STRINGS

How to read the Strings

In [ ]:
```
There are three types of strings
1.single quotes
2.double quotes
3.triple quotes
```

In [5]:
```
string1='python'   # Single quote
string1
```
Out[5]:
```
'python'
```

In [6]:
```
string2="python"    # Double quotes
string2
```
Out[6]:
```
'python'
```

In [ ]:
```
## Triple quotes is to create multi-line strings and docstrings.We can create triple quotes using double (""" """) as well as
## single (''' ''') quotes.
```

In [ ]:
```
* Doc string is used to say same information about your python code
"""
im creating a while condition
arguements: None
return:    None
"""
def hello():
    print("good morning")
```

In [10]:
```
string3='hello "python"'
print(string3)
```
```
hello "python"
```

type

type() is a built-in function that is used to return the type of data stored in the objects or variables in the program.

In [11]:
```
string1
```
Out[11]:
```
'python'
```

In [12]:
```
type(string1)
```
Out[12]:
```
str
```

len

- len() returns the number of items in an object.

In [13]:
```
len(string1)          # python #
```
Out[13]:
```
6
```

max

Max() is built-in tool that returns the largest item in an iterable or the largest of two or more arguements.

In [14]:
```
string1='pP'
max(string1)     # python

# ASCII
#'A':
```
Out[14]:
```
'p'
```

min

Min() returns the item with the lowest value,or the item with the lowest value in an iterable.

In [15]:
```
string1='pP'
min(string1)
```
Out[15]:
```
'P'
```

Ord – Chr

The chr() function takes an integer and converts it to a character, so it returns a character string
The ord() function takes a single Unicode character and returns an integer value.

In [1]:
```
ord('p')  # it will provide ascii value of charord
```
Out[1]:
```
112
```

In [2]:
```
ord('p'),ord('y'),ord('t'),ord('h'),ord('o'),ord('n')
```
Out[2]:
```
(112, 121, 116, 104, 111, 110)
```

In [3]:
```
chr(112),chr(121),chr(116),chr(104),chr(111),chr(110)
```
Out[3]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

Addition of Two strings (concatenation)

You can concatenate two different strings together and also the same string to itself multiple times using + operator respectively.

In [16]:
```
str1='hai'
str2='how'
str1+str2
```
Out[16]:
```
'haihow'
```

In [10]:
```
string1+string1+string1
```
Out[10]:
```
'haihaihai'
```

Multiplication

You can use the * operator to multiply not only numbers but also lists and strings.

In [17]:
```
3*str2
```
Out[17]:
```
'howhowhow'
```

Subtraction

You can use the operator for subtraction is"-" .It subtracts the second value from the first one.

In [7]:
```
string1-string2
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[7], line 1
----> 1 string1-string2

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

Division

Divides the number on its left by the number on its right,rounds down the answer,and returns a whole number.

In [8]:
```
string1/string2
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[8], line 1
----> 1 string1/string2

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

In operator (for loop)

In Operator determines whether a given value is a constituent element of a sequence such as a string,array,list,or tuple.

In [11]:
```
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 [12]:
```
string1='python'

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

# i in string1
```
Out[12]:
```
True
```

In [13]:
```
for i in string1:
    print(i)

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

In [ ]:
```
- 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
```

Index

Index refers to the process of accessing a specific element in a sequence,such as a string or list,using its position or index number

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

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

In [14]:
```
string1[0],string1[1],string1[2],string1[3],string1[4],string1[5]
```
Out[14]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

In [15]:
```
name='python class'

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

   # i='p' name[' p ']='p'
# if you want to print any number
```
```
the index no of 'p' is 0
the index no of 'y' is 1
the index no of 't' is 2
the index no of 'h' is 3
the index no of 'o' is 4
the index no of 'n' is 5
the index no of ' ' is 6
the index no of 'c' is 7
the index no of 'l' is 8
the index no of 'a' is 9
the index no of 's' is 10
the index no of 's' is 11
```

In [16]:
```
'the index of p is 0'
'the index of y is 1'
```
Out[16]:
```
'the index of y is 1'
```

In [17]:
```
for i in range(len(string1)):   # i means numbers 0
    print('the index of {} is {}'.format (string1[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 [ ]:
```
## Positive Index
## Negative Index
```

In [18]:
```
string1='python'
for i in range(-6,0):
    print("the positive index of {} is {}".format(string1[i],i))
```
```
the positive index of p is -6
the positive index of y is -5
the positive index of t is -4
the positive index of h is -3
the positive index of o is -2
the positive index of n is -1
```

In [19]:
```
string1='python'
for i in range(-6,0):
    print("the negative index of {} is {}".format(string1[i],i-6))
```
```
the negative index of p is -12
the negative index of y is -11
the negative index of t is -10
the negative index of h is -9
the negative index of o is -8
the negative index of n is -7
```

MUTABLE CONDITION

- Mutable and immutable concept
- Mutable ==== We can change
- Immutable==== we can not change
- Strings are immutable

In [20]:
```
string1='python'
# i want change 'p' ====== 'p'
# o/p:     'python'
string1[0]='p'
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[20], 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 [21]:
```
list1=[100,200,300]  # 100 ======1000
list1[0]=1000
list1
```
Out[21]:
```
[1000, 200, 300]
```

Slice

The format for list slicing is [start:stop:step].

start is the index of the list where slicing starts.
stop is the index of the list where slicing ends.
step allows you to select nth item within the range start to stop.

In [ ]:
```
String Methods
- capitalize/upper/lower/casefold
- index/find
- count
- replace
- lstrip/rstrip/strip
- startswith/endswith
- isalpha/isnumeric/isalnum
- split
```

Capitalize

Capitalize() method is a bulit-in string function in python that is used to modify the case of the characters in a string.

In [39]:
```
string1='welcome'
help(string1.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 rest lower
    case.

```

In [40]:
```
string1='welcome'
string1.capitalize()
```
Out[40]:
```
'Welcome'
```

Upper

upper() method returns uppercase string from the given string.It converts all lowercase characters to uppercase characters.

In [41]:
```
string1.upper()
```
Out[41]:
```
'WELCOME'
```

Lower

Lower() method returns string class method that converts all the uppercase characters in the string into lowercase characters and returns a new string.

In [42]:
```
string1.lower()
```
Out[42]:
```
'welcome'
```

In [43]:
```
string1='welcome'
print(string1.capitalize())
print(string1.upper())
print(string1.lower())
```
```
Welcome
WELCOME
welcome
```

Index

The process of accessing a specific element in a sequence, such as a string or list, using its position or index number.

In [44]:
```
string1='welcome python'
string1.index('c')

# index of 'c'
```
Out[44]:
```
3
```

In [45]:
```
sring1='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')
```
```
0
```
Out[45]:
```
0
```

In [46]:
```
for i in range(len(string1)):
    if string[i]=='a':
        print(i)
```
```
1
8
```

In [47]:
```
string1='hai hai hai hai'
i1=(string1.index('a'))
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)
```
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[47], line 6
      4 i3=(string1.index('a',i2+1))
      5 i4=(string1.index('a',i3+1))
----> 6 i5=(string1.index('a',i4+1))
      7 print(i1,i2,i3,i4,i5)

ValueError: substring not found
```

In [48]:
```
string1='hai hai hai hai hai'
# i want to first occurance of 'a'
string1.index('a')
# second occurance of 'a'
string1.index('a',i1+1)
string1.index('a',i2+1)
```
Out[48]:
```
9
```

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

Find

Use to find the index of the first occurrence of a substring from the given string.

In [50]:
```
# take one string1
# string1.find()
# apply shift+tab
# road what is is says
# implement that

string1='hai hai'
string1.find('z')      # no error

# if substring not found it returns -1

string1.index('z')
# value error: sustring not found

string1.count('z')    # no error
# returns
```
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[50], line 12
      8 string1.find('z')      # no error
     10 # if substring not found it returns -1
---> 12 string1.index('z')
     13 # value error: sustring not found
     15 string1.count('z')

ValueError: substring not found
```

Count

The count() method returns the number of times a specified value appears in the string.

In [51]:
```
string1='hai how are you'
# how many 'a's are there
```

In [52]:
```
string1.count('a')
```
Out[52]:
```
2
```

In [53]:
```
count=0
for i in string1:
    if i=='a':
        count+=1
```

In [ ]:
```
string1='ola ola ola'
```

In [54]:
```
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 [55]:
```
'ola ola ola'.count('a')
```
Out[55]:
```
3
```

In [56]:
```
string1='ola ola ola'
count=0
for i in string1:
    if i=='a':
        count+=1
print(count)
```
```
3
```

Replace

The replace() method replaces a specified phrase with another specified phrase.

In [57]:
```
string1='welcome'
# replace 'l' with 'L'
string1.replace('1','L')
```
Out[57]:
```
'welcome'
```

In [58]:
```
string1='welcome'
# replace 'l' with 'L'
string1.replace('1','@')
```
Out[58]:
```
'welcome'
```

In [59]:
```
string1='restart rrr'
# replace 'r' with '$'
string1.replace('r','$',1)
```
Out[59]:
```
'$estart rrr'
```

Lstrip/Rstrip/Strip

The lstrip(s) (left strip) function removes leading whitespace (on the left) in the string.
The rstrip(s) (right strip) function removes the trailing whitespace (on the right).

In [5]:
```
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 strip
# if you want to remove the spaces only right side then use rstrip:right strip
```

In [6]:
```
print(str1.strip())
print(str1.lstrip())
print(str1.rstrip())
```
```
hello how are you
hello how are you
 hello how are you
```

In [7]:
```
print(str1.strip())
print(str2.lstrip())
print(str3.rstrip())
```
```
hello how are you
hello how are you
hello how are you
```

In [8]:
```
str3
```
Out[8]:
```
'hello how are you '
```

In [25]:
```
str1='%%%%hello%%%%'
str1.strip('%')
```
Out[25]:
```
'hello'
```

Startswith – Endswith

The Python startswith() function checks if a string starts with a specified substring.
The Python endswith() checks if a string ends with a substring.

In [9]:
```
str1='hai how are you'
```

In [10]:
```
str1.startswith('hai')
str1.startswith('h')
```
Out[10]:
```
True
```

In [11]:
```
str1.startswith('hai')
```
Out[11]:
```
True
```

In [12]:
```
str1.endswith('hai')
```
Out[12]:
```
False
```

isalpha/isnumeric/isalnum

The isalpha() method returns True if all the characters are alphabet letters (a-z)
The isnumeric() method returns True if all the characters are numeric (0-9), otherwise False
The isalnum() method returns True if all the characters are alphanumeric, meaning alphabet letter(a-z)and numbers(0-9)

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

In [21]:
```
string1='abc'
string2='123'
string3='abc123'
string4='ABC'
string1.isnumeric()
       .isalpha
       .isnumeric
       .isupper
       .islower
```
```
  Cell In[21], line 6
    .isalpha
    ^
IndentationError: unexpected indent
```

In [15]:
```
str1='90hai hello 8 888how are you'
str1.isalnum()
```
Out[15]:
```
False
```

In [20]:
```
str1='90hai hello 8 888how are you'
str1.isnumeric()
```
Out[20]:
```
False
```

Split

The split() method splits a string into a list. You can specify the separator, default separator is any whitespace

In [22]:
```
str1='hai how are you'
str1.split()

# if i not provide anything inside brackets space
```
Out[22]:
```
['hai', 'how', 'are', 'you']
```

In [23]:
```
str1='hai how,are you'
str1.split(',')
```
Out[23]:
```
['hai how', 'are you']
```

In [24]:
```
str1='hai how,are you'
str1.split('a')  #h i how, re you
```
Out[24]:
```
['h', 'i how,', 're you']
```