1- PYTHON VARIABLE

In [ ]:

In [1]:
```
import sys
sys.version
```

Out[1]:
```
'3.13.9 | packaged by Anaconda, Inc. | (main, Oct 21 2025, 19:09:58) [MSC v.192
9 64 bit (AMD64)]'
```

In [2]:
```
v = 5
v
```

Out[2]: 5

5 = v

In [3]:
```
v2 = 10
v2
```

Out[3]: 10

In [4]:
```
nit = 20
NIT
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[4], line 2
      1 nit = 20
----> 2 NIT

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

In [ ]:
```
nit
```

In [ ]:
```
v@ = 60
v@
```

In [ ]:
```
v_ = 60
v_
```

In [ ]:
```
nit@ = 70
nit@
```

In [ ]:
```
nit_ = 70
nit_
```

In [ ]:
```
import keyword
keyword.kwlist
```

In [ ]:
```
if = 75
if
```

In [ ]:
```
len(keyword.kwlist)
```

PYTHON VARIABLE DONE

In [ ]:
```
x = 10
x
```

In [ ]:
```
id(x)
```

In [ ]:
```
y = 15
id(y)
```

In [ ]:
```
z = 20
id(z)
```

In [ ]:
```
i = 5
i
```

In [ ]:
```
type(i)
```

In [ ]:
```
f = 3.4
f
```

In [ ]:
```
type(f)
```

In [ ]:
```
gold = 127000.67
gold
```

In [ ]:
```
petrol = 110.34
petrol
```

In [ ]:
```
2e0
```

In [ ]:
```
2E1
```

In [ ]:
```
3E3
```

In [ ]:
```
4a4  # float only e leter is allowed thats it
```

In [ ]:
```
# INT FLOAT IS COMPLETED
```

In [ ]:
```
b = true
```

In [ ]:
```
b = True
b
```

In [ ]:
```
b1 = False
b1
```

In [ ]:
```
b2 = None
b2
```

In [ ]:
```
True + True
```

In [ ]:
```
True - False
```

In [ ]:
```
False - True
```

In [ ]:
```
True * True - False
```

In [ ]:
```
False / True
```

In [ ]:
```
False // True
```

In [ ]:
```
True + True
True * False
True - True + True
```

In [ ]:
```
print(True + True)
print(True * False )
print(True - True)
```

In [ ]:
```
# COMPLEX DATA TYPES

c = 10 + 20j
c
```

In [ ]:
```
type(c)
```

In [ ]:
```
c.real
```

c.image

In [ ]:
```
d = 20 + 30j
d
```

In [ ]:
```
s = 'welcom to nit'
s
```

In [ ]:
```
type(s)
```

In [ ]:
```
s1 = "welcom to nit"
s1
```

In [ ]:
```
s2 = '''welcom to nit'''
s2
```

string indexing

In [ ]:
```
s = 'welcome'
s
```

In [ ]:
```
len(s)
```

In [ ]:
```
#  package (collection of module )
#  module (collection of fun )
#  function (2 types of fun ){1 inbuil fun - print(), id(), type} {2 user define
```

In [ ]:
```
# python index beguns with o




name = "Python"
print(name[0])  # P
print(name[1])  # y
print(name[2])  # t
print(name[3])  # h
print(name[4])  # o
print(name[5])  # n
```

In [ ]:
```
type(s)
```

In [ ]:
```
print(s[0])
print(s[1])
print(s[2])
print(s[3])
print(s[4])
print(s[5])
```

In [ ]:
```
print(s[-1])
print(s[-2])
print(s[-3])
print(s[3])
print(s[4])
print(s[5])
```

slicing

In [ ]:
```
##  1:5 == print element from 1sst index to to 5th(n-1 formula)
#left indexalways start from 0
# right starts from n-1 formula
```

In [ ]:
```
index = 'welcome'
index
```

In [ ]:
```
index[1:5]
```

In [ ]:
```
index[:]
```

In [ ]:
```
index[4:10]
```

In [ ]:
```
index[:10]
```

In [ ]:
```
index[10]
```

In [ ]:
```
index[0:6:1]
```

In [ ]:
```
index[5:7]
```

In [ ]:
```
#BACKWORD IDEXING
index[-5]
```

In [ ]:
```
index[1:-2]
```

In [ ]:
```
i5 = 'nareshit'
i5
```

In [ ]:
```
i5[0:8]
```

In [ ]:
```
i5[0:7:3]
```

In [ ]:
```
i5[2:-1]
```

In [ ]:
```
i5[::1]
```

In [ ]:
```
i5[::2]
```

In [ ]:
```
i5[::3]
```

In [ ]:
```
i5[::-1]
```

In [ ]:
```
i5[::-2]
```

In [ ]:
```
i5[::-3]
```

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```
s[:]
```

In [ ]:
```
s[:5]
```

In [ ]:
```
s[1:3]
```

In [ ]:
```
s1 = 'MANISHCHOUDHARI'
s1
```

In [ ]:
```
len(s1)
```

In [ ]:
```
s1[:]
```

In [ ]:
```
s1[6:]
```

In [ ]:
```
s1[1:15]
```

In [ ]:
```
s1[0:15:2]
```

In [ ]:
```
s1[1:17:4]
```

In [ ]:
```
s[0:8:4]
```

In [ ]:
```
s[-2]
```

In [ ]:
```
s[-5:-1]
```

In [ ]:
```
s[-5:-1]
```

In [ ]:
```
s2 = 'devilchoudhariinthehouse'
s2
```

In [ ]:
```
s2[::1] # advanced slicing
```

In [ ]:
```
s2[::2]
```

In [ ]:
```
s2[::3]
```

In [ ]:
```
s2[::-1]
```

In [ ]:
```
s2[::-2]
```

In [ ]:
```
s2[::-5]
```

In [ ]:
```
s2[::-9]
```

In [ ]:
```

```

TYPE CASTING

In [ ]:
```
int(2.3)  #float to int
```

In [ ]:
```
int(True) #bool to int
```

In [ ]:
```
int(False)
```

In [ ]:
```
int('10')
```

In [ ]:
```
# all other datatype to float

float(10)
```

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

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

In [ ]:
```
# other datatype to complex datatype
```

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

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

In [ ]:
```
complex(9.7)
```

In [ ]:
```
complex(9.7,  3.4)
```

In [ ]:
```
complex(1,  2.3)
```

In [ ]:
```
complex(False, True)
```

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

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

In [ ]:
```
bool( 3.4)
```

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

In [ ]:

In [ ]:

In [ ]:

DATA STRUCTURE

In [ ]:
```
#dataype -- user define only one value { int , float , string , boolean}
#datastructur -- user declare more than one value {list, tuple, set ,dictionary
#array -- 2d matrix

#tensor - collection of arrays or matrix / libreary called pytorch()
#
#
```

In [ ]:
```
''' DATA 1 structured data -- supervised learning= labeled data , csv , excel fi
          used by (bank , insurence, healthcare)

 2 unstructured data -- unsupervised learning=unllabled data ,image , audio, vid
   used by (netflix , amazon, ipl live , fashion )

machine learning == machine will learn historical structure data
```

In [ ]:
```
a = 5
b = 5.5
c = True
d = 'nit'
print(a,b,d)
```

LIST

In [ ]:
```
l = []
l
## list is dedined with squarer bracket []
```

In [ ]:
```
type(l)
```

In [ ]:
```
print(type(l), len(l))
```

In [ ]:
```
## in python for caslling function we only use ' . and tab'

l1 = [10, 20, 30, 40]
l1
```

In [ ]:
```
len(l1)
```

In [ ]:
```
type(l1)
```

In [ ]:
```
print(l)
print(l1)
```

In [ ]:
```
l == l1
```

In [ ]:
```
l != l1
```

In [ ]:
```
l.append(100)
l.append(12.5)
l.append(True)
l.append(1+2j)
l.append([1,2,3,])   # list in a list is called neasted list

''' >  list []
    >  list is growable (mutable)
    >  append() will allowed -- user to add thw element or value to at the end o
    >  duplication is allowed in list
    >  multiple datatype  you can declare inside the list
    > * list,append takes only one argument
    >
    >

    '''
```

In [ ]:
```
l
```

In [ ]:
```
 l.append(100)
```

In [ ]:
```
l
```

In [ ]:
```
print(l)
print(l1)
```

In [ ]:
```
print(len(l))
print(len(l1))
```

In [ ]:
```
l.count(100)
```

In [ ]:
```
l.count(12.5)
```

In [ ]:
```
l.count(1000)
```

In [ ]:
```
l2 =
```

In [ ]:
```
l2 = l1.copy()
l2
```

In [ ]:
```
l2
```

In [ ]:
```
print(l)
print(l1)
print(l2)
```

In [ ]:
```
l1.append(50)
```

In [ ]:
```
l1
```

In [ ]:
```
l2
```

In [ ]:
```
l2.append(50)
```

In [ ]:
```
l2
```

In [ ]:
```
l2.clear()
```

In [ ]:
```
l2
```

In [ ]:
```
print(l)
print(l1)
print(l2)
```

In [ ]:
```
l2 = l1.copy()
l2
```

In [ ]:
```
l.extend(l1) # this function used to extend an list
```

In [ ]:
```
l
```

In [ ]:
```
l.index(10) # return index value
```

In [ ]:
```
l.index(100)
```

In [ ]:
```
l1.insert(1, 15)  #
```

In [ ]:
```
l1
```

In [ ]:
```
l1.insert(3 , 25)
l1
```

In [ ]:
```
l.pop() # by default last element will remove value first then it will print the
```

In [ ]:
```
l
```

In [ ]:
```
l.pop()
```

In [ ]:
```
l.pop()
```

In [ ]:
```
l.pop(3)  # it will remove 3rd index position value
```

In [ ]:
```
l
```

In [ ]:
```
l.remove(True)                    #
```

In [ ]:
```
l
```

In [ ]:
```

```

In [ ]:
```
print(l)
print(l1)
print(l2)
```

In [ ]:
```
l
```

In [ ]:
```
l.remove([1, 2, 3])
```

In [ ]:
```
l
```

In [ ]:
```
l.sort()   # Sort the list in ascending order and return None
```

In [ ]:
```
l
```

In [ ]:
```
l3 = ['a', 2.3, 1+2j, True]
l3
```

In [ ]:
```
l3.sort() # sirt is applicable only if there is same type of data type , will no
```

In [ ]:
```
# parameter tunning -- system given bydefault parameter
# hyper parameter  tunning -- user change the system parameter
```

In [ ]:
```
l1.sort(revers=True)
```

In [ ]:
```
for i in l:
    print(i)
```

In [ ]:
```
for i in enumerate(l):          # enumerate == The enumerate object yields pairs
                                #                 defaults to zero) and a value y
    print(i)
```

In [ ]:
```
for i in l1:print(i)
```

In [ ]:
```

```

In [ ]:
```
all(l)
```

In [ ]:
```
any(l)
```

In [ ]:
```
l.append(0)
```

In [ ]:
```
l
```

In [ ]:
```
all(l)
```

In [ ]:
```
any(l)
```

In [ ]:
```
# list slicing [:]== print all the element or all value
              # [3:]== print the value from third index
               # [:3]== print the value from (n-1 ) 3-1 till 2nd index
                #[1:10:3]
                #[::2]
                #[::3]
                #[::-1]
                #[::-2]
```

In [ ]:
```
l1
```

In [ ]:
```
l1[:]
```

In [ ]:
```
l1[3:]
```

In [ ]:
```
l1[:3]
```

In [ ]:
```
l1[1:5:3]
```

In [ ]:
```
l1[::2]
```

In [ ]:
```
l1[::3]
```

In [ ]:
```
l1[::-1]
```

In [ ]:
```
l1[::-2]
```

In [ ]:
```
l1[20]
```

In [ ]:
```
l1
```

In [ ]:
```
len(l1)
```

In [ ]:
```
l1[:5]
```

In [ ]:
```
l1[2:6]
```

In [ ]:
```
l1
```

In [ ]:
```
l1[1:6:3]
```

In [ ]:
```
l1
```

In [6]:
```
l1
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[6], line 1
----> 1 l1

NameError: name 'l1' is not defined
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

old practice

In [ ]:
```
l  = []
l
```

In [ ]:
```
type(l)
```

In [ ]:
```
len(l)
```

In [ ]:
```
l.append(10)
```

In [ ]:
```
l
```

In [ ]:
```
l.append(20)
l.append(30)
l.append(50)
```

In [ ]:
```
l
```

In [ ]:
```

```

In [ ]:
```
l = []
```

In [ ]:
```
l.append(10)
l.append('nit')
l.append(1+2j)
l.append(2.3)
l.append(True )
```

In [ ]:
```
l
```

In [ ]:
```
l1 = [10, 20, 30, 40, 50, [1,2,3], 'hello', False, 2+3j, 3.4]
l1
```

In [ ]:
```
print(l)
print(l1)
```

In [ ]:
```
print(len(l))
print(len(l1))
```

In [ ]:
```
print(type(l))
print(type(l1))
```

In [ ]:
```
l
```

In [ ]:
```
l[0] = 100
```

In [ ]:
```
l
```

In [ ]:
```
l2 = []
l2
```

In [ ]:
```
l3 = l1.copy()
l3
```

In [ ]:
```
l1
```

In [ ]:
```
l3.remove([1,2,3])
```

In [ ]:
```
l3
```

In [ ]:
```
l1
```

In [ ]:
```
print(l)
print(l1)
print(l2)
print(l3)
```

In [ ]:
```
l2.extend(l)
```

In [ ]:
```
l2
```

In [ ]:
```
l2.extend(l1)
```

In [ ]:
```
l2
```

In [ ]:
```
l2.extend(l1)
```

In [ ]:
```
l2
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

PYTHON OPERATOR

In [ ]:
```
# OPERATOR

# ARITHMATIC OPERATOR +, -,   *, /, //
```

```
x = 2
x
```

In [ ]:
```
x = x + 2
x
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```
#ASSIGNMENT OPERATOR
```

In [ ]:
```
# unary operator
n = 7
n
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

print()

In [ ]:
```
num1 = 20
num2 = 30
add=num1+num2

print('the addition of ',num1, '&' ,num2, 'is=' ,add)
```

In [ ]:
```
num1 = 20
num2 = 30
num3 = 23
add=num1+num2+num3

print('the addition of {} and {}  and {} is= {}'.format(num1,num2,num3,add))
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

ADVANCEA string

In [ ]:
```
# string we cant multiply but can add it
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```
round(15.2)
```

In [ ]:
```
round(15.8)
```

In [ ]:
```
help()
```

In [ ]:
```
'doesn/t'
```

In [ ]:
```
"doesnt"
```

In [ ]:
```
s = 'first line./nsecond line.'
s
```

In [ ]:
```
3*'un' + 'ium'
```

In [ ]: