How to read the strings

In [1]:
```
string1='python'   # single quote
string1
```

Out[1]:
```
'python'
```

In [2]:
```
string2="python"  # double quotes
string2
```

Out[2]:
```
'python'
```

In [ ]:
```
### Triple quotes

# Doc string is used to say some information about your python code
```

In [ ]:
```
"""
im creating a hello function
arguments: None
return:   None
"""
def hello():
    print("good moring")
```

In [ ]:
```
'hello python'  # I want highlite the python
```

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

```
hello "python"
```

In [7]:
```
string4="hello 'python'"
print(string4)
```

```
hello 'python'
```

- type
- len
- max
- min

type:

In [8]:
```
string1
```

Out[8]:
```
'python'
```

In [9]:
```
type(string1)   # str
```

Out[9]:
```
str
```

len

In [10]:
```
len(string1)
```

Out[10]:
```
6
```

max-min

In [11]:
```
string1='pP'
max(string1)   # python

# ASCII
# 'A': 65    'a':97
```

Out[11]:
```
'p'
```

ord-chr

In [12]:
```
ord('p')  # It will provide ascii value of char
```

Out[12]:
```
112
```

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

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

In [16]:
```
string1='python'
max(string1),min(string1)
```

Out[16]:
```
('y', 'h')
```

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

Out[15]:
```
(112, 121, 116, 104, 111, 110)
```

In [22]:
```
chr(112),chr(121),chr(116),chr(104),chr(111),chr(110)
```

Out[22]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

In [25]:
```
for i in range(len('python')):
    print(i)
```

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

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

in

In [27]:
```
string1='python'

'p' in string1
'y' in string1
't' in string1
'h' in string1
'o' in string1
'n' in string1

#i in string1
```

Out[27]:
```
True
```

In [28]:
```
for i in string1:
    print(i)
```

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

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

if you want print the letters using for loop go for in operator

In [29]:
```
print(ord('p'))
print(ord('y'))
print(ord('t'))
print(ord('h'))
print(ord('o'))
print(ord('n'))

print(ord(i))
```

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

In [31]:
```
for i in string1:
    print("the ascii value of {} is {}".format(i,ord(i)))
    # The ascii value of p is 112
```

```
the ascii value of p is 112
the ascii value of y is 121
the ascii value of t is 116
the ascii value of h is 104
the ascii value of o is 111
the ascii value of n is 110
```

In [ ]:
```
# Ascii value of A to Z
```

In [32]:
```
for i in 'ABCDEFGHIJKLMNOPQRSTUVWXYZ':
    print("the ascii value of {} is {}".format(i,ord(i)))
```

```
the ascii value of A is 65
the ascii value of B is 66
the ascii value of C is 67
the ascii value of D is 68
the ascii value of E is 69
the ascii value of F is 70
the ascii value of G is 71
the ascii value of H is 72
the ascii value of I is 73
the ascii value of J is 74
the ascii value of K is 75
the ascii value of L is 76
the ascii value of M is 77
the ascii value of N is 78
the ascii value of O is 79
the ascii value of P is 80
the ascii value of Q is 81
the ascii value of R is 82
the ascii value of S is 83
the ascii value of T is 84
the ascii value of U is 85
the ascii value of V is 86
the ascii value of W is 87
the ascii value of X is 88
the ascii value of Y is 89
the ascii value of Z is 90
```

In [33]:
```
# package_name: string
import string
```

In [34]:
```
dir(string)
```

Out[34]:
```
['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 [35]:
```
# see the output of : ascii_uppercase
string.ascii_uppercase
```

Out[35]:
```
'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

In [36]:
```
for i in string.ascii_uppercase:
    print("the ascii value of {} is {}".format(i,ord(i)))
```

```
the ascii value of A is 65
the ascii value of B is 66
the ascii value of C is 67
the ascii value of D is 68
the ascii value of E is 69
the ascii value of F is 70
the ascii value of G is 71
the ascii value of H is 72
the ascii value of I is 73
the ascii value of J is 74
the ascii value of K is 75
the ascii value of L is 76
the ascii value of M is 77
the ascii value of N is 78
the ascii value of O is 79
the ascii value of P is 80
the ascii value of Q is 81
the ascii value of R is 82
the ascii value of S is 83
the ascii value of T is 84
the ascii value of U is 85
the ascii value of V is 86
the ascii value of W is 87
the ascii value of X is 88
the ascii value of Y is 89
the ascii value of Z is 90
```

In [37]:
```
for i in string.ascii_lowercase:
    print("the ascii value of {} is {}".format(i,ord(i)))
```

```
the ascii value of a is 97
the ascii value of b is 98
the ascii value of c is 99
the ascii value of d is 100
the ascii value of e is 101
the ascii value of f is 102
the ascii value of g is 103
the ascii value of h is 104
the ascii value of i is 105
the ascii value of j is 106
the ascii value of k is 107
the ascii value of l is 108
the ascii value of m is 109
the ascii value of n is 110
the ascii value of o is 111
the ascii value of p is 112
the ascii value of q is 113
the ascii value of r is 114
the ascii value of s is 115
the ascii value of t is 116
the ascii value of u is 117
the ascii value of v is 118
the ascii value of w is 119
the ascii value of x is 120
the ascii value of y is 121
the ascii value of z is 122
```

In [38]:
```
for i in string.punctuation:
    print("the ascii value of {} is {}".format(i,ord(i)))
```

```
the ascii value of ! is 33
the ascii value of " is 34
the ascii value of # is 35
the ascii value of $ is 36
the ascii value of % is 37
the ascii value of & is 38
the ascii value of ' is 39
the ascii value of ( is 40
the ascii value of ) is 41
the ascii value of * is 42
the ascii value of + is 43
the ascii value of , is 44
the ascii value of - is 45
the ascii value of . is 46
the ascii value of / is 47
the ascii value of : is 58
the ascii value of ; is 59
the ascii value of < is 60
the ascii value of = is 61
the ascii value of > is 62
the ascii value of ? is 63
the ascii value of @ is 64
the ascii value of [ is 91
the ascii value of \ is 92
the ascii value of ] is 93
the ascii value of ^ is 94
the ascii value of _ is 95
the ascii value of ` is 96
the ascii value of { is 123
the ascii value of | is 124
the ascii value of } is 125
the ascii value of ~ is 126
```

In [39]:
```
# what is the start and end of ascii numbers?

ord('a')
```

Out[39]:
```
97
```

In [49]:
```
for i in range(983,1000):
    print(i,chr(i))

# start with 33
# end with 126
# ascii:
```

```
983 ϗ
984 Ϙ
985 ϙ
986 Ϛ
987 ϛ
988 Ϝ
989 ϝ
990 Ϟ
991 ϟ
992 Ϡ
993 ϡ
994 Ϣ
995 ϣ
996 Ϥ
997 ϥ
998 Ϧ
999 ϧ
```

In [50]:
```
string.ascii_letters
```

Out[50]:
```
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

In [51]:
```
string.printable

# digts
# lower
# upp
# punct
```

Out[51]:
```
'0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!"#$%&\'()*
+,-./:;<=>?@[\\]^_`{|}~ \t\n\r\x0b\x0c'
```

In [54]:
```
# WAP ask the user find number of 'a' letters in a given strinh
# string='hai how are you and how do you do'
# ans:3

# count=0
# first iterate the string using in and for loop
# print each letter ==== > i
# apply the if condition  i=='a'
#                         count+=1
string='hai how are you and how do you do'
count=0
for i in string:
    #print(i)
    if i=='a':
        count=count+1

print(count)

# step-1:  i='h'  'h'=='a'  F
# step-2:  i='a'   'a'=='a' T === > count=1
```

```
3
```

In [55]:
```
# WAP count the number of vowels in a given string
# string='hai how are you'
# 7

string='hai how are you'
count=0
for i in string:
    #print(i)
    if i in 'aeiou':
        count=count+1

print(count)
```

```
7
```

In [60]:
```
# WAP count the number of unique vowels in a given string
# ans:5
string1=''
for i in 'python':
    string1=string1+i

print(string1)
```

```
python
```

In [61]:
```
string='hai how are you'
str1=''
count=0
for i in string:
    if i in 'aeiou':
        count=count+1

print(count)
```

```
7
```

concatenation

In [62]:
```
str1='hai'
str2='how'
str1+str2
```

Out[62]:
```
'haihow'
```

In [ ]:
```
str1-str2 #
str1*str2 #
str1/str2 #
```

In [66]:
```
str1-str2
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[66], line 1
----> 1 str1-str2

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

In [68]:
```
#can't multiply sequence by non-int of type 'str'

3*str1
```

Out[68]:
```
'haihaihai'
```

In [69]:
```
str1+str1+str1
```

Out[69]:
```
'haihaihai'
```

In [ ]:
```
- how to read

- single/double/triple

- type

- len

- in (how to iterate through for loop)

- max

- min

- concatenation
```

index

In [70]:
```
string1='python'  # 6 letters
```

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

In [75]:
```
string1[0],string1[1],string1[2],string1[3],string1[4],string1[5]

string1[i]
```

Out[75]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

In [78]:
```
for i in range(5):
    print(string1[i])
#i=0 ====== string1[0]====='p'
#i=1======= string1[1]====='y'

# i ======= will give index
# string[i] ======= letters
```

```
p
y
t
h
o
```

In [81]:
```
string1='python'

# i want to print letters using in opertor
# i want to print letters using 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 [83]:
```
name='python class'

for i in name:
    print("The index no of '{}' is {}".format(name[i],i))

    # i='p'  name['p'],'p'

# if you want print only letter : in
# if you want print index: range
# if you want print index as well as letter: range
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[83], line 4
      1 name='python class'
      3 for i in name:
----> 4     print("The index no of '{}' is {}".format(name[i],i))

TypeError: string indices must be integers, not 'str'
```

- How to read the strings
- doc string
- type/len/max/min
- concatenation
- index

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

# I want to print the letters using for loop
# in
# range
```

In [11]:
```
for i in string1:   # i means each letter
    print(i,end=' ')


for i in range(len(string1)):  # i means numbers o
    print(i,string1[i])
```

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

In [12]:
```
'the index of p is 0'
'the index of y is 1'
```

Out[12]:
```
'the index of y is 1'
```

In [36]:
```
for i in range(len(string1)):  # i means numbers o
    print('the postive index of {} is {}'.format(string1[i],i))
```

```
the postive index of p is 0
the postive index of y is 1
the postive index of t is 2
the postive index of h is 3
the postive index of o is 4
the postive index of n is 5
```

In [ ]:
```

-6  -5   -4    -3   -2   -1
p    y    t     h    o    n
0    1    2     3    4    5 ======= > pos
```

In [15]:
```
string1[-1]
```

Out[15]:
```
'n'
```

In [22]:
```
string1[-6], string1[-5],string1[-4],string1[-3],string1[-2],string1[-1]
```

Out[22]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

In [27]:
```
# iterate the for loop on string1
# print the letters using negative index
# the idea is first you need to print the numers between -6 to -1 using for

string1='python'
for i in range(-len(string1),0):
    print("the negative index of {} is {}".format(string1[i],i))
```

```
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 [32]:
```
string1='python'
for i in range(len(string1)):
    print("the negative index of {} is {}".format(string1[i],i-6))
```

```
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 [35]:
```
string1='python'
for i in range(len(string1)):
    print("the postive index is {} and negative index is {} for {}".format(
```

```
the postive index is 0 and negative index is p for -6
the postive index is 1 and negative index is y for -5
the postive index is 2 and negative index is t for -4
the postive index is 3 and negative index is h for -3
the postive index is 4 and negative index is o for -2
the postive index is 5 and negative index is n for -1
```

In [31]:
```
#for i in range(-6,0):print(i,end=' ')
for i in range(0,6):print(i-6,end=' ')  # 0 --------- -6   i-6
                                        #  1 --------- -5
```

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

In [ ]:
```
for i in range(len(string1)):
    print('the postive index of {} is {}'.format(string1[i],i))


for i in range(len(string1)):
    print("the negative index of {} is {}".format(string1[i],i-6))


for i in range(len(string1)):
    print("the postive index is {} and negative index is {} for {}".format(
```

In [40]:
```
for i in range(-len(string1),0):
    print('the postive index of {} is {}'.format(string1[i],i+6))


for i in range(-len(string1),0):
    print("the negative index of {} is {}".format(string1[i],i))


for i in range(-len(string1),0):
    print("the postive index is {} and negative index is {} for {}".format(
```

```
the postive index of p is 0
the postive index of y is 1
the postive index of t is 2
the postive index of h is 3
the postive index of o is 4
the postive index of n is 5
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
the postive index is 0 and negative index is p for -6
the postive index is 1 and negative index is y for -5
the postive index is 2 and negative index is t for -4
the postive index is 3 and negative index is h for -3
the postive index is 4 and negative index is o for -2
the postive index is 5 and negative index is n for -1
```

In [39]:
```
for i in range(-len(string1),0):
    print(i,i+6)
```

```
-6 0
-5 1
-4 2
-3 3
-2 4
-1 5
```

In [43]:
```
i=0
while i<len(string1):
    print("the postive index is {} and negative index is {} for {}".format(
    i=i+1
```

```
the postive index is 0 and negative index is -6 for p
the postive index is 1 and negative index is -5 for y
the postive index is 2 and negative index is -4 for t
the postive index is 3 and negative index is -3 for h
the postive index is 4 and negative index is -2 for o
the postive index is 5 and negative index is -1 for n
```

In [47]:
```
string1='hai how are you'
count=0
for i in string1:
    if i=='a':
        count+=1
print(count)
```

```
2
```

In [46]:
```
string1='hai how are you'
count=0
for i in range(len(string1)):
    if string1[i]=='a':
        count+=1
print(count)
```

```
2
```

Mutable and immutable concept

mutable ===== we can change

immutable ===== we can not change the value by using index operations

strings are immutable

In [50]:
```
string1='python'
# I want change  'p'  ====== 'P'
# o/p: 'Python'
string1[0]='P'
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[50], 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 [53]:
```
list1=[100,200,300]  # 100 ====== 1000
list1[0]=1000
list1
```

Out[53]:
```
[1000, 200, 300]
```

slice

In [ ]:
```
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 [55]:
```
string1='hai how are you'
string1[2:10]
# start=2
# stop=10-1=9
# postive
```

Out[55]: 'i how ar'

In [56]:
```
string1='hai how are you'
string1[2:10:3]
# start=2
# dire=+  step=3
# stop=10-1=9
# 2  5  8
```

Out[56]: 'ioa'

In [57]:
```
string1[:]

# nothing mentioned means take start and last letter
```

Out[57]: 'hai how are you'

In [58]:
```
string1[::]
# start= 0
# stop= last letter len(string1)
# step= +1
```

Out[58]: 'hai how are you'

In [ ]:
```
string1='hai how are you'
string1[2:10]
string1[2:10:3]
string1[:]
string1[::]
```

In [59]:
```
string1[2:10:-3]
```

Out[59]: ''

In [60]:
```
for i in range(2,10,-3):print(i)
```

In [61]:
```
len(string1)
```

Out[61]: 15

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 [65]:
```
string1[8:-15:-2]  # start=8  stop =-15+1=-14  step=2
```

Out[65]:
```
'awhi'
```

In [69]:
```
string1[-2:-14:-2]   # empty   i

# start=2
# stop=-14+1=-13
#
# ' '
```

Out[69]:
```
'o r 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]
```

In [63]:
```
for i in range(2,-14,-2):print(i,end=' ')
```

```
2 0 -2 -4 -6 -8 -10 -12
```

In [ ]:
```

```

- reading methods
- single/double/triple(doc string)
- type
- len
- max
- min
- concatenation
- subtraction/mult/div
- in
- index
- mutable
- slice

String Methods

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

In [70]:
```
dir('')
```

```
Out[70]: ['__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']
```

capitalize

In [77]:
```
string1='welcome'
string1.capitalize()
```

Out[77]:
```
'Welcome'
```

upper

In [78]:
```
string1.upper()
```

Out[78]:
```
'WELCOME'
```

lower

In [79]:
```
string1.lower()
```

Out[79]:
```
'welcome'
```

In [80]:
```
string1='weLCome'
print(string1.capitalize())
print(string1.upper())
print(string1.lower())
```

```
Welcome
WELCOME
welcome
```

In [81]:
```
string1='hai how are you'
# how many 'a' s are there
```

In [82]:
```
string1.count('a')
```

Out[82]: 2

In [83]:
```
count=0
for i in string1:
    if i=='a':
        count+=1
```

In [89]:
```
string1='welcome'
# output: 'weLcome'
# idea:
# str1=we
# str2=come
# index
# slice
# concatenation= str1+'L'+str2
```

In [93]:
```
string1='welcome'
str1=string1[:2]
str2=string1[3:]
str1+'L'+str2
```

Out[93]: 'weLcome'

In [94]:
```
string1.replace('l','L')
```

Out[94]: 'weLcome'

casefold

In [1]:
```
string1='WelCome'
string1.casefold()
```

Out[1]: 'welcome'

In [2]:
```
string2='welcome'
string2.casefold()
```

Out[2]: 'welcome'

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 [8]:
```
string1='ola ola ola'
print(string1.count('ola'))  # 3
print(string1.count('ola ')) # 2
print(string1.count('ol'))  # 3
print(string1.count('oa'))  # check 'o' after 'a' is there?
```

```
3
2
3
0
```

In [19]:
```
string1='ola ola ola'
# ola   ola   ola
# 012 3 456 7 8910
# we are counting the number of 'a' from index 4
print(string1.count('a',4))
print(string1.count('a',6))
print(string1.count('a',4,6))  # 4 and 5
print(string1.count('a',4,7))  # 4 5 6
print(string1.count('A',4))    # 0
print(string1.count('A'.lower(),4))  # 2
print(string1.count('a'.upper(),4,7))
```

```
2
2
0
1
0
2
0
```

In [ ]:
```
'ola ola ola'.count('a')
```

In [ ]:
```
# sir any ignore case function
```

In [21]:
```
string1='ola ola ola'
count=0
for i in string1:
    if i=='a':
        count+=1

print(count)
```

```
3
```

In [22]:
```
# How many 'ola' are there
# do till now what we completed use that only except count method
count = 0
word = ''
for i in string1:
    if i != ' ':
        word+=i
    if word == 'ola':
        count+=1
        word=''

count
```

Out[22]: 3

In [23]:
```
string1='ola ola ola'
count=0
for i in range(len(string1)):
    if string1[i:i+3]=='ola':
        count+=1

print(count)
# i=0   string1[0:3]: 0 1 2  : ola
# i=1   string1[1:4]: 1 2 3  : la
```

3

In [24]:
```
string1.count('a') # this as developer
string1.count('ola') # as developer mode
```

Out[24]: 3

replace

In [29]:
```
string1='welcome'
# replace 'l' with 'L'
string1.replace('l','@')
```

Out[29]: 'we@come'

In [30]:
```
string1='restart'
# replace 'r' with '$'
string1.replace('r','$')
```

Out[30]: '$esta$t'

In [40]:
```
string1='restart'
# replace 'r' with '$'
string1.replace('r','$',1)
```

Out[40]: '$estart'

In [ ]:
```
string1='restart'
# o/p: 'resta$t'

# Do only with what we completed till now
```

In [26]:
```
len(string1)
max(string1)  # string1.max()
```

Out[26]: 11

In [45]:
```
string1='restart'
s1=string1[0]
s2=string1[1:].replace('r','$')
s1+s2
```

Out[45]: 'resta$t'

In [49]:
```
string1[::-1].replace('r','$',1)[::-1]
```

Out[49]: 'resta$t'

index

In [50]:
```
string1='welcome python'

# index of 'c'
string1.index('c')
```

Out[50]: 3

In [51]:
```
string1.index('z') # sub string not found
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[51], line 1
----> 1 string1.index('z')

ValueError: substring not found
```

In [53]:
```
string1='hai how are you and'
# how many 'a' are there
# what are the indexes of 'a'

count=0
for i in string1:
    if i=='a':
        count+=1

print(count)

string1.count('a')
```

```
3
```

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

In [54]:
```
for i in range(len(string1)):
    if string1[i]=='a':
        print(i)
```

```
1
8
16
```

In [55]:
```
string1.index('a')  # only first occurence
```

Out[55]:
```
1
```

In [65]:
```
string1='hai hai hai hai hai'
i1=string1.index('a') # 1
i2=string1.index('a',i1+1)
i3=string1.index('a',i2+1)
i4=string1.index('a',i3+1)
i5=string1.index('a',i4+1)
print(i1,i2,i3,i4,i5)
```

```
1 5 9 13 17
```

In [76]:
```
string1='hai hai hai hai hai'
# i want to first occurence of 'a'
i1=string1.index('a')
# second occurence of 'a'
i2=string1.index('a',i1+1) # 5
string1.index('a',i2+1)

string1.index('a',string1.index('a')+1)
string1.index('a',string1.index('a',string1.index('a')+1)+1)
```

Out[76]:
```
9
```

In [66]:
```
string1.index('a',string1.index('a')+1)
```

Out[66]:
```
5
```

In [71]:
```
string1.index('a',string1.index('a',string1.index('a',string1.index('a')+1)+
#string1.index('a',string1.index('a',string1.index('a')+1)+1)+1
string1.index('a',string1.index('a',string1.index('a',string1.index('a',str
```

Out[71]: 17

In [78]:
```
# function in function
for i in string1:
    if i=='a':
        index=string1.index(i)
        print(string1.index(i,index+1))
```

```
5
5
5
5
5
```

In [87]:
```
string1='welcome helo hello'
string1.index('l')
string1.index('l',string1.index('l')+1)
string1.index('l',string1.index('l',string1.index('l')+1)+1)
#string1.index('l',string1.index('l',string1.index('l')+1)+1)
```

Out[87]: 15

find

In [ ]:
```
# take one string1
# string1.find()
# apply shift+tab
# read waht it is says
# implement that
```

In [92]:
```
string1='hai hai'
string1.find('z')  # No error
# if substring not found it returns -1

string1.index('z')
# ValueError: substring not found

string1.count('z')  # No error
# returns zero
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
Cell In[92], line 6
      2 string1.find('z')  # No error
      4 # if substring not found it returns -1
----> 6 string1.index('z')
      8 # ValueError: substring not found
     10 string1.count('z')

ValueError: substring not found
```

In [ ]:
```
- capitalize/upper/lower/casefold

- index/find

- count

- replace
```

In [1]:
```
str1='hai how are you , im good'
str1.index('i') # first occurence
```

Out[1]: 2

In [3]:
```
str1.index('i',str1.index('i')+1)
```

Out[3]: 18

strip-lstrip-rstrip

In [4]:
```
str1=' hello how are you '
str2=" hello how are you"
str3="hello how are you "

# I want remove the spaces
# If you want to remove the spaces both side use strip method
# If you want to remove the spaces only left side then use lstrip: left str
# If you want to remove the spaces only right side then use rstrip: Right s
```

In [5]:
```
print(str1.strip())
print(str1.lstrip())
print(str1.rstrip())
```

```
hello how are you
hello how are you
 hello how are you
```

In [6]:
```
print(str1.strip())
print(str2.lstrip())
print(str3.rstrip())
```

```
hello how are you
hello how are you
hello how are you
```

In [8]:
```
print(str3)
```

```
hello how are you
```

In [9]:
```
str3
```

Out[9]:
```
'hello how are you '
```

In [14]:
```
#Wap to extract

str1='python.anaconda@nareshit.com'
str2='omkar.nallagoni@cognizant.com'
str3='virat.kohli@bcci.com'
# Extract first name: python
# Extract second name: anaconda
# Extract company name: nareshit

# Do not count it
# Use methods find/index
```

In [27]:
```
str1[0:6],str1[7:15],str1[16:24]
str1.find('.')
```

Out[27]:
```
6
```

In [28]:
```
str1[0:str1.find('.')]

str1[7:15]
```

Out[28]:
```
'python'
```

In [ ]:
```

```

In [11]:
```
str1 ='omkar.nallagoni@cognizant.com'
print("First Name:",str1[:str1.find('.')])
print("Second Name:",str1[str1.find('.')+1:str1.find('@')])
print("Second Name:",str1[str1.find('@')+1:str1.find('.',str1.find('.')+1)]
```

```
First Name: omkar
Second Name: nallagoni
Second Name: cognizant
```

In [13]:
```
str1='omkar.nallagoni@cognizant.com'
num1=str1.find('.')
f_name=str1[:num1]
print(f_name)
num2=str1.find('@')
s_name=str1[num1+1:num2]
print(s_name)
num3=str1.find('.',str1.find('.')+1)
c_name=str1[num2+1:num3]
print(c_name)
```

```
omkar
nallagoni
cognizant
```

In [36]:
```
str1='a.b@c.com'
first_dot=str1.index('.')

second_dot=str1.index('.',first_dot+1) # after completion of first dot

str1[str1.find('@')+1:second_dot]
```

Out[36]: 'c'

In [45]:
```
str1='3.1489'
# Extract 3
# Extract 1489

str1[str1.index('.')-1]
str1[str1.index('.')+1:]  # str1[<first>:<last>]
```

Out[45]: '1489'

In [44]:
```
str1.index('.')+1
```

Out[44]: 1

startswith- endswith

In [46]:
```
str1='hai how are you'
```

In [53]:
```
str1.startswith('hai how are you')
#str1.startswith('h')
```

Out[53]: True

In [50]:
```
str1.endswith('you')
```

Out[50]: True

In [54]:
```
str1.startswith(str1)

#str1.endswith(str1)
```

Out[54]: True

In [55]:
```
str1.endswith(str1)
```

Out[55]: True

In [56]:
```
dir('')
```

Out[56]:
```
['__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 [ ]:
```
'isalnum',
 'isalpha',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
 'isupper',
```

In [ ]:
```
# take one string1
# string1.isalnum()
# isalpha
# isnumeric
# isupper
# islower
```

In [58]:
```
string1='abc'
string2='123'
string3='abc123'
string4='ABC'
string1.isnumeric()
       .isaplha
        .isalnum
        .isupper
        .islower
```

Out[58]:
```
False
```

In [59]:
```
str1='90hai hello 8 888how are you'
str1.isalnum()
```

Out[59]:
```
False
```

In [ ]:
```
str1='abc.123'
```

split

In [69]:
```
str1='hai howw are you'
str1.split()
# if i not provide anything inside brackets: space
# output is in list format
```

Out[69]:
```
['hai', 'howw', 'are', 'you']
```

In [62]:
```
str1='hai how,are you'
str1.split(',')
```

Out[62]:
```
['hai how', 'are you']
```

In [63]:
```
str1='hai how,are you'
str1.split('a') #h i how, re you
```

Out[63]:
```
['h', 'i how,', 're you']
```

In [66]:
```
str1='hai how,are you'
str1.split() #h i how, re you
```

Out[66]:
```
['hai', 'how,are', 'you']
```

In [64]:
```
str1='%%%hello%%%%'
str1.strip('%')
```

Out[64]:
```
'hello'
```

In [ ]:
```
- capitalize/upper/lower/casefold

- index/find

- count

- replace

- lstrip/rstrip/strip

- startswith/endswith

- isalpha/isnumeric/isalnum

- split
```

In [ ]:
```
# WAP to identify the longest word and shortest word in a given sentence

# WAP to identify the most repated word in a given stentence

# WAP out of 3 strings 'hyd','blr','mumbai' which is greater and which lowe
```

In [ ]:
```
Basic python======= assignment

certication


Writeup on list
```

- how to read the strings
- single quotes/double quotes/triple quotes
- docstring
- type
- max
- min
- len
- addition of two strings(concatenation)
- multiplication
- subtraction
- division
- in opertor (for loop)
- index
- mutable condition
- slice
- stringe methods:
  - capitalize/upper/lower/casefold
  - index/find
  - count
  - replace
  - lstrip/rstrip/strip
  - startswith/endswith
  - isalpha/isnumeric/isalnum
  - split

In [ ]:

In [ ]:

In [ ]: