Heading 1

Heading 2

Heading 3 ¶

Heading 4

Heading 5

This is Markdown cell

In [12]:
```
# This is a single comment

''' This is multiline
comments.'''

print("Hello")
```

executed in 7ms, finished 16:30:02 2024-08-05

```
Hello
```

Basics

- Comments
- Identifiers & Variables
- Keywords
- Data Types
- Operators
- I/O statements
- Conditional Statements
- Loops (for,while)
- Functions
- Modules
- File Handling
- Exception Handling
- OOPS

Database

- Mysql
- MongoDB(React)

Framework

- Django

Identifiers and Variables:

- Identifiers are the variable name.
- Variable is used to store the value.

NOTE: id() is method used to find the memory address of any variable.

Rules for creating Identifier or Variable name

- Identifiers should not start with a number.
- Identifiers can be alphanumeric (combination of alphabets and numbers).
- Identifiers should not contain any special characters except underscore "_".
- We can start a variable name using underscore "_".

In [3]:
```
a=10
print(a,id(a))

b="social"
print(b,id(b))
```

executed in 8ms, finished 15:31:25 2024-08-06

```
10 140718947550280
social 2454380577904
```

In [10]:
```
abc_xyz=100
print(abc_xyz)
```

executed in 7ms, finished 15:34:25 2024-08-06

```
100
```

In [11]:
```
_xyz=190
print(_xyz)
```

executed in 6ms, finished 15:35:30 2024-08-06

```
190
```

Keywords : These are the reserved words used for specific purpose.

['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

- we have a total of 35 keywords in python.

In [17]:
```
import keyword
print(len(keyword.kwlist))
```

executed in 7ms, finished 15:39:35 2024-08-06

```
35
```

Data Types:

- Numeric Data Type:
  - In this we have int, float and complex.

In [25]:
```
a=10,20
b=10.5

print(a,type(a))
print(b,type(b))

c=5-6j
print(c,type(c))

print("real:",c.real)
print("imag:",c.imag)
```

executed in 9ms, finished 15:47:18 2024-08-06

```
2
<class 'tuple'>
(5-6j) <class 'complex'>
real: 5.0
imag: -6.0
```

- Sequence Data Type
  - In this we have list,tuple and range.
  - Syntax: range(start,end,stepvalue)

In [31]:
```
l1=[1,2,3,"hello","abc",19.5,"social",100]
print(l1,type(l1))

l1[4]="bye"

print(l1)
t1=(1,2,3,4,56,"heloo")
print(t1,type(t1))
print(t1)
```

executed in 10ms, finished 15:53:11 2024-08-06

```
[1, 2, 3, 'hello', 'abc', 19.5, 'social', 100] <class 'list'>
[1, 2, 3, 'hello', 'bye', 19.5, 'social', 100]
(1, 2, 3, 4, 56, 'heloo') <class 'tuple'>
(1, 2, 3, 4, 56, 'heloo')
```

In [45]:
```
## Range

ra=range(1,50)
# print(ra,type(ra))

r=range(1,50,3)
# print(r,type(ra))
```

executed in 8ms, finished 15:57:25 2024-08-06

In [46]:
```
for i in ra:
    print(i,end=" ")

print("="*50)
for i in r:
    print(i,end=" ")
```

executed in 8ms, finished 15:57:40 2024-08-06

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 ================
==================================
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49
```

3.Set Data Type

- Same like list , tuple and range set also stores multiple values in it.

In [49]:
```
s1={1,2,3,4,5,6,7,8,"abc",1,2,3}
print(s1,type(s1))
```

executed in 8ms, finished 16:03:51 2024-08-06

```
{1, 2, 3, 4, 5, 6, 7, 8, 'abc'} <class 'set'>
```

In [53]:
```
s2={100,2,1,44,0,"True",True,False,"Abc"}
print(s2)

for i in s2:
    print(i)
```

executed in 8ms, finished 16:07:50 2024-08-06

```
{0, 1, 2, 'True', 100, 'Abc', 44}
0
1
2
True
100
Abc
44
```

4.Map Data Type

- Map data type stores thevalues in the form of key and value pairs.
- We call this map data type as Dictionary

In [70]:
```
d1={12:"vinay",2:30,3:"vinay",4:"Hari"}
print(d1,type(d1))

# d1["sal"]=40000
# d1['location']="Hyderabad"

# print(d1)

# print(d1['name'])
# print(d1['age'])
```

executed in 8ms, finished 16:23:31 2024-08-06

```
{12: 'vinay', 2: 30, 3: 'vinay', 4: 'Hari'} <class 'dict'>
```

Q) Remove Duplicates from the given List.

In [72]:
```
arr=[1,2,3,4,5,1,2,3,6,7,9,54,74,3,5]
s1=set(arr)
print(list(s1))
```

executed in 7ms, finished 16:27:30 2024-08-06

```
[1, 2, 3, 4, 5, 6, 7, 9, 74, 54]
```

Type Casting: Conversion of one data type to another data type is called Type Casting or type conversion.

- We have two types of type casting:
  1. Implicit Type casting
  2. Explicit Type casting

In [74]:
```
a=10
b=1.6

print(int(a+b))
```

executed in 8ms, finished 16:28:56 2024-08-06

```
11
```

5.String Data Type

In [81]:
```
a="Hello"
b='socialtek'

c=''' this is
string data
 type '''

print(a,type(a))
print(b,type(b))
print("="*50)
print(c,type(c))
```

executed in 9ms, finished 16:35:26 2024-08-06

```
Hello <class 'str'>
socialtek <class 'str'>
==================================================
 this is
string data
 type  <class 'str'>
```

6.Boolean Data Type

In [90]:
```
a=True
b=False

print(a,type(a))
print(b,type(b))

print(bool(1))
print(bool(0))
print(bool(-10))
print(bool(""))
print(bool(" "))
print(bool("True"))
print(bool(False))
```

executed in 10ms, finished 16:40:35 2024-08-06

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

7.None Data Type

- It stores empty Values in it.

In [93]:
```
a=None

print(a,type(a))
```

executed in 7ms, finished 16:44:35 2024-08-06

```
None <class 'NoneType'>
```

In [ ]:
```

```