Type Casting For Floating

In [1]:
```
float(20) # In this Scenario integear
```

Out[1]: 20.0

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

Out[3]: 1.0

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

Out[4]: 0.0

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

Out[6]: 10.0

Complex Number

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

Out[8]: (10+0j)

In [9]:
```
complex(10,20)
```

Out[9]: (10+20j)

In [11]:
```
complex(True)
```

Out[11]: (1+0j)

In [12]:
```
complex(False)
```

Out[12]: 0j

In [14]:
```
complex(3.2,56)
```

Out[14]: (3.2+56j)

In [15]:
```
#Boolen UseCase
```

In [17]:
```
boolen(10)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[17], line 1
----> 1 boolen(10)

NameError: name 'boolen' is not defined
```

In [25]:
```
bool(10) # In this Case Scenario Nonzero Value Always so Output is True
```

Out[25]: True

In [29]:
```
bool(0) # In this Case valye is zero so out put is false
```

Out[29]: False

In [30]:
```
bool(1j)
```

Out[30]: True

In [31]:
```
bool(1+2j)
```

Out[31]: True

In [32]:
```
bool(0j)
```

Out[32]: False

In [33]:
```
bool('ten')
```

Out[33]: True

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

Out[34]: True

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

Out[35]: False

String-Indexing,Advnaced Slicing

In [43]:
```
a1='HELLOPYTHON'
```

In [44]:
```
a1[:]
```

Out[44]: 'HELLOPYTHON'

In [46]:
```
a1[10]
```

Out[46]: 'N'

In [48]:
```
a1[-9]
```

Out[48]: 'L'

In [49]:
```
a1[-4]
```

Out[49]: 'T'

In [50]:
```
for i in a1:
    print(i)
```

```
H
E
L
L
O
P
Y
T
H
O
N
```

In [52]:
```
a1[::-1]
```

Out[52]: 'NOHTYPOLLEH'

In [53]:
```
a1[0:5]
```

Out[53]: 'HELLO'

In [54]:
```
a1[0:-1]
```

Out[54]: 'HELLOPYTHO'

In [56]:
```
a1[1:20]
```

Out[56]: 'ELLOPYTHON'

In [57]:
```
a1[0:20]
```

Out[57]: 'HELLOPYTHON'

In [3]:
```
a1:'hellopython'
```

In [14]:
```
a1
```

Out[14]: 'hellopython'

In [5]:
```
a1='hellopython'
```

In [6]:
```
a1
```

Out[6]: 'hellopython'

In [7]:
```
a1[:5]
```

Out[7]: 'hello'

In [8]:
```
a1[5:]
```

Out[8]: 'python'

In [9]:
```
a1[0:11:2] # In this case 2 steps are gap
```

Out[9]: 'hloyhn'

In [10]:
```
a1[0:11:2]
```

Out[10]: 'hloyhn'

In [13]:
```
a1[0:11:5]
```

Out[13]: 'hpn'

In [15]:
```
a1[::-2]
```

Out[15]: 'nhyolh'

In [16]:
```
a1[::-3]
```

Out[16]: 'ntoe'

Print Statement Condition

In [17]:
```
num1=20
num2=30
add=num1+num2
print('The addition of {} and {} is= {}'.format(num1,num2,add))
```

```
The addition of 20 and 30 is= 50
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: