Data Types

- integer : int
- float
- string
- list
- tuple
- set
    - frozen set
- dictionary
- bool
- complex
- range
- byte
- byte array
- memory view
- None type

- myth
    - only 14-15 data type available

$integer$

In [1]:
```
number=100
number
```

Out[1]:
```
100
```

In [3]:
```
type(number)
```

Out[3]:
```
int
```

numbers system

- Binary number system
- Decimal number system

- Octa number system
- hexa number system

In [6]:
```
9*9
5*5
81*25
```

Out[6]: 2025

In [8]:
```
40*40+20*20+5*5
```

Out[8]: 2025

In [9]:
```
sum([i**3 for i in range(1,10)])
```

Out[9]: 2025

Binary number system

- it is denoted with **0b** or **0B**
- Binary means two
- we have generally 0 to 9 digits available
- binary means only twi digits allowed : 0 and 1
- ex: 0b101 0b111 0b000 0B001
- not valid : 0b123 0B1000012

In [15]:
```
0b111,0b101
```

Out[15]: (7, 5)

In [14]:
```
0b11101
```

Out[14]: 29

In [ ]:
```
2^2   2^1    2^0
4      2      1
###################################
0      0      0     0
0      0      1     1
0      1      0     2
0      1      1     3
1      0      0     4
1      0      1     5
1      1      0     6
1      1      1     7
```

In [ ]:
```
2^3   2^2   2^1    2^0
8     4      2      1
    ###################################
```

```
0   0     0     0     0
0   0     0     1     1
0   0     1     0     2
0   0     1     1     3
0   1     0     0     4
0   1     0     1     5
0   1     1     0     6
0   1     1     1     7
1   0     0     0     8
1   0     0     1     9
1   0     1     0     10
1   0     1     1     11
1   1     0     0     12
1   1     0     1     13
1   1     1     0     14
1   1     1     1     15
```

octa number system

- it is denoted with **0o** or **0O**
- octa means 8
- we have generally 0 to 9 digits available
- octa means only twi digits allowed : 0 1 2 3 4 5 6 7
- ex: 0o567 0o111 0o001 0O672
- not valid : 0o768 0O169

In [16]:
```
0o123
```

Out[16]: 83

hexa number system

- it is denoted with **0x** or **0X**
- hexa means 16
- we have generally 0 to 9 digits available
- hexa means only twi digits allowed : 0 to 9 A to F
- A: 10 B:11 C: 12 D: 13 E: 14 F:15
- ex: 0xabc 0XA1
- not valid : 0xefg

In [17]:
```
0xabc
```

Out[17]: 2748

2025

- binary
- oct
- hexa

In [18]:
```
0b11111101001
```

Out[18]: 2025

In [19]:
```
0b11111101001
```

Out[19]: 2025

In [22]:
```
0X7E9,0o3751
```

Out[22]: (2025, 2025)

float

In [23]:
```
number=10.5
type(number)
```

Out[23]: float

e represntation

In [24]:
```
1e1
```

Out[24]: 10.0

In [25]:
```
1e2
```

Out[25]: 100.0

In [26]:
```
1e3 # 1*1000
```

Out[26]: 1000.0

In [27]:
```
2e4   # 2* 10000
```

Out[27]: 20000.0

In [ ]:
```
1e1   # 1*10=10
1e2   # 1*100=100
2e3   # 2*1000=2000
4e4   # 4*10000=40000
```

In [29]:
```
print(1e-1)   # 1/10=0.1
print(1e-2)   # 1/100=0.01
print(2e-3)   # 2/1000=0.002
print(4e-4)   # 4/10000=0.0004
```

```
0.1
0.01
0.002
0.0004
```

In [ ]:
```
# integer data type
# float data type
```

boolean

In [1]:
```
val=True
val
```

Out[1]:
```
True
```

In [2]:
```
type(val)
```

Out[2]:
```
bool
```

In [3]:
```
val1=False
type(val1)
```

Out[3]:
```
bool
```

Strings

- English format representation
- strings represent with quotes
- we can use single quotes, double quotes also triple quotes

In [4]:
```
name='python'
name
```

Out[4]:
```
'python'
```

In [5]:
```
type(name)
```

Out[5]:
```
str
```

In [6]:
```
print(name)
# when we print we will not able to see the quotes
```

```
python
```

In [7]:
```
val='10'
type(val)
```

Out[7]:
```
str
```

In [8]:
```
val1=10
type(val1)
```

Out[8]:
```
int
```

In [9]:
```
name="naresh it"
name
# but the output displayed as single quotes
```

Out[9]:
```
'naresh it'
```

In [10]:
```
type(name)
```

Out[10]:
```
str
```

In [11]:
```
print(name)
```

```
naresh it
```

In [14]:
```
str1="i love 'python'"
print(str1)
```

```
i love 'python'
```

In [15]:
```
str1='i love "python"'
print(str1)
```

```
i love "python"
```

- Keep entire string in Double quotes then highlite the word with single quote vice versa

triple quotes

- triple quotes can not use in coding part
- triple quotes means conveying the inforamtion
- this is process is called as **Doc string**
- in jupyter notebook we have markdown option
- but in vscode or pycharm there is no markdown option
- so user will convey the information by providing triple quotes

In [ ]:
```
"""hai
today im learning python
its easy"""
```

In [ ]:
```
true=True  # works
true='True' # w
true=10.5 # w
True=10 # f
True=True # f
```

In [16]:
```
True=10
```

```
  Cell In[16], line 1
    True=10
    ^
SyntaxError: cannot assign to True
```

In [17]:
```
True=True
```

```
  Cell In[17], line 1
    True=True
    ^
SyntaxError: cannot assign to True
```

In [ ]: