OPRATIONS ON STRINGS

Repeating the Strings

Example:

```
str="Good Afternoon"
print(str*2)
s=str[2:7]*3 #repeat 2nd to 6th characters for 3 times
print(s)
```

Concatenation of Strings

- **'+' Operator**
- If we want to concatenate strings in different lines, we can use **parenthesis**.

Example:

```
s1='Python'
s2="Lecture"
s3=s1+s2 #concatenate s1 and s2
print(s3)

s=('Hello'
   'World')
print(s)
```

Iterating through a String

Example:

```
#iterating through a string
count=0
for letter in 'Hello World':
   if(letter=='l'):
      count=count+1
print(count,'letters found')
```

String Membership Test (Checking Membership)

- The operator 'in' and 'not in' make case sensitive comparisons.

Example:

```
'a' in 'program'
True
'at' not in 'battle'
False
```

Comparing Strings

- Relational Operators
- They return Boolean value.

Example:

```
s1='Hello'
s2="World"
if(s1==s2):
   print('Both are same')
else:
   print('Not same')

s1='Hello'
s2="World"
if(s1<s2):
   print('s1 less than s2')
else:
   print('s1 graeter than or equal to s2')
```

Removing Spaces from a String

- The **rstrip() method** removes the spaces which are at the right side of the string.
- The **lstrip() method** removes the spaces which are at the left side of the string.
- The **strip() method** removes the spaces from both the sides of the strings.
- These methods do not remove spaces which are in the middle of the string.

Example:

```
name='  Vaibhav Sharma  '
print(name.strip()) #removes spacs from both sides
print(name.rstrip()) #removes spacs at right
print(name.lstrip()) #removes spacs at left
```

Finding Sub Strings

- The **find()** and **index()** methods search for the sub string from the beginning of the main string.

- The **rfind()** and **rindex()** methods search for the sub string from right to left.
- The find() method returns -1 if the sub string is not found in the main string.
- The index() method returns **'Value Error'** exception if the sub string is not found.

Syntax: mainstring.find(substring, beginning, ending)

- The same format is used for other methods also.

Example: #To find the first occurrence of sub string in a main string

```
        str=input("Enter the string : ")
sub=input("Enter the sub-string : ")

n=str.find(sub,0,len(str))

if n==-1:
   print("Sub string not found")
else:
   print("Sub string found at position :",n+1)
```

Counting Substrings in a String

- Count() to count the number of occurrences of a sub string.
- **Syntax: stringname.count(substring)**
- **Stringname.count(substring, beg, end)**

Example:

```
str='Good Morning'
n=str.count('Good')
print(n)

n1=str.count('o',0,5)
print(n1)

n1=str.count('o',0,len(str))
print(n1)
```

Replacing the String with another String

- **Syntax: stringname.replace(old,new)**

Example:

```
oldstring='I like Maggi'
```

```
print(oldstring)
newstring=oldstring.replace('like','love')
print(newstring)
```

Splitting & Joining Strings

- **Split()**
- Pieces are returned as a list.
- **Syntax: str.split(‘,’)**

Example:

```
word="Hello Good Morning"
print(word.split(' '))
print(word.split('o'))

#To accept and display a group of numbers
str=input('Enter numbers separated by space:')
lst=str.split(" ")

for i in lst:
   print(i)
```

- **Join()**
- You can add any character into the string.
- **Syntax: separator.join(str)**

Example:

```
str = ('one','two','three')
str1 = "-".join(str)
print(str1)

str = ('one','two','three')
sep=":"
str1 = sep.join(str)
print(str1)
```

Changing Case of the String (Changing Upper and Lower Case Strings)

- Upper(), lower(), swapcase(), title().

Example:

```
str="This is the python lecture"
print(str.upper())
print(str.lower())
print(str.swapcase())
print(str.title())
print(str.capitalize())
```

Changing or Checking Starting and Ending of a String

- Startswith () and endswith()
- **Syntax: str.startswith(substring)**
    - **Str.endswitg(substrion)**

Example:

```
str='This is Python Lab Session'
print(str.startswith('This'))
print(str.endswith('Session'))
print(str.endswith('Python'))
```

String Testing Methods

-True or False

- Isdigit()
- Isalnum()
- Isalpha()
- Islower()
- Isupper()
- Istitle()
- Isspace()

Reversing String

```
string="12345"
print(''.join(reversed(string)))
```

BUILT-IN FUNCTIONS TO WORK WITH PYTHON

- Enumerate() returns an enumerate object.
- It contains the index and value of all the items in the strings pairs.
- Len() method

Example:

```
str='Hello'
#enumerate()
list_enumerate=list(enumerate(str))
print('list(enumerate(str)', list_enumerate)
#character count
print('len(str)',len(str))
```

Python String Formatting

- Presenting a string in a clearly understandable manner.
- The replacement fields are denoted by curly braces { } that contain names or indexes.
- These names or indexes represent the order of the values.

Syntax: 'format string with replacement fields'.format(values)

Example:

```
id=40
name='Ram'
sal=30000.75

str='{},{},{}'.format(id,name,sal)
print(str)

str='{}-{}-{}'.format(id,name,sal)
print(str)
```

Escape Sequence

```
print("He said,"What's there?") # syntax Error

#Using triple quotes
print('''He said,"What's there?"''')

#escaping single quotes
print('He said,"What\'s there?"')

#escaping double quotes
print("He said,\"What's there?\"")
```

Raw string to ignore escape sequence

- R or r in front of the string.

Sorting Strings

- Sort() and sorted() method
- **Syntax: str.sort()**
- **Syntax: Str1.sorted()**

Example:

```
#sorting a group of strings
str=[]
n=int(input('How many strings ?'))

#append strings to str array
for i in range(n):
   print('Enter Strings:', end='')
   str.append(input())

#For sorting
str.sort()
str1=sorted(str)

#display
print('Sorted list:')
for i in str1:
   print(i)
```

Searching in the strings

```
For I in range(len(str)):
        If s==str[i]: #if s is matching with str[i] then found.
```

Some other Examples(IDLE)

```
"Hello".lower()
"HEllo".upper()
"This will split all words into a list".split()
' '.join(['Hello','Good','Morning'])
"Good Morning".find('oo')
'Hello Everone'.replace('Hello','Bye')
```