Computational Thinking and Programming - 1

Python - Strings

Definition

A Python string is a **sequence of characters stored in contiguous memory locations**.It is an immutable data type.

String is a sequence which is made up of one or more UNICODE characters. Here the character can be a letter, digit, whitespace or any other symbol. A string can be created by enclosing one or more characters in single, double or triple quote.

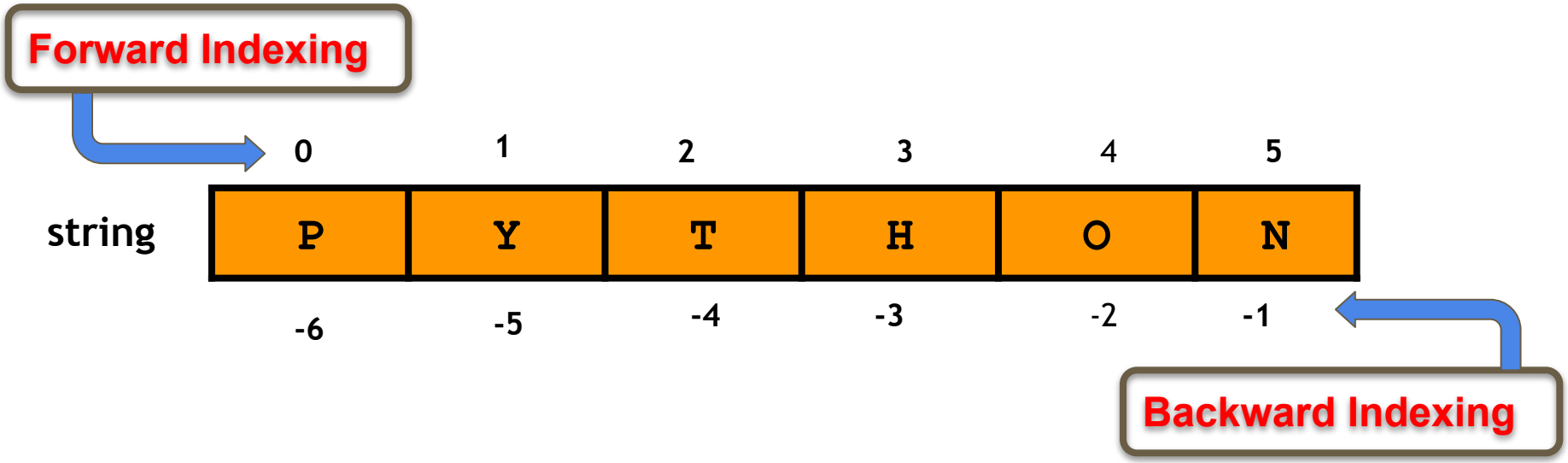
Each character can be individually accessed using its index. These characters can be accessed using either forward indexing or backward indexing.

You can access any character as **<stringname>[index].**

The index can be a forward index or a backward index.

- **0, 1, 2,.... In the forward direction**
- **-1, -2, -3….. In the backward direction**

Indexing



Traversing a string

Individual character of a string can be accessed using the method of indexing. Each character of a string is referred by an integer which is known as Index or subscript. The index or subscript starts from 0 and must be an integer.

Traversing a string means accessing all the elements of the string one after another by using the subscript or index where the index or subscript is given within [ ] with the stringname.

The index used to access character of a string can be positive or negative integers.

- Positive subscript helps in accessing the string from the beginning
- Negative subscript helps in accessing the string from end
- Subscript 0 or -n (where n is the length of the string) displays the first character, 1 or -(n-1) displays the second character and so on.

Traversing a string

String (str)

0	1	2	3	4	5	6	7
c	o	m	p	u	t	e	r
-8	-7	-6	-5	-4	-3	-2	-1

print(str[0])	Displays first character from beginning
print(str[7])	Displays last character
print(str[-8])	Displays first character from begining
print(str[-4])	Displays 4th character from the end

Assigning a string

String is a collection of ASCII/Unicode characters. Its content can represented within a set of single or double quotes.

```
Name='Amar Shah'
City="Hyderabad"
Place="King's Palace"
Title='The Dream of "Doll" uncovered'
print(Name,"Lived in the", Place,"at", City)
print("His best book was entitled as",Title)
```

Output

```
Amar Shah Lived in the King's Palace at Hyderabad
His best book was entitled as The Dream of "Doll" uncovered
```

Assigning a string

String can also be created using Escape sequence characters as a part of the string. The following example shows the use of \n and \t escape sequence characters for tab and new line

```
message = "Programming Language:\t Python\n Developed
by Guido Van Rossum"

print(message)
```

Output

```
Programming Language:    Python
 Developed by Guido Van Rossum
```

Assigning a string

Python does not support a character data type, these are treated as strings of length one.

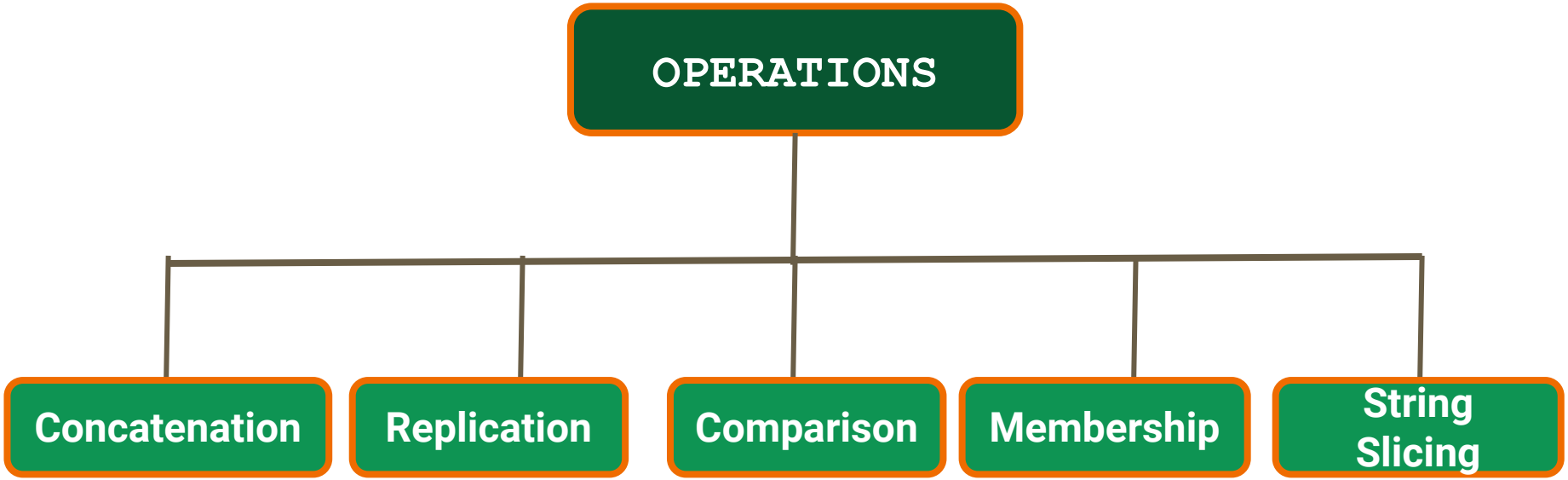
For example :

```
>>grade="A"
>>ch="*"
```

A string without any character inside is known as Empty String. When an Empty string is displayed using print(), a blank space gets displayed.

```
>>ch=""
>>print(ch)
''
```

Operations on Strings



Concatenation

The + operator takes two strings as operands and results a new string by joining the two operand strings.

Syntax : **string literal 1/object 1+string literal 2/object 2**

Example :

```
>>a="Computer"
>>b='Science'
>>print(a+b)
```

Output

```
ComputerScience
```

Replication

The * operator as a replication operator needs two operand i.e a string and an integer. Python replicates the string operand the given number of times.

Syntax : **string literal /object * integer data item**

Example :

```
>> a="Computer"
>>print(a*3)
```

Output

```
ComputerComputerComputer
```

Comparison

All relational and comparison operators (==, >, <, <=, >=, !=) can be used to compare two strings .

The comparison will be done on the basis of ASCII value of the character. Python compares first element of each string. If they are the same ,it goes on to the next element and so on .

Example : If there are three strings ,

s1="science"

s2="science"

s3="Science"

>>> s1==s2 True	>>> s1>s3 True	>>> s1==s3 False

Common Characters and Ordinal Values

CHARACTERS	ORDINAL VALUES
0 to 9	48 to 57
“A” to “Z”	65 to 90
“a” to “z”	97 to 122

Python provides a built in function ord() that takes in a single character and returns the ordinal value.

Syntax : ord(<single-character>)

```
>>ord("a")

97
```

ch=’a’
ord(ch)

Membership

The operators in and not in are used to check whether a string exists in the given string or not. The operands of in and not in operators are strings.

Syntax : <string1> in <string2>

<string1> not in <string2>

Example :

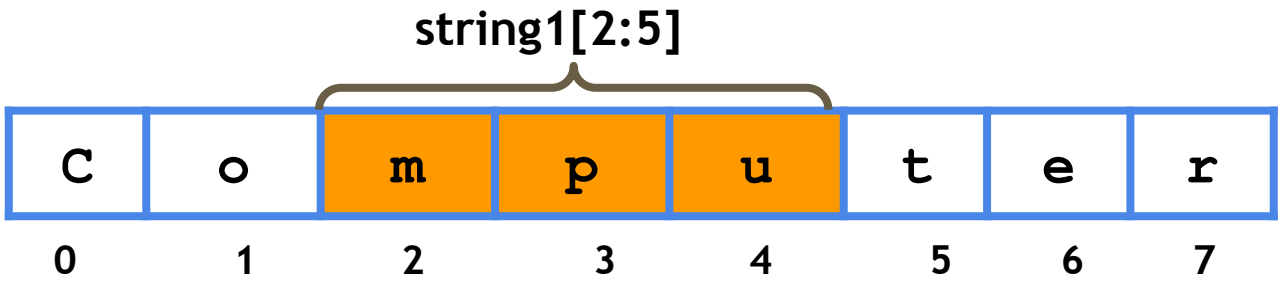
>>"t" in "technology" True	>>"s" in "image" False
>>"t" not in "first" False	>>"m" not in "operate" True

String Slicing

String slicing refers to the part of the string or a subset of a string.This subset of a string (the substring) is termed as slice. To extract the slice two indexes are given in square bracket separated by a colon (:).

Syntax : string_name [start : end : step]
where start, end and step are integers
start represents the starting index of string
end denotes the end index of string which is not inclusive
step denotes the distance between the elements to be sliced

Example :



String Slicing

Python will return a slice of the substring by returning the character falling between the **start** and **end-1** position

However , while slicing string, start, end and step can be omitted.

- When **start** position is omitted, **Python considers it to be 0.**
- When **end** position is omitted, **Python will extract the string starting from start position till the end of the string.**
- When both are omitted, Python will extract the entire string.
- When step is omitted, **Python assigns it as 1**
- When start value >stop value, step should be negative

String Slicing

The start and end values can be –ve integers in which Python counts from the right hand side.

Example :

```
>>> str= "Computer Science"
>>> print(str[-3:])                nce
>>> print(str[:-3])                Computer Scie
>>>print(str[-6:-1])               cienc
>>> print(str[-10:-2:2])           e ce
>>> print(str[-2:-15:-2])          cec eum #when start>stop,
                                   step should be -ve
>>> str1=str[-1:-17:-1]            'ecneicS retupmoC'
                                   #Reverse of str stored in str1
```

String Slicing

Index out of bound causes errors with strings but slicing a string outside the bounds does not cause error

Example:

h	e	l	l	o
0	1	2	3	4
-5	-4	-3	-2	-1

s="hello"

print(s[5]) -> will cause error as 5 is invalid index-out-of-bound

But, print(s[4:8]) -> valid

print(s[5:10]) -> valid NO OUTPUT ... but also not showing any error msg

String Slicing

	0	1	2	3	4	5	6	7
String (str)	C	o	m	p	u	t	e	r
	-8	-7	-6	-5	-4	-3	-2	-1

Output

print(str[3:7])	pute
print(str[4:])	uter
print(str[:6])	Comput
print(str[1::2])	optr
print(str[-3:])	ter

Answer the following :

String (str)

C	o	m	p	u	t	e	r		S	c	i	e	n	c	e
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

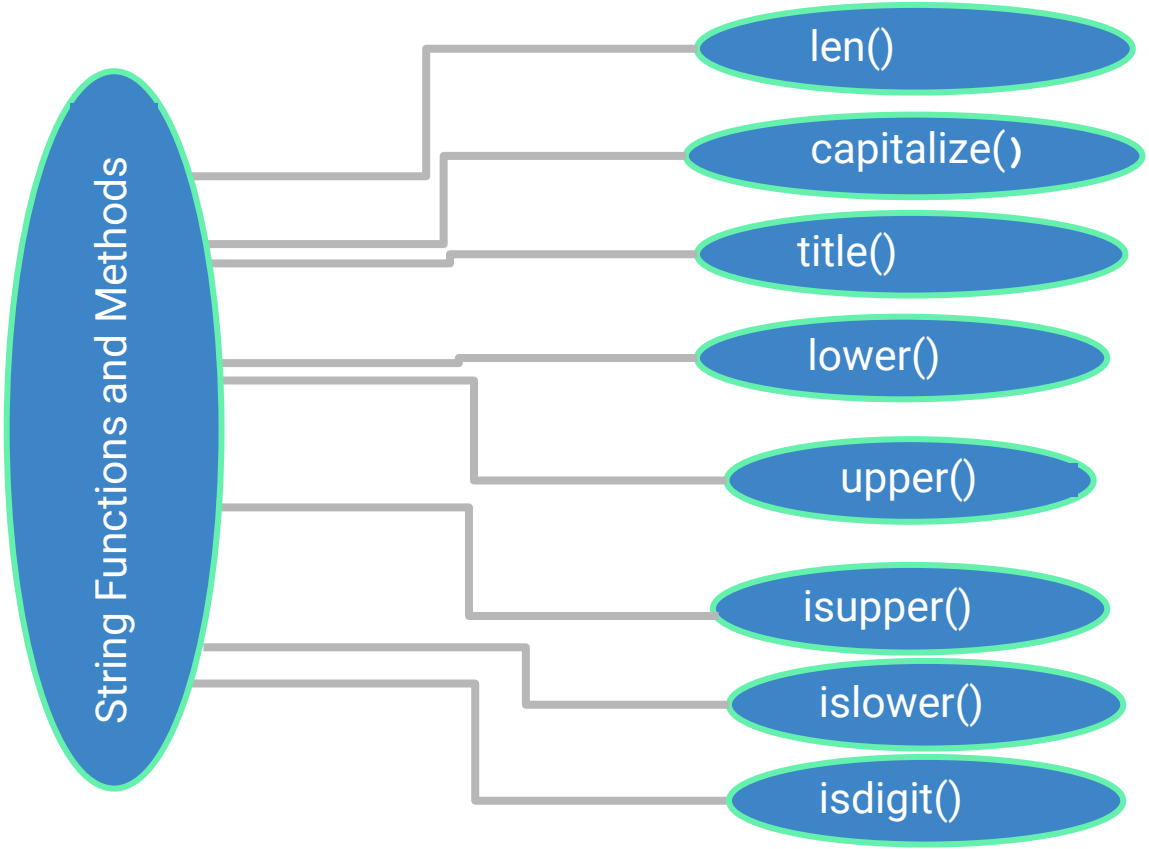
```
>>> str= "Computer Science"
>>> print( str[3:7] )
>>> print( str[4:] )
>>> print( str[0:] )
>>> print( str[:6] )
>>> print( str[:7] )
>>> print( str[1:10:2] )
>>> print( str[4: :2] )
>>> print( str[:10:2] )
>>> print( str[10:2:-1] )
```

pute
uter Science
Computer Science
Comput
Compute
optr S
ue cec
Cmue S
cS retup

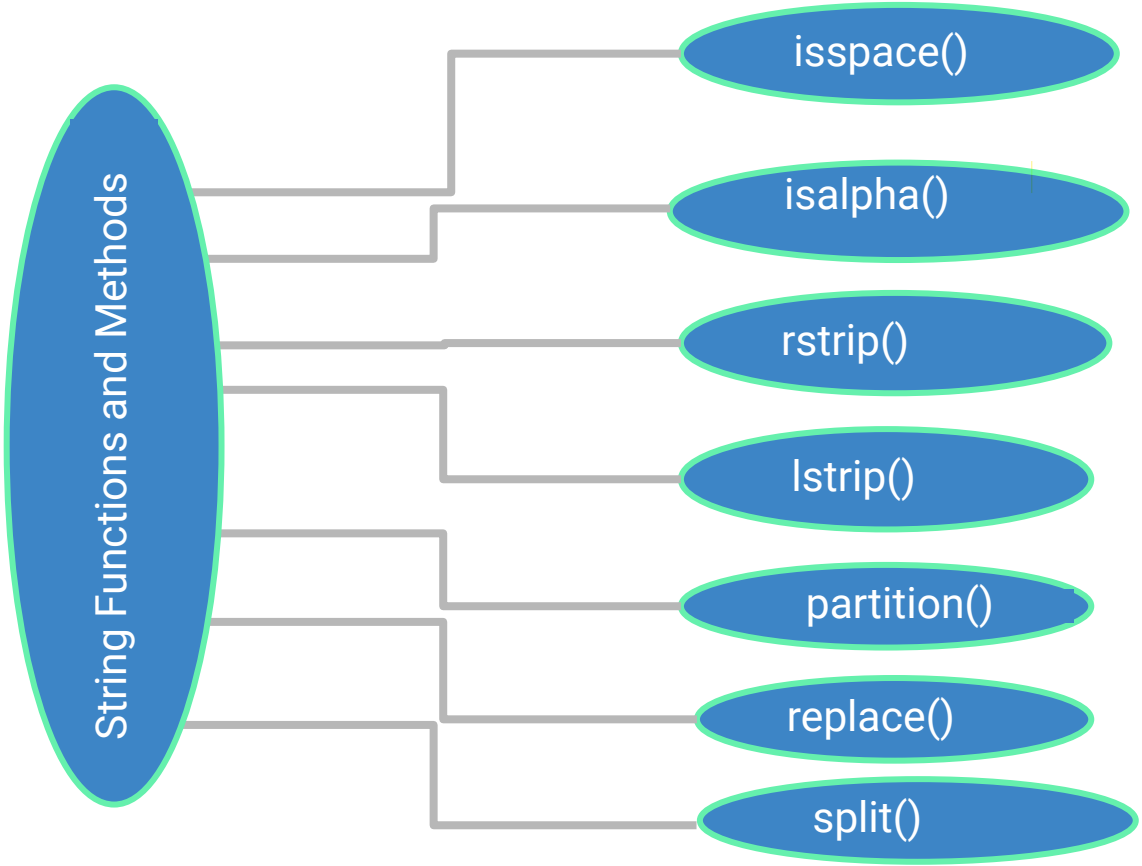
Solution:

```
>>> str= "Computer Science"
>>> print( str[3:7] )                    pute
>>> print( str[4:] )                     uter Science
>>> print( str[0:] )                     Computer Science
>>> print( str[:6] )                     Comput
>>> print( str[:7] )                     Compute
>>> print( str[1:10:2] )                 optrS
>>> print( str[4: :2] )                  ue cec ( #end  value is omitted )
>>> print( str[:10:2] )                 Cmue
>>> print( str[10:2:-1] )                cS retup (# when start>stop, step
                                         has to be negative)
```

String Functions & Methods



String Functions & Methods



String Functions and Methods

1. **len()**

   Returns the length of the string or the number of characters in the string

   Syntax : len(stringname)

Example

```
>>> str="i love india"
>>> print(len(str))
12
```

String Functions and Methods

2. capitalize()

Returns the copy of the string with the first character capitalised.

Syntax : **stringname.capitalize()**

Example

```
>>> str="hello"
>>>print(str.capitalize())
Hello
```

String Functions and Methods

3. title()

Returns a string which has first letter in each word is uppercase and all remaining letters are lowercase.

Syntax : stringname.title()

Example

```
>>> str="hello how are you"
>>>print(str.title())
Hello How Are You
```

String Functions and Methods

4. upper()

Returns an exact copy of the string with all the letters in uppercase

Syntax : stringname.upper()

Example

```
>>> str="learn python"
>>>print(str.upper())
LEARN PYTHON
```

String Functions and Methods

5. lower()

Returns an exact copy of the string with all the letters in lowercase

Syntax : stringname.lower()

Example

```
>>> str="PYTHON"
>>>print(str.lower())
python
```

String Functions and Methods

6. isupper()

Returns True if all the characters in the string are uppercase letters.

Syntax : **stringname.isupper()**

Example

```
>>>word='ABC'
>>>print(word.isupper())
True
```

String Functions and Methods

7. islower()

Returns True if all the characters in the string are lowercase letters.

Syntax : **stringname.islower()**

Example

```
>>>word='xyz'
>>>print(word.islower())
True
```

String Functions and Methods

8. isdigit()

Returns True if all the characters in the string are digits.

Syntax : **stringname.isdigit()**

Example

```
>>>word='123'
>>>print(word.isdigit())
True
```

String Functions and Methods

9. isalpha()

Returns True if all the characters in the string are alphabets.

Syntax : **stringname.isalpha()**

Example

```
>>>word='alpha123'
>>>print(word.isalpha())
False
>>>word='abcdef'
>>>print(word.isalpha())
True
```

String Functions and Methods

10. isalnum()

Returns True if all the characters in the string are alphabets or digits.

Syntax : **stringname.isalnum()**

Example

```
>>>word='alpha123'
>>>print(word.isalnum())
True
>>>word='abc123'
>>>print(word.isalnum())
True
```

String Functions and Methods

11. isspace()

Returns True if all the characters in the string are white space.

Syntax : **stringname.isspace()**

Example

```
>>>word='alpha123'
>>>print(word.isspace())
False
>>>word=' '
>>>print(word.isspace())
True
```

String Functions and Methods

12. split()

Returns a list of strings after breaking the given string by the specified separator.

Syntax : **stringname.split([separator,maxsplit])**

Example

```
>>>word='DPS Mathura Road Delhi'
>>>print(word.split())
['DPS', 'Mathura', 'Road', 'Delhi']
```

String Functions and Methods

Example

```
>>>word='DPS,Mathura,Road,Delhi'

>>>print(word.split(',',0))

['DPS,Mathura,Road,Delhi']
```

Example

```
>>>word='DPS,Mathura,Road,Delhi'

>>>print(word.split(',',2))

['DPS', 'Mathura', 'Road,Delhi']
```

Max splits

Split function

```
T = "LEARN PYTHON"

#Splits at space

print(T.split())

print(T.split('R'))

print(T.split('O'))
```

```
["LEARN", "PYTHON"]

['LEA', 'N PYTHON']

['LEARN PYTH','N']
```

String Functions and Methods

13. strip()

Returns a copy of string after removing leading or trailing characters passed as argument. If no argument is provided, whitespaces are taken by default.

Syntax : stringname.strip(character)

Example

```
>>>word='Python Programming'
>>>print(word.strip('g'))
'Python Programmin'
```

```
>>>word='  Python Programming   '
>>>print(word.strip())
'Python Programming'
```

String Functions and Methods

14. lstrip()

Returns a copy of string after removing leading characters passed as argument. If no argument is provided, whitespaces are taken by default.

Syntax : **stringname.lstrip(character)**

Example

```
>>>word='Python Programming'
>>>print(word.lstrip('P'))
'ython Programming'
```

String Functions and Methods

15. rstrip()

Returns a copy of string after removing trailing characters passed as argument. If no argument is provided, whitespaces are taken by default.

Syntax : **stringname.rstrip(character)**

Example

```
>>>word='Python Programming'
>>>print(word.rstrip('g'))
'Python Programmin'
```

String Functions and Methods

16. partition()

Returns a tuple of strings after splitting the string at first occurrence of Argument. Returns a tuple of length 3.

Syntax : **stringname.partition(character)**

Example

```
>>>word='Python is interesting'
>>>print(word.partition('is'))
('Python','is','interesting')
```

partition ()

```
p=”peace on earth”
print(p.partition(' on '))
print(p.partition('moon '))
print(p.partition('peace '))
print(p.partition('e'))
print(p.partition('h'))
```

```
('peace', ' on ', 'earth')
('peace on earth', '', '')
('', 'peace ', 'on earth')
('p', 'e', 'ace on earth')
('peace on eart', 'h', '')
```

String Functions and Methods

17. find()

Returns the lowest index in the string where the substring is found or within the slice range (if specified. If not specified, the range is taken as start - 0th index and end - last index. Returns -1 if sub is not found.

Syntax : String.find (substring[, start[, end]])

Example

```
>>>word="We learn C++"
>>>print(word.find("learn")
3
```

```
>>>word="We learn C++"
>>>print(word.find("yearn")
-1
```

String Functions and Methods

18. count()

Returns the number of occurrences of a substring in the given string. Returns 0 if not found.

Syntax : String.count (substring [, start[, end]])

Example

```
>>>s = 'I like programming and animation both'
>>>print(s.count("like")
1
```

String Functions and Methods

19. replace()

Returns copy of string which has replaced all old substring with new substring.

Syntax : **str.replace(oldsubstring,new substring)**

Example

```
>>>word='We learn C++'
>>>print(word.replace("C++","PYTHON"))

We Learn PYTHON
```

Question 1

Program to traverse a string inputted by user and display it.

```
str1=input("Enter the string :")
for i in str1:
    print(i)
```

```
str1=input("Enter the string :")
for i in range(len(str1)):
    print(str1[i])
```

Output :

Enter the string : PYTHON
P
Y
T
H
O
N

Question 2

Program to input a string and display the string in reverse order.

```
str1=input("Enter the string :")
l=len(str1)
for i in range(l-1,-1,-1):
    print(str1[i])
```

```
str1=input("Enter the string :")
str2=str1[::-1]
print(str2)
```

Output :

Enter the string : PYTHON
N
O
H
T
Y
P

Question 3

Program to input a string and then print the number of uppercase letters, lowercase letters, alphabets and digits.

```
str= input("Enter string : ")
cntalpha=0
cntdigit=0
cntupper=0
cntlower=0
for ch in str:
     if ch.islower() :
          cntlower+=1
     elif ch.isupper() :
          cntupper+=1
     elif ch.isdigit() :
          cntdigit+=1
     if ch.isalpha() :
          cntalpha+=1
print("No of alphabets : ",cntalpha)
print("No of digits : " ,cntdigit)
print("No of uppercase characters : ",cntupper)
print("No of lowercase characters : ",cntlower)
```

Output :

Enter string:EAT 123 code
No of alphabets : 6
No of digits : 3
No of uppercase characters: 3
No of lowercase characters : 4

Question 4

Write a program that reads a string and creates a new string by capitalising every alternate letter in the string.

```
str = input("enter a string ")
length = len(str)
newstr=''
for i in range(0,length):
    if i%2==0:
        newstr=newstr+str[i].upper()
    else:
        newstr+=str[i]
print(newstr)
```

Output :

Enter string:apple
ApPlE

Question 5

Program to input a string and then check if it a palindrome or not.

```
str = input("Enter a string :")
length = len(str)
valid = True
for i in range(length//2):
    if str[i] != str[length - i - 1]:
        valid = False
        break
if valid:
    print(str," is a palindrome")
else:
    print(str," is not a palindrome")
```

Output :

Enter string:KANAK
KANAK is a palindrome

Question 6

Program that reads a string with multiple words and creates a new string which capitalises the first character of each word.

```
str = input("Enter the string ")
length = len(str)
newstr = ''
newstr +=str[0].upper()
i=1
while i < length:
    if str[i] == ' ' and str[i+1] != ' ':
        newstr+=str[i]
        newstr+=str[i+1].upper()
        i+=2
    else:
        newstr+=str[i]
        i+=1
print("New string is ", newstr)
```

Output :

Enter string: i learn python
New string is : I Learn Python

Question 7

Program to accept a string and count and display the number of words which are starting with a vowel

```
str=input("Enter a string:")
l=len(str)
p=0
ctr=0
while p<=l-1:
    if p==0:
        if str[p] in "aeiouAEIOU" :
            ctr+=1
    elif str[p] == " " and str[p+1] != " ":
        if str[p+1] in "aeiouAEIOU" :
            ctr+=1
    p+=1
print("No. of words starting with vowel - ",ctr)
```

Output :

Enter a string : i eat food
No of words starting with vowel - 2

Question 7

Program to accept a string and count and display the number of words which are starting with a vowel

```
str=input("Enter a string:")
l=len(str)
s=str.split()
ctr=0
for i in s:
    if i[0] in "aeiouAEIOU":
        ctr+=1
print("No. of words starting with vowel - ",ctr)
```

Output :

Enter a string : i eat food
No of words starting with vowel - 2

Question 8

Program that reads a string and a substring, and then display the number of occurrences of that given substring in the string.

```
string=input("Enter a string - ")
sub=input("Enter a substring - ")
lenstr=len(string)
lensub=len(sub)
start=ctr=0
end=lenstr
while True:
     pos = string.find(sub,start,end)
     if pos != -1:
          ctr+=1
          start=pos+lensub
     else:
          break
     if start>=lenstr:
          break
print("No of occurences of ",sub, "=", ctr)
```

Output :

Enter a string: i learn learn python

Enter a substring: learn
No of occurrences of learn=2

Question 9

Write the equivalent code of islower() for a string. Accept a string. It should print True if the string has only lowercase letter.

```
str=input("Enter a String:")
length = len(str)
valid=True
for i in range(length):
    if (str[i]>="A" and str[i]<="Z"):
        valid=False
        break

print(valid)
```

Output :

Enter a String: money
True

Question 10

Write the equivalent code of upper() for a string. Accept a string. It should convert all lowercase letters to uppercase.

```
str=input("enter String - ")
newstr=""
for ch in str:
    if ch >="a" and ch<="z":
        newstr+=chr(ord(ch)-32)
    else:
        newstr=newstr+ch

print("New String :",newstr)
```

chr() converts ascii to char

ord() outputs ascii value of charcchar

Output :

Enter a character:
programming
New String :PROGRAMMING

Question 11

Write a program to enter a string and form an integer by extracting all digits from the string in the order they occurred.

```
str=input("enter string-")
num=0
for ch in str:
    if ch.isdigit():
        num = num*10+int(ch)
print("String entered:",str)
print("Extracted number:",num)
```

Output :

Enter string : raj456
String entered : raj456
Extracted number : 456

Question 12

Write a program to accept an identifier and check for its validity. An identifier is valid if it is starting with a letter or underscore (_) followed by underscore (_) and any alphanumeric character.

```
import keyword
str=input("Enter a Identifier: ")
valid = True; valid=str[0].isalpha() or str[0]=='_'
if keyword.iskeyword(str):
    valid = False
length = len(str);
if valid==True:
    for i in range(1,length):
        if not str[i].isalnum() and str[i]!="_":
            valid=False
            print(str, " cannot be an identifier ")
            break
    else:
        print(str, " is a Valid identifier")
else:
    print(str, " cannot be an identifier ")
```

Output :

Enter a Identifier : _ROLLNO
Valid

Question 13

Write a program that accepts a string. Encrypt the string on the basis of given criteria and print the encrypted string:

- Every letter and digit should be replaced by the next character.
- 9 should be replaced by 0 and
- Z/z should be replaced by A/a.
- All spaces should be changed to *.
- All other special characters should remain unchanged

Solution

```
str=input("Enter a string-")
newstr=""
newch=""
for ch in str:
    if ch.isalnum():
        if ch=="z":
            newch="a"
        elif ch =="9":
            newch="0"
        else:
            n=ord(ch)+1
            newch=chr(n)
    elif ch.isspace():
        newch="*"
    else :
        newch=ch
    newstr+=newch
print("Original string -",str)
print("Changed String -" ,newstr)
```

Question 14

Write a program to input the full name (First name, Middle name (optional), and Last name) of a person, and then display the name in abbreviated form.

For example:

(i) If the name is “Saif Ali Khan”, the output should be “S. A. Khan”

(ii) If the name is “Priyanka Chopra”, the output should be P. Chopra

```
name = input("enter your name : ")
l=name.split()
newstr=''
for i in range(len(l)-1):
    newstr = newstr +  l[i][0].capitalize() + '.'
newstr = newstr + l[len(l) - 1].title()
print("Required name is : ",newstr)
```

Another solution:

```
fn=input("Enter first name: ")
mn=input("Enter middle name: ")
ln=input("Enter last name: ")
if (fn=='' or ln==''):
    print("first name and last are mandatory")
else:
if (mn==''):
    shortName=fn[0]+'. '+ln
else:
    shortName=fn[0]+'. '+mn[0]+'. '+ln
print(shortName)
```

Output :

```
Enter first name:Saif
Enter middle name:Ali
Enter last name:Khan

S.A.Khan
```

Another solution:

```
name = input("enter your name: ")
x=name[::-1]
m = x.find(" ")
str1= name[0].capitalize()+". "
for i in range(len(name)-m-1):
    if name[i]==" ":
        str1+=name[i+1].capitalize()+". "
str1+=name[len(name)-m:].capitalize()
print("short form of name is",str1)
```

Programs

1. Write a program to input a string and a character and count the number of times a character appears in the string.
2. Write a program to input n number of strings. Accept n from user. Count the number of strings which have more than 5 characters in it (Do not use any string function)
3. Write a program to accept a string. Create a new string with all the consonants deleted from the string.
4. Write a Python program to remove the nth index character from a nonempty string. If input string is “python” and index is 3, Expected output :”pyton”
5. Write a menu driven program to accept a string and do the following operation as per the choice of the user

   a. Display length of string
   b. Display no. of words
   c. Display no. of vowels
   d. Reverse string

Find the output :

A = "Passionate Programmer"

```
(i) print(len(A))
(ii) print(A[3])
(iii) print(A[-3])
(iv) print(A[3:])
(v) print(A[-3:])
(vi) print(A[::3])
(vii) print(A[3::])
(viii) print(A[3::-2])
```

21
s
m
sionate Programmer
mer
Psnerrm
sionate Programmer
sa

```
(ix) print(A[-3::-2])
(x) print(A[::-3])
(xi) print(A[3:-3:3])
(xii) print(A[3:-3:-3])
(xiii) print(len(A[3:]))
(xiv) print(len(A[-3:]))
(xv) print(len(A[::-3]))
```

magr tnisP
rmgPtos
snerr
balnk
18
3
7

Solution :

```
(i) print(len(A))      21
(ii) print(A[3])       s
(iii) print(A[-3])     m
(iv) print(A[3:])      sionate Programmer
(v) print(A[-3:])      mer
(vi) print(A[::3])     Psnerrm
(vii) print(A[3::])    sionate Programmer
(viii) print(A[3::-2])     sa
```

```
(ix) print(A[-3::-2])      magr tnisP
(x) print(A[::-3])          rmgPtos
(xi) print(A[3:-3:3])       snerr
(xii) print(A[3:-3:-3]) blank string
(xiii) print(len(A[3:])) 18
(xiv) print(len(A[-3:]))  3
(xv) print(len(A[::-3]))  7
```

A = "Passionate Programmer"

Find the output :

```
s="ComPutEr"
for i in s:
    if i in 'aeiou':
        print('*',end='')
    elif i.isupper():
        print(i,end='')
    else: print('-')
```

C*-
P*-
E-

Solution:

```
s="ComPutEr"
for i in s:
    if i in 'aeiou':
        print('*',end='')
    elif i.isupper():
        print(i,end='')
    else: print('-')
```

```
C*-
P*-
E-
```

Find the output :

```
Name = "ComPUteR"
for x in range(0,len(Name)):
    if Name[x].islower():
        print(Name[x].capitalize(), end='')
    elif Name[x].isupper():
        if x%2==0:
            print (chr(ord(Name[x])+32),end='*')
        else:
            print(Name[x-1])
```

c*O Mm
u*T Ee

Solution:

```
Name = "ComPUteR"
for x in range(0,len(Name)):
    if Name[x].islower():
        print(Name[x].capitalize(), end='')
    elif Name[x].isupper():
        if x%2==0:
            print (chr(ord(Name[x])+32),end='*')
        else:
            print(Name[x-1])
```

```
c*OMm
u*TEe
```

Find the output :

```
Text1="AIsScE 2019"
Text2=""
i=0
while i < len(Text1):
    if Text1[i] >="0" and Text1[i] <"9":
        val=int(Text1[i])
        val=val+1
        Text2=Text2+str(val)
    elif Text1[i] >="A" and Text1[i] <="Z":
        Text2=Text2+(Text1[i+1])
    else:
        Text2=Text2+"*"
    i+=1
    print(Text2)
```

Solution :

```
Text1="AIsScE 2019"
Text2=""
i=0
while i < len(Text1):
    if Text1[i] >="0" and Text1[i] <"9":
        val=int(Text1[i])
        val=val+1
        Text2=Text2+str(val)
    elif Text1[i] >="A" and Text1[i] <="Z":
        Text2=Text2+(Text1[i+1])
    else:
        Text2=Text2+"*"
    i+=1
    print(Text2)
```

```
I
Is
Is*
Is*c
Is*c*
Is*c*
Is*c* *
Is*c* *3
Is*c* *31
Is*c* *312
Is*c* *312*
```

Find the output

1.
```
word = 'DPS:Mathura:Road'
print(word.split(':'))                    # Splitting at ':'
```

2.
```
word = 'CatBatSatFatOr'
for i in range(0, len(word), 3):          # Splitting at 3
    print(word[i:i+3])
```

3.
```
txt = "apple#banana#cherry#orange"
x = txt.split("#", 1)                     # Splitting at '#'
print(x)
```

Solution:

```
1.  word = 'DPS:Mathura:Road'
    print(word.split(':'))                  # Splitting at ':'

2.  word = 'CatBatSatFatOr'
    for i in range(0, len(word), 3):        # Splitting at 3
        print(word[i:i+3])
3.  txt = "apple#banana#cherry#orange"
    x = txt.split("#", 1)          # Splitting at '#'
    print(x)
```

['DPS','Mathura','Road']

Cat
Bat
Sat
Fat
Or

['apple','banana#cherry#orange']

Happy Learning!!!

Thank you