```
Out[19]: (234, 'microsoft', 'accenture')
```

```
In [21]: icici[0]
```

```
Out[21]: 234
```

```
In [29]: t1 = (3,4,5,6,7)
         t1
```

```
Out[29]: (3, 4, 5, 6, 7)
```

```
In [31]: t1 * 3
```

```
Out[31]: (3, 4, 5, 6, 7, 3, 4, 5, 6, 7, 3, 4, 5, 6, 7)
```

```
In [33]: t1.index(4)
```

```
Out[33]: 1
```

```
In [35]: t1.index(4)
```

```
Out[35]: 1
```

```
In [37]: t1[2:]
```

```
Out[37]: (5, 6, 7)
```

```
In [39]: t1[:25]
```

```
Out[39]: (3, 4, 5, 6, 7)
```

SET

```
In [41]: s = {}
         s
```

```
Out[41]: {}
```

```
In [43]: type(s)
```

```
Out[43]: dict
```

```
In [47]: s1= set()
         s1
```

```
Out[47]: set()
```

```
In [49]: type(s1)
```

```
Out[49]: set
```

```
In [51]: l10 = list()
         type(l10)
```

```
Out[51]: list
```

```
In [55]: s2 = {11, 35, 20, 76, 97}
         s2
```

```
Out[55]: {11, 20, 35, 76, 97}
```

```
In [57]: s = {'m', 'z', 'r'}
         s
```

```
Out[57]: {'m', 'r', 'z'}
```

```
In [59]: s3 = {1, 'f', 12.9, (1 + 9j)}
         s3
```

```
Out[59]: {(1+9j), 1, 12.9, 'f'}
```

```
In [63]: s3.add(1000)
         s3
```

```
Out[63]: {(1+9j), 1, 1000, 12.9, 'f'}
```

```
In [67]: s5 = s3.copy()
         s5
```

```
Out[67]: {(1+9j), 1, 1000, 12.9, 'f'}
```

```
In [69]: s4 = {1, 'm', 3.3, 1+2j}
         len(s4)
```

```
Out[69]: 4
```

```
In [71]: s4[0]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[71], line 1
----> 1 s4[0]

TypeError: 'set' object is not subscriptable
```

```
In [77]: s3.add('o')
         s3
```

```
Out[77]: {(1+9j), 1, 1000, 12.9, 'f', 'o'}
```

```
In [79]: s3.pop()
```

```
Out[79]: 1
```

```
In [81]: s3
```

```
Out[81]: {(1+9j), 1000, 12.9, 'f', 'o'}
```

```
In [93]: s3 = {'m', 'o', 'r', 'z'}
         s3
```

```
Out[93]:  {'m', 'o', 'r', 'z'}

In [95]:  s3.pop()

Out[95]:  'o'

In [87]:  s3

Out[87]:  {'m', 'r', 'z'}

In [97]:  s2 = {3, 10, 20, 200}
          s2

Out[97]:  {3, 10, 20, 200}

In [103…  # s2.remove(200)
          s2

Out[103…  {3, 10, 20}

In [105…  s2.discard(20)

In [107…  s2

Out[107…  {3, 10}

In [109…  s2.discard(20)

In [111…  s2

Out[111…  {3, 10}

In [200…  s5 = {2,3,'nit', 1+2j, True, 45.8}
          s5

Out[200…  {(1+2j), 2, 3, 45.8, True, 'nit'}

In [202…  type(s5)

Out[202…  set

In [204…  2 in s5

Out[204…  True

In [206…  200 in s5

Out[206…  False

In [208…  s5

Out[208…  {(1+2j), 2, 3, 45.8, True, 'nit'}

In [210…  for i in s5:
              print(i)
```

```
True
2
3
nit
(1+2j)
45.8
```

In [212…

```
for i in enumerate(s5):
    print(i)
```

```
(0, True)
(1, 2)
(2, 3)
(3, 'nit')
(4, (1+2j))
(5, 45.8)
```

In [216…

```
s5.update([1,2,3])
s5
```

Out[216…

```
{(1+2j), 2, 3, 45.8, True, 'nit'}
```

In [220…

```
s5.update([100, 200, 300])
s5
```

Out[220…

```
{(1+2j), 100, 2, 200, 3, 300, 45.8, True, 'nit'}
```

In [224…

```
s5.remove(True)
s5
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[224], line 1
----> 1 s5.remove(True)
      2 s5

KeyError: True
```

In [226…

```
s6 = s5.copy()
```

In [228…

```
s6
```

Out[228…

```
{(1+2j), 100, 2, 200, 3, 300, 45.8, 'nit'}
```

set operation

In [123…

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

In [117…

```
A.union(B)
```

Out[117…

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

In [119…

```
A.union(B,C)
```

Out[119…

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

In [121…

```
A.union(B,C, 56)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[121], line 1
----> 1 A.union(B,C, 56)

TypeError: 'int' object is not iterable
```

In [125…

```
A | B
```

Out[125…

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

In [127…

```
A | B| C
```

Out[127…

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

In [129…

```
A & B
```

Out[129…

```
{4, 5}
```

In [131…

```
A.intersection(B)
```

Out[131…

```
{4, 5}
```

In [133…

```
A.intersection(C)
```

Out[133…

```
set()
```

In [135…

```
B.intersection(C)
```

Out[135…

```
{8}
```

In [137…

```
A.intersection(B, C)
```

Out[137…

```
set()
```

In [139…

```
A - B
```

Out[139…

```
{1, 2, 3}
```

In [ ]:

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

In [143…

```
A.symmetric_difference(B)
```

Out[143…

```
{1, 2, 3, 6, 7, 8}
```

In [230…

```
C - A
```

Out[230…

```
{8, 9, 10}
```

In [145…

```
B ^ C
```

```
Out[145…   {4, 5, 6, 7, 9, 10}
```

```
In [149…   A1 = {1,2,3,4,5,6,7,8,9}
           B1 = {3,4,5,6,7,8}
           C1 = {10,20,30,40}
```

```
In [151…   B1.issubset(A1)
```

```
Out[151…   True
```

```
In [153…   A1.issubset(B1)
```

```
Out[153…   False
```

```
In [155…   A1.issuperset(B1)
```

```
Out[155…   True
```

```
In [159…   B1.isdisjoint(A1)
```

```
Out[159…   False
```

```
In [161…   C1.isdisjoint(B1)
```

```
Out[161…   True
```

```
In [165…   A2 = {1,2,3,4,5,6,7,8,9}
           B2 = {13,14,15,16,17,18}
           C2 = {10,20,30,40}
```

```
In [167…   B2.isdisjoint(C2)
```

```
Out[167…   True
```

```
In [169…   B2.issubset(A2)
```

```
Out[169…   False
```

```
In [171…   A2.issuperset(B2)
```

```
Out[171…   False
```

```
In [173…   for i in enumerate(A):
               print(i)
```

```
(0, 1)
(1, 2)
(2, 3)
(3, 4)
(4, 5)
```

```
In [175…   list(enumerate(A))
```

```
Out[175…   [(0, 1), (1, 2), (2, 3), (3, 4), (4, 5)]
```

```
In [ ]:    # DICTIONARY
In [177…  d = {}
          type(d)

Out[177…  dict

In [179…  d1 = {1:'one', 2: 'two', 3: 'three'}
          d1

Out[179…  {1: 'one', 2: 'two', 3: 'three'}

In [181…  d1.keys()

Out[181…  dict_keys([1, 2, 3])

In [183…  d1.values()

Out[183…  dict_values(['one', 'two', 'three'])

In [185…  d1.items()

Out[185…  dict_items([(1, 'one'), (2, 'two'), (3, 'three')])

In [187…  len(d1.items())

Out[187…  3

In [189…  d1[1]

Out[189…  'one'

In [191…  d1[3]

Out[191…  'three'

In [193…  d1

Out[193…  {1: 'one', 2: 'two', 3: 'three'}

  In [ ]:

  In [ ]:

  In [ ]:

  In [ ]:
```