In [9]:
```
import sys
sys.version
```

Out[9]:
```
'3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 6
4 bit (AMD64)]'
```

python variables=identifier=object

In [12]:
```
a=8
a
```

Out[12]:
```
8
```

In [14]:
```
import keyword
keyword.kwlist
```

Out[14]:
```
['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 [16]:
```
len(keyword.kwlist)
```

Out[16]: 35

14/10/25 VARIABLE DECLARATION

In [5]:
```
var="hello"
VAR
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[5], line 2
      1 var="hello"
----> 2 VAR

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

In [3]:
```
#case sensitive
var1="hello"
var1
```

Out[3]: 'hello'

In [9]:
```
VAR2="anjali"
var2
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[9], line 2
      1 VAR2="anjali"
----> 2 var2

NameError: name 'var2' is not defined
```

In [11]:
```
VAR3="ANJALI"
VAR3
```

Out[11]: 'ANJALI'

In [13]:
```
#we should not use special characters while declaring a variable
v@r1="python"
v@r1
```

```
  Cell In[13], line 2
    v@r1="python"
    ^
SyntaxError: cannot assign to expression here. Maybe you meant '==' instead of '='?
```

In [17]:
```
a$1="hello"
a$1
```

```
  Cell In[17], line 1
    a$1="hello"
     ^
SyntaxError: invalid syntax
```

In [19]:
```
a&bc="world"
a&bc
```

```
  Cell In[19], line 1
    a&bc="world"
    ^
SyntaxError: cannot assign to expression here. Maybe you meant '==' instead of '='?
```

In [21]:
```
a.="anjali"
a.
```

```
  Cell In[21], line 1
    a.="anjali"
      ^
SyntaxError: invalid syntax
```

In [ ]:
```
#we use underscore while declaring a variable
```

In [40]:
```
variable_name="apple"
variable_name
```

Out[40]:
```
'apple'
```

In [42]:
```
var_123=1000
var_123
```

Out[42]:
```
1000
```

In [25]:
```
#we should not start with numbers but end with numbers while declaring a variable
123var="anjali"
123var
```

```
  Cell In[25], line 2
    123var="anjali"
      ^
SyntaxError: invalid decimal literal
```

In [27]:
```
10var="python"
10var
```

```
  Cell In[27], line 1
    10var="python"
     ^
SyntaxError: invalid decimal literal
```

In [29]:
```
0var="anjali"
0var
```

```
  Cell In[29], line 1
    0var="anjali"
    ^
SyntaxError: invalid decimal literal
```

In [31]:
```
var11="anjali"
var11
```

Out[31]: 'anjali'

In [34]:
```
var22=100
var22
```

Out[34]: 100

In [36]:
```
var23iable=100
var23iable
```

Out[36]: 100

In [38]:
```
v1r=5000
v1r
```

Out[38]: 5000

In [44]:
```
#we should not use key word names as a variable name
False=0
False
```

```
  Cell In[44], line 2
    False=0
    ^
SyntaxError: cannot assign to False
```

In [46]:
```
false=0
false
```

Out[46]: 0

In [48]:
```
True=1
True
```

```
  Cell In[48], line 1
    True=1
    ^
SyntaxError: cannot assign to True
```

In [50]:
```
true=1
true
```

Out[50]: 1

In [52]:
```
None="None"
None
```

```
  Cell In[52], line 1
    None="None"
    ^
SyntaxError: cannot assign to None
```

In [54]:
```
none="None"
none
```

Out[54]: 'None'

In [56]:
```
while="it is a loop"
while
```

```
  Cell In[56], line 1
    while="it is a loop"
          ^
SyntaxError: invalid syntax
```

In [60]:
```
if="condition"
if
```

```
  Cell In[60], line 1
    if="condition"
       ^
SyntaxError: invalid syntax
```

In [62]:
```
and="python"
and
```

```
  Cell In[62], line 1
    and="python"
    ^
SyntaxError: invalid syntax
```

In [64]:
```
return="python"
return
```

```
  Cell In[64], line 1
    return="python"
           ^
SyntaxError: invalid syntax
```

In [66]:
```
continue=100
continue
```

```
  Cell In[66], line 1
    continue=100
             ^
SyntaxError: invalid syntax
```

In [68]:
```
#we should not give space instead of space we use underscore
variable name="anjali"
variable name
```

```
  Cell In[68], line 1
    variable name="anjali"
              ^
SyntaxError: invalid syntax
```

In [70]:
```
variable_name="anjali"
variable_name
```

Out[70]: 'anjali'

15/10/25 DATA TYPES

In [77]:
```
#Integer datatype
num=100
num
```

Out[77]: 100

In [79]:
```
type(num)
```

Out[79]: int

In [87]:
```
#Float datatype
num1=23.65
num1
```

Out[87]: 23.65

In [125…
```
a=10
b=20.3
c=a+b
c
```

Out[125… 30.3

In [127…
```
type(c)
```

Out[127… float

In [89]:
```
type(num1)
```

Out[89]: float

In [101…
```
num3=1e0
num3
```

Out[101… 1.0

In [103…
```
type(num3)
```

Out[103… float

In [105…
```
num4=2e0
num4
```

Out[105… 2.0

In [107…
```
type(num4)
```

Out[107… float

In [109…
```
num5=2e10
num5
```

Out[109… 20000000000.0

In [111…
```
#boolean datatype
True
```

Out[111… True

In [113…
```
False
```

Out[113… False

In [115…
```
True+True
```

Out[115… 2

In [117…
```
True+False
```

Out[117… 1

In [119…
```
False+False
```

Out[119… 0

In [121…
```
False+True
```

Out[121… 1

In [129…
```
True*False
```

Out[129… 0

In [131…
```
True/False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[131], line 1
----> 1 True/False

ZeroDivisionError: division by zero
```

In [133…
```
False/True
```

Out[133… 0.0

In [139…
```
False//True
```

Out[139… 0

In [135…
```
True%False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[135], line 1
----> 1 True%False

ZeroDivisionError: integer modulo by zero
```

In [137…
```
False%True
```

Out[137…
```
0
```

In [141…
```
#String datatype
str=hello
str
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[141], line 2
      1 #String
----> 2 str=hello
      3 str

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

In [143…
```
str1='hello'
str1
```

Out[143…
```
'hello'
```

In [147…
```
str2="hello python"
str2
```

Out[147…
```
'hello python'
```

In [145…
```
str3='''Python is a popular programming language. It was created by Guido van Rossu

It is used for:

web development (server-side),
software development,
mathematics,
system scripting.'''
str3
```

Out[145…
```
'Python is a popular programming language. It was created by Guido van Rossum, and
released in 1991.\n\nIt is used for:\n\nweb development (server-side),\nsoftware d
evelopment,\nmathematics,\nsystem scripting.'
```

In [149…
```
#complex datatype
a=1+2j
a
```

Out[149…
```
(1+2j)
```

In [165…
```
type(a)
```
Out[165… complex

In [153…
```
a.real
```
Out[153… 1.0

In [155…
```
a.imag
```
Out[155… 2.0

In [159…
```
b=10+20j
c=20+30j
d=b+c
d
```
Out[159… (30+50j)

In [167…
```
type(d)
```
Out[167… complex

In [161…
```
e=b-c
e
```
Out[161… (-10-10j)

In [169…
```
type(e)
```
Out[169… complex

In [163…
```
e1=c-b
e1
```
Out[163… (10+10j)

In [171…
```
type(e1)
```
Out[171… complex

In [173…
```
f=b*c
f
```
Out[173… (-400+700j)

16/10/25
TYPE CASTING

ALL OTHER DATATYPES TO INTEGER

In [4]:
```
#float to integer
int(2.45)
```

Out[4]: 2

In [8]:
```
#boolean to integer
int(True)
```

Out[8]: 1

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

Out[10]: 0

In [12]:
```
int(0)
```

Out[12]: 0

In [14]:
```
int(1)
```

Out[14]: 1

In [16]:
```
#text string to integer(not possible)
int("anjali")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[16], line 1
----> 1 int("anjali")

ValueError: invalid literal for int() with base 10: 'anjali'
```

In [18]:
```
#number string to integer
int("10")
```

Out[18]: 10

In [20]:
```
#complex to integer
int(10+20j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[20], line 1
----> 1 int(10+20j)

TypeError: int() argument must be a string, a bytes-like object or a real number, no
t 'complex'
```

In [22]:
```
int(10+20)
```

Out[22]: 30

ALL OTHER DATATYPES TO FLOAT

In [24]:
```
#integer to float
float(10)
```

Out[24]: 10.0

In [26]:
```
float(200.00)
```

Out[26]: 200.0

In [28]:
```
float(20.435)
```

Out[28]: 20.435

In [30]:
```
#boolean to float
float(True)
```

Out[30]: 1.0

In [32]:
```
float(False)
```

Out[32]: 0.0

In [34]:
```
# text string to float(not possible)
float("anjali")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[34], line 1
----> 1 float("anjali")

ValueError: could not convert string to float: 'anjali'
```

In [36]:
```
#number string to float
float("10")
```

Out[36]: 10.0

In [38]:
```
#complex to float(not possible)
float(10+20j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[38], line 1
----> 1 float(10+20j)

TypeError: float() argument must be a string or a real number, not 'complex'
```

In [40]:
```
float(10+20)
```

Out[40]: 30.0

In [42]:
```
float(10+20+30+40)
```

Out[42]: 100.0

ALL OTHER DATATYPES TO BOOLEAN

In [44]:
```
#integer to bool
bool(10)
```

Out[44]: True

In [46]:
```
bool()
```

Out[46]: False

In [48]:
```
bool(0)
```

Out[48]: False

In [50]:
```
bool(12.87)
```

Out[50]: True

In [52]:
```
#string to bool
bool("anjali")
```

Out[52]: True

In [54]:
```
bool("10")
```

Out[54]: True

In [56]:
```
#complex to bool
bool(10+20j)
```

Out[56]: True

In [58]:
```
bool(10+20)
```

Out[58]: True

ALL OTHER DATATYPES TO STRING

In [60]:
```
#integer to string
str(10)
```

Out[60]: '10'

In [62]:
```
str(200)
```

Out[62]: '200'

In [64]:
```
#float to string
```

```
In [64]: str(39.54)
```

```
Out[64]: '39.54'
```

```
In [66]: str(200.000)
```

```
Out[66]: '200.0'
```

```
In [68]: #bool to string
         str(True)
```

```
Out[68]: 'True'
```

```
In [70]: STR(10)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[70], line 1
----> 1 STR(10)

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

```
In [72]: str(10)
```

```
Out[72]: '10'
```

```
In [74]: str("anjali")
```

```
Out[74]: 'anjali'
```

```
In [76]: str(True)
```

```
Out[76]: 'True'
```

```
In [78]: str(False)
```

```
Out[78]: 'False'
```

```
In [80]: #complex to string
         str(10+20j)
```

```
Out[80]: '(10+20j)'
```

```
In [82]: str(10+20)
```

```
Out[82]: '30'
```

ALL OTHER DATATYPES TO COMPLEX

```
In [95]: #integer to complex
         complex(10)
```

Out[95]: (10+0j)

In [97]:
```
complex(10+20)
```

Out[97]: (30+0j)

In [99]:
```
complex(10,20)
```

Out[99]: (10+20j)

In [101…
```
#float to complex
complex(49.43)
```

Out[101… (49.43+0j)

In [103…
```
complex(23.64,87.87)
```

Out[103… (23.64+87.87j)

In [105…
```
#bool to complex
complex(True)
```

Out[105… (1+0j)

In [107…
```
complex(False)
```

Out[107… 0j

In [109…
```
#text string to complex(not possible)
complex("anjali")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[109], line 1
----> 1 complex("anjali")

ValueError: complex() arg is a malformed string
```

In [111…
```
#number string to complex
complex("10")
```

Out[111… (10+0j)

In [113…
```
#should not take 2 string arguments
complex("10","20")
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[113], line 1
----> 1 complex("10","20")

TypeError: complex() can't take second arg if first is a string
```

17/10/25 INDEXING

In [2]:
```
str="anjali"
str[0]
```

Out[2]: 'a'

In [4]:
```
str[1]
```

Out[4]: 'n'

In [6]:
```
str[3]
```

Out[6]: 'a'

In [8]:
```
str[-1]
```

Out[8]: 'i'

In [10]:
```
str[-5]
```

Out[10]: 'n'

In [14]:
```
print(str[-1])
print(str[-2])
print(str[-3])
print(str[-4])
print(str[-5])
print(str[-6])
```

```
i
l
a
j
n
a
```

In [16]:
```
print(str[0])
print(str[1])
print(str[2])
print(str[3])
print(str[4])
print(str[5])
```

```
a
n
j
a
l
i
```

In [18]:
```
str
```

Out[18]: 'anjali'

In [20]:
```
len(str)
```

Out[20]: 6

In [22]:
```
id(str)
```

Out[22]: 2953141096912

SLICING

In [26]:
```
str="anjali"
str[:]
```

Out[26]: 'anjali'

In [28]:
```
str[::]
```

Out[28]: 'anjali'

In [32]:
```
str[0:]
```

Out[32]: 'anjali'

In [34]:
```
str[:3]
```

Out[34]: 'anj'

In [36]:
```
str[4:]
```

Out[36]: 'li'

In [38]:
```
str[::2]
```

Out[38]: 'ajl'

In [40]:
```
str[::-2]
```

Out[40]: 'ian'

In [44]:
```
str
```

Out[44]: 'anjali'

In [48]:
```
str[0:2:2]
```

Out[48]: 'a'

In [50]:
```
str[3:6:-1]
```

Out[50]: ''

In [54]:
```
str[2::2]
```

Out[54]:
```
'jl'
```

In [58]:
```
str[::-1]
```

Out[58]:
```
'ilajna'
```

In [60]:
```
str[::1]
```

Out[60]:
```
'anjali'
```

18/10/25 OPERATORS

ARITHMATIC OPERATORS

In [64]:
```
a=10
b=20
a+b
```

Out[64]:
```
30
```

In [66]:
```
a-b
```

Out[66]:
```
-10
```

In [68]:
```
a*b
```

Out[68]:
```
200
```

In [70]:
```
a/b
```

Out[70]:
```
0.5
```

In [72]:
```
a//b
```

Out[72]:
```
0
```

In [78]:
```
a%b
```

Out[78]:
```
10
```

In [80]:
```
a**b
```

Out[80]:
```
100000000000000000000
```

ASSIGNMENT OPERATORS

In [102…
```
x=10
x+=10
x
```

Out[102… 20

In [104…
```
x-=10
x
```

Out[104… 10

In [106…
```
x*=10
x
```

Out[106… 100

In [114…
```
y=20
y/=10
y
```

Out[114… 2.0

In [118…
```
z=20
z//=10
z
```

Out[118… 2

In [122…
```
a=100
a%=10
a
```

Out[122… 0

In [124…
```
z**=10
z
```

Out[124… 1024

RELATIONAL OPERATORS

In [127…
```
num1=200
num2=300
num1>num2
```

Out[127… False

In [129…
```
num1<num2
```

Out[129… True

In [131…
```
num1>=num2
```

Out[131… False

In [133…
```
num1<=num2
```

Out[133… True

In [135…
```
num1==num2
```

Out[135… False

In [137…
```
num1!=num2
```

Out[137… True

In [139…
```
num1=300
```

In [141…
```
num1==num2
```

Out[141… True

In [143…
```
num1!=num2
```

Out[143… False

In [145…
```
num1>=num2
```

Out[145… True

In [147…
```
num1<=num2
```

Out[147… True

LOGICAL OPERATORS

In [154…
```
x=10
print(x>20 and x<30)
```

False

In [156…
```
print(x>5 and x<20)
```

True

In [158…
```
print(x>20 or x<30)
```

True

In [168…
```
print(x>5 or x<20)
```

True

In [162…
```
print(x>20 or x<5)
```

False

In [172…
```
print(not(x>5 and x<20))
```

False

In [174…
```
print(not(x>20 or x<30))
```

```
False
```

In [178…
```
print(not(x>20 and x<30))
```

```
True
```

UNARY OPERATOR

In [181…
```
x=10
print(-(x))
```

```
-10
```

In [183…
```
print(-x)
```

```
-10
```

In [185…
```
y=100
-y
```

Out[185… -100

In [ ]: