Strings

- Collection of characters or Sequence of characters or Set of characters.
- we can define a string by using single , double or triple quotes.
  - single or double --> single line strings
  - triple --> multiline strings.

In [5]:
```
s1 = 'python workshop'
s2 = "this is a single line string"
```

In [6]:
```
print(s1)
```

```
python workshop
```

In [7]:
```
print(s2)
```

```
this is a single line string
```

In [10]:
```
s3 = '''This is
the last session
of this workshop.'''
print(s3)
```

```
This is
the last session
of this workshop.
```

In [11]:
```
s4 = """this is
also a multi
line string"""
print(s4)
```

```
this is
also a multi
line string
```

In [12]:
```
for i in s4:
    print(i)
```

```
t
h
i
s

i
s


a
l
s
o

a

m
u
l
t
i


l
i
n
e

s
t
r
i
n
g
```

In [14]:
```
first_name = input("Enter ur firstname")
last_name = input("Enter ur lastname")
# string concatenation --> adding two strings
print(first_name+' '+last_name)
```

```
Enter ur firstnamejohn
Enter ur lastnamesmith
john smith
```

In [16]:
```
# 'vits' i want to print this string 6 times
'vits '*6
```

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

String indexing and slicing

- string indexing starts from 0
- types of indexing:
    1. positive or forward indexing
        - left to right
        - index starts from 0 to end of the string
    2. negative or backward indexing
        - from right ot left
        - index starts from -1
- str_name[index_value]
- Immutable object.

Immutable vs Mutable

- Immutable : We can't change or modify anything after it's declaration.
    - integers
    - float values
    - strings
    - tuple
- Mutable : We can update or add or delete anything from the given object at any time.
    - lists
    - sets
    - dictionaries

In [18]:
```
s = 'visvodaya institute'
s[4] # positive
s[-15] # _ve
```

Out[18]: 'o'

In [20]:
```
s[-4]
s[15]
```

Out[20]: 't'

Slicing

- slicing means dividing the main string into sub-strings.
- str_name[index] --> it will return char at index
- str_name[start:end] --> it will return sequence of characters from start index to (end-1) index.
- str_name[start:end:incre/decre]

In [21]:
```
print(s)
```

```
visvodaya institute
```

In [22]:
```
1 s[1:3]
```

Out[22]: 'is'

In [23]:
```
1 s[5:9]
```

Out[23]: 'daya'

In [24]:
```
1 # a i
2 s[8:11]
```

Out[24]: 'a i'

In [27]:
```
1 print(s)
```

```
visvodaya institute
```

In [29]:
```
1 # a i using negative indexing
2 s[-11:-8]
```

Out[29]: 'a i'

In [30]:
```
1 s[-8:-11]
```

Out[30]: ''

In [31]:
```
1 s[:6] # by default starts from 0
```

Out[31]: 'visvod'

In [32]:
```
1 s[8:] # default end value is len(s)-1
```

Out[32]: 'a institute'

In [33]:
```
1 # len() --> it will return the total number of
2 # characters present in the given string.
3 len('hello world!')
```

Out[33]: 12

In [34]:
```
1 max(s) # return the char with highest ascii value
```

Out[34]: 'y'

In [35]:
```
1 min(s)
```

Out[35]: ' '

In [36]:
```
ord(' ')
```

Out[36]: 32

In [37]:
```
l = 'andhra pradesh'
l[::]
```

Out[37]: 'andhra pradesh'

In [38]:
```
l[:]
```

Out[38]: 'andhra pradesh'

In [42]:
```
l[0::]
```

Out[42]: 'andhra pradesh'

In [40]:
```
l[:15]
```

Out[40]: 'andhra pradesh'

In [44]:
```
l
```

Out[44]: 'andhra pradesh'

In [45]:
```
l[::2]
```

Out[45]: 'adr rds'

In [46]:
```
l[1::2]
```

Out[46]: 'nhapaeh'

In [49]:
```
# heapahn
l[::-2]
```

Out[49]: 'heapahn'

In [55]:
```
l
```

Out[55]: 'andhra pradesh'

In [56]:
```
l[-14:-8:3] #ah
```

Out[56]: 'ah'

In [58]:
```
l[-2:-10:-4] #
```

Out[58]:
```
'sr'
```

In [59]:
```
l[::-1]
```

Out[59]:
```
'hsedarp arhdna'
```

In [ ]:
```
pop
lol
wow
mom
dad
tenet
level
eye
malayalam
racecar
```

In [62]:
```
s = input("Enter any string: ")
if s==s[::-1]:
    print("Palindrome")
else:
    print("Not a palindrome")
```

```
Enter any string: levels
Not a palindrome
```

String methods

- dir(str)
- str_name.method_name(parameters)

In [64]:
```
print(dir(str))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewa
rgs__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__l
e__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce
__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__siz
eof__', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'cou
nt', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'inde
x', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'i
slower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join',
'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rind
ex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startsw
ith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
```

In [65]:
```
a = 'elEctroNICs EnGIneErInG'
len(a)
```

Out[65]: 23

In [66]:
```
a.index('o')
```

Out[66]: 6

In [71]:
```
a.index('E')
```

Out[71]: 2

In [72]:
```
a.index('q') # left to right
```

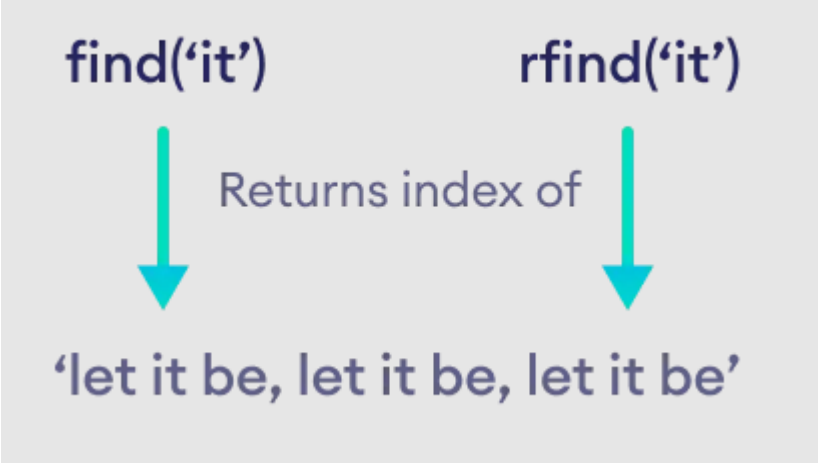
...

In [73]:
```
a.rindex('E') # right to left
```

Out[73]: 18

In [76]:
```
a.index('ro')
```

Out[76]: 5



In [79]:
```
a
```

Out[79]: 'elEctroNICs EnGIneErInG'

In [80]:
```
a.rfind('G')
```

Out[80]: 22

In [81]:
```
1 a.find('G')
```

Out[81]: 14

In [84]:
```
1 a.find('Cs')
```

Out[84]: 9

In [85]:
```
1 a.rfind('Cs')
```

Out[85]: 9

In [86]:
```
1 # count() --> it will return the number of
2 # occurances of particular character.
3 a.count('E')
```

Out[86]: 3

In [87]:
```
1 a.count('r')
```

Out[87]: 2

In [88]:
```
1 a.count('M')
```

Out[88]: 0

In [89]:
```
1 # capitalize()
2 a.capitalize()
```

Out[89]: 'Electronics engineering'

In [90]:
```
1 a.upper()
```

Out[90]: 'ELECTRONICS ENGINEERING'

In [91]:
```
1 a.lower()
```

Out[91]: 'electronics engineering'

In [92]:
```
1 a.swapcase()
```

Out[92]: 'ELeCTROnicS eNgiNEeRiNg'

In [96]:
```
1 s = "elec12tRonIcs aNd cO*mmUni45cation enGin16eEriNg"
2 s.title()
```

Out[96]: 'Elec12Tronics And Co*Mmuni45Cation Engin16Eering'

In [95]:
```
1 s.split()
```

Out[95]: ['elec12tRonIcs', 'aNd', 'cOmmUni45cation', 'enGin16eEriNg']

In [97]:
```
1 a
```

Out[97]: 'elEctroNICs EnGIneErInG'

In [98]:
```
1 a.split()
```

Out[98]: ['elEctroNICs', 'EnGIneErInG']

In [106]:
```
1 a.split('n')
```

Out[106]: ['elEctroNICs E', 'GI', 'eErI', 'G']

In [108]:
```
1 a.split('E',6)
```

...

In [103]:
```
1 a.partition('n')
```

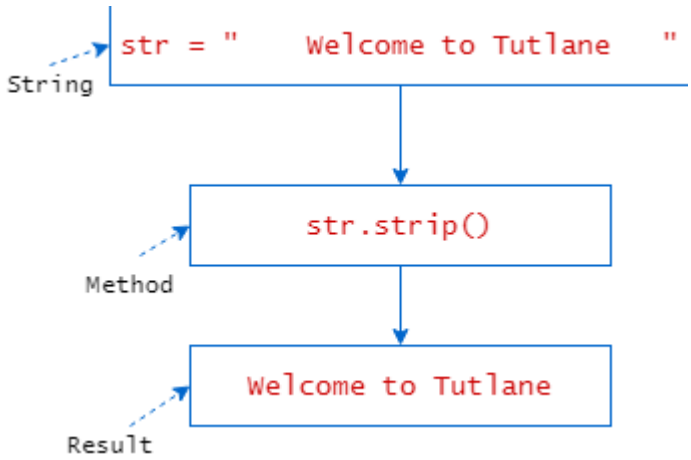
Out[103]: ('elEctroNICs E', 'n', 'GIneErInG')

In [104]:
```
1 a.rpartition('n')
```

Out[104]: ('elEctroNICs EnGIneErI', 'n', 'G')

In [ ]:
```
1
```

In [ ]:
```
1
```



In [109]:
```
k = '   vits college   '
k.strip()
```

Out[109]: 'vits college'

In [110]:
```
k.lstrip()
```

Out[110]: 'vits college   '

In [111]:
```
k.rstrip()
```

Out[111]: '   vits college'

In [114]:
```
j = '@@ @@@hello vits@@ @'
j.strip('@')
```

Out[114]: ' @@@hello vits@@ '

In [113]:
```
print(j)
```

...

In [119]:
```
v = 'nellore'
'11'.join(v)
```

Out[119]: 'n11e11l11l11o11r11e'

In [120]:
```
k
```

Out[120]: '   vits college   '

In [124]:
```
k.replace('e','E')
```

Out[124]: '   vits collEgE   '

In [125]:
```
# center,zfill,ljust,rjust
c = 'visvodaya'
c.zfill(20)
```

Out[125]: '00000000000visvodaya'

In [127]:
```
c.center(20,'@')
```

Out[127]: '@@@@@visvodaya@@@@@'

In [130]:
```
c.center(21,'%')
```

Out[130]: '%%%%%visvodaya%%%%%'

In [135]:
```
1 c.center(12,'@')
```

Out[135]: '@visvodaya@@'

In [136]:
```
1 c.ljust(25)
```

Out[136]: 'visvodaya   '

In [137]:
```
1 c.rjust(23)
```

Out[137]: '   visvodaya'

In [138]:
```
1 c.rjust(24,'O')
```

Out[138]: 'OOOOOOOOOOOOOOOvisvodaya'

In [139]:
```
1 c.ljust(21,'O')
```

Out[139]: 'visvodaya000000000000'

In [140]:
```
1 l
```

Out[140]: 'andhra pradesh'

In [141]:
```
1 l.islower()
```

Out[141]: True

In [142]:
```
1 l.isupper()
```

Out[142]: False

In [143]:
```
1 'hEloo'.isupper()
```

Out[143]: False

In [144]:
```
1 ' '.isspace()
```

Out[144]: True

In [147]:
```
1 'Python3.10'.isalpha()
```

Out[147]: False

In [148]:
```
1 'Python310'.isalnum()
```

Out[148]: True

In [149]:
```
1 'Python3.10'.isalnum()
```

Out[149]: False

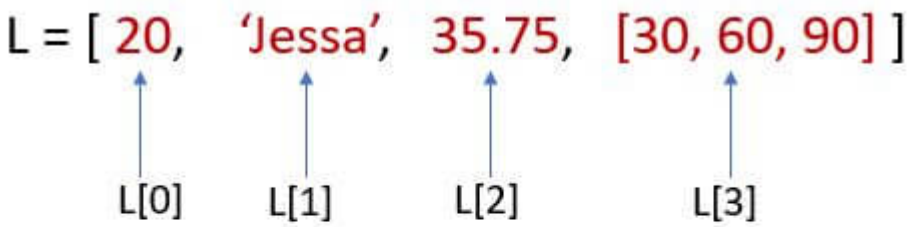
In [ ]:
```
1
```

Name	Purpose
`len(s)`	Calculate the length of the string `s`
`+`	Add two strings together
`*`	Repeat a string
`s.find(x)`	Find the first position of `x` in the string `s`
`s.count(x)`	Count the number of times `x` is in the string `s`
`s.upper()` `s.lower()`	Return a new string that is all uppercase or lowercase
`s.replace(x, y)`	Return a new string that has replaced the substring `x` with the new substring `y`
`s.strip()`	Return a new string with whitespace stripped from the ends
`s.format()`	Format a string's contents

Data Structures in python

- List
- Tuple
- Set
- Dictionaries

Lists



- ✓ **Ordered**: Maintain the order of the data insertion.
- ✓ **Changeable**: List is mutable and we can modify items.
- ✓ **Heterogeneous**: List can contain data of different types
- ✓ **Contains duplicate**: Allows duplicates data

In [154]:
```
l = [16,42,93,44,5.44,16,44,'mercy','avin','green','avin']
print(l)
```

```
[16, 42, 93, 44, 5.44, 16, 44, 'mercy', 'avin', 'green', 'avin']
```

In [153]:
```
l[0]
```

Out[153]:
```
16
```

In [155]:
```
l[4:8]
```

Out[155]:
```
[5.44, 16, 44, 'mercy']
```

In [156]:
```
type(l)
```

Out[156]:
```
list
```

In [157]:
```
l[::-2]
```

Out[157]:
```
['avin', 'avin', 44, 5.44, 93, 16]
```

In [158]:
```
dir(list)
```

. . .

In [159]:
```
l.append(100)
```

In [164]:
```
print(l)
```

```
[16, 42, 93, 44, 5.44, 16, 44, 'mercy', 'avin', 'green', 'avin', 100, 'pink',
'white', ['pink', 'white']]
```

In [161]:
```
1 l.extend(['pink','white'])
```

In [163]:
```
1 l.append(['pink','white'])
```

In [165]:
```
1 l[-1]
```

Out[165]:
```
['pink', 'white']
```

In [166]:
```
1 l.insert(10,1000)
```

In [167]:
```
1 print(l)
```

...

In [168]:
```
1 l[3] = 'vits'
2 print(l)
```

...

In [182]:
```
1 print(l)
```

```
[42, 93, 'vits', 5.44, 16, 44, 'mercy', 'green', 1000, 'avin', 100]
```

In [176]:
```
1 l.pop()
```

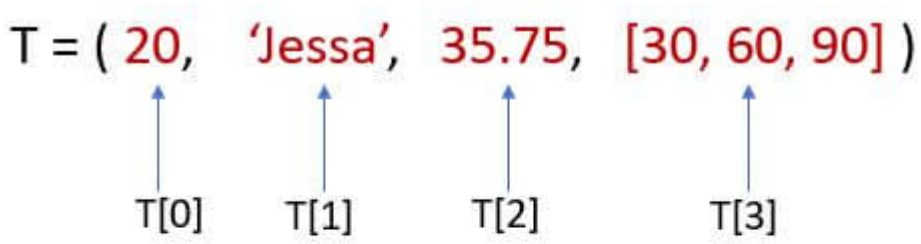
Out[176]:
```
'pink'
```

In [180]:
```
1 l.remove('16')
```

...

In [181]:
```
1 l.remove(16)
```

Tuple



- ✓ **Ordered**: Maintain the order of the data insertion.
- ✓ **Unchangeable**: Tuples are immutable and we can't modify items.
- ✓ **Heterogeneous**: Tuples can contains data of types
- ✓ **Contains duplicate**: Allows duplicates data

In [183]:
```
t= (1,2,3,4,5)
print(t)
```

```
(1, 2, 3, 4, 5)
```

In [184]:
```
dir(tuple)
```

...

In [185]:
```
t.index(4)
```

Out[185]: 3

In [186]:
```
names = ('nandini','lakshman','seshu','yogi','mrudula','poojitha')
print(names)
```

```
('nandini', 'lakshman', 'seshu', 'yogi', 'mrudula', 'poojitha')
```

In [187]:
```
names.index('yogi')
```

Out[187]: 3

In [188]:
```
names[1:4]
```

Out[188]: ('lakshman', 'seshu', 'yogi')

In [189]:
```
names[::-3]
```

Out[189]: ('poojitha', 'seshu')

In [190]:
```
names *3
```

...

In [191]:
```
1 t+names
```

...

In [194]:
```
1 for i in names:
2     print(i,end=" ")
```

```
nandini lakshman seshu yogi mrudula poojitha
```

In [196]:
```
1 for i in range(len(names)-1,0,-1):
2     if names[i]=='mrudula':
3         break
4     print(names[i],end=" ")
```

```
poojitha
```

In [ ]:
```
1
```

Sets

S = { 20, 'Jessa', 35.75}

- ✓ **Unordered**: Set doesn't maintain the order of the data insertion.
- ✓ **Unchangeable**: Set are immutable and we can't modify items.
- ✓ **Heterogeneous**: Set can contains data of all types
- ✓ **Unique**: Set doesn't allows duplicates items

In [197]:
```
1 s = {1,11,45,1,76,89,23,11,89}
2 print(s)
```

```
{1, 11, 76, 45, 23, 89}
```

In [200]:
```
1 s.add(10)
```

In [201]:
```
1 s
```

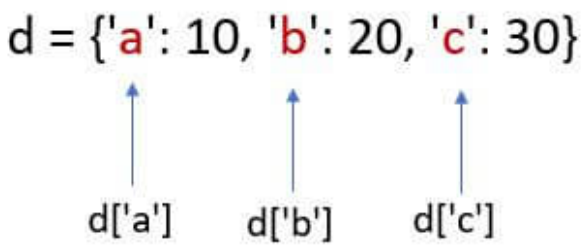
Out[201]:
```
{1, 10, 11, 23, 45, 76, 89, 100}
```

In [202]:
```
1 dir(set)
```

...

Dictionaries

Unordered collections of unique values stored in (Key-Value) pairs.



- ✓ **Unordered**: The items in dict are stored without any index value
- ✓ **Unique**: Keys in dictionaries should be Unique
- ✓ **Mutable**: We can add/Modify/Remove key-value after the creation

In [ ]:
```
1
```