join() method

This method is used to join collection of strings into one string using separator

Syntax:
Variable-name=separator.joint(iterable)

separator : it is string which is added between two string
Iterable: it is collection which consist of more than one string

Example:

```
>>> A=["naresh","ramesh","kishore"]
>>> s1=" ".join(A)
>>> print(A)
['naresh', 'ramesh', 'kishore']
>>> print(s1)
naresh ramesh kishore
>>> B=["a","b","c","d","e"]
>>> s2=",".join(B)
>>> print(B)
['a', 'b', 'c', 'd', 'e']
>>> print(s2)
a,b,c,d,e
>>> C=["10","20","30"]
>>> s3=":".join(C)
>>> print(s3)
10:20:30
```

Example:

```
# Reverse Words in a Given String in Python
# input : python is simple language
# output: language simple is python

s1="python is simple language"
A=s1.split()
print(A)
A=A[::-1]
print(A)
```

```
s2=" ".join(A)
print(s2)
```

Output

```
['python', 'is', 'simple', 'language']
['language', 'simple', 'is', 'python']
language simple is python
```

Example:

```
# Python program to print even length words in a string

str1="python is simple language than CPP"
A=str1.split()
for s in A:
    if len(s)%2==0:
        print(s)
```

Output

```
 python
is
simple
language
than
```

Example:

```
    # Python – Uppercase Half String

str1="python"
half1=str1[:len(str1)//2]
half2=str1[len(str1)//2:]
str2=half1.upper()+half2.lower()
str3=half1.lower()+half2.upper()
print(str1)
print(str2)
print(str3)
```

Output

```
python
PYThon
```

python

Example:

```
# Python program to capitalize the first and last character of
# each word in a string

str1="python programming"
A=str1.split()
C=[]
for word in A:
    B=list(word)
    B[0]=B[0].upper()
    B[-1]=B[-1].upper()
    s="".join(B)
    C.append(s)


str2=" ".join(C)
print(str2)
```

Output

```
['python', 'programming']
['PythoN', 'ProgramminG']
PythoN ProgramminG
```

String strip methods

These methods are used to removing leading or trailing characters from string.

1. lstrip()
2. rstrip()
3. strip()

lstrip() : This removing leading characters from string (the characters found at left side)

Syntax:

Variable-name=stringname.lstrip(chars=' ')

Default character removed from left side of string is space

```
>>> str1="  nit"
>>> str2=str1.lstrip()
>>> print(len(str1))
5
>>> print(len(str2))
3
>>> print(str1)
  nit
>>> print(str2)
nit
>>> str3="  abc xyz pqr"
>>> str4=str3.lstrip()
>>> print(str3)
  abc xyz pqr
>>> print(str4)
abc xyz pqr
>>> str5="***nit"
>>> str6=str5.lstrip("*")
>>> print(str5)
***nit
>>> print(str6)
nit
>>> str7="$%***nit"
>>> str8=str7.lstrip("*$%")
>>> print(str7)
$%***nit
>>> print(str8)
nit
```

rstrip() : This method is used to remove trailing characters (The characters found at right side)

Syntax:
variable-name=string-name.rstrip(chars=' ')

by default rstrip remove space characters found at right side.

```
>>> s1="nit    "
>>> s2="nit"
>>> s1==s2
False
>>> s1==s2.rstrip()
False
>>> s1.rstrip()==s2
True
>>> s3=s1.rstrip()
>>> print(len(s1),len(s2))
7 3
>>> print(len(s3))
3
>>> print(s1,s2,s3,sep="\n")
nit
nit
nit
>>> s4="nit****"
>>> s5=s4.rstrip("*")
>>> print(s4)
nit****
>>> print(s5)
nit
>>> s6="nit***$$##@@***"
>>> s7=s6.rstrip("*#@$")
>>> print(s6)
nit***$$##@@***
>>> print(s7)
Nit
```

strip() : This method is used to remove leading and trailing characters from string. Default characters removed is space characters

Syntax:
variable-name=string-name.strip(chars=' ')

```
>>> str1="  nit   "
>>> str2="nit"
>>> str1==str2
False
```

```
>>> str1.strip()==str2
True
>>> str3=str1.strip()
>>> print(str1)
  nit
>>> print(str3)
nit
>>> s1="***nit****"
>>> s2=s1.strip("*")
>>> print(s1)
***nit****
>>> print(s2)
nit
>>> s3="***$$nit&&&**%%"
>>> s4=s3.strip("*$&%")
>>> print(s3)
***$$nit&&&**%%
>>> print(s4)
nit
>>> s5="www.nareshit.com"
>>> s6=s5.strip("w.com")
>>> print(s5)
www.nareshit.com
>>> print(s6)
nareshit
```

Example:

```
# Login Application
uname=input("UserName :")
pwd=input("Password :")

if uname.strip()=="nit" and pwd.strip()=="n123":
    print("Welcome")
else:
    print("Invalid username or password")
```

Output

```
UserName :nit
Password :n123
Welcome
```

UserName : nit
Password :n123
Invalid username or password

UserName :nit
Password :n123
Welcome

UserName : nit
Password : n123
Welcome

String Searching methods

1. startswith
2. endswith

These methods returns boolean values(True/False)

startswiths(prefix): This method find prefix at the beginning of the string if found it returns True else False

```
>>> s1="naresh"
>>> s1.startswith("n")
True
>>> s1.startswith("x")
False
```

Example:

```
names=["naresh","ramesh","kishore","raman","rajesh","kishore"]
for name in names:
    b=name.startswith("r")
    if b:
        print(name)


for name in names:
    b=name.startswith(("r","k"))
    if b:
        print(name)
```

Output
ramesh
raman
rajesh
ramesh
kishore
raman
rajesh
kishore

endswith(suffix): This method find given suffix at the end of the string, if found return True else False

Syntax: variable-name=stringname.endswith(suffix)

```
>>> s1="python"
>>> s1.endswith("n")
True
>>> email="naresh@nareshit.in"
>>> email.endswith("in")
True
>>> email.endswith("com")
False
```

Example:

```
names=["naresh","ramesh","kishore","raman","rajesh","kiran"]
for name in names:
    b=name.endswith("h")
    if b:
        print(name)

print("===========")
for name in names:
    b=name.endswith(("h","n"))
    if b:
        print(name)
```

Output
naresh

ramesh
rajesh
==========
naresh
ramesh
raman
rajesh
kiran

Partition methods

String data type provides 2 partition methods

1. partition
2. rpartition

partition methods returns tuple contains 3 values

1. value before separator
2. separator
3. value after separator

```
>>> s1="a,b,c,d,e"
>>> t1=s1.partition(",")
>>> print(t1)
('a', ',', 'b,c,d,e')
>>> t2=s1.rpartition(",")
>>> print(t2)
('a,b,c,d', ',', 'e')
```

Alignment methods

1. ljust()
2. rjust()
3. center()