



Chapter 11  
Strings

Computer Science  
Class XI ( As per CBSE Board)

Visit : [python.mykvs.in](http://python.mykvs.in) for regular updates

# String



String is a sequence of characters, which is enclosed between either single ( ' ') or double quotes ( " " ), python treats both single and double quotes same.





## Creating String

Creation of string in python is very easy.

e.g.

```
a='Computer Science'
```

```
b="Informatics Practices"
```

## Accessing String Elements

e.g.

```
str='Computer Sciene'  
print('str-', str)  
print('str[0]-', str[0])  
print('str[1:4]-', str[1:4])  
print('str[2:]-', str[2:])  
print('str *2-', str *2 )  
print("str +'yes'-'", str +'yes')
```

**OUTPUT**

```
('str-', 'Computer Sciene')  
( 'str[0]-', 'C')  
( 'str[1:4]-', 'omp')  
( 'str[2:]-', 'mputer Sciene')  
( 'str *2-', 'Computer  
ScieneComputer Sciene')  
("str +'yes'-'", 'Computer  
Scieneeyes')
```

# String



## Iterating/Traversing through string

Each character of the string can be accessed sequentially using for loop.

e.g.

```
str='Computer Sciene'  
for i in str:  
    print(i)
```

OUTPUT



C  
o  
m  
p  
u  
t  
e  
r  
  
S  
c  
i  
e  
n  
e

## String comparison

We can use ( > , < , <= , >= , == , != ) to compare two strings. Python compares string lexicographically i.e using ASCII value of the characters.

Suppose you have str1 as "Maria" and str2 as "Manoj" . The first two characters from str1 and str2 ( M and M ) are compared. As they are equal, the second two characters are compared. Because they are also equal, the third two characters ( r and n ) are compared. And because 'r' has greater ASCII value than 'n' , str1 is greater than str2 .

e.g.program

	<u>OUTPUT</u>
<code>print("Maria" == "Manoj")</code>	False
<code>print("Maria" != "Manoj")</code>	True
<code>print("Maria" &gt; "Manoj")</code>	True
<code>print("Maria" &gt;= "Manoj")</code>	True
<code>print("Maria" &lt; "Manoj")</code>	False
<code>print("Maria" &lt;= "Manoj")</code>	False
<code>print("Maria" &gt; "")</code>	True



## Updating Strings

String value can be updated by reassigning another value in it.

e.g.

```
var1 = 'Comp Sc'  
var1 = var1[:7] + ' with Python'  
print ("Updated String :- ",var1 )
```

OUTPUT

```
('Updated String :- ', 'Comp Sc with Python')
```

## String Special Operators

e.g.

a="comp"

B="sc"

Operator	Description	Example
+	<b><u>Concatenation</u></b> – to add two strings	a + b = comp sc
*	Replicate same string multiple times ( <b><u>Repetition</u></b> )	a*2 = compcomp
[]	Character of the string	a[1] will give o
[ : ]	Range Slice –Range string	a[1:4] will give omp
in	<b><u>Membership</u></b> check	p in a will give 1
not in	Membership check for non availability	M not in a will give 1
%	Format the string	

`print ("My Subject is %s and class is %d" % ('Comp Sc', 11))`



## Format Symbol

**%s** -string conversion via `str()` prior to formatting

**%i** -signed decimal integer

**%d** -signed decimal integer

**%u** -unsigned decimal integer

**%o** -octal integer

**%x** -hexadecimal integer (lowercase letters)

**%X** -hexadecimal integer (UPPERcase letters)

**%e** -exponential notation (with lowercase 'e')

**%E** -exponential notation (with UPPERcase 'E')

**%f** -floating point real number

**%c** -character

**%G** -the shorter of **%f** and **%E**





## Triple Quotes

It is used to create string with multiple lines.

e.g.

```
Str1 = """This course will introduce the learner to text  
mining and text manipulation basics. The course begins  
with an understanding of how text is handled by python"""
```

## String functions and methods

a="comp"

b="my comp"

Method	Result	Example
len()	Returns the length of the string	r=len(a) will be 4
str.capitalize()	To capitalize the string	r=a.capitalize() will be "COMP"
str.title()	Will return title case string	
str.upper()	Will return string in upper case	r=a.upper() will be "COMP"
str.lower()	Will return string in lower case	r=a.lower() will be "comp"
str.count()	will return the total count of a given element in a string	r=a.count('o') will be 1
str.find(sub)	To find the substring position(starts from 0 index)	r=a.find('m') will be 2
str.replace()	Return the string with replaced sub strings	r=b.replace('my','your') will be 'your comp'

## String functions and methods

a="comp"

Method	Result	Example
<code>str.index()</code>	Returns index position of substring	<code>r=a.index('om')</code> will be 1
<code>str.isalnum()</code>	String consists of only alphanumeric characters (no symbols)	<code>r=a.isalnum()</code> will return True
<code>str.isalpha()</code>	String consists of only alphabetic characters (no symbols)	
<code>str.islower()</code>	String's alphabetic characters are all lower case	
<code>str.isnumeric()</code>	String consists of only numeric characters	
<code>str.isspace()</code>	String consists of only whitespace characters	
<code>str.istitle()</code>	String is in title case	
<code>str.isupper()</code>	String's alphabetic characters are all upper case	

## String functions and methods

a="comp"

Method	Result	Example
<code>str.lstrip(char)</code> <code>str.rstrip(char)</code>	Returns a copy of the string with leading/trailing characters removed	<code>b='**comp';</code> <code>r=b.lstrip()</code> will be 'comp'
<code>str.strip(char)</code>	Removes specific character from leading and trailing position	
<code>str.split()</code>	Returns list of strings as splitted	<code>b='my comp';</code> <code>r=b.split()</code> will be ['my','comp']
<code>str.partition()</code>	Partition the string on first occurrence of substring	<code>b='my comp';</code> <code>r=b.partition('comp')</code> will be ['my','comp']



## #Python Program to calculate the number of digits and letters in a string

```
string=raw_input("Enter string:")
count1=0
count2=0
for i in string:
    if(i.isdigit()):
        count1=count1+1
    count2=count2+1
print("The number of digits is:")
print(count1)
print("The number of characters is:")
print(count2)
```

## Searching for Substrings

E.g. program

```
s = "welcome to python"  
print(s.endswith("thon"))  
print(s.startswith("good"))  
print(s.find("come"))  
print(s.find("become"))  
print(s.rfind("o"))  
print(s.count("o"))
```

OUTPUT

```
True  
False  
3  
-1  
15  
3
```

METHOD NAME	METHODS DESCRIPTION:
<code>endswith(s1: str): bool</code>	Returns True if strings ends with substring s1
<code>startswith(s1: str): bool</code>	Returns True if strings starts with substring s1
<code>count(substring): int</code>	Returns number of occurrences of substring the string
<code>find(s1): int</code>	Returns lowest index from where s1 starts in the string, if string not found returns -1
<code>rfind(s1): int</code>	Returns highest index from where s1 starts in the string, if string not found returns -1