```
var_i = -10
print(var_i)
```

```
     -10
```

```
octal_v = 0o25
hex_v = 0x1F

print("Octal Value:", octal_v)
print("Hexadecimal Value:", hex_v)
```

```
     Octal Value: 21
     Hexadecimal Value: 31
```

```
floating_V = 3.14
print(floating_V)
```

```
     3.14
```

```
complex_v = 3+5j
print(complex_v)
'''
a=3
b=5
print(complex(a,b))
'''
```

```
     (3+5j)
     '\na=3\nb=5\nprint(complex(a,b))\n'
```

```
int_v = 10
long_v = 2560000
float_v = 2.44
str_v = "Tilak's Phone"

print(int_v)
print(long_v)
print(float_v)
print(str_v)
```

```
     10
     2560000
     2.44
     Tilak's Phone
```

```
float_1 = 3.14
float_2 = 10.25
print(float_1)
print(float_2)
del float_1
#print(float_1) -- throws an error
```

```
     3.14
     10.25
```

```
#7
```

! /usr/bin/python

Its called as a `Shebang line` in unix operation system. In Unix by default it uses the `Bash` to execute the scripe. So to change specify which bash the OS should use for the script we specify the `Shebang line` as the 1st line in that script. so it sets the required scripting tool.

```
str_v = "DeepSphere.AI"

print(str_v)
print(str_v[0])
print(str_v[2:5])
print(str_v[2:])
print(str_v*2)
print(str_v+" Entreprise AI and IoT for Analytics")
```

```
DeepSphere.AI
D
epS
epSphere.AI
DeepSphere.AIDeepSphere.AI
DeepSphere.AI Entreprise AI and IoT for Analytics
```

```
list_1 = [1,3,5,"LV",8,9]

print(list_1)
print(list_1[0])
print(list_1[1:3])
print(list_1[2:])
print(list_1 *2)
print(list_1 + [2,4])
```

```
[1, 3, 5, 'LV', 8, 9]
1
[3, 5]
[5, 'LV', 8, 9]
[1, 3, 5, 'LV', 8, 9, 1, 3, 5, 'LV', 8, 9]
[1, 3, 5, 'LV', 8, 9, 2, 4]
```

```
tup_1 = (2,4,6,8,10)

print(tup_1)
print(tup_1[0])
print(tup_1[1:3])
print(tup_1[2:])
print(tup_1*2)
print(tup_1 + ("LV","DeepSphere"))
```

```
(2, 4, 6, 8, 10)
2
(4, 6)
(6, 8, 10)
(2, 4, 6, 8, 10, 2, 4, 6, 8, 10)
(2, 4, 6, 8, 10, 'LV', 'DeepSphere')
```

```
dict_1 = {"Movie":"Bahubali","Hero":"Prabhas","Director":"SSR","MD":"MMK"}

print(dict_1["Movie"])
print(dict_1["Movie"],dict_1["Hero"])
print(dict_1)
print(dict_1.keys())
print(dict_1.values())
```

```
Bahubali
Bahubali Prabhas
{'Movie': 'Bahubali', 'Hero': 'Prabhas', 'Director': 'SSR', 'MD': 'MMK'}
dict_keys(['Movie', 'Hero', 'Director', 'MD'])
dict_values(['Bahubali', 'Prabhas', 'SSR', 'MMK'])
```

```
my_set = set()

print(type(my_set))
```

```
<class 'set'>
```

```
sets = [{1,2,3},{"vk","LV","DSAI"},{4,5,6}]
print(sets)

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

```
[{1, 2, 3}, {'DSAI', 'LV', 'vk'}, {4, 5, 6}]
{1, 2, 3}
{'DSAI', 'LV', 'vk'}
{4, 5, 6}
```

```
tups = [(11,13,15),("LV","DSAI")]

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

```
[(11, 13, 15), ('LV', 'DSAI')]
(11, 13, 15)
('LV', 'DSAI')
```

```
dicts = [{1:"LV",2:"DSAI"},{"Company":"LatentView","Loc":"Taramani"}]
print(dicts)

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

```
[{1: 'LV', 2: 'DSAI'}, {'Company': 'LatentView', 'Loc': 'Taramani'}]
{1: 'LV', 2: 'DSAI'}
{'Company': 'LatentView', 'Loc': 'Taramani'}
```

```
sets_tups = [{1,2,3},{"LV","DSAI"},(2.3,4.5,6.7,("x",'y','z'))]
print(sets_tups)

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

```
[{1, 2, 3}, {'DSAI', 'LV'}, (2.3, 4.5, 6.7, ('x', 'y', 'z'))]
{1, 2, 3}
{'DSAI', 'LV'}
(2.3, 4.5, 6.7, ('x', 'y', 'z'))
```

```
dicts_sets_tups = [{1:"LV",2:"DSAI"},{5,6,6,7},('x','y','z')]

print(dicts_sets_tups)

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

```
[{1: 'LV', 2: 'DSAI'}, {5, 6, 7}, ('x', 'y', 'z')]
{1: 'LV', 2: 'DSAI'}
{5, 6, 7}
('x', 'y', 'z')
```

```
#18
```

List: A list is an ordered collection of items. --Lists are mutable. -- accessed by their index. EX: list_1 = [1, 2, 3, 4, 5]

Set: A set is an unordered collection of unique items. -- do not allow duplicate elements. -- Sets are mutable -- defined using curly braces { } or set() function. Ex: set1 = {1, 2, 3, 4, 5}

Tuple: A tuple is an ordered collection of items. -- Tuples are immutable. -- Elements are accessed by their index. -- are defined using parentheses ( ). Ex: tuple = (1, 2, 3, 4, 5)

Dictionary: A dictionary is an unordered collection of key-value pairs. -- Each key in a dictionary must be unique. -- Dictionaries are mutable. -- Elements in a dictionary are accessed by their keys rather than their index. -- are defined using curly braces { }, with key-value pairs separated by colons : Ex: my_dict = {1:"LV, 2: "DSAI"}

```
#19
```

Lists: Use lists when you need an ordered collection of items and you may need to modify the contents -- Lists are suitable for situations where duplicates are allowed and maintaining the order of elements is important. -- Commonly used for storing and manipulating sequential data, such as user input, sensor readings, or file contents.

Sets: when you need an unordered collection of unique items. Sets are ideal for checking membership or removing duplicates from a collection of items. If you need to perform set operations like union, intersection, or difference, sets are the appropriate choice. Suitable for tasks like finding unique elements in a list, checking for common elements between collections, or eliminating duplicates.

Tuples: when you want an immutable ordered collection of elements. -- Tuples are handy for representing fixed collections of items, such as coordinates, constants, or record-like structures where the elements have a specific meaning. -- They are often used for returning multiple values from a function or passing data to functions where you want to ensure it remains unchanged.

Dictionaries Use dictionaries when you need to associate keys with values and require fast lookups based on the keys. -- Dictionaries are suitable for scenarios where you have a mapping between unique identifiers (keys) and associated data (values). -- If you need to quickly retrieve or update values based on some key, dictionaries offer efficient performance. Commonly used for tasks like storing configurations, representing structured data, or implementing key-value caches.

```
x = 5
x_float = float(x)
print("Integer to Float:", x_float)


y = 3.14
y_int = int(y)
print("Float to Integer:", y_int)




z = 10
z_str = str(z)
print("Integer to String:", z_str)


s_list = [1, 2, 3, 4, 5]
s_tuple = tuple(s_list)
print("List to Tuple:", s_tuple)


s_tuple = (1, 2, 3, 4, 5)
s_list = list(s_tuple)
print("Tuple to List:", s_list)


x = 65
char_x = chr(x)
print("Integer to Character:", char_x)

d_list = [('a', 1), ('b', 2), ('c', 3)]
d_dict = dict(d_list)
print("List of Tuples to Dictionary:", d_dict)
```