PYTHON PROGRAMMING LANGUAGE

In [2]:
```
'WELCOME TO PYTHON'
```

Out[2]:
```
'WELCOME TO PYTHON'
```

In [3]:
```
20
```

Out[3]:
```
20
```

In [4]:
```
10+10-(10*3)-3
```

Out[4]:
```
-13
```

In [5]:
```
10+10-10*3-3
```

Out[5]:
```
-13
```

numbers are called as operand

+,-,* are operators

Arithmetic Operator

1. ARITHMETIC OPERATOR (_,-,,/,%,%%,*,^)
2. ASSIGNMENT OPERATOR (=)
3. RELATIONAL OPERATOR
4. LOGICAL OPERATOR
5. UNARY OPERATOR

In [8]:
```
10**3 #POWER
```

Out[8]:
```
1000
```

In [7]:
```
10***2
```

```
  Cell In[7], line 1
    10***2
        ^
SyntaxError: invalid syntax
```

In [9]:
```
10/5  #float division
```

Out[9]:
```
2.0
```

In [10]:
```
10//5  # int division
```

Out[10]:
```
2
```

In [11]:
```
10%5  #moduls division
```

Out[11]: 0

In [12]:
```
15%6
```

Out[12]: 3

In [14]:
```
15 / 6
```

Out[14]: 2.5

In [15]:
```
2^2
```

Out[15]: 0

In [16]:
```
5^2
```

Out[16]: 7

Assignment Operator

In [28]:
```
x=18
x
# x is called variable or object or identifier
```

Out[28]: 18

In [29]:
```
x+=2
x
```

Out[29]: 20

In [30]:
```
x+=2
x
```

Out[30]: 22

In [31]:
```
x+=2
x
```

Out[31]: 24

In [32]:
```
x-=4
x
```

Out[32]: 20

In [33]:
```
x-=4
x
```

Out[33]: 16

In [34]:
```
x+=2
x
```

Out[34]: 18

In [35]:
```
x
```

Out[35]: 18

In [36]:
```
x*=2
x
```

Out[36]: 36

In [37]:
```
x/=2
x
```

Out[37]: 18.0

In [39]:
```
x=x//2
x
```

Out[39]: 4.0

Unary operator

In [40]:
```
n=8
n
```

Out[40]: 8

In [42]:
```
m=-n
m
```

Out[42]: -8

In [43]:
```
nareshit
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[43], line 1
----> 1 nareshit

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

In [45]:
```
"narestit" # string always in between " ",' ', ''' '''
```

Out[45]: 'narestit'

In [46]:
```
len('nareshit')
```

Out[46]: 8

In [47]:
```
a="Rajesh"
```

In [48]:
```
a
```

Out[48]: 'Rajesh'

In [50]:
```
len(a) # returns the length of string
```

Out[50]: 6

In [51]:
```
a[4]
```

Out[51]: 's'

In [52]:
```
a
```

Out[52]: 'Rajesh'

In [53]:
```
for i in a:
    print(i)
```

```
R
a
j
e
s
h
```

In [54]:
```
a[0:4]  # slicing of a string
```

Out[54]: 'Raje'

In [57]:
```
a[::-1]  # printing of a string in reverse order
```

Out[57]: 'hsejaR'

python variable (identifier or object)

In [58]:
```
va=5
va
```

Out[58]: 5

In [60]:
```
id(va)  c
```

Out[60]: 140706135943736

In [62]:
```
1nit=19
1nit    # variable never starts with digits but it can ends with digit
```

```
  Cell In[62], line 1
    1nit=19
    ^
SyntaxError: invalid decimal literal
```

In [63]:
```
nit3=3
```

In [64]:
```
nit3
```

Out[64]: 3

In [67]:
```
nit3=46   # python is a case sensitive while calling the variables and so on.
NIT3
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[67], line 2
      1 nit3=46   # python is a case sensitive while calling the variables and so
on.
----> 2 NIT3

NameError: name 'NIT3' is not defined
```

In [68]:
```
nit3
```

Out[68]: 46

In [69]:
```
v$=90   # special character is not allowed except undrscore(_)
v$
```

```
  Cell In[69], line 1
    v$=90
     ^
SyntaxError: invalid syntax
```

In [70]:
```
v_=89
v_
```

Out[70]: 89

key words or reserved words never be delared as variables

python key words (35)

In [72]:
```
import keyword
keyword.kwlist
```

Out[72]:
```
['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']
```

In [74]:
```
def=89
def
```

```
  Cell In[74], line 1
    def=89
       ^
SyntaxError: invalid syntax
```

In [75]:
```
Def=89
Def
```

Out[75]:
```
89
```

In [76]:
```
5a=45
5a
```

```
  Cell In[76], line 1
    5a=45
    ^
SyntaxError: invalid decimal literal
```

In [78]:
```
len(keyword.kwlist)
```

Out[78]: 35

In [81]:
```
a=3
b=9
c=5

print(a)
print(b)
print(c)    # we use print keyword when we need the output more than 1 result at
```

```
3
9
5
```

In [82]:
```
import sys
sys.version
```

Out[82]: '3.12.7 | packaged by Anaconda, Inc. | (main, Oct 4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64)]'

completed variable concept in python

Father of python is Guido van Rossum

python name comes from -- serial complete monty python flying circus

In [83]:
```
a=5
a
```

Out[83]: 5

In [85]:
```
#python is
PVM(python virtual machine)
platform independent (we can execute the code in several system)
Python is dynamic programming language
```

```
  Cell In[85], line 2
    PVM(python virtual machine)
        ^
SyntaxError: invalid syntax. Perhaps you forgot a comma?
```

In [86]:
```
a=34.56
a
```

Out[86]: 34.56

In [87]:
```
type(a)
```

Out[87]: float

libraries SciPy---take care of statistics Sypder--ML NumbPy---matrix we need to convert images to matrix

running the code line by line is interpreter

running the total code is compiler

python variable---> variablename=value

In [88]:
```
a=10 # (a-variable 10-value) 10-integer value
 # type(a)---int
'''types of data
int- value without decimal
float-value with decimal
string-value with "|""|''' '''
bool- True or False
complex--a+bj'''
```

Out[88]:
```
'types of data \nint- value without decimal\nfloat-value with decimal\nstring-v
alue with "|""|\nbool- True or False\ncomplex--a+bj'
```

Python datatypes --->

--integer datatypes --float datatypes --Bool data types --string datatypes -- complex datatypes

In [91]:
```
a=45
print(a)
print(type(a))
```

```
45
<class 'int'>
```

In [93]:
```
print(type(a))  # <class 'int'> is inbuilt class
```

```
<class 'int'>
```

In [94]:
```
petrol=110.56
type(petrol)
```

Out[94]:
```
float
```

In [96]:
```
i1,i2=20,40
print(i1,i2)
print(i2)
print(i1 + i2)
print(i1 - i2)
print(i1 * i2)
print(i1 / i2)
```

```
20 40
40
60
-20
800
0.5
```

string (non-technical--text, technical--string)

In [97]:
```
s = nareshit
s
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[97], line 1
----> 1 s=nareshit
      2 s

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

In [98]:
```
s = 'nareshit'
s
```

Out[98]:
```
'nareshit'
```

In [99]:
```
type(s)
```

Out[99]:
```
str
```

In [100…
```
s1 = "nareshit"
s1
```

Out[100…
```
'nareshit'
```

In [103…
```
s2= '''naresh it technology
      datascience, ai student  -- 6 month i will change your brain'''
s2
```

Out[103…
```
'naresh it technology\n      datascience, ai student  -- 6 month i will change
your brain'
```

string--- for single line--- which ''|"" multiline--'" '''

string-->

assign text to the variable define ''| " "|''' '''(multiline comments

string indexing-->

forward indexing--left to right(begin with 0) backward indexing---right to left(begins with -1)

In [108…
```
print(s[0])
print(s[-1])
print(s[3])
```

```
n
t
e
```

In [109…
```
s[:]
```

Out[109… 'nareshit'

In [110…
```
s[2:5]
```

Out[110… 'res'

21st april

In [112…
```
i=5.5
type(i)
```

Out[112… float

LIST are defined in []

In [113…
```
l=[]
l
```

Out[113… []

In [114…
```
type(l)
```

Out[114… list

In [117…
```
l.append(10)
l
```

Out[117… [10]

In [122…
```
l.append(20)
l.append(30)
l.append(40)
l.append(50)
l.append(60)
```

In [119…
```
l
```

Out[119… [10, 20, 30, 40, 50]

In [120…
```
l.append(70,80,90) # only one argument is valid
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[120], line 1
----> 1 l.append(70,80,90)

TypeError: list.append() takes exactly one argument (3 given)
```

In [123…
```
l
```
Out[123… [10, 20, 30, 40, 50, 20, 30, 40, 50, 60]

In [124…
```
l
```
Out[124… [10, 20, 30, 40, 50, 20, 30, 40, 50, 60]

append() adding element at last only one element can be added at a time list is growable duplicate is allowed

In [125…
```
len(l)
```
Out[125… 10

In [127…
```
l1=l.copy()
```

In [128…
```
l1
```
Out[128… [10, 20, 30, 40, 50, 20, 30, 40, 50, 60]

In [131…
```
l1.append(2.3)
l1.append('nit')
l1.append(1-2j)
l1.append([1,2,3])
```

In [132…
```
print(l)
print(l1) # list inside list=nested list, for inside for is nested for
```
```
[10, 20, 30, 40, 50, 20, 30, 40, 50, 60]
[10, 20, 30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j), [1, 2, 3]]
```

In [134…
```
print(len(l))
print(len(l1)) # in list multiple data types can be defined
```
```
10
14
```

In [135…
```
l==l1
```
Out[135… False

In [137…
```
l
```
Out[137… [10, 20, 30, 40, 50, 20, 30, 40, 50, 60]

In [139…
```
l[:]
```
Out[139… [10, 20, 30, 40, 50, 20, 30, 40, 50, 60]

In [140…
```
l[0]
```
Out[140… 10

In [141…
```
l[0]=100 # this is called mutable(changeable)
```

In [142…
```
l
```
Out[142… [100, 20, 30, 40, 50, 20, 30, 40, 50, 60]

In [143…
```
l[-1]=200
l
```
Out[143… [100, 20, 30, 40, 50, 20, 30, 40, 50, 200]

In [144…
```
l[3:]
```
Out[144… [40, 50, 20, 30, 40, 50, 200]

In [146…
```
l[10:]
```
Out[146… []

In [148…
```
l[:10]
```
Out[148… [100, 20, 30, 40, 50, 20, 30, 40, 50, 200]

In [150…
```
l[-1]
```
Out[150… 200

In [151…
```
l2.clear()
```

In [152…
```
l2
```
Out[152… []

In [153…
```
del l2
```

In [154…
```
l2
```
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[154], line 1
----> 1 l2

NameError: name 'l2' is not defined
```

In [155…
```
l1
```
Out[155… [10, 20, 30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j), [1, 2, 3]]

In [157…
```
l1[0:12:5]
```
Out[157… [10, 20, 2.3]

In [158…
```
l1
```
Out[158…
```
[10, 20, 30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j), [1, 2, 3]]
```

In [159…
```
l1[0:11:3]  # this is called step indexing
```
Out[159…
```
[10, 40, 30, 60]
```

In [160…
```
l1.index('nit')
```
Out[160…
```
11
```

In [162…
```
l1.index(2.3)
```
Out[162…
```
10
```

list is mutable(we can modify any value) list[]

called list inbuild function using .tab append()--add the element at last, duplicate is allowed, indexing and slicing is allowed, copy list, clear the elements from the list, del keyword list is growable

functions covered (append,clear,index)

22nd

In [2]:
```
print(l)
print(l1)
```
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[2], line 1
----> 1 print(l)
      2 print(l1)

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

In [3]:
```
print(l1)
```
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[3], line 1
----> 1 print(l1)

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

In [4]:
```
l1=[10, 20, 30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j), [1, 2, 3]]
l=[10, 20, 30, 40, 50, 20, 30, 40, 50, 60]
```

In [12]:
```
print(l1[11][0])
print(l1[11][1])
```

```
print(l1[11][2])
```

```
n
i
t
```

In [7]:
```
l1.count(10)
```

Out[7]: 1

In [8]:
```
l1
```

Out[8]: [10, 20, 30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j), [1, 2, 3]]

In [10]:
```
l1.remove([1,2,3]) #removing the element from the list(it removes the first occu
```

In [11]:
```
l1
```

Out[11]: [10, 20, 30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j)]

In [13]:
```
l1.remove(10)
```

In [14]:
```
l1
```

Out[14]: [20, 30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j)]

In [15]:
```
l1.remove(20)
l1
```

Out[15]: [30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j)]

In [16]:
```
l
```

Out[16]: [10, 20, 30, 40, 50, 20, 30, 40, 50, 60]

In [18]:
```
l.append(25)
l
```

Out[18]: [10, 20, 30, 40, 50, 20, 30, 40, 50, 60, 25, 25]

In [19]:
```
l.insert(2,29)  # at insert function by defaut user need to pass 2 arguments, 1s
l
```

Out[19]: [10, 20, 29, 30, 40, 50, 20, 30, 40, 50, 60, 25, 25]

In [21]:
```
l.pop() # pop will remove the last element in the list
```

Out[21]: 25

In [22]:
```
l
```

Out[22]: [10, 20, 29, 30, 40, 50, 20, 30, 40, 50, 60, 25]

In [23]:
```
l1
```

Out[23]: [30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit', (1-2j)]

In [24]:
```
l1.pop()
l1
```

Out[24]:
```
[30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit']
```

pop(3) will remove index wise(value of the 3rd index)

pop() by default last value will be removed

remove by element number

In [26]:
```
l1
```

Out[26]:
```
[30, 40, 50, 20, 30, 40, 50, 60, 2.3, 'nit']
```

In [27]:
```
len(l)
```

Out[27]:
```
12
```

In [28]:
```
len(l)
```

Out[28]:
```
12
```

In [29]:
```
l2=l1.extend(l)
```

In [31]:
```
l3=l1.copy()
```

In [33]:
```
l3
```

Out[33]:
```
[30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 2.3,
 'nit',
 10,
 20,
 29,
 30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 25]
```

In [36]:
```
l4=l1.extend(l)
```

In [37]:
```
l4
```

In [38]:
```
l3
```

Out[38]:
```
[30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 2.3,
 'nit',
 10,
 20,
 29,
 30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 25]
```

In [39]:
```
l3.extend(l1)
```

In [40]:
```
l3
```

Out[40]:
```
[30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 2.3,
 'nit',
 10,
 20,
 29,
 30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 25,
 30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 2.3,
 'nit',
 10,
 20,
 29,
 30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 25,
 10,
 20,
 29,
 30,
 40,
 50,
 20,
 30,
 40,
 50,
 60,
 25]
```

In [41]:
```
l7=[1,2]
l7
```

Out[41]: [1, 2]

In [ ]:
```
l1.sort
```