In [ ]:
```
#strings
#lists
#dictionary
#tuple
#set
```

how to read the strings

In [1]:
```
string1="python"
string1
```

Out[1]: 'python'

In [3]:
```
string1='python'
string1
```

Out[3]: 'python'

In [ ]:
```
### triple quotes
#doc string is used to say some information about your python code
```

In [5]:
```
"""
im creating a hello function
agrumnets: none
return: none
"""
def hello():
    print("gm")
```

In [ ]:
```
complex()    #shift tab
```

In [ ]:
```
'hello python'    #i want to highlight the python word
```

In [6]:
```
string3='hello "python"'
print(string3)
```

```
hello "python"
```

In [7]:
```
string3="hello 'python'"
print(string3)
```

```
hello 'python'
```

-type -len -max -min

type

In [9]:
```
string1
```

Out[9]: 'python'

In [10]:
```
type(string1)    #str
```

Out[10]:
```
str
```

In [11]:
```
len(string1)   #6 letters
```

Out[11]:
```
6
```

In [12]:
```
max(string1)    #alphabets order
```

Out[12]:
```
'y'
```

In [ ]:
```
string1='p'    #complete
max(string1)
#ASCII
#A=65 a:95
```

In [13]:
```
ord('P')
```

Out[13]:
```
80
```

In [14]:
```
ord('p')
```

Out[14]:
```
112
```

In [15]:
```
string1='python'
max(string1)
```

Out[15]:
```
'y'
```

In [16]:
```
ord('p'),ord('y'),ord('t'),ord('h'),ord('o'),ord('n')
```

Out[16]:
```
(112, 121, 116, 104, 111, 110)
```

In [19]:
```
chr(112),chr(121),chr(116),chr(104),chr(111),chr(110)
```

Out[19]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

how can we implement loop on the string

In [21]:
```
for i in range(len('python')):
    print(i)
```

```
0
1
2
3
4
5
```

In [ ]:
```
#i want to print p y t h o n
```

in operator

In [22]:
```
string1='python'
'p' in string1
```

Out[22]: True

In [23]:
```
'y' in string1
```

Out[23]: True

In [24]:
```
't' in string1
```

Out[24]: True

In [25]:
```
'h' in string1
```

Out[25]: True

In [26]:
```
'o' in string1
```

Out[26]: True

In [27]:
```
'n' in string1   #instead of writing 6 times other way is generalized metho
```

Out[27]: True

In [ ]:
```
#i in string1  #generalized experssion for varring and common letters
```

In [28]:
```
for i in (string1):
    print(i)     #in operator implemneting in for loop
```

```
p
y
t
h
o
n
```

note

-range() :you need to provide numbers inside the range -in : is used only for strings

In [ ]:
```
--if u want to print the letters using for loop go for in operator
```

In [106]:
```
print(ord('p'))
print(ord('y'))
print(ord('t'))
print(ord('h'))
print(ord('o'))
print(ord('n'))
```

```
112
121
116
104
111
110
```

In [30]:
```
for i in (string1):
    print(ord(i))
```

```
112
121
116
104
111
110
```

In [33]:
```
for i in (string1):
    print("the ascii value of{} is{}".format(i,ord(i)))
```

```
the ascii value ofp is112
the ascii value ofy is121
the ascii value oft is116
the ascii value ofh is104
the ascii value ofo is111
the ascii value ofn is110
```

In [107]:
```
#acii value of a to z
for i in range(65,91):
    print("the ascii value of{} is{}".format(chr(i),i))
```

```
the ascii value ofA is65
the ascii value ofB is66
the ascii value ofC is67
the ascii value ofD is68
the ascii value ofE is69
the ascii value ofF is70
the ascii value ofG is71
the ascii value ofH is72
the ascii value ofI is73
the ascii value ofJ is74
the ascii value ofK is75
the ascii value ofL is76
the ascii value ofM is77
the ascii value ofN is78
the ascii value ofO is79
the ascii value ofP is80
the ascii value ofQ is81
the ascii value ofR is82
the ascii value ofS is83
```

In [109]:
```
for i in "ABCDEFGHIJKLMNOPQRSTUVWXY":
    print("the ascii value of{} is{}".format(i,ord(i)))
```

```
the ascii value ofA is65
the ascii value ofB is66
the ascii value ofC is67
the ascii value ofD is68
the ascii value ofE is69
the ascii value ofF is70
the ascii value ofG is71
the ascii value ofH is72
the ascii value ofI is73
the ascii value ofJ is74
the ascii value ofK is75
the ascii value ofL is76
the ascii value ofM is77
the ascii value ofN is78
the ascii value ofO is79
the ascii value ofP is80
the ascii value ofQ is81
the ascii value ofR is82
the ascii value ofS is83
the ascii value ofT is84
the ascii value ofU is85
the ascii value ofV is86
the ascii value ofW is87
the ascii value ofX is88
the ascii value ofY is89
```

In [37]:
```
import string
```

In [38]:
```
dir(string)
```

Out[38]:
```
['Formatter',
 'Template',
 '_ChainMap',
 '__all__',
 '__builtins__',
 '__cached__',
 '__doc__',
 '__file__',
 '__loader__',
 '__name__',
 '__package__',
 '__spec__',
 '_re',
 '_sentinel_dict',
 '_string',
 'ascii_letters',
 'ascii_lowercase',
 'ascii_uppercase',
 'capwords',
 'digits',
 'hexdigits',
 'octdigits',
 'printable',
 'punctuation',
 'whitespace']
```

In [41]:
```
string.ascii_uppercase
```

Out[41]:
```
'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

In [110]:
```
for i in string.ascii_uppercase:
    print("the ascii value of{} is{}".format(i,ord(i)))
```

```
the ascii value ofA is65
the ascii value ofB is66
the ascii value ofC is67
the ascii value ofD is68
the ascii value ofE is69
the ascii value ofF is70
the ascii value ofG is71
the ascii value ofH is72
the ascii value ofI is73
the ascii value ofJ is74
the ascii value ofK is75
the ascii value ofL is76
the ascii value ofM is77
the ascii value ofN is78
the ascii value ofO is79
the ascii value ofP is80
the ascii value ofQ is81
the ascii value ofR is82
the ascii value ofS is83
the ascii value ofT is84
the ascii value ofU is85
the ascii value ofV is86
the ascii value ofW is87
the ascii value ofX is88
the ascii value ofY is89
the ascii value ofZ is90
```

In [111]:
```
for i in string.ascii_lowercase:
    print("the ascii value of{} is{}".format(i,ord(i)))
```

```
the ascii value ofa is97
the ascii value ofb is98
the ascii value ofc is99
the ascii value ofd is100
the ascii value ofe is101
the ascii value off is102
the ascii value ofg is103
the ascii value ofh is104
the ascii value ofi is105
the ascii value ofj is106
the ascii value ofk is107
the ascii value ofl is108
the ascii value ofm is109
the ascii value ofn is110
the ascii value ofo is111
the ascii value ofp is112
the ascii value ofq is113
the ascii value ofr is114
the ascii value ofs is115
the ascii value oft is116
the ascii value ofu is117
the ascii value ofv is118
the ascii value ofw is119
the ascii value ofx is120
the ascii value ofy is121
the ascii value ofz is122
```

In [45]:
```
string.punctuation
```

Out[45]:
```
'!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

In [46]:
```
#start and end of ascii numbers
ord('a')
```

Out[46]:
```
97
```

In [48]:
```
chr(1)
```

Out[48]:
```
'\x01'
```

In [50]:
```
for i in range(1,50):
    print(i,chr(i))
```

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

11 
12 
13 
14 
15 
16 
17 
18 
19 
20 
21 
22 
23 
24 
25 
26 
27 
28 
29 
30 
31 
32 
33 !
34 "
35 #
36 $
37 %
38 &
39 '
40 (
41 )
42 *
43 +
44 ,
45 -
46 .
47 /
48 0
49 1
```

In [51]:
```
for i in range(100,150):
    print(i,chr(i))
```

```
100 d
101 e
102 f
103 g
104 h
105 i
106 j
107 k
108 l
109 m
110 n
111 o
112 p
113 q
114 r
115 s
116 t
117 u
118 v
119 w
120 x
121 y
122 z
123 {
124 |
125 }
126 ~
127 
128 
129 
130 
131 
132 
133 
134 
135 
136 
137 
138 
139 
140 
141 
142 
143 
144 
145 
146 
147 
148 
149 
```

In [52]:
```
for i in range(33,126):
    print(i,chr(i))
#start with 33
#end with 126
#ascii:93
```

```
33 !
34 "
35 #
36 $
37 %
38 &
39 '
40 (
41 )
42 *
43 +
44 ,
45 -
46 .
47 /
48 0
49 1
50 2
51 3
52 4
53 5
54 6
55 7
56 8
57 9
58 :
59 ;
60 <
61 =
62 >
63 ?
64 @
65 A
66 B
67 C
68 D
69 E
70 F
71 G
72 H
73 I
74 J
75 K
76 L
77 M
78 N
79 O
80 P
81 Q
82 R
83 S
84 T
85 U
86 V
87 W
88 X
89 Y
90 Z
91 [
92 \
93 ]
```

```
94 ^
95 _
96 `
97 a
98 b
99 c
100 d
101 e
102 f
103 g
104 h
105 i
106 j
107 k
108 l
109 m
110 n
111 o
112 p
113 q
114 r
115 s
116 t
117 u
118 v
119 w
120 x
121 y
122 z
123 {
124 |
125 }
```

In [53]:
```
string.ascii_letters
```

Out[53]:
```
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

In [54]:
```
string.printable
#digits
#lower
#upp
#punctuation
```

Out[54]:
```
'0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!"#$%&\'()*
+,-./:;<=>?@[\\]^_`{|}~ \t\n\r\x0b\x0c'
```

In [114]:
```
#wap ask the user find number of a letters in a given string
#string=hai how are you and how do u do
count1=0
string4="hai how are you and how do u do"
for i in string4:
    if i=="a":
        print(i)
        count1=count1+1
print(count1)
```

```
a
a
a
3
```

In [76]:
```
#wap count the number of vowels in a given string
string5="hai how are you"
count=0
for i in string5:
    if i in"aeiou":
        count=count+1
print(count)
```

```
7
```

In [137]:
```
#wap count the number of vowels in a given string
string5="hai how are you"
str6=""
count=0
for i in string5:
    if i in"aeiou":
        count=count+1
        str6=str6+i
print(count)
print(str6)
```

```
7
aioaeou
```

In [140]:
```
string6="hai how are you"
str7=""
count1=0
for i in string6:
    print(i)
    if i in"aeiou" and i not in str7:
        count1=count1+1
        str7=str7+i
print(str7)
print(count1)
```

```
h
a
i

h
o
w

a
r
e

y
o
u
aioeu
5
```

In [143]:
```
string6="hai how are you"
str7=""
count1=0
for i in string6:
    print(i)
    if i in"aeiou" :
          if i not in str7:
            count1=count1+1
            str7=str7+i
print(str7)
print(count1)
```

```
h
a
i

h
o
w

a
r
e

y
o
u
aioeu
5
```

In [131]:
```
str1='hai'
str2='how'
str1+str2
```

Out[131]:
```
'haihow'
```

In [90]:
```
(str1*str2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[90], line 1
----> 1 (str1*str2)

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

In [89]:
```
3*str2
```

Out[89]:
```
'howhowhow'
```

In [ ]:
```
-how to read
-single/double/triple
-type
-len
-max
-in
-concatenation
```

index

In [91]:
```
string1="python"
```

In [ ]:
```
p y t h o n
0 1 2 3 4 5
```

In [97]:
```
string1[0],string1[1],string1[2],string1[3],string1[4],string1[5]
string1[i]
```

Out[97]:
```
'o'
```

In [93]:
```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

In [104]:
```
for i in range(5):
    print(string1[i])
```

```
p
y
t
h
o
```

In [105]:
```
strin1="python"
#i want to print letters using in operator
#i wnt to print letters range operator
for i in string1:
    print(i)
for i in range(len(string1)):
    print(i,string1[i])
```

```
p
y
t
h
o
n
0 p
1 y
2 t
3 h
4 o
5 n
```

In [ ]:
```
#if youn want to print only letters:in
#if you want to print only idex:range
#if you want to print both index and range:use range
```

10-11-2023

In [2]:
```
string1='python'
#i want to print the letters using for loop
#in
#range
```

In [6]:
```
for i in string1:     #i means each letter
    print(i,end=" ")
for i in range(len('python')):  #i means numbers 0 to len-1
    print(i)
```

```
p y t h o n 0
1
2
3
4
5
```

In [8]:
```
for i in string1:     #i means each letter
    print(i,end=" ")
for i in range(len('python')):  #i means numbers 0 to len-1
    print(i,string1[2])
```

```
p y t h o n 0 t
1 t
2 t
3 t
4 t
5 t
```

In [10]:
```
string1[2]
```

Out[10]:
```
't'
```

In [14]:
```
#the index of p is zero
#the index of y is 1
for i in string1:     #i means each letter
    print(i,end=" ")
for i in range(len('python')):  #i means numbers 0 to len-1
    print("the index of {} is{}".format(string1[i],i))
```

```
p y t h o n the index of p is0
the index of y is1
the index of t is2
the index of h is3
the index of o is4
the index of n is5
```

In [ ]:
```
-6-5-4-3-2-1==========>negative index
p y  t h o n
0 1 2 3 4 5=========>positive index
```

In [15]:
```
string1[-6],string1[-5],string1[-4],string1[-3],string1[-2],string1[-1]
```

Out[15]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

In [21]:
```
#iterate the for loop on string1
#print the letters using negative index
#the idea is first you need print the numbers between-6 to -1 using for loo
string2="python"
for i in string2:
    print(i,end=" ")
for i in range(-6,0):
    print(i,string2[i])
```

```
p y t h o n -6 p
-5 y
-4 t
-3 h
-2 o
-1 n
```

In [24]:
```
string2="python"
for i in string2:
    print(i,end=" ")
for i in range(-len(string2),0):
    print("the negative index {} is{}".format(string2[i],i))
```

```
p y t h o n the negative index p is-6
the negative index y is-5
the negative index t is-4
the negative index h is-3
the negative index o is-2
the negative index n is-1
```

In [26]:
```
for i in range(-6,0):
    print(i,end="")
```

```
-6-5-4-3-2-1
```

In [29]:
```
for i in range(6,0):
    print(i,end="")
```

In [30]:
```
for i in range(0,6):
    print(i-6,end=" ")
```

```
-6 -5 -4 -3 -2 -1
```

In [34]:
```
string2="python"
for i in string2:
    print(i,end=" ")
for i in range(len(string2)):
    print("the negative index {} is{}".format(string2[i],i-6))
```

```
p y t h o n the negative index p is-6
the negative index y is-5
the negative index t is-4
the negative index h is-3
the negative index o is-2
the negative index n is-1
```

In [38]:
```
string2="python"
for i in string2:
    print(i,end=" ")
for i in range(len(string2)):
    print("the  positive index is{} and negative index is {}".format(i,stri
```

```
p y t h o n the  positive index is0 and negative index is p
the  positive index is1 and negative index is y
the  positive index is2 and negative index is t
the  positive index is3 and negative index is h
the  positive index is4 and negative index is o
the  positive index is5 and negative index is n
```

In [41]:
```
string2="python"
for i in string2:
    print(i,end=" ")
for i in range(-len(string2),0):
    print("the negative index {} is{}".format(string2[i],i+6))


string2="python"
for i in string2:
    print(i,end=" ")
for i in range(-len(string2),0):
    print("the negative index {} is{}".format(string2[i],i))


string2="python"
for i in string2:
    print(i,end=" ")
for i in range(-len(string2),0):
    print("the  positive index is{} and negative index is {}".format(i+6,st
```

```
p y t h o n the negative index p is0
the negative index y is1
the negative index t is2
the negative index h is3
the negative index o is4
the negative index n is5
p y t h o n the negative index p is-6
the negative index y is-5
the negative index t is-4
the negative index h is-3
the negative index o is-2
the negative index n is-1
p y t h o n the  positive index is0 and negative index is p
the  positive index is1 and negative index is y
the  positive index is2 and negative index is t
the  positive index is3 and negative index is h
the  positive index is4 and negative index is o
the  positive index is5 and negative index is n
```

In [61]:
```
#string2="python"
#for i in string2:
 #   print(i,end=" ")
#for i in range(-len(string2),0):
 #   print("the negative index {} is{}".format(string2[i],i+6))
#while loop
i=0
string2="python"
while (i<len(string2)):
    print("the negative index {} is{}".format(i,i-6,string2[i],))
    i=i+1
```

```
the negative index 0 is-6
the negative index 1 is-5
the negative index 2 is-4
the negative index 3 is-3
the negative index 4 is-2
the negative index 5 is-1
```

In [68]:
```
string3="hai how are you"
count=0
for i in range(len(string3)):
    if string3[i]=="a":
        count+=1
print(count)
```

```
2
```

In [ ]:
```
mutable and immutable concept
mutable=======we can change
immutable=====we can not change the value index operation
strings are immutable
```

In [69]:
```
string1="python"
#i want change 'p'==========="P"
#o/p:"Python"
string1[0]="p"
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[69], line 4
      1 string1="python"
      2 #i want change 'p'==========="P"
      3 #o/p:"Python"
----> 4 string1[0]="p"

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

In [70]:
```
list1=[100,200,300] #100=====1000
list1[0]=1000
list1
```

Out[70]:
```
[1000, 200, 300]
```

slice

In [71]:
```
string1="hai how are you"
string1[2:10]
```

Out[71]:
```
'i how ar'
```

In [73]:
```
string1[2:10:3]
```

Out[73]:
```
'ioa'
```

In [78]:
```
string1[:] #full string will print
```

Out[78]:
```
'hai how are you'
```

In [75]:
```
string1[::]  #full string will print
```

Out[75]:
```
'hai how are you'
```

In [76]:
```
string1[2:10:-3]
```

Out[76]:
```
''
```

In [77]:
```
for i in range(2,10,-3):
    print(i)
```

In [ ]:
```
 -15 -14 -13  -12   -11 -10 -9   -8 -7  -6  -5  -4  -3  -2  -1
   h   a   i         h    o   w      a   r   e       y   o   u
  0    1   2   3     4    5    6   7  8   9   10  11  12  13 14
```

In [119]:
```
string1="hai how are you"
string1[-10]
```

Out[119]:
```
'o'
```

In [ ]:
```
string1[2:14:2] #p
string1[2:14:-2]#np
string1[2:-14:2]#np
string1[2:-14:-2]#
string1[-2:14:2]#p
string1[-2:-14:2]#np
string1[-2:-14:-2]#p
```

In [82]:
```
string1[2:-14:-2]
#start=2
#stop=-14+1=-13
#2
```

Out[82]:
```
'i'
```

In [83]:
```
string1[3:-14:-2]
#start=2
#stop=-14+1=-13
#2
```

Out[83]:
```
' '
```

In [ ]:
```
-reading methods
-single/double/triple(doc string)
-type/len/max/min
-concatenation
-subtraction/mul/div
-in
-index
-mutable
-slice
```

string methods

In [ ]:
```
import <package_name>
dir(<package_name>)
help(<package_name.<method_name>)
```

In [84]:
```
dir('')
```

Out[84]:
```
['__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']
```

capitalized

In [85]:
```
string1="welcome"
string1.capitalize()
#only starting letter will be captale
```

Out[85]: 'Welcome'

In [86]:
```
string1="welcome"
string1.upper()
```

Out[86]: 'WELCOME'

In [87]:
```
string1.lower()
```

Out[87]: 'welcome'

In [90]:
```
string1="weLCome"
print(string1.capitalize())
print(string1.upper())
print(string1.lower())
```

```
Welcome
WELCOME
welcome
```

In [92]:
```
string1="hai how are you"
string1.count('a')
```

Out[92]: 2

In [95]:
```
count=0
for i in string1:

    if i=="a":
        count+=1
print(count)
```

2

In [115]:
```
#string1="welcome"
#welcome
#index
#slice
#concatenation

string1="welcome"
str1=(string1[:2])
str2=(string1[3:])
str1+"L"+str2
```

Out[115]: 'weLcome'

In [116]:
```
string1.replace("l","L")
```

Out[116]: 'weLcome'

In [120]:
```
string1[-2:14:2]#p
```

Out[120]: 'o'

In [ ]:
```
 -15 -14 -13  -12   -11 -10 -9   -8 -7  -6  -5  -4  -3  -2  -1
   h   a   i         h    o   w      a   r   e       y   o   u
  0    1   2   3     4    5    6   7  8   9   10  11  12  13 14
```

14-11-2023

casefold

In [1]:
```
string1="welcome"
string1.casefold()
```

Out[1]: 'welcome'

In [2]:
```
string2="welcome"
string2
```

Out[2]: 'welcome'

lower case and casefold both are same

In [ ]:
```
-capitalize -first letter as capital
-upper     -all letters are in upper case
-lower     -all letters are in lower case
-casefold   -case less comparision(lower case)
```

count

In [3]:
```
string1="hai how are you"#how many "A" are there
string1.count("a")
```

Out[3]: 2

In [4]:
```
string1.count("hai")
```

Out[4]: 1

In [12]:
```
string1="ola ola ola"
print(string1.count('ola'))   #3
print(string1.count(' ola'))  #2
print(string1.count('ol'))   #3
print(string1.count('oa'))   #0
```

```
3
2
3
0
```

In [19]:
```
string1='ola ola ola'
#ola   ola    ola
#012 3 456 7  8910
print(string1.count("a",4))   #we are counting the number of "a" from index
print(string1.count("a",6))
print(string1.count("a",4,6))
print(string1.count("a",4,7))
```

```
2
2
0
1
```

In [20]:
```
print(string1.count("A".lower(),4))
```

```
2
```

In [22]:
```
string1="ola ola ola"
count=0
for i in string1:
    if i=="a":
        count=count+1
print(count)
```

```
3
```

In [30]:
```
string1="ola ola ola"
count=0
for i in range(len(string1)):
    if string1[i:i+3]=="ola":
        count=count+1
print(count)
```

```
3
```

In [ ]:
```
#i=0 string1[0:3]: 0 1 2:ola
#i=1 string1[1:4]: 1 2 3:la
#single letter in operator
#multi letters slice operator
```

In [31]:
```
string1.count("a")   #this a developer
string1.count("ola")   #as developer mode
```

Out[31]:
```
3
```

replace

In [32]:
```
string1="welcome"
#replace "l" with "L"
string1.replace("l","L")
```

Out[32]:
```
'weLcome'
```

In [33]:
```
string1="restart"
string1.replace("r","$")
```

Out[33]:
```
'$esta$t'
```

In [34]:
```
string1="restart"
string1.replace("r","$",1)
#str.replace() will not take key word arguments ---count=1
```

Out[34]:
```
'$estart'
```

In [35]:
```
string1="restart  rrr"
string1.replace("r","$",-1)
#-1 means all the occuranceses
```

Out[35]:
```
'$esta$t  $$$'
```

In [47]:
```
string1="restart"
s1=string1[0]
s2=string1[1:].replace("r","$")
s1+s2
```

Out[47]:
```
'resta$t'
```

In [48]:
```
string1[::-1].replace("r","$",1)[::-1]
```

Out[48]:
```
'resta$t'
```

index

In [53]:
```
string1="welcome python"
string1.index("c")
```

Out[53]:
```
3
```

In [54]:
```
string1="welcome python"
string1.index("z")
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[54], line 2
      1 string1="welcome python"
----> 2 string1.index("z")

ValueError: substring not found
```

In [56]:
```
string1="hai how are you and"
count=0
for i in string1:
    if i=="a":
        count+=1
print(count)
string1.count("a")
```

```
3
```

Out[56]:
```
3
```

In [60]:
```
for i in range(len(string1)):
    if string1[1]=="a":
        print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
```

In [62]:
```
string1.index("a")#only first occurance
```

Out[62]:
```
1
```

In [63]:
```
string1="hai hai hai hai hai"
i1=string1.index("a")
i2=string1.index("a",i1+1) #1+1=5
i3=string1.index("a",i2+1)  #5+1===9
i4=string1.index("a",i3+1)   #9+1===10
i5=string1.index("a",i4+1)
print(i1,i2,i3,i4,i5)
```

```
1 5 9 13 17
```

In [64]:
```
string1.index("a",string1.index("a")+1)
```

Out[64]:
```
5
```

In [74]:
```
string1="welcome helo"
string1.index("l")
string1.index("l",string1.index("l")+1)
string1.index("l",string1.index("l",string1.index("l")+1)+1)
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[74], line 4
      2 string1.index("l")
      3 string1.index("l",string1.index("l")+1)
----> 4 string1.index("l",string1.index("l",string1.index("l")+1)+1)

ValueError: substring not found
```

find

In [79]:
```
string1="hello"
string1.find("e")
```

Out[79]: 1

In [81]:
```
string1="hello"
string1.find("z")
#if substring not found it returns -1
string1.index("z")
#valueerror:substring not found
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last
t)
Cell In[81], line 4
      2 string1.find("z")
      3 #if substring not found it returns -1
----> 4 string1.index("z")

ValueError: substring not found
```

In [82]:
```
string1.count("z")
#zero
```

Out[82]: 0

In [ ]:
```
-capitalize/upper/lower/casefold
-index/find
-count
-replace
```