16th-to-28th-oct-work-done

November 1, 2024

python 1st code

[235]:
```
10+5
```

[235]: 15

[236]:
```
10-5
```

[236]: 5

[237]:
```
10*5
```

[237]: 50

[238]:
```
10/5
```

[238]: 2.0

[239]:
```
10//5
```

[239]: 2

[240]:
```
(10+5)-7+6
```

[240]: 14

[241]:
```
5+5*5
```

[241]: 30

[242]:
```
(5+5)*5
```

[242]: 50

[243]:
```
_+3
```

[243]: 53

[244]:
```
import sys
sys.version
```

[244]:
```
'3.12.4 | packaged by Anaconda, Inc. | (main, Jun 18 2024, 15:03:56) [MSC v.1929
64 bit (AMD64)]'
```

[245]:
```
1+1
2+1
3+1
4+1
```

[245]:
```
5
```

[246]:
```
print(1+1)
```

```
2
```

[247]:
```
a = 10
b = 20
c = a+b
print(c)
```

```
30
```

[248]:
```
print(a)
print(b)
print(c)
```

```
10
20
30
```

[249]:
```
100 = d
```

```
  Cell In[249], line 1
    100 = d
    ^
SyntaxError: cannot assign to literal here. Maybe you meant '==' instead of '='
```

[460]:
```
print(10)
print(10,20)
print('python')
print(10,20,'python')
```

```
10
10 20
```

```
python
10 20 python
```

[464]:
```
num1=20
num2=30
add=num1+num2
print(add)
```

```
50
```

[470]:
```
-+3
```

[470]:
```
-3
```

[468]:
```
_+3
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[468], line 1
----> 1 _+3

TypeError: can only concatenate str (not "int") to str
```

[474]:
```
import keyword
keyword.kwlist
```

[474]:
```
['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']
```

[478]:
```
len(keyword.kwlist)
```

[478]: 35

1 22nd Oct

2 Python variable concept = python identifier concept

.syntax of define variable || (variable name = value) || (identifier = value)

[480]:
```
NIT = 15
NIT
```

[480]: 15

[482]:
```
NIT = 20
NIT
```

[482]: 20

[484]:
```
v = 15
v
```

[484]: 15

[486]:
```
print(v)
print(NIT)
```

```
15
20
```

[488]:
```
NIT
```

[488]: 20

[490]:
```
1var = 20
1var
```

```
  Cell In[490], line 1
    1var = 20
    ^
SyntaxError: invalid decimal literal
```

[492]:
```
var1 = 20
var1
```

[492]: 20

[494]:
```
var$ = 56
var$
```

```
  Cell In[494], line 1
    var$ = 56
        ^
SyntaxError: invalid syntax
```

[496]:
```
x_train, x_test = 70,50,40
print(x_train)
print(x_test)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[496], line 1
----> 1 x_train, x_test = 70,50,40
      2 print(x_train)
      3 print(x_test)

ValueError: too many values to unpack (expected 2)
```

[498]:
```
x_train, x_test = 80, 20
print(x_train)
print(x_test)
```

```
80
20
```

[503]:
```
a,b,c,d=10,20,30,40
print(a)
print(b)
print(c)
print(d)
```

```
10
20
30
40
```

[501]:
```
aaaaaaaaaaaaaaaaaaaaa = 78
a
```

[501]: 10

3 PYTHON DATA TYPES

-INT - value without decimal # -FLOAT - Value with decimal # -BOOL - True or False # -STRING - 'nit' or "nit" # -COMPLEX - (a + bj)

[505]:
```
i = 25
i
```

[505]: 25

[507]:
```
type(i)
```

[507]: int

[509]:
```
print(type(i))
```

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

4 23rd Oct Python Datatypes

[520]:
```
i = 30
```

[522]:
```
i1,i2=20,30
```

[524]:
```
i - i2 + i1
```

[524]: 20

[528]:
```
i - (i2+i1)
print(i)
```

```
print(i1)
print(i2)
```

```
30
20
30
```

5 integer datatype completed

[530]:
```
f = 110.23
```

[532]:
```
type(f)
```

[532]:
```
float
```

[ ]:
```
f1, f2, f3 = 2.3, 3.4, 5.1
```

[534]:
```
print(f)
print(f1)
print(f2)
print(f3)
```

```
110.23
2.3
3.4
5.1
```

[ ]:
```
f1, f2, f3 = 2.3,3.4,5.6
print(f1)
print(f2)
print(f3)
```

[536]:
```
1f = 1e0
```

```
  Cell In[536], line 1
    1f = 1e0
    ^
SyntaxError: invalid decimal literal
```

[538]:
```
f1 = 1e0
f1
```

[538]:
```
1.0
```

[540]:
```
f2=2e1
f2
```

[540]: 20.0

[542]:
```
f3 = 3e2
f3
```

[542]: 300.0

[544]:
```
f4 = 3e3
f4
```

[544]: 3000.0

[546]:
```
f5 = 2.4e2
f5
```

[546]: 240.0

[548]:
```
f6 = 2a3
```

```
  Cell In[548], line 1
    f6 = 2a3
         ^
SyntaxError: invalid decimal literal
```

6 Bool or Boolean

[551]:
```
b= true
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[551], line 1
----> 1 b= true

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

[553]:
```
b = True
```

[555]:
```
b1 = false
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
```

```
Cell In[555], line 1
----> 1 b1 = false

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

[557]:
```
b1 = False
```

[559]:
```
b2 = True
print(b)
```

```
True
```

[561]:
```
b = True
b1 = False
print(b)
print(b1)
```

```
True
False
```

[563]:
```
True + False
```

[563]: 1

[565]:
```
True - False
```

[565]: 1

[567]:
```
False - True
```

[567]: -1

[569]:
```
True + True + True + False
```

[569]: 3

[571]:
```
False * True
```

[571]: 0

[573]:
```
True * True
```

[573]: 1

[575]:
```
False / True
```

[575]: 0.0

[577]:
```
True/False
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
Cell In[577], line 1
----> 1 True/False

ZeroDivisionError: division by zero
```

7 complex datatypes

[580]:
```
c = 10 + 20j
c
```

[580]:
```
(10+20j)
```

[582]:
```
c = 1 + 20j
c
```

[582]:
```
(1+20j)
```

[586]:
```
type(c)
```

[586]:
```
complex
```

[588]:
```
import keyword
keyword.kwlist
```

[588]:
```
['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']
```

[590]:
```
if = 45
```

```
  Cell In[590], line 1
    if = 45
       ^
SyntaxError: invalid syntax
```

[592]:
```
p,q,r=20
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[592], line 1
----> 1 p,q,r=20

TypeError: cannot unpack non-iterable int object
```

[594]:
```
p,q,r = 20,20,20
```

[596]:
```
c = 1+20j
c
```

[596]:
```
(1+20j)
```

[598]:
```
c.imag
```

[598]: 20.0

[600]:
```
c.imag
```

[600]: 20.0

[602]:
```
c.real
```

[602]: 1.0

[604]:
```
c1 = 10+20j
c2 = 30 + 40j
c1
```

[604]: (10+20j)

[606]:
```
c1+c2
```

[606]: (40+60j)

[608]:
```
c1 = 10 + 20j
c2 = 30 + 40j
c1+c2
```

[608]: (40+60j)

[612]:
```
print(c1-c2)
```

```
(-20-20j)
```

[614]:
```
import sys
import keyword
import operator
from datetime import datetime
import os
```

[616]:
```
print(keyword.kwlist)
```

```
['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']
```

[618]:
```
len(keyword.kwlist)
```

[618]: 35

[620]:
```
import = 123
```

```
  Cell In[620], line 1
    import = 123
           ^
SyntaxError: invalid syntax
```

[622]:
```
p = 20
q = 20
r = q
p , type(p), hex(id(p))
```

[622]:
```
(20, int, '0x7ffd23933c18')
```

[624]:
```
p,q,r = 20
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[624], line 1
----> 1 p,q,r = 20

TypeError: cannot unpack non-iterable int object
```

[630]:
```
p,q,r = 20,20,20
hex(id(p)), hex(id(q)), hex(id(r))
```

[630]:
```
('0x7ffd23933c18', '0x7ffd23933c18', '0x7ffd23933c18')
```

[628]:
```
p = 20
q = p + 10 #variable overwriting
print(q)
```

```
30
```

[634]:
```
intvar = 10
floatvar = 2.57
strvar = 'Python Language'

print(intvar)
print(floatvar)
print(strvar)
```

```
10
2.57
Python Language
```

[636]:
```
p,q,r = 20
print(p,q,r)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[636], line 1
----> 1 p,q,r = 20
      2 print(p,q,r)

TypeError: cannot unpack non-iterable int object
```

[638]:
```
p1 = p2 = p3 = p4 = 44 #all variables pointing to same value
print(p1,p2,p3,p4)
```

```
44 44 44 44
```

[640]:
```
print(sys.getsizeof(p1))
```

```
28
```

[642]:
```
print(p1,"is Integer?", isinstance(p1,int))
```

```
44 is Integer? True
```

[644]:
```
val2 = 23.4
print(val2)
print(type(val2))
print(sys.getsizeof(val2))
print(val2,"is Float?", isinstance(val2,float))
```

```
23.4
<class 'float'>
24
23.4 is Float? True
```

[646]:
```
sys.getsizeof(complex())
```

[646]:
```
32
```

[648]:
```
sys.getsizeof(int())
```

[648]:
```
28
```

[650]:
```
sys.getsizeof(float())
```

[650]:
```
24
```

[652]:
```
bool1 = True
bool2 = False
```

[654]:
```
sys.getsizeof(bool)
```

[654]: 432

[656]:
```
print(type(bool1))
```

```
<class 'bool'>
```

[658]:
```
isinstance(bool1,bool)
```

[658]: True

[660]:
```
isinstance(bool2,bool)
```

[660]: True

[662]:
```
bool(0)
```

[662]: False

[664]:
```
bool(1)
```

[664]: True

[666]:
```
bool(None)
```

[666]: False

[668]:
```
bool(False)
```

[668]: False

8 String Creation

[671]:
```
#define string using double quotes
str = "HELLO PYTHON"
print(str)
```

```
HELLO PYTHON
```

[673]:
```
#Define string using single quotes
mystr = 'Hello World'
print(mystr)
```

```
Hello World
```

[675]:
```
#define string using triple quotes
mystr1 = '''HELLO
            WORLD'''
print(mystr1)
```

```
HELLO
            WORLD
```

[677]:
```
mstr3 = """Hello
                    Python"""
print(mstr3)
```

```
Hello
                    Python
```

[679]:
```
mystr = ('Happy '
          'Monday '
           'Everyone')
print(mystr)
```

```
Happy Monday Everyone
```

[681]:
```
mstr = 'Woohoo  '
mstr = mstr*5
print(mstr)
```

```
Woohoo  Woohoo  Woohoo  Woohoo  Woohoo
```

[683]:
```
mstr = 'Wow ' *5
print(mstr)
```

```
Wow Wow Wow Wow Wow
```

[685]:
```
len(mstr)
```

[685]:
```
20
```

[687]:
```
str
```

[687]:
```
'HELLO PYTHON'
```

[689]:
```
str = "HELLO WORLD"
str[0]
```

[689]:
```
'H'
```

[691]:
```
str[5]
```

[691]: ' '

[693]:
```
str[4]
```

[693]: 'O'

[695]:
```
str[-2]
```

[695]: 'L'

[697]:
```
str[len(str) - 5]
```

[697]: 'W'

[699]:
```
str[-1]
```

[699]: 'D'

[701]:
```
str[0:5]
```

[701]: 'HELLO'

[703]:
```
str
```

[703]: 'HELLO WORLD'

[705]:
```
str[-4]
```

[705]: 'O'

[707]:
```
str = "HELLO PYTHON"
print(str)
```

HELLO PYTHON

[709]:
```
str[-4]
```

[709]: 'T'

[711]:
```
str[-4:0]
```

[711]: ''

[713]:
```
str[-9:]
```

[713]:
```
'LO PYTHON'
```

[715]:
```
str[-9:0]
```

[715]:
```
''
```

[717]:
```
str[3:9]
```

[717]:
```
'LO PYT'
```

[719]:
```
str[:4]
```

[719]:
```
'HELL'
```

[723]:
```
del str
print(str)
```

```
<class 'str'>
```

[725]:
```
print(str)
```

```
<class 'str'>
```

[727]:
```
#String Concatenation
s1 = "Hellooo"
s2 = " World"
s3 = s1+s2
print(s3)
```

```
Hellooo World
```

9 DataType Interview questions

[730]:
```
int(12.3)
```

[730]:
```
12
```

[732]:
```
int(True)
```

[732]:
```
1
```

[734]:
```
int("1")
```

[734]:
```
1
```

[736]:
```
type(np.nan)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[736], line 1
----> 1 type(np.nan)

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

[739]:
```
type(12.3)
```

[739]:
```
float
```

[741]:
```
print(True*2)
```

```
2
```

[743]:
```
poll_data = 7
type(poll_data)
```

[743]:
```
int
```

[745]:
```
set(range(9))
```

[745]:
```
{0, 1, 2, 3, 4, 5, 6, 7, 8}
```

[747]:
```
list(range(9))
```

[747]:
```
[0, 1, 2, 3, 4, 5, 6, 7, 8]
```

[749]:
```
tuple(range(9))
```

[749]:
```
(0, 1, 2, 3, 4, 5, 6, 7, 8)
```

[751]:
```
dict(range(9))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[751], line 1
----> 1 dict(range(9))

TypeError: cannot convert dictionary update sequence element #0 to a sequence
```

[753]:
```
obj_data = ()
type(obj_data)
```

[753]:
```
tuple
```

10 Strings

[756]:
```
s = 'nit'
```

[758]:
```
s
```

[758]:
```
'nit'
```

[760]:
```
type(s)
```

[760]:
```
str
```

[762]:
```
s1 = "hello python"
```

[764]:
```
s1
```

[764]:
```
'hello python'
```

[766]:
```
s2 = '''nit
            hello python'''
s2
```

[766]:
```
'nit\n            hello python'
```

[768]:
```
s1
```

[768]:
```
'hello python'
```

[770]:
```
s1[-4]
```

[770]:
```
't'
```

[772]:
```
s1[4]
```

[772]:
```
'o'
```

[774]:
```
s[-7]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[774], line 1
----> 1 s[-7]

IndexError: string index out of range
```

[776]:
```
s1
```

[776]: 'hello python'

[778]:
```
s1[-7]
```

[778]: ' '

[780]:
```
s
```

[780]: 'nit'

[782]:
```
print(s[0])
print(s[1])
print(s[2])
```

```
n
i
t
```

11 String slicing

[788]:
```
s1
```

[788]: 'hello python'

[790]:
```
s1[:]
```

[790]: 'hello python'

[792]:
```
s1[2:7]
```

[792]: 'llo p'

[794]:
```
s2
```

[794]: 'nit\n   hello python'

[796]:
```
s3 = 'dataanalyst'
s3
```

[796]: 'dataanalyst'

[798]:
```
s[0:10]
```

[798]: 'nit'

[800]:
```
s3[0:10]
```

[800]: 'dataanalys'

[802]:
```
s3[0:11]
```

[802]: 'dataanalyst'

[804]:
```
s3
```

[804]: 'dataanalyst'

[806]:
```
s3[12]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[806], line 1
----> 1 s3[12]

IndexError: string index out of range
```

[808]:
```
s3[9:12]
```

[808]: 'st'

[810]:
```
s3[0:11:2]
```

[810]: 'dtaayt'

[812]:
```
s3[0:11:3]
```

[812]: 'daas'

[814]:
```
s3[2:-2]
```

[814]: 'taanaly'

[816]:
```
print(s)
print(s1)
```

```
nit
hello python
```

[818]:
```
print(s3)
```

```
dataanalyst
```

[820]:
```
for i in s3:
    print(i)
```

```
d
a
t
a
a
n
a
l
y
s
t
```

[822]:
```
import keyword
len(keyword.kwlist)
```

[822]: 35

12 python typecasting or type conversion

[825]:
```
int(2.3)
```

[825]: 2

[827]:
```
int
```

[827]: int

[831]:
```
int(True) #bool to int
```

[831]: 1

[833]:
```
int(1+2j)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[833], line 1
----> 1 int(1+2j)

TypeError: int() argument must be a string, a bytes-like object or a real␣
 ↪number, not 'complex'
```

[835]:
```
int('10')
```

[835]: 10

[837]:
```
int('ten')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[837], line 1
----> 1 int('ten')

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

[856]:
```
s2 = 'nit'
s2
```

[856]: 'nit'

[858]:
```
s2
```

[858]: 'nit'

[860]:
```
del s2
```

[862]:
```
np.nan
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[862], line 1
----> 1 np.nan

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

[864]:
```
type(np.nan)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[864], line 1
----> 1 type(np.nan)

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

[866]:
```
import numpy as np
a = np.nan
```

[868]:
```
type(a)
```

[868]: float

[870]:
```
s1 = 10
s2 = 20
s3 = 30
add = s1+s2+s3
print('The sum of',s1,',',s2,'and',s3,'is:',add)
print(f'The sum of {s1},{s2} and {s3} is: {add}')
print('The sum of {},{} and {} is: {}'.format(s1,s2,s3,add))
#using format methof,f
```

```
The sum of 10 , 20 and 30 is: 60
The sum of 10,20 and 30 is: 60
The sum of 10,20 and 30 is: 60
```

[872]:
```
print("Hello!")
print("How are you?")
```

```
Hello!
How are you?
```

[874]:
```
print("Hello!",end =' ')
print("How are you?")
```

```
Hello! How are you?
```

[876]:
```
print("Jan","Feb","Mar","Apr",sep = '-->')
```

```
Jan-->Feb-->Mar-->Apr
```

[878]:
```
print("Sun","Mon","Tue",sep = '#')
```

```
Sun#Mon#Tue
```

[880]:
```
print(1,2,end = ' ')
print(3,'.',sep = '')
```

```
1 2 3.
```

[882]:
```
'Hello' in 'Hello World'
```

[882]: True

[884]:
```
'Hello' not in 'Hello World'
```

[884]: False

[892]:
```
text = "Python Language"
text.replace("Python", "Programming")
```

[892]:
```
'Programming Language'
```

[895]:
```
fruits = "apple, mango, apple, orange"
fruits.replace("apple", "cherry", 3)
```

[895]:
```
'cherry, mango, cherry, orange'
```

[899]:
```
fruits = "apple, mango, apple, orange"
print(fruits)
```

```
apple, mango, apple, orange
```

[901]:
```
fruits.replace("apple","jackfruit",100)
```

[901]:
```
'jackfruit, mango, jackfruit, orange'
```

[903]:
```
print(fruits)
```

```
apple, mango, apple, orange
```

######Manipulating strings

[906]:
```
print("Hello \'how are you?\'")
```

```
Hello 'how are you?'
```

[908]:
```
print("Hello \"how are you?\"")
```

```
Hello "how are you?"
```

[910]:
```
print("Hello. \\What is your name?\\")
```

```
Hello. \What is your name?\
```

[912]:
```
print("Hello.\\")
```

```
Hello.\
```

[916]:
```
print("Hello\r1")
```

```
1ello
```

[918]:
```
print("Hai bye\r6")
```

```
6ai bye
```

[920]:
```
s = (r'haihello world\bbye')
print(s)
```

```
haihello world\bbye
```

[922]:
```
p = "333"
print(p.isalnum())
```

```
True
```

[924]:
```
p = "hello World"
print(p.upper())
```

```
HELLO WORLD
```

[926]:
```
m = "33 33"
p.isalnum()
```

[926]: False

[930]:
```
m = "sefs"
m.isdecimal()
```

[930]: False

[932]:
```
m1 = "sdfv\tdf"
m1.isspace()
```

[932]: False

[934]:
```
r = "Hello World"
r.istitle()
```

[934]: True

[936]:
```
'Hello world'.startswith('uuuu')
```

[936]: False

[938]:
```
"hai hello".endswith('helloy7987')
```

[938]: False

[940]:
```
'ABC'.join(['Hai','Hello','Bye'])
```

[940]: 'HaiABCHelloABCBye'

[942]:
```
' ,'.join(['Hai','Hello','Bye'])
```

[942]: 'Hai ,Hello ,Bye'

[944]:
```
'Hai im studing python'.split('i')
```

[944]:
```
['Ha', ' ', 'm stud', 'ng python']
```

[946]:
```
'Hello World'.rjust(20,'&')
```

[946]:
```
'&&&&&&&&&Hello World'
```

[948]:
```
'Hello world'.ljust(30,'-')
```

[948]:
```
'Hello world------------------'
```

[952]:
```
'Hello world'.center(30,'*')
```

[952]:
```
'*********Hello world**********'
```

[954]:
```
s = '       Hello World    '
s.strip()
```

[954]:
```
'Hello World'
```

[950]:
```
spam = 'SpamSpamBaconSpamEggsSpamSpam'
spam.strip('amS')
```

[950]:
```
'pamSpamBaconSpamEggsSpamSp'
```

[956]:
```
sent = ("Thids isdsdss dsfdfal")
sent.count('d',6,9)
```

[956]:
```
1
```

[958]:
```
text = 'Hello World Hello What World'
text.replace("World","Hai",2)
```

[958]:
```
'Hello Hai Hello What Hai'
```

[976]:
```
class1 = 'Four'
school = 'XYZ'
print('My class is {} and im in school {}'.format(class1, school))
```

```
My class is Four and im in school XYZ
```

[986]:
```
name = 'Teju'
section = 'A'
(
f'My name is {name}. '
f'Im in section {section}'
)
```

[986]:
```
'My name is Teju. Im in section A'
```

[990]:
```
p = "hai world"
p.capitalize()
```

[990]:
```
'Hai world'
```

[992]:
```
min([2,5,6,3,1])
```

[992]:
```
1
```

[994]:
```
s = 'abc'
s*3
```

[994]:
```
'abcabcabc'
```

[996]:
```
s[0]
```

[996]:
```
'a'
```

[1000]:
```
s.center(2)
```

[1000]:
```
'abc'
```

[1002]:
```
a = 'entrance'
a.isspace()
```

[1002]:
```
False
```

[1004]:
```
a = 'Abc'
a.islower()
```

[1004]:
```
False
```

[1006]:
```
complex(True)
```

[1006]:
```
(1+0j)
```

[1008]:
```
complex(False)
```

[1008]:
```
0j
```

[1010]:
```
bool(1)
```

[1010]:
```
True
```

[1012]:
```
bool(0)
```

[1012]:
```
False
```

[1014]:
```
bool(2.3)
```

[1014]:
```
True
```

[1016]:
```
bool( )
```

[1016]:
```
False
```

[1020]:
```
bool('nit')
```

[1020]:
```
True
```

[1024]:
```
bool(0+0j)
```

[1024]:
```
False
```

[1026]:
```
print(str(2))
```

```
2
```

[1030]:
```
print(str(2.3))
```

```
2.3
```

[1032]:
```
print(str(True))
```

```
True
```

[1034]:
```
print(str(1+2j))
```

```
(1+2j)
```

13 Python type casting (convert all other datatype of one datatype)

[1046]:
```
index = 'HELLOPYTHON'
index
```

[1046]:
```
'HELLOPYTHON'
```

[1048]:
```
index[:]
```

[1048]:
```
'HELLOPYTHON'
```

[1050]:
```
index[::-1]
```

[1050]: 'NOHTYPOLLEH'

[1052]:
```
index[::-2]
```

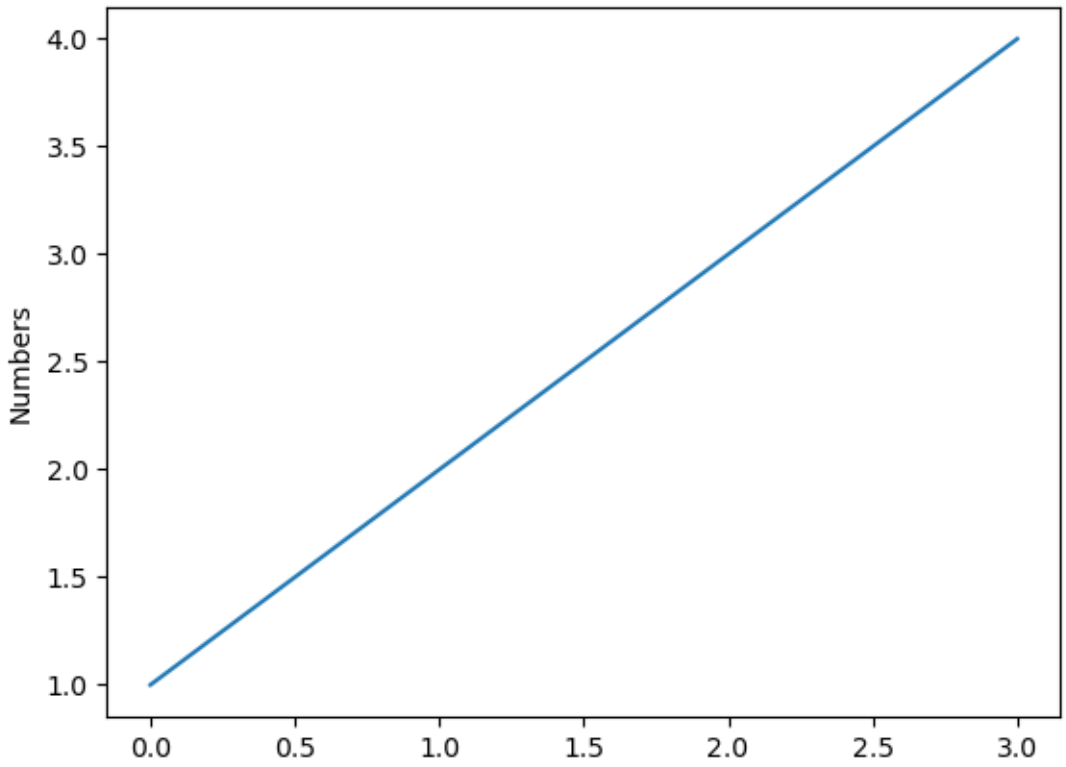
[1052]: 'NHYOLH'

[1054]:
```
index[1:10:3]
```

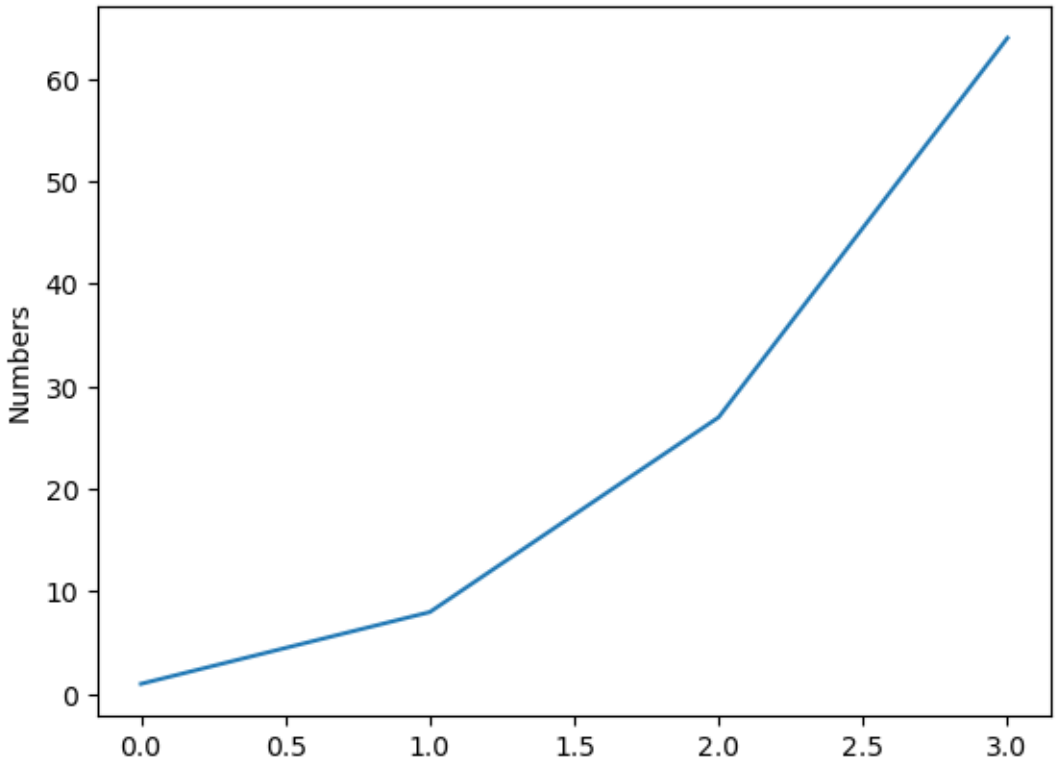
[1054]: 'EOT'

14 Python type casting completed

[1057]:
```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4])
plt.ylabel('Numbers')
plt.show()
```



[1059]:
```
import matplotlib.pyplot as plt
plt.plot([1,8,27,64])
plt.ylabel('Numbers')
plt.show()
```



15 26th - DATA STRUCTURE

[1062]:
```
l = []
l
```

[1062]:
```
[]
```

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

[1064]:
```
l.append(10)
l
```

[1064]:
```
[10]
```

[1068]:
```
l.append(50,30)
l
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1068], line 1
----> 1 l.append(50,30)
```

```
      2 l

TypeError: list.append() takes exactly one argument (2 given)
```

[1070]:
```
l.append([50,30])
print(l)
```

```
[10, [50, 30]]
```

[1072]:
```
len(l)
```

[1072]: 2

[1076]:
```
l = []

l.append(10)
l.append(20)
l.append(30)
l.append(2.3)
l.append(1+2j)
l.append(True)
l.append('nit')
l
```

[1076]: [10, 20, 30, 2.3, (1+2j), True, 'nit']

[ ]:
```
l.count(20)
```

[1078]:
```
len(l)
```

[1078]: 7

[1080]:
```
l.append(10)
l
```

[1080]: [10, 20, 30, 2.3, (1+2j), True, 'nit', 10]

[1082]:
```
l1 = l.copy()
l
```

[1082]: [10, 20, 30, 2.3, (1+2j), True, 'nit', 10]

[ ]:
```
l1.remove(20)
l
```

[1086]:
```
l[:]
```

[1086]:
```
[10, 20, 30, 2.3, (1+2j), True, 'nit', 10]
```

[1088]:
```
l[4]
```

[1088]:
```
(1+2j)
```

[1092]:
```
l.count(20)
```

[1092]:
```
1
```

[1094]:
```
l = [20,30,2.3,(1+2j),True,'nit',10]
print(l)
```

```
[20, 30, 2.3, (1+2j), True, 'nit', 10]
```

[1096]:
```
l.count(20)
```

[1096]:
```
1
```

[1098]:
```
l1 = l.copy()
print(l1)
```

```
[20, 30, 2.3, (1+2j), True, 'nit', 10]
```

[1100]:
```
l.count(30)
```

[1100]:
```
1
```

[1102]:
```
l.append(20)
print(l)
```

```
[20, 30, 2.3, (1+2j), True, 'nit', 10, 20]
```

[1104]:
```
l
```

[1104]:
```
[20, 30, 2.3, (1+2j), True, 'nit', 10, 20]
```

[1106]:
```
l1
```

[1106]:
```
[20, 30, 2.3, (1+2j), True, 'nit', 10]
```

16 28th - list data structure

[1111]:
```
l2 = []
l2
```

[1111]:
```
[]
```

[1113]:
```
l2.append(1)
l2.append(2.3)
l2.append(True)
l2.append(1+2j)
l2.append('nit')
l2
```

[1113]: [1, 2.3, True, (1+2j), 'nit']

[1115]:
```
l3 = l2.copy()
l3
```

[1115]: [1, 2.3, True, (1+2j), 'nit']

[1117]:
```
l2
```

[1117]: [1, 2.3, True, (1+2j), 'nit']

[1119]:
```
l3
```

[1119]: [1, 2.3, True, (1+2j), 'nit']

[1121]:
```
len(l3)
```

[1121]: 5

[1127]:
```
l3.clear()
```

[1129]:
```
len(l3)
```

[1129]: 0

[1131]:
```
del l3
```

[1133]:
```
l2
```

[1133]: [1, 2.3, True, (1+2j), 'nit']

[1135]:
```
l3
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1135], line 1
----> 1 l3

NameError: name 'l3' is not defined
```

[1137]:
```
l2
```

[1137]:
```
[1, 2.3, True, (1+2j), 'nit']
```

[1139]:
```
l2.remove(2.3)
```

[1141]:
```
l2
```

[1141]:
```
[1, True, (1+2j), 'nit']
```

[1145]:
```
l2.append(1)
```

[1148]:
```
l2
```

[1148]:
```
[1, True, (1+2j), 'nit', 1]
```

[1152]:
```
l3 = []
l3.append(10)
l3
```

[1152]:
```
[10]
```

[1154]:
```
l2
```

[1154]:
```
[1, True, (1+2j), 'nit', 1]
```

[1158]:
```
l3.extend(l2)
l3
```

[1158]:
```
[10, 1, True, (1+2j), 'nit', 1]
```

[1166]:
```
l3.index(10)
```

[1166]:
```
0
```

[1170]:
```
l3.index(True)
```

[1170]:
```
1
```

[1172]:
```
l3.index('nit')
```

[1172]:
```
4
```

[1174]:
```
l3.index(1+2j)
```

[1174]:
```
3
```

[1177]:
```
l2
```

[1177]:
```
[1, True, (1+2j), 'nit', 1]
```

[1179]:
```
l3
```

[1179]:
```
[10, 1, True, (1+2j), 'nit', 1]
```

[1181]:
```
l2
```

[1181]:
```
[1, True, (1+2j), 'nit', 1]
```

[1183]:
```
l2.index('nit')
```

[1183]:
```
3
```

[1185]:
```
l3
```

[1185]:
```
[10, 1, True, (1+2j), 'nit', 1]
```

[1187]:
```
l3.insert(3,False)
```

[1189]:
```
l3
```

[1189]:
```
[10, 1, True, False, (1+2j), 'nit', 1]
```

[1191]:
```
l3.insert(1,2)
```

[1193]:
```
l3
```

[1193]:
```
[10, 2, 1, True, False, (1+2j), 'nit', 1]
```

[1195]:
```
l3.insert(5,'programming')
```

[1197]:
```
l3
```

[1197]:
```
[10, 2, 1, True, False, 'programming', (1+2j), 'nit', 1]
```

[1199]:
```
l3.remove(2)
l3.remove("programming")
l3
```

[1199]:
```
[10, 1, True, False, (1+2j), 'nit', 1]
```

[1201]:
```
l3.insert(6,"programming")
l3
```

[1201]: [10, 1, True, False, (1+2j), 'nit', 'programming', 1]

[1203]:
```
l3.pop()
```

[1203]: 1

[1205]:
```
l3.pop()
```

[1205]: 'programming'

[1207]:
```
l3
```

[1207]: [10, 1, True, False, (1+2j), 'nit']

[1209]:
```
l3.append('programming')
l3
```

[1209]: [10, 1, True, False, (1+2j), 'nit', 'programming']

[1217]:
```
l3
```

[1217]: [10, 1, True, False, 'programming']

[1221]:
```
l3.insert(4,'nit')
l3
```

[1221]: [10, 1, True, False, 'nit', 'programming']

[1223]:
```
l4 = [10,100,3,45,76,24]
l4
```

[1223]: [10, 100, 3, 45, 76, 24]

[1225]:
```
l4.sort()
```

[1227]:
```
l4
```

[1227]: [3, 10, 24, 45, 76, 100]

[1229]:
```
l4.sort(reverse = True)
l4
```

[1229]: [100, 76, 45, 24, 10, 3]

[1231]:
```
l5 = ['z','m','c','w']
l5
```

[1231]:
```
['z', 'm', 'c', 'w']
```

[1233]:
```
l5.sort()
l5
```

[1233]:
```
['c', 'm', 'w', 'z']
```

[1235]:
```
l5.sort(reverse = True)
l5
```

[1235]:
```
['z', 'w', 'm', 'c']
```

[1237]:
```
l6 = [1,2,3, 'a','z','w']
l6
```

[1237]:
```
[1, 2, 3, 'a', 'z', 'w']
```

[1239]:
```
l6.sort()
l6
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1239], line 1
----> 1 l6.sort()
      2 l6

TypeError: '<' not supported between instances of 'str' and 'int'
```

[1241]:
```
l2
```

[1241]:
```
[1, True, (1+2j), 'nit', 1]
```

[1243]:
```
l3
```

[1243]:
```
[10, 1, True, False, 'nit', 'programming']
```

[1245]:
```
l3.reverse()
l3
```

[1245]:
```
['programming', 'nit', False, True, 1, 10]
```

[1247]:
```
l2
```

[1247]:
```
[1, True, (1+2j), 'nit', 1]
```

[1249]:
```
l2[3]
```

[1249]:
```
'nit'
```

[1251]:
```
print(l2[3][0])
print(l2[3][1])
print(l2[3][2])
```

```
n
i
t
```

17 Using Strip function

[1]:
```
txt = "   abc def ghi    "
txt.lstrip()
```

[1]:
```
'abc def ghi    '
```

[3]:
```
txt = "   abc def ghi    "
txt.rstrip()
```

[3]:
```
'   abc def ghi'
```

[5]:
```
txt = "   abc def ghi    "
txt.strip()
```

[5]:
```
'abc def ghi'
```

18 Using Escape Character

[10]:
```
mystr = "My favourite TV series is "Game of Thrones""
print(mystr)
```

```
  Cell In[10], line 1
    mystr = "My favourite TV series is "Game of Thrones""
                                        ^
SyntaxError: invalid syntax
```

[12]:
```
mystr = "My favourite TV series is \"Game of Thrones\""
print(mystr)
```

```
My favourite TV series is "Game of Thrones"
```

19 List Creation

[15]:
```
list1 = []
print(type(list1))
```

```
<class 'list'>
```

[17]:
```
list2 = [10,30,60]
```

[19]:
```
list3 = [10.77,30.66,60.89]
```

[21]:
```
list4 = ['one','two','three']
```

[23]:
```
list5 = [100,'Roja',[50,100],[150,60]]
```

[25]:
```
list6 = [100,'Roja',19.53]
```

[27]:
```
list7 = ['Roja',34,[50,300],[30,60],{'John','David'}]
```

[29]:
```
len(list7)
```

[29]: 5

[31]:
```
len(list6)
```

[31]: 3

20 List Indexing

[33]:
```
list7[0][3]
```

[33]: 'a'

[38]:
```
list6[2]
```

[38]: 19.53

[40]:
```
list6[-1]
```

[40]: 19.53

21 Add, Remove and Change Items

[43]:
```
mylist = ['one','two','three','four','five','six','seven','eight']
mylist
```

[43]:
```
['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

[57]:
```
mylist.remove('nine')
mylist
```

[57]:
```
['one',
 'two',
 'ten',
 'three',
 'four',
 'five',
 'six',
 'seven',
 'eight',
 'nine',
 'nine']
```

[59]:
```
mylist.remove('nine')
mylist
```

[59]:
```
['one', 'two', 'ten', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

[61]:
```
mylist.append('ten')
mylist
```

[61]:
```
['one',
 'two',
 'ten',
 'three',
 'four',
 'five',
 'six',
 'seven',
 'eight',
 'nine',
 'ten']
```

[65]:
```
mylist.pop()
```

[65]:
```
'ten'
```

[67]:
```
mylist
```

[67]:
```
['one', 'two', 'ten', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

[69]:
```
mylist.pop(2)
mylist
```

[69]:
```
['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

[71]:
```
del mylist[1]
mylist
```

[71]:
```
['one', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

[73]:
```
mylist[0] = 1
mylist[7] = 8
mylist
```

[73]:
```
[1, 'three', 'four', 'five', 'six', 'seven', 'eight', 8]
```

[75]:
```
mylist.clear()
mylist
```

[75]:
```
[]
```

[79]:
```
myl = [1,4,'abc',6,'xyz',55.6,9]
myl
```

[79]:
```
[1, 4, 'abc', 6, 'xyz', 55.6, 9]
```

[81]:
```
mylist2 = myl.copy()
```

[83]:
```
mylist2
```

[83]:
```
[1, 4, 'abc', 6, 'xyz', 55.6, 9]
```

[89]:
```
id(mylist),id(myl),id(mylist2)
```

[89]:
```
(2301953660288, 2301953968704, 2301922893504)
```

[113]:
```
m1 = [1,2,3,5]
m1
```

[113]:
```
[1, 2, 3, 5]
```

[103]:
```
m2 = []
m2 = m1
```

[107]:
```
m1
```

[107]:
```
[1, 2, 3, 5]
```

[109]:
```
m2
```

[109]:
```
[1, 2, 3, 5]
```

[111]:
```
id(m1),id(m2),id(mystr)
```

[111]:
```
(2301954048896, 2301954048896, 2301898299344)
```

[175]:
```
print(type(range(5)))
```

```
<class 'range'>
```

[177]:
```
type(10)
```

[177]:
```
int
```

[ ]: