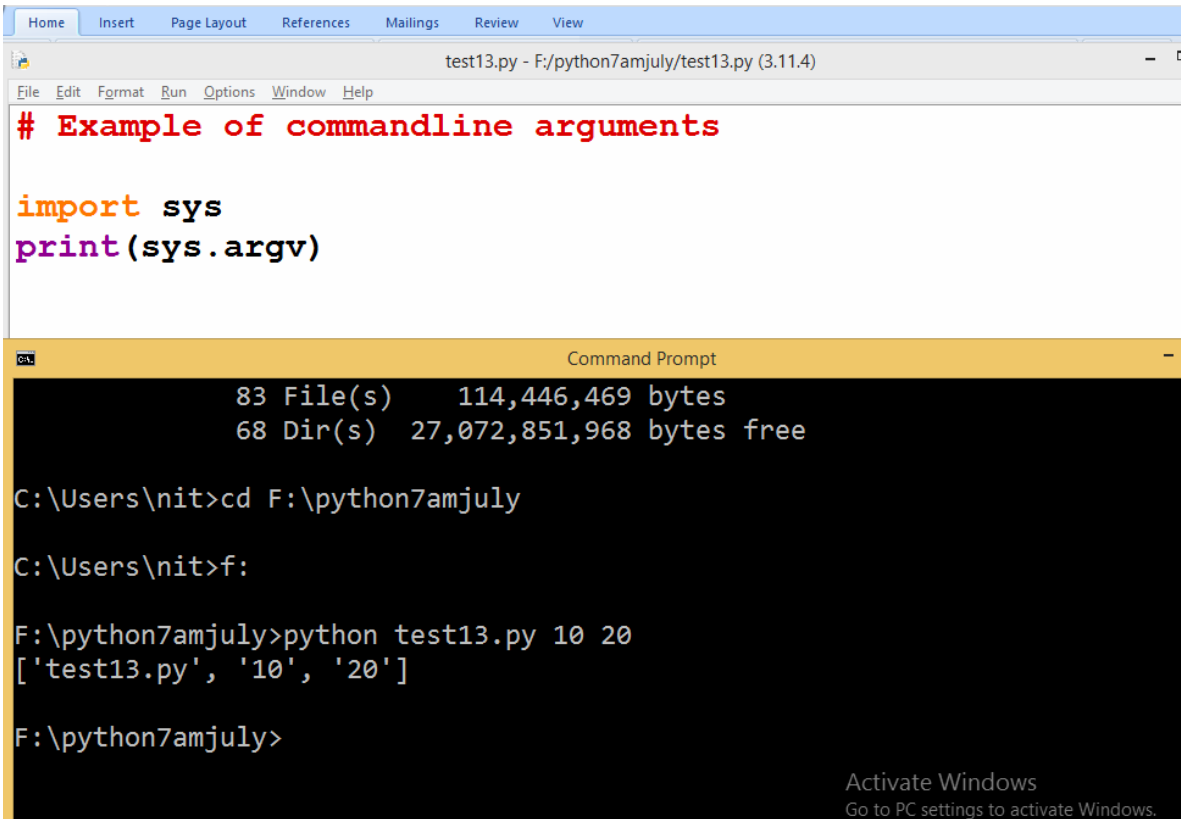
Command line arguments

The values given from command prompt to program are called command line arguments.

Command line arguments are stored inside a predefined variable called “argv”. It is in a library called sys.

Applications:

1. **developing commands**
2. **utility programs**



Example:

test14.py - F:/python7amjuly/test14.py (3.11.4)

```python
# Write a program to add two integers to using
# commandline argument

import sys
n1=int(sys.argv[1])
n2=int(sys.argv[2])
n3=n1+n2
print(n1,n2,n3)
```

Command Prompt

```
C:\Users\nit>f:

F:\python7amjuly>python test13.py 10 20
['test13.py', '10', '20']

F:\python7amjuly>python test14.py 100 200
100 200 300

F:\python7amjuly>
```

Example:

test15.py - F:/python7amjuly/test15.py (3.11.4)

```python
# write a program to read student details from
# command line arguments

import sys
rollno=int(sys.argv[1])
name=sys.argv[2]
course=sys.argv[3]
fee=float(sys.argv[4])

print(rollno,name,course,fee,sep="\n")
```

Command Prompt

```
F:\python7amjuly>python test15.py 101 naresh python 4000.0
101
naresh
python
4000.0

F:\python7amjuly>
```

Program with command line arguments is tested from IDLE using
The following option

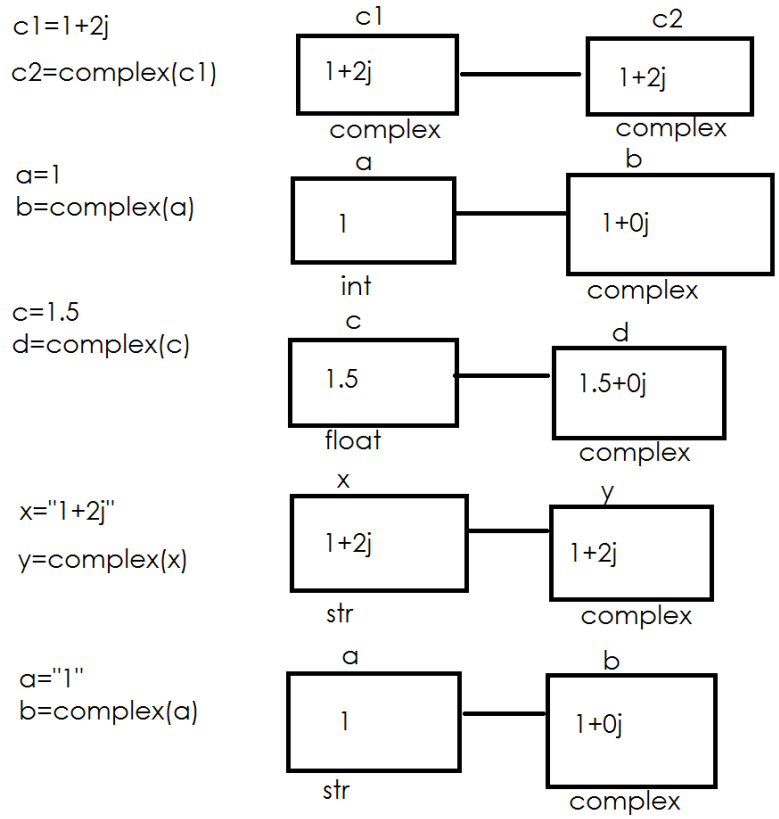
Run → Run Customized

complex()

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
5. bool to complex

Syntax1: complex(value)
Syntax2: complex(real=0.0,img=0.0)




```
>>> a=complex(1+2j)
>>> b=complex(1)
```

```
>>> c=complex(2)
>>> d=complex(1.2)
>>> e=complex("1+2j")
>>> f=complex("1")
>>> g=complex("1.2")
>>> h=complex("1j")
>>> i=complex(True)
>>> j=complex(False)
>>> print(a,b,c,d,e,f,g,h,i,j,sep="\n")
(1+2j)
(1+0j)
(2+0j)
(1.2+0j)
(1+2j)
(1+0j)
(1.2+0j)
1j
(1+0j)
0j
```

Example:

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


c1=input("Enter First Complex Number ")
c2=input("Enter Second Complex Number ")
c3=complex(c1)+complex(c2)

print(c1,c2,c3)
```

Output:

```
Enter First Complex Number 1+2j
Enter Second Complex Number 1+1j
1+2j 1+1j (2+3j)
```

bool() function

This function performs the following conversions

1. bool to bool
2. int to bool

3. float to bool

Syntax: bool(value)

```
>>> b1=bool(1)
>>> print(b1)
True
>>> b2=bool(0)
>>> print(b2)
False
>>> b3=bool(65)
>>> print(b3)
True
>>> b4=bool(1.0)
>>> print(b4)
True
>>> b5=bool(0.0)
>>> print(b5)
False
>>> b6=bool(1+2j)
>>> print(b6)
True
>>> b7=bool("A")
>>> print(b7)
True
>>> b8=bool("False")
>>> print(b8)
True
>>> b9=bool("naresh")
>>> print(b9)
True
>>> b10=bool("0")
>>> print(b10)
True
>>> ord("A")
65
>>> ord("F")
70
>>> ord("0")
48
```

```
>>> chr(65)
'A'
>>> chr(70)
'F'
>>> chr(48)
'0'
```