

WORK WITH PYTHON NUMBERS

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

```
Out[1]: 5
```

```
In [2]: 5+5
```

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

```
In [3]: -5-5
```

```
Out[3]: -10
```

```
In [4]: 5+6-7*3-7
```

```
Out[4]: -17
```

```
In [5]: 5+6-7*(3-7)
```

```
Out[5]: 39
```

```
In [6]: _
```

```
Out[6]: 39
```

```
In [7]: _+1
```

```
Out[7]: 40
```

```
In [8]: _=4
```

```
In [9]: _-4
```

```
Out[9]: 0
```

```
In [ ]: a=3  
        b=4
```

```
In [10]: int.__add__(a,b)
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[10], line 1  
----> 1 int.__add__(a,b)  
NameError: name 'a' is not defined
```

```
In [ ]: a=3  
        b=4
```

```
In [11]: int.__sub__(a,b)
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[11], line 1  
----> 1 int.__sub__(a,b)  
  
NameError: name 'a' is not defined
```

```
In [12]: c=suunitha  
        d=venu  
        int.__add__(c,d)
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[12], line 1  
----> 1 c=suunitha  
      2 d=venu  
      3 int.__add__(c,d)  
  
NameError: name 'suunitha' is not defined
```

```
In [13]: str.__add__(c,d)
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[13], line 1  
----> 1 str.__add__(c,d)  
  
NameError: name 'c' is not defined
```

```
In [ ]:
```

works with text

```
In [14]: Naresh IT
```

```
Cell In[14], line 1  
    Naresh IT  
      ^  
SyntaxError: invalid syntax
```

```
In [15]: 'Naresh IT'
```

```
Out[15]: 'Naresh IT'
```

```
In [16]: "Naresh IT"
```

```
Out[16]: 'Naresh IT'
```

```
In [17]: '''Naresh IT'''
```

```
Out[17]: 'Naresh IT'
```

```
In [18]: 'Naresh  
        Technology.'
```

Cell In[18], line 1

'Naresh

^

SyntaxError: unterminated string literal (detected at line 1)

In [19]: "Naresh
Technology"

Cell In[19], line 1

"Naresh

^

SyntaxError: unterminated string literal (detected at line 1)

In [20]: '''Naresh
Technology'''

Out[20]: 'Naresh \n Technology'

In []:

In []:

28th variables

In [1]: v=10
v

Out[1]: 10

In [2]: id(v)

Out[2]: 140717069481160

In []:

In [3]: nit=8
NIT

NameError Traceback (most recent call last)

Cell In[3], line 2

1 nit=8

----> 2 NIT

NameError: name 'NIT' is not defined

In []:

In [4]: 8=nit

Cell In[4], line 1

8=nit

^

SyntaxError: cannot assign to literal here. Maybe you meant '==' instead of '='?

In []:

In [5]: `8nit=10`

```
Cell In[5], line 1
    8nit=10
      ^
SyntaxError: invalid decimal literal
```

In []:

In [8]: `nit8=20`
`nit8`

Out[8]: 20

In []:

In [9]: `nit$=50`
`nit$`

```
Cell In[9], line 1
    nit$=50
      ^
SyntaxError: invalid syntax
```

In []:

In [10]: `nit_=78`
`nit_`

Out[10]: 78

In []:

In [11]: `import keyword`
`keyword.kwlist`

```
Out[11]: ['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 []:

```
In [12]: def=50
         def
```

Cell In[12], line 1

def=50

^

SyntaxError: invalid syntax

In []:

```
In [13]: DEF=60
         DEF
```

Out[13]: 60

In []:

```
In [14]: 3+4
```

Out[14]: 7

```
In [15]: 3=4  
         4=5
```

Cell In[15], line 1

```
3=4  
^
```

SyntaxError: cannot assign to literal here. Maybe you meant '==' instead of '='?

```
In [16]: 3+4  
         4+5
```

Out[16]: 9

```
In [ ]:
```

```
In [17]: 3+4  
         4+5  
         5+6
```

Out[17]: 11

```
In [ ]:
```

```
In [18]: print(3+4)  
         print(4+5)  
         print(5+6)
```

7

9

11

```
In [ ]:
```

```
In [ ]:
```

1st Python datatypes

```
In [1]: i=100  
        i
```

Out[1]: 100

```
In [2]: type(i)
```

Out[2]: int

```
In [ ]:
```

```
In [ ]:
```

```
In [1]: j=200.56  
        j
```

Out[1]: 200.56

```
In [2]: type(j)
```

```
Out[2]: float
```

```
In [ ]:
```

```
In [3]: f1=1e0  
f1
```

```
Out[3]: 1.0
```

```
In [4]: f2=1e1  
f2
```

```
Out[4]: 10.0
```

```
In [5]: f3=1e2  
f3
```

```
Out[5]: 100.0
```

```
In [7]: f4=2e3  
f4
```

```
Out[7]: 2000.0
```

```
In [ ]:
```

```
In [8]: import keyword  
keyword.kwlist
```

```
Out[8]: ['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 [ ]:
```

```
In [9]: b = True  
b
```

```
Out[9]: True
```

```
In [10]: b = True  
b1 = False  
b + b1
```

```
Out[10]: 1
```

```
In [11]: print(b+b1)  
print(b-b1)  
print(b*b1)
```

```
1  
1  
0
```

```
In [12]: False / True
```


Out[12]: 0.0

```
In [13]: c=False  
c
```

Out[13]: False

```
In [14]: print(True)
```

True

```
In [ ]:
```

```
In [1]: False/True
```

Out[1]: 0.0

```
In [3]: False//True
```

Out[3]: 0

```
In [2]: True/False
```

```
-----  
ZeroDivisionError                                Traceback (most recent call last)  
Cell In[2], line 1  
----> 1 True/False  
  
ZeroDivisionError: division by zero
```

```
In [ ]:
```

complex

```
In [4]: c=10+20j  
c
```

Out[4]: (10+20j)

```
In [5]: type(c)
```

Out[5]: complex

```
In [ ]:
```

```
In [6]: c.real
```

Out[6]: 10.0

```
In [8]: c.imag
```

Out[8]: 20.0

```
In [10]: False+True
```

Out[10]: 1

In []:

In [13]: `c1=10+20.3j`
`c1`

Out[13]: (10+20.3j)

In [15]: `c2=30+53.8`
`c2`

Out[15]: 83.8

In [16]: `c1+c2`

Out[16]: (93.8+20.3j)

In []:

string

In [74]: `name='sunitha'`
`name`

Out[74]: 'sunitha'

In [75]: `type(name)`

Out[75]: str

In []:

In [76]: `name[1]`

Out[76]: 'u'

In [78]: `name[0]`

Out[78]: 's'

In [79]: `name[-1:0]`

Out[79]: ''

In [80]: `name[0:-1]`

Out[80]: 'sunith'

In [81]: `name[1:-1]`

Out[81]: 'unith'

```
In [82]: name[10]
```

```
-----  
IndexError                                Traceback (most recent call last)  
Cell In[82], line 1  
----> 1 name[10]  
  
IndexError: string index out of range
```

```
In [83]: name[-10]
```

```
-----  
IndexError                                Traceback (most recent call last)  
Cell In[83], line 1  
----> 1 name[-10]  
  
IndexError: string index out of range
```

```
In [84]: name[:]
```

```
Out[84]: 'sunitha'
```

```
In [85]: name[2:]
```

```
Out[85]: 'nitha'
```

```
In [86]: name[:-1]
```

```
Out[86]: 'sunith'
```

```
In [87]: name[:5]
```

```
Out[87]: 'sunit'
```

```
In [ ]:
```

```
In [88]: name
```

```
Out[88]: 'sunitha'
```

```
In [89]: name[1:-2:3]
```

```
Out[89]: 'ut'
```

```
In [ ]:
```

backward index

```
In [90]: name[-1]
```

```
Out[90]: 'a'
```

```
In [91]: name[-2:]
```

```
Out[91]: 'ha'
```

```
In [92]: name[-2:0]
```

```
Out[92]: ''
```

```
In [93]: name[-2:-1]
```

```
Out[93]: 'h'
```

```
In [ ]:
```

```
In [42]: movie = '''Hero: Yash (as Rocky)
Heroine: Srinidhi Shetty
Villain: Garuda
Music: Ravi Basrur

Story in short:

Rocky, born in poverty, grows up to become a powerful gangster in Mumbai.
He is sent to assassinate Garuda, the ruthless ruler of the Kolar Gold Fields.
The film ends with Rocky killing Garuda, setting the stage for Chapter 2.'''
```

```
In [43]: movie
```

```
Out[43]: 'Hero: Yash (as Rocky)\nHeroine: Srinidhi Shetty\nVillain: Garuda\nMusic: Ravi
Basrur\n\nStory in short:\n\nRocky, born in poverty, grows up to become a power
ful gangster in Mumbai. \n\nHe is sent to assassinate Garuda, the ruthless rule
r of the Kolar Gold Fields. \nThe film ends with Rocky killing Garuda, setting
the stage for Chapter 2.'
```

```
In [ ]:
```

```
In [ ]:
```

2nd python typecasting

convert all datatypes to int except complex and text string

```
In [44]: int(100.5)
```

```
Out[44]: 100
```

```
In [45]: int(100.9)
```

```
Out[45]: 100
```

```
In [ ]:
```

```
In [47]: int(True)
```

Out[47]: 1

In [48]: `int(False)`

Out[48]: 0

In []:

In [49]: `int(10+20j)`

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[49], line 1  
----> 1 int(10+20j)  
  
TypeError: int() argument must be a string, a bytes-like object or a real number,  
not 'complex'
```

In []:

In [50]: `int("10")`

Out[50]: 10

In [51]: `int("ten")`

```
-----  
ValueError                                Traceback (most recent call last)  
Cell In[51], line 1  
----> 1 int("ten")  
  
ValueError: invalid literal for int() with base 10: 'ten'
```

In []:

cast other datatypes to float

In [53]: `float(25)`

Out[53]: 25.0

In [54]: `float(25,39)`

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[54], line 1  
----> 1 float(25,39)  
  
TypeError: float expected at most 1 argument, got 2
```

In []:

In [55]: `float(10+20j)`

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[55], line 1  
----> 1 float(10+20j)  
  
TypeError: float() argument must be a string or a real number, not 'complex'
```

In []:

In [56]: `float("10")`

Out[56]: 10.0

In [57]: `float("ten")`

```
-----  
ValueError                                Traceback (most recent call last)  
Cell In[57], line 1  
----> 1 float("ten")  
  
ValueError: could not convert string to float: 'ten'
```

In []:

cast other datatypes to complex

In [58]: `complex(10)`

Out[58]: (10+0j)

In [59]: `complex(20,10)`

Out[59]: (20+10j)

In [60]: `complex(20,10,5)`

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[60], line 1  
----> 1 complex(20,10,5)  
  
TypeError: complex() takes at most 2 arguments (3 given)
```

In []:

In [61]: `complex(2.9)`

Out[61]: (2.9+0j)

In []:

In [62]: `complex(2.0+37.8)`

Out[62]: (39.8+0j)

```
In [63]: complex(True,False)
```

```
Out[63]: (1+0j)
```

```
In [64]: complex(False,True)
```

```
Out[64]: 1j
```

```
In [ ]:
```

```
In [65]: complex('10','20')
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[65], line 1  
----> 1 complex('10','20')  
  
TypeError: complex() can't take second arg if first is a string
```

```
In [ ]:
```

```
In [66]: complex("10",20)
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[66], line 1  
----> 1 complex("10",20)  
  
TypeError: complex() can't take second arg if first is a string
```

```
In [ ]:
```

```
In [67]: complex("10")
```

```
Out[67]: (10+0j)
```

```
In [68]: complex(20,'10')
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[68], line 1  
----> 1 complex(20,'10')  
  
TypeError: complex() second arg can't be a string
```

```
In [ ]:
```

```
In [69]: complex('ten')
```

```
-----  
ValueError                                Traceback (most recent call last)  
Cell In[69], line 1  
----> 1 complex('ten')  
  
ValueError: complex() arg is a malformed string
```

```
In [ ]:
```

cast other datatypes to boolean

```
In [95]: bool()
```

```
Out[95]: False
```

```
In [96]: bool(10)
```

```
Out[96]: True
```

```
In [97]: bool(9.8)
```

```
Out[97]: True
```

```
In [100... bool(0)
```

```
Out[100... False
```

```
In [101... bool(10+20j)
```

```
Out[101... True
```

```
In [102... bool("sunitha")
```

```
Out[102... True
```

```
In [ ]:
```

```
In [103... print(10) # 1 arg  
print(10,20) # 2 arg  
print('python') # string arg  
print(10,20,'python') # 3 arg
```

```
10  
10 20  
python  
10 20 python
```

```
In [ ]:
```

```
In [106... num1=10  
num2=20  
add=num1+num2  
print('The addition of', num1, 'and', num2, 'is:', add)
```

```
The addition of 10 and 20 is: 30
```

```
In [ ]:
```

```
In [109... num1,num2, num3=10,20,30  
add=num1+num2+num3  
print('The addition of {} and {} and {} is {}'.format(num1,num2,num3,add))
```

```
The addition of 10 and 20 and 30 is 60
```



```
In [111... print(f'The addition of {num1} and{num2} and {num3} is {add}')
```

The addition of 10 and20 and 30 is 60

```
In [ ]:
```

```
In [117... print('hello')
print('good morning')
```

hello
good morning

```
In [ ]:
```

```
In [119... print('hello',end='_')
print('good morning')
```

hello_good morning

```
In [ ]:
```

```
In [120... print('*')
print('***',end="")
print('*****')
```

*

```
In [ ]:
```

```
In [121... print('hello','hai','how are you',sep='--->')
```

hello--->hai--->how are you

```
In [122... print('hello','hai','how are you',sep='$-$')
```

hello\$-\$hai\$-\$how are you

```
In [ ]:
```

3rd python datastructure

```
In [123... l =[]
l
```

```
Out[123... []
```

```
In [124... type(l)
```

```
Out[124... list
```

```
In [125... len(l)
```

```
Out[125... 0
```

```
In [126... l
```

Out[126... []

```
In [129... 1.append(10)
1
```

Out[129... [10]

In []:

```
In [131... 1.append(20)
1.append(30)
1.append(40)
```

```
In [132... 1
```

Out[132... [10, 20, 30, 40, 20, 30, 40]

```
In [170... l2=l.copy()
l2
```

Out[170... [10, 20, 30, 40, 20, 30, 40, 20.3, 'hi', (20+10j), True]

In []:

```
In [138... print(l)
print(l2)
```

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

In []:

```
In [140... 1.append(20.3, 'hi', 20+10j, True)
```

```
-----
TypeError                                Traceback (most recent call last)
Cell In[140], line 1
----> 1 1.append(20.3, 'hi', 20+10j, True)

TypeError: list.append() takes exactly one argument (4 given)
```

In []:

```
In [141... 1.append(20.3)
1.append('hi')
1.append(20+10j)
1.append(True)
```

```
In [142... 1
```

Out[142... [10, 20, 30, 40, 20, 30, 40, 20.3, 'hi', (20+10j), True]

```
In [143... print(l)
print(l2)
```

```
[10, 20, 30, 40, 20, 30, 40, 20.3, 'hi', (20+10j), True]
[10, 20, 30, 40, 20, 30, 40]
```

In []:

In [147... `l2.clear()`

In [148... `l2`

Out[148... `[]`

In [149... `del l2`

In [150... `l2`

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[150], line 1  
----> 1 l2  
  
NameError: name 'l2' is not defined
```

In [151... `l`

Out[151... `[10, 20, 30, 40, 20, 30, 40, 20.3, 'hi', (20+10j), True]`

In [152... `l.count(10)`

Out[152... `1`

In [153... `l.count(20)`

Out[153... `2`

In [154... `l[:]`

Out[154... `[10, 20, 30, 40, 20, 30, 40, 20.3, 'hi', (20+10j), True]`

In [155... `l[1:]`

Out[155... `[20, 30, 40, 20, 30, 40, 20.3, 'hi', (20+10j), True]`

In [156... `l[5:]`

Out[156... `[30, 40, 20.3, 'hi', (20+10j), True]`

In [157... `l[:-3]`

Out[157... `[10, 20, 30, 40, 20, 30, 40, 20.3]`

In []:

In [158... `l[::-2]`

Out[158... `[True, 'hi', 40, 20, 30, 10]`

In [159... `l.index('hi')`

Out[159... 8

In [162... `l[8]`

Out[162... 'hi'

In []:

In [163... 1

Out[163... [10, 20, 30, 40, 20, 30, 40, 20.3, 'hi', (20+10j), True]

In [164... `l[:: -2]`

Out[164... [True, 'hi', 40, 20, 30, 10]

In [165... `l[::2]`

Out[165... [10, 30, 20, 40, 'hi', True]

In []:

In [169... `l[3:14:7]`

Out[169... [40, True]

In []:

In []: