Strings

- strings can instialise in 3 ways
- one is single quotes
- double quotes
- triple quotes
- triple quotes and double quotes are common way to present
  - Triple quotes means **doc string**
- String can be anything that is written in quotes

In [1]:
```
str1='python'
str1
```

Out[1]: 'python'

In [2]:
```
str2="PYTHON"
str2
```

Out[2]: 'PYTHON'

In [4]:
```
str3='10.5'
str3
```

Out[4]: '10.5'

In [5]:
```
str4="""True"""
str4
```

Out[5]: 'True'

In [6]:
```
str5="@$f"
str5
```

Out[6]: '@$f'

triple quotes

- triple quotes is used to convey the information
- it is also called **doc string**
- also called **multi line**

In [7]:
```
str7="""Hello how are you
        I am good"""
print(str7)
```

```
Hello how are you
        I am good
```

In [9]:
```
str8='I like "python"'
str8
```

Out[9]:
```
'I like "python"'
```

In [11]:
```
str8="I 'like' python"
str8
```

Out[11]:
```
"I 'like' python"
```

- type
- print
- len
- max
- min
- reversed
- sorted
- in
- index
- slice

type operator

- type is a keyword
- It will give the type of value
- ex: str1="python" type(str1)

What is difference between keywords and package methods

- keyword is reserved word
- we can limited keywords(35) are fixed
- whenever keyword: keyword(value)
- before keyword there is no another statement
- there is no dot (.) before keyword
- ex: eval( ),input(),print(),range(),type()
- If you want to use methods from package first you need to import the package first

- package name.method name

```
In [14]: str9='sruthi'
         type(str9)
```

```
Out[14]: str
```

len

- len gives us how many characters are present in string
- len also calculates space

```
In [15]: str9='sruthi'
         len(str9)
```

```
Out[15]: 6
```

```
In [4]: len(' ')
```

```
Out[4]: 1
```

```
In [16]: max(str9)
```

```
Out[16]: 'u'
```

```
In [26]: ord('s'),ord('r'),ord('u'),ord('t'),ord('h'),ord('i')
```

```
Out[26]: (115, 114, 117, 116, 104, 105)
```

max

- gives max value based on ascii values

min

- gives min value based on ascii values

```
In [17]: min(str9)
```

```
Out[17]: 'h'
```

```
In [2]: str9='sruthi'
        for i in reversed(str9):
            print(i,end='')
```

```
ihturs
```

```
In [20]: sorted(str9)
```

```
Out[20]: ['h', 'i', 'r', 's', 't', 'u']
```

sum

- It will give sum of value

- But it not works for string means characters

In [28]:
```
sum('123'),sum('python')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[28], line 1
----> 1 sum('123'),sum('python')

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

reverse

In [32]:
```
str9='sruthi'
reversed(str9)
```

Out[32]:
```
<reversed at 0x2636ee7f520>
```

- when ever you see this type of output that is values
- The output soterd inside that
- it means **iterator**
- Output we can see in loop

In [1]:
```
str9=reversed('sruthi')
for i in str9:
    print(i,end=' ')
```

```
i h t u r s
```

sorted

- gives ascending order or decending order based on ascii value
- sorted has one argument called reverse
- reverse is a default argument with value False means ascending order[small to big]
- now give reverse=True and check the order it given decending order(big to small)

In [37]:
```
ord('s'),ord('r'),ord('u'),ord('t'),ord('h'),ord('i')
```

Out[37]:
```
(115, 114, 117, 116, 104, 105)
```

In [35]:
```
str9=sorted("sruthi")
str9
```

Out[35]:
```
['h', 'i', 'r', 's', 't', 'u']
```

In [ ]:
```
# if we see any / don't assign any value before / (slash)
```

In [36]:
```
sorted(str9)
```

Out[36]:
```
['h', 'i', 'r', 's', 't', 'u']
```

In [17]:
```
sorted('sruthi',reverse=False)
```

Out[17]:
```
['h', 'i', 'r', 's', 't', 'u']
```

In [44]:
```
complex()            #works
complex(2,3)          #works
complex(real=2,imag=3)  #works
complex(real=2,3) #fail(default argument should be assign last)
complex(2,imag=3)  #work
```

Out[44]:
```
(2+3j)
```

In [46]:
```
complex(2,imag=6)
```

Out[46]:
```
(2+6j)
```

In [45]:
```
complex(real=2,imag=3)
```

Out[45]:
```
(2+3j)
```

In [ ]:
```
import random
random.randint()#works
random.randint(a=10,b=20)#works
random.randint(c=10,d=51)#fail
random.randint(a=10,90)#fails
random.randint(19,b=90)#works
```

In [5]:
```
#ask the user
ord('s'),ord('r'),ord('u'),ord('t'),ord('h'),ord('i')
```

Out[5]:
```
(115, 114, 117, 116, 104, 105)
```

In [62]:
```
#wap ask the user to print how amny a's are there
str1='hello how are you, im annad'

count=0
for i in str1:
    if i=='a':
        count=count+1
print(f"the number of a's are {count}")
```

```
the number of a's are 3
```

In [14]:
```
str1='hello how are you, im annad'
count=0
for i in str1:
    if i=='e':
        count=count+1
print(f" the number of e's are {count}")
```

```
the number of e's are 2
```

index

In [18]:
```
str1='hello python'
# in python index number start with 0
str1[0]
```

Out[18]: 'h'

In [ ]:
```
              -7   -6   -5    -4  -3   -2 -1
h  e  l  l  o    p  y  t  h  o  n

0  1  2  3  4  5  6  7  8  9  10  11
```

positive index

In [24]:
```
for i in range(0,12):
    print(i,str1[i])
```

```
0 h
1 e
2 l
3 l
4 o
5
6 p
7 y
8 t
9 h
10 o
11 n
```

In [28]:
```
str2='sruthi laya'
n=len(str2)
for i in range(n):
    print(i , str2[i])
```

```
0 s
1 r
2 u
3 t
4 h
5 i
6
7 l
8 a
9 y
10 a
```

negative index

In [30]:
```
str='hello python'

for i in range(-12,0):
    print(i,str[i])
```

```
-12 h
-11 e
-10 l
-9 l
-8 o
-7
-6 p
-5 y
-4 t
-3 h
-2 o
-1 n
```

In [33]:
```python
str='hello python'
n=len(str)
for i in range(-n,0):
    print(i,str[i])
```

```
-12 h
-11 e
-10 l
-9 l
-8 o
-7
-6 p
-5 y
-4 t
-3 h
-2 o
-1 n
```

In [62]:
```python
#wap to print each letter index positive way
# positive index  h is 0

str1='Hello python'
n=len(str1)
for i in range(n):
    print(f"The positive index of {str[i]} is {i}")
```

```
The positive index of h is 0
The positive index of e is 1
The positive index of l is 2
The positive index of l is 3
The positive index of o is 4
The positive index of   is 5
The positive index of p is 6
The positive index of y is 7
The positive index of t is 8
The positive index of h is 9
The positive index of o is 10
The positive index of n is 11
```

In [34]:
```python
#wap to print each letter index negative way
# negative index  h is -12

str2='hello python'
n=len(str2)
for i in range(n):
    print(f"The negative index of {str2[i]} is {i-len(str2)}")
```

```
The negative index of h is -12
The negative index of e is -11
The negative index of l is -10
The negative index of l is -9
The negative index of o is -8
The negative index of   is -7
The negative index of p is -6
The negative index of y is -5
The negative index of t is -4
The negative index of h is -3
The negative index of o is -2
The negative index of n is -1
```

In [22]:
```python
#wap to print each letter index  both positive and negative way  in single for l
# negative index  h is -12

str1='Hello python'
n=len(str1)
for i in range(n):
    print(f"The positive index of {str1[i]} is {i} and negative index of {str1[i
```

```
The positive index of H is 0 and negative index of H is -12
The positive index of e is 1 and negative index of e is -11
The positive index of l is 2 and negative index of l is -10
The positive index of l is 3 and negative index of l is -9
The positive index of o is 4 and negative index of o is -8
The positive index of   is 5 and negative index of   is -7
The positive index of p is 6 and negative index of p is -6
The positive index of y is 7 and negative index of y is -5
The positive index of t is 8 and negative index of t is -4
The positive index of h is 9 and negative index of h is -3
The positive index of o is 10 and negative index of o is -2
The positive index of n is 11 and negative index of n is -1
```

In [24]:
```python
#wap ask the user get the indexs of occurances of 'a' given string  we must use
str1='avaliable'

count=0
for i in range(len(str1)):
    if str1[i]=='a':
        count=count+1
        print(f"The {count}  occurances  of '{str1[i]}' index is {i}")
```

```
The 1  occurances  of 'a' index is 0
The 2  occurances  of 'a' index is 2
The 3  occurances  of 'a' index is 5
```

In [24]:
```python
# Q7 wap ask the write the sum of all indexes and count of all occurances
str='avaliable'
count=0
summ = 0
for i in range(len(str)):
    if str[i]=='a':
        count = count + 1
        summ = summ + i
print(f"occurance of a is {count}")
print(f"sum of all indexes is {summ}")
```

```
occurance of a is 3
sum of all indexes is 7
```

In [67]:
```
str='sruthilaya'
count=0
count1 = 0

summ1 = 0
summ2 = 0
for i in range(len(str)):
    if str[i] in 'aeiou':
        count=count+1

        summ1  = summ1 + (i+1)
        summ2 = summ2 + (i-1)

        print(f"The vowels in given string are {str[i]},indexes of vowels are {i

print(f"The count of vowels are  {count}")
print(f"sum of vowel indexes are {summ}")
print(f"sum of after  vowel indexes are {summ1}")
print(f"sum of  before vowel indexes are {summ2}")
```

```
The vowels in given string are u,indexes of vowels are 2
The vowels in given string are i,indexes of vowels are 5
The vowels in given string are a,indexes of vowels are 7
The vowels in given string are a,indexes of vowels are 9
The count of vowels are  4
sum of vowel indexes are 7
sum of after  vowel indexes are 27
sum of  before vowel indexes are 19
```

In [28]:
```
#find the vowels in given string
# hai how are you
# find the index of vowels
# find the count of vowels
# find the sum of before index of vowels

str2='hai how are you'
for i in range(len(str2)):
    if str2[i]=='a' or str2[i]=='e' or str2[i]=='i' or str2[i]=='o' or str2[i]==
        print(f"the vowels are {str2[i]}")
```

```
the vowels are a
the vowels are i
the vowels are o
the vowels are a
the vowels are e
the vowels are o
the vowels are u
```

In [29]:
```
str2='hai how are you'
for i in range(len(str2)):
    if str2[i] in'aeiou':
        print(str2[i],end=' ')
```

```
a i o a e o u
```

In [31]:
```
str2='hai how are you'
count=0
summ=0
summ1=0
summ2=0
v='aeiou'
for i in range(len(str2)):
    if str2[i] in v:
        count=count+1
        summ= summ + i
        summ1 = summ1+(i-1)

        print(f"{str2[i]} , index of vowel  {str2[i]}  is {i}")
print(f"The   count   of   vowels   are   {count}")

print(f"The   sum    of  indexes   of     vowels    are {summ}")

print(f"the   sum    of    before  indexes  of   vowels   are {summ1}")
```

```
a , index of vowel  a  is 1
i , index of vowel  i  is 2
o , index of vowel  o  is 5
a , index of vowel  a  is 8
e , index of vowel  e  is 10
o , index of vowel  o  is 13
u , index of vowel  u  is 14
The   count   of   vowels   are   7
The   sum    of  indexes   of     vowels    are 53
the   sum    of    before  indexes  of   vowels   are 46
```

mutable-immutable

- values cannot change using index operations called as **immutable**
- **STRINGS ARE IMMUTABLE**

In [2]:
```
str1='python'      # we will not change case senstive letters
str1[0]='P'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[2], line 2
      1 str1='python'
----> 2 str1[0]='P'

TypeError: 'str' object does not support item assignment
```

In [ ]:
```
##STRINGS ARE IMMUTABLE (THEY WILL NOT CHANGE)
```

slice

- cut into pieces
- which means we can extract some part of string
- for that we need to understand the start,stop,step
- s[start:stop:step]

In [3]:
```
# -12  -11   -10   -9    -8   -7    -6    -5    -4      -3      -2       -1
# h     e    l     l      o         p      y     t      h       o        n
# 0     1    2     3      4    5    6      7     8      9       10       11
str1="hello python"
str1[2:10]

#start value=2
#last=stop-1=10-1=9
#direction=positve
```

Out[3]: 'llo pyth'

In [4]:
```
str1="hello python"
str1[2:10:2]
```

Out[4]: 'lopt'

In [6]:
```
str1="hello python"
str1[2:10:-2]
#start value=2
#direction=negative
#stop=last+1=10+1=11
#it gives empty string(not possible)
```

Out[6]: ''

In [7]:
```
str1="hello python"
str1[2:-10:-2]
```

Out[7]: ''

In [11]:
```
str1[:]
```

Out[11]: 'hello python'

In [14]:
```
str1[::]
```

Out[14]: 'hello python'

In [16]:
```
str1[:12:]
```

Out[16]: 'hello python'

In [17]:
```
str1[1:]
```

Out[17]: 'ello python'

In [18]:
```
# -12  -11   -10   -9    -8   -7    -6    -5    -4      -3      -2       -1
# h     e    l     l      o         p      y     t      h       o        n
# 0     1    2     3      4    5    6      7     8      9       10       11
str1[::-1]  # start value=0 and last value=12 step =-1
```

Out[18]: 'nohtyp olleh'

In [ ]:
```
str1[2::-2]
```

In [19]:
```
str1[::-1]
```

Out[19]:
```
'nohtyp olleh'
```

In [82]:
```
#-12   -11  -10
#  h    e    l    l    o       p   y   t   h   o   n
#  0    1    2    3    4  5    6   7   8   9  10   11
str1[2:12:2] # P
str1[2:12:-2] # NP
str1[2:-12:2] # NP
str1[2:-12:-2] # P
str1[-2:-12:-2] # P
str1[-2:12:2] # P ==============
str1[-2:-12:2] # NP
str1[:]#  ALL
str1[2:] # start=2  last=last
str1[:12] # start=0  last=12-1=11
str1[::]  # start=0  last=last step=1
str1[::-1] # reverse string
```

Out[82]:
```
'alo alo alo'
```

In [20]:
```
str1[2:12:2]
```

Out[20]:
```
'lopto'
```

In [ ]:
```
str1[2:12:-2]
```

concatenation

- adding two strings

In [26]:
```
str1="hello  " #for space
str2="  python"   # for space
str1+str2
str1+' '+str2     # for space
```

Out[26]:
```
'hello     python'
```

In [27]:
```
str1*str2      # strings are not a numbers so won't perform those operations
str1/str2
str1-str2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[27], line 1
----> 1 str1*str2

TypeError: can't multiply sequence by non-int of type 'str'
```

In [30]:
```
str1*3
```

Out[30]:
```
'hello  hello  hello  '
```

In [33]:
```
str1+str1+str1
```

Out[33]:
```
'hello  hello  hello  '
```

In [31]:
```
str2*3
```

Out[31]:
```
'  python  python  python'
```

In [34]:
```
str1-str2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[34], line 1
----> 1 str1-str2

TypeError: unsupported operand type(s) for -: 'str' and 'str'
```

In [35]:
```
str1/str2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[35], line 1
----> 1 str1/str2

TypeError: unsupported operand type(s) for /: 'str' and 'str'
```

Strings-partII string methods

- you need to treat like strings is a package
- so apply dir on any string we will get the methods

In [36]:
```
s='python'
dir(s)    # gives method
```

Out[36]:
```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rmod__',
 '__rmul__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'capitalize',
 'casefold',
 'center',
 'count',
 'encode',
 'endswith',
 'expandtabs',
 'find',
 'format',
 'format_map',
 'index',
 'isalnum',
 'isalpha',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
 'isupper',
 'join',
 'ljust',
 'lower',
```

```
 'lstrip',
 'maketrans',
 'partition',
 'removeprefix',
 'removesuffix',
 'replace',
 'rfind',
 'rindex',
 'rjust',
 'rpartition',
 'rsplit',
 'rstrip',
 'split',
 'splitlines',
 'startswith',
 'strip',
 'swapcase',
 'title',
 'translate',
 'upper',
 'zfill']
```

In [37]:
```
dir(' ')
```

Out[37]:
```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rmod__',
 '__rmul__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'capitalize',
 'casefold',
 'center',
 'count',
 'encode',
 'endswith',
 'expandtabs',
 'find',
 'format',
 'format_map',
 'index',
 'isalnum',
 'isalpha',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
 'isupper',
 'join',
 'ljust',
 'lower',
```

```
 'lstrip',
 'maketrans',
 'partition',
 'removeprefix',
 'removesuffix',
 'replace',
 'rfind',
 'rindex',
 'rjust',
 'rpartition',
 'rsplit',
 'rstrip',
 'split',
 'splitlines',
 'startswith',
 'strip',
 'swapcase',
 'title',
 'translate',
 'upper',
 'zfill']
```

captialize

In [38]:
```
str='python'
str.capitalize()
```

Out[38]: 'Python'

upper

In [39]:
```
str='python'
str.upper()
```

Out[39]: 'PYTHON'

casefold

- always return lower case

In [40]:
```
str='python'
str.casefold()
```

Out[40]: 'python'

lower

In [41]:
```
str='python'
str.lower()
```

Out[41]: 'python'

centre

In [48]:
```
str='python'         # gives spaces according to passed argument in brackets
str.center(10)
```

Out[48]: ' python '

In [50]:
```
str='python'
str.center(10,'$')
# total occupied by 10 charcters
# pt
```

Out[50]: '$$python$$'

In [52]:
```
' python'.center(2,'$')'
```

Out[52]: ' python'

In [54]:
```
 'python'.center(7,'$')
```

Out[54]: '$python'

In [67]:
```
'python'.count('python')
```

Out[67]: 1

unique vowels

- first take the given string
- next take the empty string with ex:' '
- use for loop condition for original string
- use if condition for unqiue and not in for empty string
  - is that vowel
  - is that present in that new string
- next if vowel add to empty string

wrapper unquie items

- orginal string ex:str='abcd'
- empty string ex:str2=' '

for i in str: if i is in 'aeiou' and i not in str2: str2=str2+i print(str2)

In [6]:
```
str1='12354465566648954'
str2=' '
for i in str1:
    if i in '123456789' and i not in str2:
        str2=str2+i
print(f"unique {str2}")
```

unique 12354689

In [4]:
```
#wap to find unique vowel
str1='abbccedd'
str2=' '
for i in str1:
    if i in 'aeiou' and i not in str2:
```

```
        str2=str2+i
print(str2)
```

```
ae
```

In [6]:
```
str='python learning'
str.title()
```

Out[6]:
```
'Python Learning'
```

In [7]:
```
#Q:str="hello how are you"
str1="hello how are you"
str1.upper()
```

Out[7]:
```
'HELLO HOW ARE YOU'
```

In [8]:
```
str1.capitalize()  # first letter in string is capital
```

Out[8]:
```
'Hello how are you'
```

In [9]:
```
str1.title()      #  title gives capital letters for first letter of each word i
```

Out[9]:
```
'Hello How Are You'
```

count

- there is no error in count
- if no char exist in string is given answer is 0

In [12]:
```
str1="hai how are you"
str1.count('a')            # gives the how many characters in given string
```

Out[12]:
```
2
```

In [13]:
```
str1.count('o')
```

Out[13]:
```
2
```

In [16]:
```
str1.count('a',7)
```

Out[16]:
```
1
```

In [18]:
```
str1.count('a',-5)
```

Out[18]:
```
0
```

In [19]:
```
str1.count('a',5,2)
```

Out[19]:
```
0
```

In [ ]:
```
str1="hai how are you"
str1.count('a')
```

In [9]:
```
# you want count the number of 'a' after 5th index
# you want to count the number of 'a'  between 5th and 10th index
```

```
str1='ola ola ola'
str1.count('a',3) # caluclates how number of  a's  after index 2
```

Out[9]: 2

In [12]:
```
str2='ola ola ola olaa ola'   # along with space 11 characters
str2.count('a',4,15)   # start with 4 and end with 11

# str2[4:11]
#start value=4
# stop= last-1==>11-1=10
```

Out[12]: 3

In [13]:
```
str2='ola ola ola'   # along with space 11 characters
str2.count('a',4,10)
```

Out[13]: 1

In [14]:
```
str2='ola ola ola'   # along with space 11 characters
str2.count('a',4,21)
```

Out[14]: 2

In [1]:
```
#Q without using count find the number of ola
def number():
    str='ola ola olaa'
    count = 0
    for i in range(len(str)):
        if str[i:i+3]=='ola':
            count = count + 1
    print(count)
number()
```

3

replace

In [ ]:
```
- # i want replace 'l' with 'L'
```

In [35]:
```
str1='hello'
str1.replace('l','L')
```

Out[35]: 'heLLo'

- In replace we have count=-1 as a default value
- which means it replace all the oocurance

In [37]:
```
str1.replace('l','L',2)
```

Out[37]: 'heLLo'

In [38]:
```
str1.replace('z','L',1)
```

Out[38]: 'hello'

In [ ]:
```
- Read the   **docstring**   carefully
```

In [58]:
```
#Q 14)wap ask the user
# input: str is "restart"
#oput: resta$t

str1="restart"
str1.replace('r','$',1)
```

Out[58]:
```
'$estart'
```

In [61]:
```
str1='restart'
str1.replace('r','$',0-8)
```

Out[61]:
```
'$esta$t'
```

In [72]:
```
str1[::-1].replace('r','$',1)[::-1]
```

Out[72]:
```
'resta$t'
```

In [94]:
```
str1='restart'
str1[::-1]
str1.replace('r','$',1)
```

Out[94]:
```
'$estart'
```

In [7]:
```
str1='ola ola ola ola'
l=len(str1)
count=0
for  i in range(0,l,4):
    for j in range(3,l+1):
        if str1[i:j]=='ola':
            count= count+1
print(count)
```

```
4
```

In [16]:
```
#**second method**

str1='ola ola ola ola'
count=0
for i in range(len(str1)):
    if str1[i:i+3]=='ola':
        count = count +1
count
```

Out[16]:
```
4
```

In [35]:
```
str1='jyothi jyothi jyo'
s='jyo'
count = 0
for i in range(len(str1)):
    if str1[i:i+len(s)]== s:
        count = count + 1
count
```

Out[35]: 3

In [32]:
```
str='sruthilaya sruthilaya sruthi'
s='sruthi'
count = 0
for i in range(len(str)):
    if str[i:i+len(s)]==s:
        count =count+1
count
```

Out[32]: 3

In [1]:
```
name='sruthilaya sruthilaya sruthilaya'
count =0
for i in  range(len(name)):
    if name[i:i+6]=='sruthi':
        count = count + 1
print(count)
```

3

In [95]:
```
str1='hai hai hai hai hai'

i1=str1.index('a')  # 1
i2=str1.index('a',i1+1) # 5
i3=str1.index('a',i2+1) # 9
i4=str1.index('a',i3+1)
i5=str1.index('a',i4+1)

i1,i2,i3,i4,i5
```

Out[95]: (1, 5, 9, 13, 17)

In [137…
```
str1='restart'
str2=str1[::-1]
str3=str2.replace('r','$',1)
str4=str3[::-1]
str4
```

Out[137… 'resta$t'

In [ ]:

In [ ]:

find

In [19]:
```
str1='hai hai hai'
str1.find('a')
```

Out[19]: 1

In [20]:
```
str1.find('z')  # substring is not there so gives -1
```

Out[20]: -1

In [ ]:

note(if substring is not present in main string)

- count will give zero(0)
- index will give error
- find will give -1

-replace will give same string

In [47]:
```
str='omkar.nallagoni@cognizant.com'
i=str[0:5]
#i
i1=str[6:15]
#i1
i2=str[16:25]
#i2
print(f"first_name='{i}'")
print(f"second_name='{i1}'")
print(f"company_name='{i2}'")
```

```
first_name='omkar'
second_name='nallagoni'
company_name='cognizant'
```

In [65]:
```
name='omkar.nallagoni@cognizant.com'
i=name.find('.')
i1=name.find('@',i+1)
i2=name.find('.',i1+1)
print(f"'{name[0:i]}'")
print(f"'{name[i+1:i1]}'")
print(f"'{name[i1+1:i2]}")
```

```
'omkar'
'nallagoni'
'cognizant
```

In [22]:
```
name='choutapelly.sruthilaya@gmail.com'
dot=name.find('.')  #11
sym=name.find('@')  #22
dot2=name.find('.',12) #28
print(f" sur_name: '{name[0:dot]}' ")
print(f" name    : '{name[dot+1:sym]}'")
print(f" mail    : '{name[sym+1:dot2]}'")
```

```
sur_name: 'choutapelly'
name    : 'sruthilaya'
mail    : 'gmail'
```

strip-lstrip-rstrip

In [72]:
```
name1=' python '
name2=' python'
name3='python '
name1.strip(),name1.lstrip(),name1.rstrip()
```

Out[72]:
```
('python', 'python ', ' python')
```

In [73]:
```
name2.strip(),name2.lstrip(),name2.rstrip()
```

Out[73]: ('python', 'python', ' python')

In [74]:
```
name3.strip(),name3.lstrip(),name3.rstrip()
```

Out[74]: ('python', 'python ', 'python')

In [83]:
```
name4=' python$$$'
i=name4.lstrip()
i.rstrip('$' )
```

Out[83]: 'python'

In [30]:
```
str='@@@sruthi  '
str1=str.lstrip('@@@')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[30], line 2
      1 str='@@@sruthi  '
----> 2 str1=str.lstrip('@@@'),str1.rstrip()

NameError: name 'str1' is not defined
```

split

- cutting into pieces

In [88]:
```
str='python learning'
str.split()
```

Out[88]: ['python', 'learning']

In [87]:
```
str='python'
str.split('o')
```

Out[87]: ['pyth', 'n']

In [89]:
```
str2='how how are you'
str2.split('o')
```

Out[89]: ['h', 'w h', 'w are y', 'u']

startswith and endswith

In [97]:
```
str2='data science'
str2.startswith('s')
```

Out[97]: False

In [98]:
```
str2='data science'
str2.startswith('d')
```

Out[98]: True

In [100…
```
str2='data science'
str2.endswith('s')
```

Out[100… False

In [101…
```
str2='data science'
str2.endswith('data science')
```

Out[101… True

In [102…
```
str2='data science'
str2.endswith('e')
```

Out[102… True

In [ ]: