In [2]:
```
a=10
b=20
a+b
```

Out[2]: 30

In [ ]:
```
n1=100
n2=200
```

In [4]:
```
n1+n2
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[4], line 1
----> 1 n1+n2

NameError: name 'n1' is not defined
```

In [6]:
```
omkar
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[6], line 1
----> 1 omkar

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

In [8]:
```
omkar=200
```

In [10]:
```
omkar
```

Out[10]: 200

In [14]:
```
python=100
```

In [16]:
```
python
```

Out[16]: 100

Data types

In [ ]:
```
- int (integer)

- float

- str (string)

- bool (boolean)

- list

- tuple

- dict

- set
```

```
- frozenset
- bytes
- bytearray
- memoryview
- range
- None
- complex
```

int

In [20]:
```
number=100
```

In [22]:
```
number
```

Out[22]: 100

In [24]:
```
type(number)
```

Out[24]: int

float

In [27]:
```
number_one=100.5
type(number_one)
```

Out[27]: float

string

In [30]:
```
name='naresh it'
name
```

Out[30]: 'naresh it'

In [32]:
```
name1="python"
name1
```

Out[32]: 'python'

In [34]:
```
print(name)  # when you print quotes will not display
```

naresh it

In [36]:
```
print(name1) # so dont confuse bcz quotes are not display
```

python

In [38]:
```
type(name)
```

Out[38]: str

In [40]:
```
type(name1)
```

Out[40]: str

In [ ]:
```
100 # with out decimal === integer
100.25 # with decimal ==== float
```

In [42]:
```
name='naresh it'
print(type(name))
```

```
<class 'str'>
```

In [44]:
```
name='naresh it'
type(print(name))
```

```
naresh it
```

Out[44]: NoneType

In [46]:
```
name='naresh it'
type(name)
```

Out[46]: str

- print only for to display the output
- do not perform any operation on print
- strings can write either single or double quotes
- but python by default display by using single quotes only
- when we print the string the output in with out quotes

In [53]:
```
str1="i love 'python'"
str1
```

Out[53]: "i love 'python'"

In [55]:
```
str2='i love "python"'
str2
```

Out[55]: 'i love "python"'

In [57]:
```
print(str1)
print(str2)
```

```
i love 'python'
i love "python"
```

Doc string

- triple quotes also use in strings
- triple quotes means multiline comment
- in jupyter notebook we have markdown to write the information

- but in vscode or pycharm there is no markdown option
- in order write any information we need to use triple quotes as multiline comment
- this is called as **Doc String**
- triple quotes means conveying the information

In [ ]:
```
"""
**Doc string**

- triple quotes also use in strings

- triple quotes means multiline comment

- in jupyter notebook we have markdown to write the information

- but in vscode or pycharm there is no markdown option

- in order write any information we need to use triple quotes as multiline comme

- this is called as **Doc String**

- triple quotes means conveying the information
"""
```

bool

In [63]:
```
n1=True
type(n1)
```

Out[63]:
```
bool
```

In [65]:
```
n2=False
type(n2)
```

Out[65]:
```
bool
```

In [ ]:
```
true=100    # int
true=True  # bool
true='true' # str

# true==== variable
# True === boolean
#'true' === string
```