STRINGS

- HOW TO READ THE STRINGS

- There are three types of strings
  - single quote
  - double quote
  - triple quote

In [2]:
```
# single quote example:

string1='python'          # single quote
string1
```
Out[2]:
```
'python'
```

In [5]:
```
# double quote example:

string2="python"
string2
```
Out[5]:
```
'python'
```

In [9]:
```
# triple quotes example:
"""
iam geetha reddy
from: Kadapa
"""
```
Out[9]:
```
'\niam geetha reddy \nfrom: Kadapa\n'
```

In [13]:
```
# i want to highlite the python

string1='hello "python"'
string1
```
Out[13]:
```
'hello "python"'
```

TYPE

- The Type of data strored in the Object or variable in the program.
- Type is a built-in function that is used to Return the Type

In [14]:
```
string1
```
Out[14]:
```
'hello "python"'
```

In [15]:
```
type(string1)
```
Out[15]:
```
str
```

LEN

- The len() returns the number of items in an object.
- The len() return the number of characters in the string.

In [2]:
```
string='python'
```

In [3]:
```
len(string)
```
Out[3]:
```
6
```

MAX – MIN

- The Max() returns the item with the Highest value
- The Min() returns the item with the Lowset value
- These functions are used to compute the Max and Min of values as passed in its arguments.

In [5]:
```
string='Naresh It'
max(string)
```
Out[5]:
```
't'
```

In [8]:
```
string="Naresh"
min(string)
```
Out[8]:
```
'N'
```

ORD – CHAR

- Ord() is used to convert a single unicode character into its integer representation.
- Char() is uesd to get a string representing of a character which points to a unicode code interger.

In [12]:
```
ord('P')
```
Out[12]:
```
80
```

In [22]:
```
ord('P'),ord('G'),ord('R')
```
Out[22]:
```
(80, 71, 82)
```

In [27]:
```
chr(80),chr(71),chr(82),
```
Out[27]:
```
('P', 'G', 'R')
```

concatenation

- Obtaining a new string that contains both of the original string.
- It involves joining two or more strings to create a single new string.

ADDITION OF TWO STRINGS

    - The merging of both the strings together

In [30]:
```
str1="Geetha "
str2="Reddy"
str1+str2
```
Out[30]:
```
'Geetha Reddy'
```

SUBTRACTION

In [31]:
```
str1="Geetha "
str2="Reddy"
str1-str2
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[31], line 3
      1 str1="Geetha "
      2 str2="Reddy"
----> 3 str1-str2

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

MULTIPLICATION

    - Use the * operator to multiply not only numbers but also string and list.

In [34]:
```
3*str1,3*str2
```
Out[34]:
```
('Geetha Geetha Geetha ', 'ReddyReddyReddy')
```

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

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

IN OPERATOR (FORLOOP)

    - In operator determines whether a given valuse is a constituent element of a sequence such as a string.
    - The statement return a boolen results of True or False.

INDEX

    - The process of accessing a specific element in a sequence, suac as a string,using its position or index number.
    - Indexing in pyhton starts at 0.
    - Which means that the first element in a sequence has an index of 0.
    - The second element has an index of 1, and so on.

In [38]:
```
string='python'
```

In [39]:
```
string[0],string[1],string[2],string[3],string[4],string[5]
```
Out[39]:
```
('p', 'y', 't', 'h', 'o', 'n')
```

In [40]:
```
 # i want print letters using in operator

for i in string1:
    print(i)
```
```
p
y
t
h
o
n
```

In [41]:
```
# i want print letters using in  range operator


for i in range(len(string1)):
    print(i)
```
```
0
1
2
3
4
5
```

In [42]:
```
# i want print letters like the index of p is 0'


for i in range(len(string1)):
    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
```

MUTABLE AND IMMUTABLE CONCEPT

    - MUTABLE====WE CAN CHANGE

    - IMMUTABLE === WE CANNOT CHANGE

    - STRINGS ARE IMMUTABLE

In [46]:
```
list1=[86,8,90]
list1[1]=88
list1
```
Out[46]:
```
[86, 88, 90]
```

In [45]:
```
list1=[100,200,300]   # 100=====1000
list1[0]=1000
list1
```
Out[45]:
```
[1000, 200, 300]
```

SLICE

    - The slice(5) represents a slice that start from index 0 and ends at index 5.
    - And selects every third element from the sequence.

In [54]:
```
string1="SIMPLELEARN"
print(string1[1:7:2])
```
```
IPE
```

STRINGS METHOD

    - It is a fundamental data type used to represent and manipulate textual data.
    - Strings in python are immutable.

In [57]:
```
dir('')
```
Out[57]:
```
['__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

    - Capitalize is used to convert the first char of a string to Uppercase and the remaining chat to Lowercase.

In [59]:
```
string1='india'
string1.capitalize()
```
Out[59]:
```
'India'
```

UPPER CASE

    - It converts all the Lowercase char in the string into Uppercase char and return a new string.

In [61]:
```
string1.upper()
```
Out[61]:
```
'INDIA'
```

LOWER CASE

    - It converts all the Uppercase char in th string into Lowercase char and retuen a new string.

In [62]:
```
string1.lower()
```
Out[62]:
```
'india'
```

In [64]:
```
string1='india'
print(string1.capitalize())
print(string1.upper())
print(string1.lower())
```
```
India
INDIA
india
```

COUNT

    - This method return the number of times a specified value appears in the string.

In [6]:
```
string="I LOVE MY MOM"
string.count('M')
```
Out[6]:
```
3
```

In [7]:
```
string.count('O')
```
Out[7]:
```
2
```

In [11]:
```
string1='ola ola ola'
print(string1.count('a',4))
```
```
2
```

REPLACE

    -This method replace a specified phrase with another specified phrase.

In [12]:
```
string1='Geetha'
# replace 'r' with '$'
string1.replace('a','@')
```
Out[12]:
```
'Geeth@'
```

FIND

    - This method finds the first occurrence of the specified value.

In [13]:
```
string1='hai and hai'
string1.find('i')
```
Out[13]:
```
2
```

STRIP RSTRIP LSTRIP

STRIP

    - The strip() method removes any leading, and trailing whitespaces.
    - Leading means at the beginning of the string
    - Trailing means at the end
    - Any whitespaces will be removed.

RSTRIP

    - Python rstrip() method removes all the trailing characters from the string.
    - It means it removes all the specified characters from right side of the string.
    - This method returns a string value.

LSTRIP

    - Python String lstrip() method returns a copy of the string with leading characters removed
    - (based on the string argument passed).
    - If no argument is passed, it removes leading spaces.

In [14]:
```
str1='hello how are you'
str2="hello how are you"
str3="hello how are you"
```

In [15]:
```
print(str1.strip())
print(str2.lstrip())
print(str3.rstrip())
```
```
hello how are you
hello how are you
hello how are you
```

STARTSWITH ENDSWITH

STARTWITH

    - This method is a string method that returns True
    - If the input string starts with the specified prefix(string)
    - else it returns False.

ENDSWITH

    - This method is a string method that returns True
    - If the input string ends with the specified suffix(string);
    - else it returns False.

In [18]:
```
string="I LOVE MY COUNTRY"
```

In [21]:
```
string.startswith("I")
string.startswith("I")
```
Out[21]:
```
True
```

In [22]:
```
string.startswith("I")
```
Out[22]:
```
True
```

In [23]:
```
string.startswith("LOVE")
```
Out[23]:
```
False
```

ISALPHA

    - A string for alphabetical characters and returns True only
    - if the string contains all alphabetical characters.

In [24]:
```
str1="PGR"
str1.isalpha()
```
Out[24]:
```
True
```

ISALNUM

    - Returns Trueif all the characters are alphanumeric
    - Meaning alphabet letter (a-z) and numbers (0-9).

In [29]:
```
str1='868890'
str1.isalnum()
```
Out[29]:
```
True
```

ISALNUMERIC

    - Returns True if all the characters are numeric (0-9), otherwise False.

In [32]:
```
str1="123456789"
str1.isnumeric()
```
Out[32]:
```
True
```

In [33]:
```
str1="123456789PGR"
str1.isnumeric()
```
Out[33]:
```
False
```

SPLIT

    - The string manipulation function in Python used to break down
    - A bigger string into several smaller strings is called the split() function in Python.
    - The split() function returns the strings as a list.

In [34]:
```
str1='hai how are you'
str1.split()
```
Out[34]:
```
['hai', 'how', 'are', 'you']
```

In [36]:
```
str1='hai how are you'
str1.split('a')
```
Out[36]:
```
['h', 'i how ', 're you']
```