In [43]:

```
# String: Any sequence of characters within either single quotes or double quotes is considered as a string.
# Syntax: emp1name='aravind'
```

In [44]:

```
emp1name='aravind'
print(type(emp1name))
```

```
<class 'str'>
```

In [47]:

```
# defining multi-line string literals - we can define multiline string literals by using triple single or double
quotes.
emp='''aravind is working at
software solutions'''
print(emp)
```

```
aravind is working at
software solutions
```

In [48]:

```
# how we can access characters of a string
# we can access characters of string by following ways
# 1. by using index
# 2. by using slice operator
```

In [49]:

```
# 1. By using index
# Python suppors both +ve and -ve
# + index means left to right(forward direction)
# - index means right to left(backward direction)
```

In [55]:

```
#eg
empl='aravind'
print(emp1[0])
print(emp1[-1])
print(emp1[3])
```

```
a
d
v
```

In [57]:

```
print(emp1[10]) # if we are trying to access characters of a string with out of range index then we will get Inde
xError.
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-57-ac759670d097> in <module>
----> 1 print(emp1[10]) # if we are trying to access characters of a string with out of range index
then we will get IndexError.

IndexError: string index out of range
```

In [58]:

```
# Accessing characters by using slice operator
# Syntax: s[BeginIndex:stopIndex:step]
# Begin index: from where we want to start our slice(substring)
# end index: We have to terminate the slice at endindex-1
# step: incremental value

#note: if we are not specifying begin index then it will consider from begining of the string. If we are not spec
ifying end
# index then it will consider up to end of the string.
# The default value for step is 1.
```

In [67]:

```
#eg:
emp_description='aravind writes code in python'
print(emp_description[1:8])
print(emp_description[1:8:1])
print(emp_description[1:8:2])
print(emp_description[:8])
print(emp_description[8:])
print(emp_description[::])
print(emp_description[::-1])
```

```
ravind
ravind
rvn
aravind
writes code in python
aravind writes code in python
nohtyp ni edoc setirw dnivara
```

In [68]:

```
# mathematical operators of string
# we can apply the following mathematical operators for strings
```

In [69]:

```
# + operator for concatenation
print('aravind'+'reddy')
```

```
aravindreddy
```

In [70]:

```
# * operator for repetetion
print("aravind"*3)
```

```
aravindaravindaravind
```

In [71]:

```
# note: to use + operator fro strings compulsory both arguments should be str type.
# to use * operator for strings compulsory one argumnent should be str and other argument shoulf be int.
```

In [72]:

```
# len() - we can use len() function to find number of characters in string.
emp_name='aravindreddy'
print(len(emp_name))
```

```
12
```

In [76]:

```
# checking membership - we can check whether the character or string is the member of another string or not by us
ing in and
# not in operators.
emp_name='aravindreddy'
print('a' in emp_name)
print('k' in emp_name)
```

```
True
False
```

In [77]:

```
# comparison of strings - we can use comparison operators(<,<=,>,>=) and equality operators(==,!=) for strings.
#comparison will be based on alphabetic order
```

In [78]:

```
emp1=input("Enter the emp1 name:")
emp2=input("Enter the emp2 name:")
if emp1==emp2:
    print("Both strings are equal")
elif emp1<emp2:
    print('emp1 string is less than emp2')
else:
    print('emp1 string is greater than emp2 string')
```

```
Enter the emp1 name:aravind
Enter the emp2 name:rahul
emp1 string is less than emp2
```

In [79]:

```
# removing spaces of string:
# we can use following 3 methods.
# 1. rstrip() - to remove spaces on right hand side
# 2. lstrip() - to remove spaces on left hand side
# 3. strip() - to remove spaces on both sides.
```

In [80]:

```
emp1=input("Enter the employee name:")
semp1=emp1.strip()
if semp1=='aravind':
    print('he is python developer')
elif semp1=='rahul':
    print('he is java developer')
else:
    print('he is c developer')
```

```
Enter the employee name:   aravind
he is python developer
```

In [85]:

```
# finding substrings

# 1.find(): returns index of first occurrence of given substring. if it is not avaiable then we will get -1.
# syntax: s.find(substring)
s='python is very easy to learn'
s.find('is')

#syntax : s.find(substring,beginindex,endindex)
print(s.find('e',15,20))
print(s.find('z',7,12))


# index(): index is exactly same as find() method except that if the specified substring is not available then we
will get
# ValueError
print(s.index('easy'))
```

```
15
-1
15
```

In [86]:

```
# counting the substring in given string
# we can find the no of occurences of substring present in the given string by using count() method
# Syntax : s.count(substring) ==> It will search through out the string.
# Syntax: s.count(substring,beginindex,endindex) ==> it will search from begin index to end-1 index
```

In [93]:

```
#eg:
emp1_description='he is a python developer'
print(emp1_description.count('a'))
print(emp1_description.count('o',2,13))
```

```
1
1
```

In [95]:

```
# replacing a string with another string
# Syntax : s.replace(oldstring,newstring)
# indside s , every occurrence of old string will be replaced with new string.
```

In [98]:

```
emp1_description='he is a python developer'
emp1_des=emp1_description.replace('python','java')
print(emp1_des)

print(id(emp1_description))
print(id(emp1_des))
```

```
he is a java developer
2580596943328
2580596916560
```

In [99]:

```
# string objects are immutable then how we change the content by using replace() method.
# once we creates a string object, we cannot change the content, this non changable behaviour is nothing but immu
tability.
# If we are trying to change content by using any method, then with those changes new object will be created and
changes
# won't be happened in existing object.
```

In [100]:

```
# Splitting of strings
# We can split the given string according to specified seperator by using split() method
# Syntax=s.split(seperator)
#The default seperator is space. The return type of split() method is List.
```

In [102]:

```
emp1_fullname='aravind reddy'
l=emp1_fullname.split()
for x in l:
    print(x)
```

```
aravind
reddy
```

In [104]:

```
k='20-03-2020'
s=k.split('-')
for x in s:
    print(x)
```

```
20
03
2020
```

In [106]:

```
# joining of strings - We can join a group of strings (list or tuple) wrt to given seperator
# Syntax: s=seperator.join(groupofstrings)
fullname=['aravind','reddy']
s=' '.join(fullname)
print(s)
```

```
aravind reddy
```

In [107]:

```
# changing case of string
# We can change the case of a string by following methods.
# 1.upper() - to convert all characters to uppercase
# 2.lower() - to convert all characters to lowercase
# 3.swapcase() - to convert lowercase characters to uppercase and all uppercase characters to lowercase
# 4. title() - convert first character in every word to uppercase.
# 5. capitalize() - only first character will be converted to uppercase
```

In [108]:

```
emp1='aravind'
print(emp1.upper())
print(emp1.lower())
print(emp1.swapcase())
print(emp1.title())
print(emp1.swapcase())
print(emp1.title())
print(emp1.capitalize())
```

```
ARAVIND
aravind
ARAVIND
Aravind
ARAVIND
Aravind
Aravind
```

In [109]:

```
# checking starting and ending part of string
# python contains the following methods for purpose
# Syntax: 1.s.startswith(substring)
# Syntax: 2.s.endswith(substring)
```

In [111]:

```
emp1='aravind'
print(emp1.startswith('a'))
print(emp1.endswith('a'))
```

```
True
False
```

In [118]:

```
# To check the type of characters present in string
# python contains following methods for purpose
# 1.isalnum() - returns true if all characters are alphanumeric
# 2.isalpha() - returns true if all characters are alphabets.
# 3.isdigit() - returns true if all characters are digits only.
# 4.isupper() - returns true if all characters are uppercase alphabet symbols.
# 5.islower() - returns true if all characters are lowercase alphabet symbols.
# 6.istitle() - returns true if string is in titlecase
# 7.isspace() - returns true if string contains only space
```

In [119]:

```
emp1name='aravind'
print(emp1name.isalnum())
print(emp1name.isalpha())
print(emp1name.isdigit())
print(emp1name.isupper())
print(emp1name.islower())
print(emp1name.istitle())
print(emp1name.isspace())
```

```
True
True
False
False
True
False
False
```

In [124]:

```
# formatting the strings
# we can format the strings with variable values by using replacement operator {} and format() method
#eg:
emp_name='aravind'
emp_salary=85000
emp_language='Python'
print("{} salary is {} and language of the employee is {}".format(emp_name,emp_salary,emp_language))
print("{0} salary is {1} and language of the employee is {2}".format(emp_name,emp_salary,emp_language))
print("{x} salary is {y} and language of the employee is {z}".format(x=emp_name,y=emp_salary,z=emp_language))
```

```
aravind salary is 85000 and language of the employee is Python
aravind salary is 85000 and language of the employee is Python
aravind salary is 85000 and language of the employee is Python
```

In [ ]: