- pop()

```
a = {1,2,3,4,5,6}
r = a.pop()
Print(a,r)
```

compiler deletes an element itself

Olp ⇒ {2,3,4,5,6} 1

- remove()

```
a = {10,20,30,40,50}
r = a.remove(30)
Print(a,r)
```

r = a.remove(300) ⇒ KeyError

Olp ⇒ {50,20,40,10} None

- discard()

```
a = {5,6,7,8,9}
r = a.discard(a)
Print(a,r)
```

r = a.discard(300) ⇒ {5,6,7,8,9} None

Olp ⇒ {5,6,7,8} None

→ Session - 11

Problem statement -

```
d = {'k1':[1,2,{'k2':['this is tricky',
    {'tough':[1,2,['hello']]}]}]}
```

Try to extract 'hello' from this data structure.

```
d = {'k1':[1,2,{'k2':['this is tricky',
    {'tough':[1,2,['hello']]}]}]}
print(d['k1'][2]['k2'][1]['tough'][2][0])
```

Olp = hello

extract 'tough'.

```
print(d['k1'][2]['k2'][1])
```

Olp = 'tough': [1,2,['hello']]

How to add elements to a set ?

```
a = {1,2,3,4,4,5,2,6,2,5}
a.add(7)
Print(a)
```

Olp ⇒ {1,2,3,4,5,6,7}

```
a = {1,2,3,3,2,1}
a.add(2)
print(a)
```

Olp ⇒ {1,2,3}

String Data type -

- immutable data type (Can't be added, updated or deleted).
- supports slicing

Ex - a = "neha reddy"
(indexes 0 1 2 3 4 5 6 7 8 9)

Print(a[5]) Olp ⇒ r

Print(a[4]) Olp ⇒ - Empty space

| n | e | h | a | - | r | e | d | d | y |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| -10 | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 |

Slicing -

```
a = "nehal reddy"
```

+Ve slicing `print(a[0:4])` (start index, end index) End index is not included

Olp ⇒ neha

-Ve slicing

```
a = "nehal reddy"
print(a[-5:-1])
```

→ End index not included

Olp ⇒ redd

s[100] ⇒ Error
Index out of range

a[:] ⇒ neha reddy.
a[5:] ⇒ reddy → by default last index
a[:4] ⇒ neha → by default start index

s =

| n | e | h | a | | r | e | d | d | y |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| -10 | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 |

s[0:9:2] → stride or step size
olp ⇒ n h ed (0,2,4,6,8)

- A string is a group of characters

Reversing a string -

```
a = "abcdef"
print(a[::-1])
```

olp ⇒ f e d c b a

sep end
arguments
flush

"neha" + " " + "reddy" ⇒ neha reddy

"neha" * 2 ⇒ nehaneha

```
print(1, 2, sep=' ', end = '\n')
```

space
newline character

olp ⇒ 1 2

Escape Sequences - \t, \n

String formatting -

```
a=10
b=20
print("the value of a is {} and the value of b is {}".format(a,b))
```

olp ⇒ the value of a is 10 and the value of b is 20

```
x=100
y=500
print("the value of x is {1} and the value of y is {0}".format(y,x))
```

olp ⇒ the value of x is 100 and the value of y is 500.

Key-value pair -

```
sample={'num1':100, 'num2':200}
print("the value of num1 is {num1} and the value of num2 is {num2}".format(num1=100, num2=200))
```

olp ⇒ the value of num1 is 100 and the value of num2 is 200.

f-strings -

```
sample={"num1":100, "num2":200}
print(f"the value of num1 is {sample['num1']} and the value of num2 is {sample['num2']}")
```

olp ⇒ the value of num1 is 100 and the value of num2 is 200

String methods :

1) split() method
2) Join() method

```
s = "Python, C++, Java, Go"
s_list = s.split(',')
print(s_list)
```

olp ⇒ ['Python', 'C++', 'Java', 'Go']

```
s = "Python C++ Java Go"
s_list = s.split()
print(s_list)
```

olp ⇒ ['Python', 'C++', 'Java', 'Go']