In [ ]:

```
#============= Python Part-1========================
1) syntax
2) Baic python
3) Conditional : if else
4) Functions
5) Loops: For/while

# strings
# list
# dic
# tuple
# sets
# file handing
```

In [2]:

```
name='python'
type(name)
```

Out[2]:

```
str
```

In [4]:

```
name
```

Out[4]:

```
'python'
```

In [3]:

```
name1="python"
name1
```

Out[3]:

```
'python'
```

In [5]:

```
print(name)
```

```
python
```

In [6]:

```
# doc string: triple quotes
# multilple lines
# in jupter: markdown
# notepad++
# vscode
# pycharm
string1="""hi how are you
   im good
   im learning python"""
```

In [7]:

```
string1
```

Out[7]:

```
'hi how are you\n   im good\n   im learning python'
```

- triple quotes is means doc string
- it is an informatiom about the code

In [10]:

```
"""
this program about infinite while loop
the original password='python'
function:
      arguments: None
      return: None
"""
def pwd():
    password='python'  # random number  (1,10)
    while True:
        user_pwd=input("enter pwd")  # user number= eval(input)
        if password==user_pwd:
            print('you won')
            break
        else:
            print("the password is wrong")
```

Out[10]:

```
10
```

In [12]:

```
print("hello 'python'")
```

```
hello 'python'
```

In [13]:

```
print('hello "python"')
```

```
hello "python"
```

- entire string will be in double quotes, the highlited string in single quotes
- entire string will be in single quotes, the highlited string in double quotes

In [14]:

```
string1='python'
string2=string1 # 'py'
string3='hello' # 'he'
string4=string3 # 'he'
string4=string1 # 'py'
string4=string2 # 'py'
string5=string4 #'py'
string5
```

Out[14]:

```
'python'
```

- type
- len
- max
- min

In [15]:

```
string1='python'
type(string1)
```

Out[15]:

```
str
```

In [16]:

```
len(string1)
# p y t h o n
```

Out[16]:

```
6
```

In [17]:

```
max(string1)
```

Out[17]:

```
'y'
```

In [18]:

```
min(string1)
```

Out[18]:

```
'h'
```

In

In [24]:

```
# iterate a loop on string1
# print each letter
# get the ascii value
'p' in 'python'
'y' in 'python'
't' in 'python'
'h' in 'python'
'o' in 'python'
'n' in 'python'

# i in 'python'
```

Out[24]:

```
True
```

In [26]:

```
for i in 'python':
    print(i,ord(i))
```

```
p 112
y 121
t 116
h 104
o 111
n 110
```

- type
- max
- min
- len
- in

In [28]:

```
string1='hai'
string2='python'
```

In [29]:

```
string1+string2  # string concatination
```

Out[29]:

```
'haipython'
```

```
In [30]: string1-string2
         # i can not do subtraction between two strings
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[30], line 1
----> 1 string1-string2

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

```
In [31]: string1/string2
         # i can not do division between two strings
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[31], line 1
----> 1 string1/string2

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

```
In [34]: string1*string2
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[34], line 1
----> 1 string1*string2

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

```
In [35]: 2*string2
```

```
Out[35]: 'pythonpython'
```

```
In [ ]: string1+string2
        string1-string2
        string1*string2
        string1/string2
        # read the errors
```

In [ ]:

```
# how to read the string
# single /double/triple
# triple: docstring
# type
# len
# max
# min
# + : conacte
# - :
# *
# /
```

In [1]:

```
string1='hello'
string2='python'
```

In [2]:

```
string1+string1
```

Out[2]:

```
'hellohello'
```

In [3]:

```
2*string1
```

Out[3]:

```
'hellohello'
```

indexing

In [4]:

```
name='python'
```

In [5]:

```
# how many letters are there: 6
# python index start with :0

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

In [6]:

```
name[0]
```

Out[6]:

```
'p'
```

In [7]:

```
name[1]
```

Out[7]:

```
'y'
```

In [ ]:

```
name[0]    #  'p'
name[1]    #  'y'
name[2]    #  't'
name[3]    # 'h'
name[4]    #  'o'
name[5]    # 'n'

name[i]
```

In [10]:

```
# how to print any word/sentence by using range method
for i in range(6):
    print(name[i])   # iter1:  i=0    name[0]   p
                     # iter2:  i=1    name[1]   y
```

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

In [12]:

```
name='python'
for i in range(7):
    print(name[i])   #  0   1   2   3  4  5  name[6]
```

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

```
---------------------------------------------------------------------------
-
IndexError                                Traceback (most recent call las
t)
Cell In[12], line 3
      1 name='python'
      2 for i in range(7):
----> 3     print(name[i])

IndexError: string index out of range
```

In [14]:

```
name1='hello how are you'
# spaces also consider one character
print(len(name1))
# i need to provide some number
for i in range(len(name1)):
    print(name1[i],end=' ')
```

```
17
h e l l o   h o w   a r e   y o u
```

In [ ]:

```
# wap ask the user print the letters from a given word
# using for loop
# 'the index of p is:0'
# 'the index of y is:1'

# i want letter as well as attached index number
```

In [17]:

```
name2='python'
for i in range(len(name2)):
    print('the index of {} is:{}'.format(name2[i],i))
```

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

In [18]:

```
# same code using while loop
i=0
name2='python'
while i<len(name2):
    print('the index of {} is:{}'.format(name2[i],i))
    i+=1
```

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

In [ ]:

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

In [21]:

```
name3='python'
name3[-6]
```

Out[21]:

```
'p'
```

In [ ]:

```
# wap ask the user print the letters from a given word
# using for loop
# 'the negative index of p is:-6'
# 'the negative index of y is:-5'

# i want letter as well as attached index number
```

In [32]:

```
name3='python'
for i in range(-len(name3),0):
    print("the negative index of {} is {}".format(name3[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 [34]:
```
name4='python'
for i in range(len(name4)):
    print("the negative index of {} is {}".format(name4[i],i-len(name4)))
```

```
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]:
```
# how to print -6 -5 -4 -3 -2 -1 using while loop
i=0
name4='python'
while i>-len(name4):
    print('the -ve index of {} is : {}'.format(name4[i],-len(name4)-i))
    i-=1
```

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

In [36]:
```
name='python'
i=-len(name)
while i<0:
    print("The negative Index of {} is {}".format(name[i],i))
    i=i+1
```

```
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 [ ]:
```
'the postive index is 0 and negative index is -6 for p'
'the postive index is 1 and negative index is -5 for y'


for  i   in  range():
    0      -6
    1      -5
```

In [37]:
```
name='python'
for i in range(len(name)):
    print('the positivr index is: {} the negetive index is: {} for {}'. for
```

```
the positivr index is: 0 the negetive index is: -6 for p
the positivr index is: 1 the negetive index is: -5 for y
the positivr index is: 2 the negetive index is: -4 for t
the positivr index is: 3 the negetive index is: -3 for h
the positivr index is: 4 the negetive index is: -2 for o
the positivr index is: 5 the negetive index is: -1 for n
```

In [39]:

```
name6='python'
for i in range(len(name6)):
    print("the positive index is {} and negative index is {} for {}".format
```

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

In [40]:

```
name2=input('enter a word')
for i in range(len(name2)):
    print('the possitive index is {} and negative index is {} for {}'.forma
```

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

In [41]:

```
word = "Learn Python"
for i in range(len(word)):
    print("The +ve Index is {} and -ve Index is {} for:{}".format(i,i-len(w
```

```
The +ve Index is 0 and -ve Index is -12 for:L
The +ve Index is 1 and -ve Index is -11 for:e
The +ve Index is 2 and -ve Index is -10 for:a
The +ve Index is 3 and -ve Index is -9 for:r
The +ve Index is 4 and -ve Index is -8 for:n
The +ve Index is 5 and -ve Index is -7 for: 
The +ve Index is 6 and -ve Index is -6 for:P
The +ve Index is 7 and -ve Index is -5 for:y
The +ve Index is 8 and -ve Index is -4 for:t
The +ve Index is 9 and -ve Index is -3 for:h
The +ve Index is 10 and -ve Index is -2 for:o
The +ve Index is 11 and -ve Index is -1 for:n
```

In [49]:

```
for i in range(len(name4)):
    print(i,name4[i],i-len(name4))


#======================= negative index===================
name4='python'
for i in range(len(name4)):
    print("the negative index of {} is {}".format(name4[i],i-len(name4)))

#======================= pos index===================
name4='python'
for i in range(len(name4)):
    print("the positive index of {} is {}".format(name4[i],i))
```

```
0 p -6
1 y -5
2 t -4
3 h -3
4 o -2
5 n -1
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 positive index of p is 0
the positive index of y is 1
the positive index of t is 2
the positive index of h is 3
the positive index of o is 4
the positive index of n is 5
```

In [ ]:

```
sent='hai hai how are you'
#Q1 print how many 'a' are there : 3
# count=0
# iterate the leeters through for loop
#        apply the if condition letter=='a'
#                    count+=1


#Q2 print the index of 'a':

#Q3 print the number of vowels: a  a  i  o  a  e  o  u: 9

# Q4 print the number of unique vowels:  a i   o   e  u:4
```

In [52]:

```
#Q1 print how many 'a' are there : 3
# count=0
# iterate the leeters through for loop
#        apply the if condition letter=='a'
#                    count+=1


sent='hai hai how are you'
# first iteratre letters
count=0
for i in range(len(sent)):
    if sent[i]=='a':
        count=count+1

print("no of repated a are:",count)

# step-1:   i=0    sent[0]='h'    'h'=='a'    F
# step-2:   i=1    sent[1]='a'     'a'=='a'   T ====count=0+1= 1
```

```
h
a
i

h
a
i

h
o
w

a
r
e

y
o
u
no of repated a are: 3
```

In [53]:

```
sent='hai hai how are you'
# first iteratre letters
count=0
for i in range(len(sent)):
    if sent[i]=='a':
        count=count+1
        print(i)

print("no of repated a are:",count)
```

```
1
5
12
no of repated a are: 3
```

In [57]:

```python
########################################## Q1) getting number of 'a' ######
sent='hai hai how are you'
# first iteratre letters
count=0
for i in range(len(sent)):
    if sent[i]=='a':
        count=count+1

print("no of repated a are:",count)


#################################### Q2)getting indexof 'a' ###############
sent='hai hai how are you'
# first iteratre letters
count=0
for i in range(len(sent)):
    if sent[i]=='a':
        count=count+1
        print(i)

print("no of repated a are:",count)


#################################### Q3) No of vowels in the given sentence

sent='hai hai how are you'
# first iteratre letters
count=0
for i in range(len(sent)):
    if sent[i] in 'aeiou':
        count=count+1

print("no of repated vowels are:",count)


# Q4) unique vowels
```

```
no of repated a are: 3
1
5
12
no of repated a are: 3
no of repated vowels are: 9
```

In [55]:

```python
'a' in 'aeiou'
```

Out[55]:

```
True
```

In [ ]:

```python
# take an empty string
# string1=''
# iterate through loop
#  if  'a' in 'aeiou':
#       if 'a' not in string1"
#           string1=string1+'a'    # string1='a'
```

In [ ]:

```
# check two conditions parallal
#  the given letter is in 'aeiou'
#  if that is true check about same letter it is there in empty string
#  if it is not there move to empty string
#  if it is there dont move it
```

- how to read the strings
- single /double
- triple quotes (doc string)
- type
- len
- max
- min
- conactenation (addition of strings)
- index opertaions

```
      - how to iterate the letters by using for loop range function

      -  using while loop

      - post /neg/both combined

      - counting the no of 'a'

      - index of 'a'

      - no of vowels

      - no of uniques
```

In [2]:

```
str1='hello'
str2='python'

# 30
```

In [3]:

```
str1+str2

# 45
```

Out[3]:

```
'hellopython'
```

mutabality-imutability

In [4]:

```
string1='python'
```

```
In [ ]: # I want to replace 'p'  with  'P'
        # based on index operation if you change it or not
        # if you change the value by using index operation: mutable
        # if you could not change the value by using index operation: immutable
```

```
In [6]: string1[0]='P'    #Python
```

```
---------------------------------------------------------------------------
-
TypeError                                 Traceback (most recent call las
t)
Cell In[6], line 1
----> 1 string1[0]='P'

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

Strings are immutable

```
In [7]: list1=[10,20,30]
```

```
In [9]: list1[0]=100
```

```
In [10]: list1
```

```
Out[10]: [100, 20, 30]
```

Slicing

```
In [11]: string1='hello how are you'
```

```
In [ ]: # this concept is same as range() in for loop
```

```
In [12]: string1='hello how are you'
         #string1[start:stop:step]         # range(start,stop,step)
```

```
Out[12]: 'h'
```

```
In [14]: string1[2:10]
         # what is the direction:  postive
         # start=2
         # stop=10-1=9
```

```
Out[14]: 'llo how '
```

```
In [ ]: h  e  l  l  o     h  o  w     a  r  e     y  o  u
        0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16
```

In [13]:
```
len(string1)
```

Out[13]:
```
17
```

In [15]:
```
string1[1:15:2]
# start=1
# stop=15-1=14
# post: step=2
# 1  3   5   7   9   11   13
# e  l       o       r
```

Out[15]:
```
'el o r '
```

In [16]:
```
string1[1:15:-2]
# start=1
# stop=16
# dire=-ve
```

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

In [17]:
```
string1[:]                # sting1[start:stop]
# notging is mentioned means
# postive direction
# start=0
# stop=till last charcter
```

Out[17]:
```
'hello how are you'
```

In [18]:
```
string1[::]  # string1[start:stop:step]
#string[0:lastchar:1]
```

Out[18]:
```
'hello how are you'
```

- nothing mentioned at start postition: simply satrting of letter
- nothing mentioned at stop postion: simply last letter
- nothing mentioned at step size: it is postive direction with step value +1

In [19]:
```
print(string1[0:])
print(string1[:len(string1)])
print(string1[:])
print(string1[::])
```

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

In [ ]:
```

-17  -16 -15 -14 -13 -12 -11  -10 -9 -8 -7  -6  -5  -4   -3  -2  -1
h     e   l   l   o       h    o   w     a   r   e        y   o   u
0     1   2   3   4   5   6    7   8  9  10  11  12  13   14  15  16
```

In [20]:
```
string1[-2:-15]
# start=-2
# direc
# step value : postive
# not possible
```

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

In [21]:
```
string1[-2:-15:-1]
# start=-2   dire=-ve  stop=-15+1=-14   possible
```

Out[21]:
```
'oy era woh ol'
```

In [22]:
```
string1[-15:-2:-1]
```

Out[22]:
```
''
```

In [23]:
```
string1[-15:-2:2]
```

Out[23]:
```
'lohwaey'
```

In [ ]:
```
-17  -16 -15 -14 -13 -12 -11  -10 -9 -8 -7  -6  -5  -4   -3  -2  -1
h     e   l   l   o       h    o   w     a   r   e        y   o   u
0     1   2   3   4   5   6    7   8  9  10  11  12  13   14  15  16
```

In [24]:
```
string1[8:-15:-1]

#start=8  stop =-15+1=-14  it is poss
```

Out[24]:
```
'woh ol'
```

In [26]:
```
string1[8:-10:-1]
```

Out[26]:
```
'w'
```

In [27]:
```
string1[8:-10:-2]
```

Out[27]:
```
'w'
```

In [28]:
```
string1[-15:8:3]
```

Out[28]:
```
'l '
```

In [29]:
```
# If i want reverse the string
string1[::1]

# postive
# start: start
# stop: last
```

Out[29]:
```
'hello how are you'
```

In [30]:
```
string1[::-1]
# in the negative
# start: u
# stop:h
```

Out[30]:
```
'uoy era woh olleh'
```

- how to read the strings
- single /double
- triple quotes (doc string)
- type
- len
- max
- min
- conactenation (addition of strings)
- index opertaions

```
    - how to iterate the letters by using for loop range function

    -  using while loop

    - post /neg/both combined

    - counting the no of 'a'

    - index of 'a'

    - no of vowels

    - no of uniques
```

- slicing

String methods:

In [31]:

```
dir('name')
```

```
Out[31]: ['__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 [32]:

```
name='naresh it'
```

In [33]:

```
help(name.capitalize)
```

```
Help on built-in function capitalize:

capitalize() method of builtins.str instance
    Return a capitalized version of the string.

    More specifically, make the first character have upper case and the re
st lower
    case.
```

In [34]:

```
name.capitalize()
```

Out[34]:

```
'Naresh it'
```

In [35]:

```
name='naresh it'
name.capitalize()
```

Out[35]:

```
'Naresh it'
```

In [ ]:

```
#upper
#lower
```

upper

In [36]:
```
name='naresh it'
help(name.upper)
```

```
Help on built-in function upper:

upper() method of builtins.str instance
    Return a copy of the string converted to uppercase.
```

In [41]:
```
name.upper()
```

Out[41]:
```
'NARESH IT'
```

lower

In [42]:
```
name='Naresh It'
help(name.lower)
```

```
Help on built-in function lower:

lower() method of builtins.str instance
    Return a copy of the string converted to lowercase.
```

In [43]:
```
name.lower()
```

Out[43]:
```
'naresh it'
```

In [44]:
```
name1='naresh IT'
print(name1.capitalize())
print(name1.upper())
print(name1.lower())
```

```
Naresh it
NARESH IT
naresh it
```

Casefold

In [50]:
```
name1='NareSh it'
help(name1.casefold)
print(name1.casefold())
```

```
Help on built-in function casefold:

casefold() method of builtins.str instance
    Return a version of the string suitable for caseless comparisons.

naresh it
```

In [ ]:

```
string1='hello'

#output: string1='Hello'  with out using string method
# strings immutable: index operations
# hint: using slicing and conactenation
```

In [51]:

```
string1='hello'
string2='H'
string3=string1[1:]
string2+string3
```

Out[51]:

```
'Hello'
```

In [55]:

```
string='hello hai how are you'
# how many h : 3
import time
start=time.time()
count=0
for i in string:
    if i=='h':
        count+=1

print(count)
end=time.time()
print(end-start)
```

```
3
0.0
```

In [54]:

```
string.count('h')
```

Out[54]:

```
3
```

In [ ]:

```
# In real time do we use methods?
```

In [3]:

```
name='python'
for i in name:
    print(i)

for i in range(len(name)):
    print(i,name[i])


# if you solve that 4 questions with in operator
# range  in
```

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

In [ ]:

```
Q1)  how many 'a'

Q2)  index of  'a'

Q3)  number of vowels in a given string

Q4) unique vowels
```

In [ ]:

```
# Now we are in the count method
```

In [4]:

```
string1='HAI HAI hai hai'
# How many h are there
# if this method will give 'H' and 'h'
```

In [5]:

```
string1.count('H')
```

Out[5]:

```
2
```

In [6]:

```
string1.count('h')
```

Out[6]:

```
2
```

In [ ]:

```
# WAP convert above string into lower case
# and count number of 'h'

# Do by using string methods: 4          lower and count
# Do by using with out string methods:4  for loop and if condition
```

In [12]:

```
# Method-1:
string1='HAI HAI hai hai'
string2=string1.lower()
string2.count('h')

# M-2:
string1.lower().count('h')

# M-3:
'HAI HAI hai hai'.lower().count('h')

# M-4:
'hai hai hai hai'.count('h')
```

Out[12]:

```
4
```

In [16]:

```
string1='HAI HAI hai hai'
count=0
for i in string1:
    #print(i)           # H or h
    if i=='h' or i=='H': # both conditions we are checking at a time
        count=count+1

print(count)

# or means any condition is true is fine
# and means both the conditions need be True
```

```
2
```

In [ ]:

```
'A'=65+32  ======== 'a'=97
```

In [17]:

```
ord('A')   # 32
```

Out[17]:

```
65
```

In [18]:

```
string1='HAI HAI hai hai'
count=0
for i in string1.lower():
    #print(i)           # H or h
    if i=='h': # both conditions we are checking at a time
        count=count+1

print(count)
```

```
4
```

In [ ]:

```
string1='HAI HAI hai hai'
count=0
for i in string1:
    #print(i)           # H or h
    if i=='h' or i=='H': # both conditions we are checking at a time
        count=count+1

print(count)

#=======================================================
string1='HAI HAI hai hai'
count=0
for i in string1.lower():
    #print(i)           # H or h
    if i=='h': # both conditions we are checking at a time
        count=count+1

print(count)
```

In [19]:

```
string1='HAI HAI hai hai'
count=0
for i in string1:
    #print(i)           # H or h
    if i.lower()=='h': # both conditions we are checking at a time
        count=count+1

print(count)
```

```
4
```

In [ ]:

```
string1='HAI HAI hai hai'
count=0
for i in string1:
    #print(i)           # H or h
    if i=='h' or i=='H': # both conditions we are checking at a time
        count=count+1

print(count)

#======================================================
string1='HAI HAI hai hai'
count=0
for i in string1.lower():
    #print(i)           # H or h
    if i=='h': # both conditions we are checking at a time
        count=count+1

print(count)

#======================================================
string1='HAI HAI hai hai'
count=0
for i in string1:
    #print(i)           # H or h
    if i.lower()=='h': # both conditions we are checking at a time
        count=count+1

print(count)
```

In [20]:

```
string1='HAI HAI hai hai'
count=0
for i in range(len(string1)):
    if string1[i].lower()=='h': # both conditions we are checking at a time
        count=count+1

print(count)
```

```
4
```

In [ ]:

```
# if you use range function on strings
# use index method

# if you use in operator on string : direct
```

In [24]:

```
string2='ola ola ola ola'
print(string2.count('ola')) #4
print(string2.count('o'))   #4
print(string2.count('oa'))  #0
print(string2.count('ola '))# 3
```

```
4
4
0
3
```

In [27]:

```
s_in='MobiN 2 Are'
s_out=''
for i in s_in:
    if i.isupper()==True:
        s_out+=i.lower()
    else:
        s_out+=i.upper()
print(s_out)
```

```
mOBIn 2 aRE
```

In [28]:

```
string2='ola ola ola ola'
len(string2)
# I want to know how many 'a' are there after 4th index
```

Out[28]:

```
15
```

In [62]:

```
string2.count('a',4)
# here 4 means we are counting 'a' from 4th index

# how many 'a' are there between 4  to 8th index
string2.count('a',4,9)

#ola   ola   ola  ola
#012 3 456 7 8910

# string.count(<char>,start_index)
```

Out[62]:

```
1
```

In [31]:

```
string2[4:].count('ola')
# count method is internally doing the slicing also
# not required slicing when you apply count method
```

Out[31]:

```
3
```

In [60]:

```
# wap by using for if
count=0
for i in range(len(string2)-2):  # 0  to 12
    if i>=4:
        if string2[i]+string2[i+1]+string2[i+2]=='ola': # 'o'=='ola'  F
            count=count+1

print(count)

# you need to count the number of 'o' from index 4
# charcters aswellas number
# in/range           range()
```

```
3
```

In [55]:

```
i=4 #==================== True
i=5
i=6
i=7
i=8 #==================== True
i=9
i=10
i=11
i=12 #====================== True
i=13
string2[i]+string2[i+1]+string2[i+2]
string2[13]+string2[14]+string2[15]
```

```
---------------------------------------------------------------------------
-
IndexError                                Traceback (most recent call las
t)
Cell In[55], line 11
      9 i=12 #====================== True
     10 i=13
---> 11 string2[i]+string2[i+1]+string2[i+2]

IndexError: string index out of range
```

In [57]:

```
len(string2)-2
```

Out[57]:

```
13
```

In [ ]: