

# 10th APRIL - FSDS, GENAI, AGENTIAI

```
In [1]: 9
```

```
Out[1]: 9
```

```
In [2]: 9 + 9
```

```
Out[2]: 18
```

```
In [3]: 9 + 9 - (10 - 3) + 3
```

```
Out[3]: 14
```

```
In [4]: 9 + 9 - 10 - 3 + 3
```

```
Out[4]: 8
```

numbers are called as operand

+, - \* are called operator

## ARITHMETIC OPERATOR

- 1- ARITHMETIC OPERATOR ( + , - , / , % , \*\* , \*)
- 2- ASSIGNMENT OPERATOR (=)
- 3- RELATIONAL OPERATOR
- 4- LOGICAL OPERATOR
- 5- UNARY OPERATOR

```
In [5]: 10 + 5
```

```
Out[5]: 15
```

```
In [6]: 10 - 5
```

```
Out[6]: 5
```

```
In [7]: 10 * 5
```

```
Out[7]: 50
```

```
In [8]: 10 * 2 # MULTIPLICATION
```

```
Out[8]: 20
```

```
In [9]: 10 ** 2 #POWER
```

```
Out[9]: 100
```

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

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

```
In [ ]: 10 / 5 # FLOAT DIVISION
```

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

```
In [ ]: 10 % 5
```

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

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

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

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

## Assignment operator

```
In [ ]: x = 10
x
# x is called variable or object or identifier
```

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

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

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

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

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

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

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

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

```
In [ ]: x
```

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

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

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

```
In [ ]: x
```

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

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

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

```
In [ ]: x
```

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

## unary operator

```
In [ ]: n = 7  
n
```

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

## 15th April -- Variable

```
In [ ]: 1nit = 9 # variable doesnot start with digit
```

```
In [ ]: nit1 = 9  
nit1
```

```
In [ ]: nit@ = 9 #variable shold not allows spcial character
```

```
In [ ]: nit_ = 9  
nit_
```

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

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

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

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

## 16th

```
In [ ]: a = 10
a
```

```
In [ ]: a = 20
a
```

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

## Python datatypes -->

- Integer datatypes
- float datatypes
- Bool datatypes
- string datatypes
- complex datatypes

```
In [ ]: i = 45
print(i)
print(type(i))
```

```
In [ ]: i1, i2 = 10
```

```
In [ ]: i1, i2 = 10, 20

print(i1)
print(i2)

print(i1 + i2)
print(i1 - i2)
print(i1 * i2)
print(i1 / i2)
```

## float

```
In [ ]: petrol = 110.56
petrol
```

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

**string (non techical - text, but technical we called this as string)**

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

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

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

```
In [ ]: s1 = "nareshit"  
s1
```

```
In [ ]: s2 = '''naresh it technology  
          datascience, ai student -- 6 month i will change your brian'''  
s2
```

```
In [ ]: s
```

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

```
In [ ]: s
```

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

```
In [ ]: s[2:5]
```

## 17th

```
In [ ]: true
```

```
In [ ]: True
```

```
In [ ]: false
```

```
In [ ]: False
```

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

```
In [ ]: b
```

```
In [ ]: b1
```

```
In [ ]: b + b1
```

```
In [ ]: print(b-b1)  
        print(b*b1)  
        print(b1/b) # float division  
        print(b1//b) # int division
```

```
In [ ]: type(b1)
```

## complex -- datatype we can use more math operation -- nasa, isro, robo

$a + bj$

- a - real part
- b - imaginary part
- j - square root of -1

```
In [ ]: c1 = 10 + 20j  
c1
```

```
In [ ]: type(c1)
```

```
In [ ]: c1
```

```
In [ ]: c1.real
```

```
In [ ]: c1.imag
```

```
In [ ]: c2 = 3 + 5j  
c2
```

```
In [ ]: print(c1)  
print(c2)
```

```
In [ ]: c1 + c2
```

## python data types

## TYPE CASTING OR TYPE CONVERSION

```
In [ ]: int(3.4) #float to int
```

```
In [ ]: int(3.4, 5.7)
```

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

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

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

```
In [ ]: print(int(3.4))  
print(int(True))  
print(int('10'))
```

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

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

**from all other datatype to int typecasting completed**

**complex & string with text is not allowed**

**18th - type casting continue**

```
In [ ]: float(23)
```

```
In [ ]: float(23, 56)
```

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

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

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

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

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

**from other datatypes to float except complex & 'string text'**

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

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

```
In [ ]: complex(10, 20, 30)
```

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

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

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

```
In [ ]: complex(3.2, 5.6)
```

**from other datatypes to complex & text string will not possible**

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

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

```
In [ ]: bool(3.2)
```

```
In [ ]: bool(0.0)
```

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

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

```
In [ ]: bool('ten')
```

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

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

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

## python type casting

- Variable completed
- Datatype complete
- Type casting

## STRING INDEXING & SLICING, ADVANCED SLICING

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

```
In [ ]: s9 = 'HELLOPYTHON'
```

```
In [ ]: s9
```

```
In [ ]: s9[:] # : slicing
```

```
In [ ]: for i in s9:  
    print(i)
```

```
In [ ]: s9
```

```
In [ ]: s9[11]
```

```
In [ ]: s9
```

```
In [ ]: s9[-4]
```

```
In [ ]: s9
```



```
In [ ]: s9[::-1]
```

```
In [ ]: s9
```

```
In [ ]: s9[0:5]
```

```
In [ ]: s9
```

```
In [ ]: s9[0:-1]
```

```
In [ ]: s9
```

```
In [ ]: s9[1:20]
```

```
In [ ]: s9
```

```
In [ ]: s9[:5]
```

```
In [ ]: s9
```

```
In [ ]: s9[5:]
```

```
In [ ]: s9
```

```
In [ ]: s9[0:11:2]
```

```
In [ ]: s9
```

```
In [ ]: s9[0:11:5]
```

```
In [ ]: s9[::-1]
```

```
In [ ]: s9
```

```
In [ ]: s9[::-3]
```

**string indexing & slicing we are completed & tomorrow we will be introduce python datastructure**

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

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

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

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

```
In [ ]: num1=20
        num2=30
        num3=40
        add=num1+num2+num3

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

## 21st Apr

```
In [ ]: i = 5.5
        type(i)
```

## LIST

```
In [11]: l = []
        l
```

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

```
In [12]: type(l)
```

```
Out[12]: list
```

```
In [13]: l.append(10)
```

```
In [14]: l
```

```
Out[14]: [10]
```

```
In [15]: l.append(20)
        l.append(30)
        l.append(40)
        l.append(50)
```

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

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

```
In [17]: l.append(60)
```

```
In [18]: l.append(70,80,90)
```

```
-----
TypeError                                Traceback (most recent call last)
Cell In[18], line 1
----> 1 l.append(70,80,90)

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

```
In [ ]: l
```

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

```
In [ ]: 1
```

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

```
In [ ]: l1 = 1.copy()
```

```
In [ ]: l1
```

```
In [19]: l1.append(2.3)
l1.append('nit')
l1.append(1+2j)
l1.append([1,2,3])
```

```
-----
NameError                                Traceback (most recent call last)
Cell In[19], line 1
----> 1 l1.append(2.3)
      2 l1.append('nit')
      3 l1.append(1+2j)

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

```
In [20]: print(l)
print(l1)
```

```
[10, 20, 30, 40, 50, 60]
```

```
-----
NameError                                Traceback (most recent call last)
Cell In[20], line 2
      1 print(l)
----> 2 print(l1)

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

```
In [21]: print(len(l))
print(len(l1))
```

```
6
```

```
-----
NameError                                Traceback (most recent call last)
Cell In[21], line 2
      1 print(len(l))
----> 2 print(len(l1))

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

```
In [22]: l == l1
```

```
-----
NameError                                Traceback (most recent call last)
Cell In[22], line 1
----> 1 l == l1

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

```
In [23]: l2 = 1.copy()
```

```
In [24]: l2
```

Out[24]: [10, 20, 30, 40, 50, 60]

In [25]: `l == l2`

Out[25]: True

In [26]: `print(l)  
print(l1)  
print(l2)`

[10, 20, 30, 40, 50, 60]

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[26], line 2  
      1 print(l)  
----> 2 print(l1)  
      3 print(l2)  
  
NameError: name 'l1' is not defined
```

In [27]: `l`

Out[27]: [10, 20, 30, 40, 50, 60]

In [28]: `l[:]`

Out[28]: [10, 20, 30, 40, 50, 60]

In [29]: `l[0]`

Out[29]: 10

In [30]: `l[0] = 100 # mutable`

In [31]: `l`

Out[31]: [100, 20, 30, 40, 50, 60]

In [32]: `l[-1] = 200`

In [33]: `l`

Out[33]: [100, 20, 30, 40, 50, 200]

In [34]: `l[3:]`

Out[34]: [40, 50, 200]

In [35]: `l`

Out[35]: [100, 20, 30, 40, 50, 200]

In [36]: `l[10]`

```
-----  
IndexError                                Traceback (most recent call last)  
Cell In[36], line 1  
----> 1 l[10]  
  
IndexError: list index out of range
```

In [37]: 1

Out[37]: [100, 20, 30, 40, 50, 200]

In [38]: l[10:]

Out[38]: []

In [39]: 1

Out[39]: [100, 20, 30, 40, 50, 200]

In [40]: l[:10]

Out[40]: [100, 20, 30, 40, 50, 200]

In [41]: 1

Out[41]: [100, 20, 30, 40, 50, 200]

In [42]: l[-1]

Out[42]: 200

In [43]: l2

Out[43]: [10, 20, 30, 40, 50, 60]

In [44]: l2.clear()

In [45]: l2

Out[45]: []

In [46]: del l2

In [47]: l2

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[47], line 1  
----> 1 l2  
  
NameError: name 'l2' is not defined
```

In [48]: l1

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[48], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [49]: l1[0:12:5]
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[49], line 1  
----> 1 l1[0:12:5]  
  
NameError: name 'l1' is not defined
```

```
In [50]: l1
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[50], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [51]: l1[0:11:3]
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[51], line 1  
----> 1 l1[0:11:3]  
  
NameError: name 'l1' is not defined
```

```
In [52]: 1
```

```
Out[52]: [100, 20, 30, 40, 50, 200]
```

```
In [53]: l1.index('nit')
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[53], line 1  
----> 1 l1.index('nit')  
  
NameError: name 'l1' is not defined
```

## 22nd

```
In [54]: print(l)  
         print(l1)
```

```
[100, 20, 30, 40, 50, 200]
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[54], line 2  
      1 print(l)  
----> 2 print(l1)  
  
NameError: name 'l1' is not defined
```

In [55]: l1[8]

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[55], line 1  
----> 1 l1[8]  
  
NameError: name 'l1' is not defined
```

In [56]: print(l1[8][0])  
print(l1[8][1])  
print(l1[8][2])

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[56], line 1  
----> 1 print(l1[8][0])  
      2 print(l1[8][1])  
      3 print(l1[8][2])  
  
NameError: name 'l1' is not defined
```

In [57]: l

Out[57]: [100, 20, 30, 40, 50, 200]

In [58]: l.count(100)

Out[58]: 1

In [59]: l1

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[59], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

In [60]: l1.count(10)

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[60], line 1  
----> 1 l1.count(10)  
  
NameError: name 'l1' is not defined
```

In [61]: l1

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[61], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [62]: l1.remove([1,2,3])
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[62], line 1  
----> 1 l1.remove([1,2,3])  
  
NameError: name 'l1' is not defined
```

```
In [63]: l1
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[63], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [64]: l1.remove(10)
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[64], line 1  
----> 1 l1.remove(10)  
  
NameError: name 'l1' is not defined
```

```
In [65]: l1
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[65], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [66]: l
```

```
Out[66]: [100, 20, 30, 40, 50, 200]
```

```
In [67]: l.append(25)
```

```
In [68]: l
```

```
Out[68]: [100, 20, 30, 40, 50, 200, 25]
```

```
In [69]: l.insert(2,25)
```

```
In [70]: l
```

```
Out[70]: [100, 20, 25, 30, 40, 50, 200, 25]
```



```
In [71]: l.pop()
```

```
Out[71]: 25
```

```
In [72]: l
```

```
Out[72]: [100, 20, 25, 30, 40, 50, 200]
```

```
In [73]: l1
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[73], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [74]: l1.pop()
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[74], line 1  
----> 1 l1.pop()  
  
NameError: name 'l1' is not defined
```

```
In [75]: l1
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[75], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [76]: l
```

```
Out[76]: [100, 20, 25, 30, 40, 50, 200]
```

```
In [77]: l.remove(100)  
l
```

```
Out[77]: [20, 25, 30, 40, 50, 200]
```

```
In [78]: l.pop(20)
```

```
-----  
IndexError                                Traceback (most recent call last)  
Cell In[78], line 1  
----> 1 l.pop(20)  
  
IndexError: pop index out of range
```

```
In [79]: l
```

```
Out[79]: [20, 25, 30, 40, 50, 200]
```

```
In [80]: l.pop(0)
```

Out[80]: 20

In [81]: 1

Out[81]: [25, 30, 40, 50, 200]

In [82]: 1

Out[82]: [25, 30, 40, 50, 200]

In [83]: l1

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[83], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

In [84]: len(1)

Out[84]: 5

In [85]: len(l1)

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[85], line 1  
----> 1 len(l1)  
  
NameError: name 'l1' is not defined
```

In [86]: l1

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[86], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

In [87]: l3 = l1.copy()

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[87], line 1  
----> 1 l3 = l1.copy()  
  
NameError: name 'l1' is not defined
```

In [88]: l3 == l1

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[88], line 1  
----> 1 l3 == l1  
  
NameError: name 'l3' is not defined
```

```
In [89]: l1.extend(l3)
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[89], line 1  
----> 1 l1.extend(l3)  
  
NameError: name 'l1' is not defined
```

```
In [90]: l1
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[90], line 1  
----> 1 l1  
  
NameError: name 'l1' is not defined
```

```
In [91]: l4 = [4,5,6]  
l4
```

```
Out[91]: [4, 5, 6]
```

```
In [92]: l7 = [1,2]  
l7
```

```
Out[92]: [1, 2]
```

```
In [93]: l7.extend(l4)
```

```
In [94]: l7
```

```
Out[94]: [1, 2, 4, 5, 6]
```

```
In [95]: l10 = [1,2,3]  
l10
```

```
Out[95]: [1, 2, 3]
```

```
In [96]: l10.reverse()
```

```
In [97]: l10
```

```
Out[97]: [3, 2, 1]
```

```
In [98]: l9 = [3,7,10, 1,100, 95]  
l9
```

```
Out[98]: [3, 7, 10, 1, 100, 95]
```

```
In [99]: l9.sort()
```

```
In [100... l9
```

```
Out[100... [1, 3, 7, 10, 95, 100]
```

```
In [101... 19.sort(reverse=True)
```

```
In [102... 19
```

```
Out[102... [100, 95, 10, 7, 3, 1]
```

## 23rd

```
In [104... 16 = ['z', 'x', 'b', 'd']  
16
```

```
Out[104... ['z', 'x', 'b', 'd']
```

```
In [105... 16.sort() # paramter tuning
```

```
In [106... 16
```

```
Out[106... ['b', 'd', 'x', 'z']
```

```
In [107... 16.sort(reverse=True) # hyperparameter tuning
```

```
In [108... 16
```

```
Out[108... ['z', 'x', 'd', 'b']
```

```
In [109... 16
```

```
Out[109... ['z', 'x', 'd', 'b']
```

```
In [112... for k in 16:  
    print(k)
```

```
z  
x  
d  
b
```

```
In [116... 'zz' in 16
```

```
Out[116... False
```

```
In [117... for i in enumerate(16):  
    print(i)
```

```
(0, 'z')  
(1, 'x')  
(2, 'd')  
(3, 'b')
```

```
In [118... 16
```

```
Out[118... ['z', 'x', 'd', 'b']
```

```
In [119... all(16)
```

Out[119...] True

In [120...] `any(l6)`

Out[120...] True

In [121...] `l6.append(0)`

In [124...] `l6`

Out[124...] `['z', 'x', 'd', 'b', 0]`

In [122...] `all(l6)`

Out[122...] False

In [123...] `any(l6)`

Out[123...] True

## list is completed today

In [125...] `t = ()`  
`t`

Out[125...] `()`

In [126...] `len(t)`

Out[126...] `0`

In [127...] `t1 = (10,20,30,40,50)`  
`t1`

Out[127...] `(10, 20, 30, 40, 50)`

In [128...] `t2 = ('a', 'z', 'm', 'n')`  
`t2`

Out[128...] `('a', 'z', 'm', 'n')`

In [129...] `t2.count('z')`

Out[129...] `1`

In [131...] `t2.index('a')`

Out[131...] `0`

In [133...] `t1`

Out[133...] `(10, 20, 30, 40, 50)`

```
In [134... t1[0]
```

```
Out[134... 10
```

```
In [139... t1[0] = 100
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[139], line 1  
----> 1 t1[0] = 100  
  
TypeError: 'tuple' object does not support item assignment
```

```
In [140... t1
```

```
Out[140... (10, 20, 30, 40, 50)
```

```
In [141... icic = (1234, '4th apr 1990', 1234, 'cizp346878')  
icic
```

```
Out[141... ('1234', '4th apr 1990', 1234, 'cizp346878')
```

```
In [142... icic[0] = 980
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[142], line 1  
----> 1 icic[0] = 980  
  
TypeError: 'tuple' object does not support item assignment
```

```
In [143... t1
```

```
Out[143... (10, 20, 30, 40, 50)
```

```
In [144... t3 = t1 * 3  
t3
```

```
Out[144... (10, 20, 30, 40, 50, 10, 20, 30, 40, 50, 10, 20, 30, 40, 50)
```

## tuple completed

## 24th (SET)

```
In [147... s = {}  
s
```

```
Out[147... {}
```

```
In [148... type(s)
```

```
Out[148... dict
```

```
In [150... s1 = set()  
type(s1)
```

```
Out[150... set
```

```
In [151... s1.add(10)
```

```
In [152... s1.
```

```
Out[152... {10}
```

```
In [153... s1.add(20)  
s1.add('nit')  
s1.add(True)  
s1.add(1+2j)  
s1.add(3.2)
```

```
In [154... s1
```

```
Out[154... {(1+2j), 10, 20, 3.2, True, 'nit'}
```

```
In [157... s2 = set()  
s2
```

```
Out[157... set()
```

```
In [159... s2.add(100)  
s2.add(10)  
s2.add(200)  
s2.add(9)
```

```
In [160... s2
```

```
Out[160... {9, 10, 100, 200}
```

```
In [161... s3 = set()  
s3
```

```
Out[161... set()
```

```
In [162... s3.add('z')  
s3.add('a')  
s3.add('m')  
s3.add('b')
```

```
In [163... s3
```

```
Out[163... {'a', 'b', 'm', 'z'}
```

```
In [164... print(s1)  
print(s2)  
print(s3)
```

```
{True, 3.2, (1+2j), 10, 'nit', 20}  
{200, 9, 10, 100}  
{'m', 'a', 'b', 'z'}
```

```
In [165... print(len(s3))
```

```
4
```

```
In [168... s3
```

```
Out[168... {'a', 'b', 'm', 'z'}
```

```
In [167... s3[0]
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[167], line 1  
----> 1 s3[0]  
  
TypeError: 'set' object is not subscriptable
```

```
In [169... s3[:]
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[169], line 1  
----> 1 s3[:]  
  
TypeError: 'set' object is not subscriptable
```

```
In [170... s4 = s3.copy()  
s4
```

```
Out[170... {'a', 'b', 'm', 'z'}
```

```
In [171... s3
```

```
Out[171... {'a', 'b', 'm', 'z'}
```

```
In [172... s3 == s4
```

```
Out[172... True
```

```
In [173... s4
```

```
Out[173... {'a', 'b', 'm', 'z'}
```

```
In [174... s4.clear()
```

```
In [175... s4
```

```
Out[175... set()
```

```
In [176... del s4
```

```
In [178... s4 = s2.copy()
```

```
In [179... s4
```

```
Out[179... {9, 10, 100, 200}
```



```
In [180... s4.pop()
```

```
Out[180... 200
```

```
In [181... s4
```

```
Out[181... {9, 10, 100}
```

```
In [182... s4.pop()
```

```
Out[182... 9
```

```
In [183... s3
```

```
Out[183... {'a', 'b', 'm', 'z'}
```

```
In [184... s1
```

```
Out[184... {(1+2j), 10, 20, 3.2, True, 'nit'}
```

```
In [185... s1.pop(0)
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[185], line 1  
----> 1 s1.pop(0)  
  
TypeError: set.pop() takes no arguments (1 given)
```

```
In [186... s1
```

```
Out[186... {(1+2j), 10, 20, 3.2, True, 'nit'}
```

```
In [187... s1.remove((1+2j))
```

```
In [188... s1
```

```
Out[188... {10, 20, 3.2, True, 'nit'}
```

```
In [190... for i in s4:  
          print(i)
```

```
10  
100
```

```
In [191... for i in enumerate(s4):  
          print(i)
```

```
(0, 10)  
(1, 100)
```

## set operation

```
In [192... A = {1,2,3,4,5}
          B = {4,5,6,7,8}
          C = {8,9,10}
```

```
In [194... A | B #UNION
```

```
Out[194... {1, 2, 3, 4, 5, 6, 7, 8}
```

```
In [195... B.union(C)
```

```
Out[195... {4, 5, 6, 7, 8, 9, 10}
```

```
In [196... A.union(B,C)
```

```
Out[196... {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

```
In [197... C | B | A
```

```
Out[197... {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

```
In [198... C
```

```
Out[198... {8, 9, 10}
```

```
In [204... C.update(B)
```

```
In [205... C
```

```
Out[205... {4, 5, 6, 7, 8, 9, 10}
```

```
In [206... print(A)
          print(B)
          print(C)
```

```
{1, 2, 3, 4, 5}
{4, 5, 6, 7, 8}
{4, 5, 6, 7, 8, 9, 10}
```

```
In [207... A & B
```

```
Out[207... {4, 5}
```

```
In [208... B.intersection(C)
```

```
Out[208... {4, 5, 6, 7, 8}
```

```
In [209... A & B & C
```

```
Out[209... {4, 5}
```

```
In [210... print(A)
          print(B)
          print(C)
```

```
{1, 2, 3, 4, 5}
{4, 5, 6, 7, 8}
{4, 5, 6, 7, 8, 9, 10}
```

In [211...] `A - B`

Out[211...] `{1, 2, 3}`

In [212...] `B - C`

Out[212...] `set()`

In [213...] `C - B`

Out[213...] `{9, 10}`

In [214...] `C.difference(A)`

Out[214...] `{6, 7, 8, 9, 10}`

## 25th|

In [215...] `print(A)`  
`print(B)`  
`print(C)`

`{1, 2, 3, 4, 5}`

`{4, 5, 6, 7, 8}`

`{4, 5, 6, 7, 8, 9, 10}`

In [217...] `A.difference(B)`

Out[217...] `{1, 2, 3}`

In [218...] `A.symmetric_difference(B)`

Out[218...] `{1, 2, 3, 6, 7, 8}`

In [221...] `s3`

Out[221...] `{'a', 'b', 'm', 'z'}`

In [222...] `s3.remove('x')`

```
-----  
KeyError                                Traceback (most recent call last)  
Cell In[222], line 1  
----> 1 s3.remove('x')  
KeyError: 'x'
```

In [223...] `s3.discard('x')`

In [224...] `s3.discard('z')`  
`s3`

Out[224...] `{'a', 'b', 'm'}`

# superset

# subset

# disjoint

```
In [225... a9 = {1,2,3,4,5,6,7,8,9}
b9 = {3,4,5,6,7,8}
c9 = {10,20,30,40}
```

```
In [226... b9.issubset(a9)
```

```
Out[226... True
```

```
In [227... a9.issuperset(b9)
```

```
Out[227... True
```

```
In [228... c9.issubset(b9)
```

```
Out[228... False
```

```
In [229... c9.isdisjoint(b9)
```

```
Out[229... True
```

```
In [230... c9.isdisjoint(a9)
```

```
Out[230... True
```

```
In [231... a8 = {1,2,3,4,5,6}
b8 = {7,8,9}
c8 = {10,20,30,40}
```

```
In [232... a8.issuperset(b8)
```

```
Out[232... False
```

```
In [233... b8.issubset(a8)
```

```
Out[233... False
```

```
In [234... c8.isdisjoint(b8)
```

```
Out[234... True
```

```
In [235... c8.isdisjoint(a8)
```

```
Out[235... True
```

# set we are completed

## DICTIONARY

```
In [237... d = {}  
type(d)
```

```
Out[237... dict
```

```
In [240... d = {1:'one', 2:'two', 3:'three', 4:'four'}
```

```
In [241... d
```

```
Out[241... {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

```
In [242... d.keys()
```

```
Out[242... dict_keys([1, 2, 3, 4])
```

```
In [243... d.values()
```

```
Out[243... dict_values(['one', 'two', 'three', 'four'])
```

```
In [244... d
```

```
Out[244... {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

```
In [245... d.items()
```

```
Out[245... dict_items([(1, 'one'), (2, 'two'), (3, 'three'), (4, 'four')])
```

```
In [246... len(d)
```

```
Out[246... 4
```

```
In [248... d[4]
```

```
Out[248... 'four'
```

```
In [249... d
```

```
Out[249... {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

```
In [250... d.get(1)
```

```
Out[250... 'one'
```

```
In [252... d.get('one') # wrong
```

```
In [253... d1 = d.copy()
```

```
In [254... d1
```

```
Out[254... {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

```
In [256... d1.pop(1)
```

```
Out[256... 'one'
```

```
In [257... d1
```

```
Out[257... {2: 'two', 3: 'three', 4: 'four'}
```

```
In [259... d1.popitem()
```

```
Out[259... (4, 'four')
```

```
In [260... d1.
```

```
Out[260... {2: 'two', 3: 'three'}
```

```
In [261... d
```

```
Out[261... {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

```
In [262... for i in d:
```

```
print(i)
```

```
1
2
3
4
```

```
In [264... for i in d:
```

```
print(i,':', d[i])
```

```
1 : one
2 : two
3 : three
4 : four
```

```
In [265... for i in d:
```

```
print(i,':', d)
```

```
1 : {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
2 : {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
3 : {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
4 : {1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

## Dict also we are completd

## Range

```
In [268... range(10)
```

Out[268... range(0, 10)

```
In [269... r = range(10)
r
```

Out[269... range(0, 10)

```
In [270... for i in r:
    print(i)
```

0  
1  
2  
3  
4  
5  
6  
7  
8  
9

```
In [272... list(r)
```

Out[272... [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

```
In [273... r1 = range(10,20)
r1
```

Out[273... range(10, 20)

```
In [274... list(r1)
```

Out[274... [10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

```
In [275... list(range(0,10))
```

Out[275... [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

```
In [276... list(range(10,20))
```

Out[276... [10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

```
In [277... list(range(10,100,10))
```

Out[277... [10, 20, 30, 40, 50, 60, 70, 80, 90]

```
In [278... list(range(10,100,10,5))
```

```
-----
TypeError                                Traceback (most recent call last)
Cell In[278], line 1
----> 1 list(range(10,100,10,5))

TypeError: range expected at most 3 arguments, got 4
```

In [ ]:

In [ ]: