string manuplation"

```
s1="this is my string class"
```

```
#different type function we have used in the string manuplation
```

```
# len function
# find how many space or len have of s1 string
len(s1)
```

```
23
```

```
# find fuction find is there s present in the s1 string
s1.find('s')
```

```
3
```

above 3 indicate the first s is present in the s1 at the 3 index *italicized text*

```
# find the sub string (my) in the s1 string
s1.find("my")
```

```
8
```

```
# to find different type funtion in the string put the string name and put dot and then press shift
```

```
# que find how many s is present in the s1 string
s1.count('s')
```

```
5
```

```
s1.count('z')
```

```
0
```

```
# que convert all the lower string data into upper string
s1.upper()
```

```
'THIS IS MY STRING CLASS'
```

```
s2='THIS IS MY STRING CLASS'
```

```
# que convert the s2 string in the lower word
s2.lower()
```

```
'this is my string class'
```

```
# que convert all the first word in the s1 string in the upper case
s1.title()
```

```
'This Is My String Class'
```

```
a="ashish"
```

```
# que convert the a string in the 3 times and print
print(a * 3)
```

```
ashishashishashish
```

```
#
```

Date-31 jan 2023

list

```
l=[1,345,45,"sudh",True,5+7j,345.56]
```

```
# find the type of l
type(l)
```

```
list
```

```
# que find 1 from the list l
l[0]
```

```
1
```

```
# que find the value from the list l 1,345,45 from the l list or create a list of 1,345,45 value
l[0:3]
```

```
[1, 345, 45]
```

```
# que find the last word from the list of l
l[-1]
```

```
345.56
```

```
# que reversed the list l
l[::-1]
```

```
[345.56, (5+7j), True, 'sudh', 45, 345, 1]
```

```
# que find data from list l which is available on the even place in the list
l[0:7:2]
```

```
[1, 45, True, 345.56]
```

```
# que find the data from the list l which is available on the odd place in the list L
l[0:7:3]
```

```
[1, 'sudh', 345.56]
```

```
# Que extract word su from the list l
l[3][0:2]   #
```

```
'su'
```

```
# que extract tr from the true list
str(l[4])[0:2]
```

```
'Tr'
```

```
# que find how many no of element in the list
len(l)
```

```
7
```

```
# que append a 2 in the list l
l.append(2)
```

```
l
```

```
[1, 345, 45, 'sudh', True, (5+7j), 345.56, 2, 2]
```

```
l1=[1,2,3,4,5]
# que clear number from the list

l1.clear()


l1
```

```
[]
```

```
# Que Copy the list in the a
a=l.copy()


l2=[3,4,5]

# que store l2 list in the l list
l.append(l2)
l
```

```
[1, 345, 45, 'sudh', True, (5+7j), 345.56, 2, 2, [3, 4, 5], [3, 4, 5]]
```

```
# Remove the l2 from the list
l.remove(l2)


l
```

```
[1, 345, 45, 'sudh', True, (5+7j), 345.56, 2, 2, [3, 4, 5]]
```

```
#  Que find the 4 from the l list
l[9][1]
```

```
4
```

```
l.extend('4')


l
```

```
[1, 345, 45, 'sudh', True, (5+7j), 345.56, 2, 2, [3, 4, 5], '4']
```

```
# que insert 345 in the list l by the seperate way
l.extend('345')


l
```

```
[1,
 345,
 45,
 'sudh',
 True,
 (5+7j),
 345.56,
 2,
 2,
 [3, 4, 5],
 '4',
 3,
 4,
 5,
 '3',
 '4',
 '5']
```

```
l2=[1,2,3]

# QUE insert word ash at index 1
l2.insert(1,'ash')



l2
```

```
[1, 'ash', 2, 3]
```

```
l2.append([2,3,4])
```

```
l2
```

```
[1, 'ash', 2, 3, [2, 3, 4]]
```

```
# Remove 4 from the list l2
l2[4].remove(4)
```

```
l2
```

```
[1, 'ash', 2, 3, [3]]
```

```
# que reversed l list
l.reverse()
```

```
l
```

```
['5',
 '4',
 '3',
 5,
 4,
 3,
 '4',
 [3, 4, 5],
 2,
 2,
 345.56,
 (5+7j),
 True,
 'sudh',
 45,
 345,
 1]
```

```
l3=[4,5,66,78,9,90]
# Que sort l3 list in systmatic way
```

```
l3.sort()
```

```
l3
```

```
[4, 5, 9, 66, 78, 90]
```

```
l4=['a','c','z','j','f']
```

```
l4.sort()
```

```
l4
```

```
['a', 'c', 'f', 'j', 'z']
```

tupple Data type

```
t=(2,3,4,5,2,"sudh",45.56,False,(45+457j),[3,4,5])
```

```
# fuction in tupple
```

```
# count how many 2 in the t tupple
```

```
t.count(2)
```

```
2
```

```
t.index('sudh')
```

```
5
```

Set data type

```
s3={324,456,456,'sudh',45+45j,34.465,(3,4,5)}
```

```
s3 # set is helping in finding the unique value
```

```
{(3, 4, 5), (45+45j), 324, 34.465, 456, 'sudh'}
```

```
l5=[1,2,3,4,5,5,6,6,2,3,44,5]
```

```
# Que find the unique value from the list
```

```
l6=set(l5)
```

```
l5=list(l6)
l5
```

```
[1, 2, 3, 4, 5, 6, 44]
```

Class 01-Feb-2023

dic Data type

```
d2={'name':'sudhanshu','Email':'ash@gmail.com','phone no':9891139045}
```

```
# que find then name from the dic
d2['name']
```

```
'sudhanshu'
```

```
# que find the phone no of the user
d2['phone no']
```

```
9891139045
```

```
d3={'company':'pwskills','course':['web dev','data science','java with dsa system design']}
```

```
# que find extract coursse data science from above dic
d3['course'][1]
```

```
'data science'
```

```
d4={'number':[2,34,3,34,34],'assignment':(1,2,3,4,5,6),'launch_date':{28,12,14},'class_time':{'web developmen':1,'Data sceience':2}}
```

```
# que extract value 2 from the d4 dic
d4['class_time']['Data sceience']
```

```
2
```

```
# que add a key mentor in the d4 dic
d4['mentor']=['anshish','rakesh','sanjay']
```

```
d4
```

```
{'number': [2, 34, 3, 34, 34],
 'assignment': (1, 2, 3, 4, 5, 6),
 'launch_date': {12, 14, 28},
 'class_time': {'web developmen': 1, 'Data sceience': 2},
 'mentor': ['anshish', 'rakesh', 'sanjay']}
```

```
# que find all key name in d4 dic

d4.keys()
```

```
dict_keys(['number', 'assignment', 'launch_date', 'class_time', 'mentor'])
```

if and else statement

Double-click (or enter) to edit

```
marks=int(input())

if marks>=80:
  print('you are qualify for Ao batch')
elif marks>=60 and marks<80:
  print('you are qualify for A1 batch')
elif marks>=40 and marks<60:
  print('you are qualify for A batch')
else:
  print('you are not qualify for the for any batch')
```

```
67
you are qualify for A1 batch
```

```
# multiple condition in the if else

price=int(input('enter you price'))

if price>=1000:
  print('i will not purchase')
  if price>900:
    print('i will not purchase')
    if price>=800:
      print('i will not purchase dude')
```

```
enter you price1100
i will not purchase
i will not purchase
i will not purchase dude
```

Loop

```
l=[1,2,3,4,5,6,7,8,9]

# Que add 1 in the all element in the l list

b=[]
for i in l:
   b.append(i+1)
print(b)
```

```
[2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
l=['sudh','kumar','pwskills','course']

# Que convert the all small letter in the Capital form

b=[]
for i in l:
  b.append(i.upper())

print(b)
```

```
['SUDH', 'KUMAR', 'PWSKILLS', 'COURSE']
```

```
l=[1,2,3,4,5,'Ash','Rajput',4,5,6,7]
```

```
# que separate numerical and catogorical value from the l list to different list

num=[]
cat=[]

for i in l:
  if type(i)==int or type(i)==float:

    num.append(i)
  else:
    cat.append(i)

print(num)
print(cat)
```

```
[1, 2, 3, 4, 5, 4, 5, 6, 7]
['Ash', 'Rajput']
```

```
l=[1,2,3,4,5]

# que print each no from l list with there data type

for i in l:
  print(i,type(i))
```

```
1 <class 'int'>
2 <class 'int'>
3 <class 'int'>
4 <class 'int'>
5 <class 'int'>
```

break condition in for loop

1. break in the python generally break the loop where condition mention

```
l=['ash','kumar','rajput']

# que use loop for print above value without rajput and kumar

for i in l:
  if i=='kumar':
    break
  print(i)
```

```
ash
```

Continue funtion

1.when we use continue function in the for loop when condition reach in the for loop and continue fuction is executed so on that time it automatically excute the for loop again and it not print that i value

```
# que print all value from the list l except kumar from l

for i in l:
  if i=='kumar':
    continue
  print(i)
```

```
ash
rajput
```

range function

1.Range function are the function which generally generate a data between define range

```
range(5)
```

```
range(0, 5)
```

```
# que create a list by using range funtion in which we have 5 digit
```

```
list(range(5))
```

```
[0, 1, 2, 3, 4]
```

```
# que create a list with alternative no
```

```
list(range(0,9,2))
```

```
[0, 2, 4, 6, 8]
```

```
l=['ash','rajput','kumar','sanjay']
```

```
# Que element from list l from the opposite direction
```

```
for i in range(len(l)-1,-1,-1):
  print(l[i])
```

```
sanjay
kumar
rajput
ash
```

```
l2=[1,2,0,3,4,5,6,6,7,8,8,9]
```

```
for i in l2:
  print(l2[0:len(l2)-1:2])
  break
```

```
[1, 0, 4, 6, 7, 8]
```

```
l2[0:112]
```

```
[1, 0, 4, 6, 7, 8]
```

```
# que find the sum of l2 list
sum=0
for i in l2:
  sum+=i
print(sum)
```

```
59
```

```
sum(list(l2))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-179-0115d0556098> in <module>
----> 1 sum(list(l2))

TypeError: 'int' object is not callable

SEARCH STACK OVERFLOW
```

```
l=[1,2,3,4,4,5]
```

```
d={'name':'sudh','class':'data science master','topic':['python','stats','machine learning','DL','cv','nlp','resume','interview']}
```

```
d.keys()
```

```
    dict_keys(['name', 'class', 'topic'])
```

```
for i in d.keys():
  print(d[i])
```

```
    sudh
    data science master
    ['python', 'stats', 'machine learning', 'DL', 'cv', 'nlp', 'resume', 'interview']
```

```
d['name']
```

```
    'sudh'
```

while loop

1 difference between for loop and while loop is that while loop is never check any condition where for loop each time check condition which you have mentioned in the while loop time

```
a=1

while a<=10:
  print(a)
  a=a+1
```

```
    1
    2
    3
    4
    5
    6
    7
    8
    9
    10
```

```
from collections import Counter
# take a input from the user print the sum of that number using while loop

n=int(input('hi dude please enter your no'))
a=0
b=1

while b <=n:
  a+=b
  b+=1
print(a)
```

```
    hi dude please enter your no5
    15
```

```
3*2*1
```

```
    6
```

```
# que find the factorial of the no without while loop

a= int(input())
b=[]
c=1

if a>0:
  for i in range(a,1,-1):
    b.append(i)
  for j in b:
    c=c*i
    i+=1
else:
  print('dude enter a number that is greate then 0')

print(c)
```

```
  # i++
# print(b)



# que find the factorial of the no with using while loop
n=int(input("enter a number"))
f=1
while n>0:
  f=f*n
  n=n-1
f



# finonacci


n=int(input())
a,b = 0,1
c=0

while c<n:
  print(a)
  c=a+b
  a=b
  b=c
  c+=1




# que print a table of the user provide a number
a=int(input())
i=1
while i<10:
  result=a*i
  print(a,"*",i,"=",result)
  i=i+1
```

```
4
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
```

Comprehension

```
l=[1,2,3,4,5,6]
l1=[]
for i in l:
  l1.append(i**2)
print(l1)
```

```
[1, 4, 9, 16, 25, 36]
```

```
# write above code in one line

[i**2 for i in l]
```

```
[1, 4, 9, 16, 25, 36]
```

```
# que find the even no from the l list
[i for i in l if i %2==0]
```

```
[2, 4, 6]
```

```
a=['ash','kumar','rajput']

[i.upper() for i in a]
```

```
['ASH', 'KUMAR', 'RAJPUT']
```

function

```
def test9(a):
  """this is my function to extract num data from list"""
  l=[]
  for i in a:
    if type(i)==list:
      for j in i:
        l.append(j)
    else:
      if type(i)==int or type(i)==float:
        l.append(i)
  return l
```

```
# note *args is used to pass multiple data in the function or mean to it used for the dynamic function

def test1(*args):
  return args
```

```
type(test1)
```

```
function
```

```
test1(1,2,3)

print(test1('a','b','c'))
```

```
('a', 'b', 'c')
```

```
def test2(*args,b):
  return args, b
```

```
test2(1,2,3,4,b=5)
```

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

**kwargs

is a function which use to form a function which help to pass multiplt key value inside a function

```
def test4(**kwargs):
  return kwargs
```

```
type(test4())
```

```
dict
```

```
test4(a=[1,2,3,4],b='ash')
```

```
{'a': [1, 2, 3, 4], 'b': 'ash'}
```

```
x`
```

```
test15(a=[1,2,3,4],b='sudh',c=23.45)
```

```
{'a': [1, 2, 3, 4], 'b': 'sudh', 'c': 23.45}
```

Genarater function

- Generate function help to generate numerical value between define range it is different from range function due to memory consumption in the range function memory consuption is more but in generator function mermory consumption of machine is less
- we can make generater function by the help of yield

lambda function is called annamoulys function becase this function call without name because first we make a function then insert into a variable that is called annolmous function

```
def test_fib(n):
  a,b=0,1
  for i in range(n):
    yield a
    a,b=b,a+b


for i in test_fib(10):
  print(i)
```

```
0
1
1
2
3
5
8
13
21
34
```

```
def count_test(n):
  count=1
  while count <=n:
    yield count
    count=count+1



for i in count_test(5):
  print(i)
```

```
1
2
3
4
5
```

lambda function

lambda is function that is called the one line function or abnomous function

Double-click (or enter) to edit

```
lambda (first return input value):(return output value that you want)
```

```
  File "<ipython-input-15-f4d1d170f31d>", line 1
    lambda (first return input value):(return output value that you want)
           ^
SyntaxError: invalid syntax
```

SEARCH STACK OVERFLOW

```
n=3
p=2
```

```
# write a function for the square of the number
```

```
# without lambda function
def test(n,p):
  return n**p
```

```
test(3,2)
```

```
9
```

```
a=lambda n,p:n**p
```

```
a(3,2)
```

```
9
```

```
# write a function which add two no using lambda
b=lambda x,y:x+y
b(2,3)
```

```
5
```

```
# write a function for adding the two no
c=lambda x,y:x*y
c(3,4)
```

```
12
```

```
# write a function that which help to find big numerical value
finding_max=lambda x,y:x if x>y else y
finding_max(3,4)
```

```
4
```

```
# Write a function which help to return a lenght of string
c=lambda s:len(s)
c('rakesh')
```

```
6
```

map,reduce,filter function

map function map all the value present in map with the given function in the map

```
l=[2,3,4,5,6]
```

```
def test(a):
  a1=[]
  for i in a:
    a1.append(i**2)
  return a1
```

```
test(l)
```

```
[4, 9, 16, 25, 36]
```

```
# we can same thing with the help of map function
def test1(a2):
  return a2**2

list(map(test1,l))
```

```
[4, 9, 16, 25, 36]
```

```
# we can do same thing with help of lambda function
c=lambda x: x**2
```

```
list(map(c,l))
```

```
[4, 9, 16, 25, 36]
```

```
# addintion of two list using map and lambda function
a=[1,2,3,4,5,6]
b=[4,5,6,7,8,9]

c=lambda a,b:a+b

list(map(c,a,b))
```

```
[5, 7, 9, 11, 13, 15]
```

```
# que convert a string into upper case using lambda and map function
s="ashish"

b=lambda s: s.upper()

list(map(b,s))
```

```
['A', 'S', 'H', 'I', 'S', 'H']
```

Reduced

reduce is also work like a map which work on the whole data or it consolidate whole data value into one value

```
from functools import reduce
```

```
from functools import reduce
l=[1,2,3,4,5,6]
c=lambda x,y:x+y
reduce(c,l)
```

```
21
```

```
# Note -in reduce function we have not take more then two Perameter
```

Double-click (or enter) to edit

```
# que find the maximum value from the l list
a=lambda x,y:x if x>y else y
reduce(a,l)
```

```
6
```

Filter function

- Filter function help to filter the value from the given data

```
l=[1,2,3,4,5,6,7,8]

# que find the even value from the list
c=lambda x:x%2==0
list(filter(c,l))
```

```
[2, 4, 6, 8]
```

```
a=23
b=44
b=2+3j
a=b.imag
print(b-a)
```

```
(-1+3j)
```

```
a="ash"
b=1
print(a+str(b))
```

```
ash1
```

```
pw-skills=5
```

```
  File "<ipython-input-42-dac1b08cb419>", line 1
    pw-skills=5
    ^
SyntaxError: cannot assign to operator
```

SEARCH STACK OVERFLOW

```
a='pwskills'
a[0]='P'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-44-92f62c074699> in <module>
      1 a='pwskills'
----> 2 a[0]='P'

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

SEARCH STACK OVERFLOW

```
i=0
while i<5:
  print(i)
  i+=1
```

```
0
1
2
3
4
```

```
s="i want to become a data scientis"

for i in s.split():
  i.capitalize()
  print(i.title())
```

```
I
Want
To
Become
A
Data
Scientis
```

OOPS Programming

"OOPS" stand for object-oriented Programming System

```
a=1
```

```
print(type(a))
```

```
<class 'int'>
```

```
type(a)
```

```
int
```

```
print(type("Pwskills"))
```

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

```
# class is also know as variable or object or instances  or also know as blueprint of object
```

Addvantage of use of class in industry

1) in real world in industry we have multiple module for the product like frontend module,backend module ,database module etc in each module we have wright different code like in the frontend we right frontend Related code in the backend we can wright backend related code in database we can wright data base realted code if we wanted to code related to particular module so on that time we define class those person who wright any code related to module like frontend related code they can mention that front class and related to that class they mention that code so class help to increase code resuseabilty and less complexity etc

```
class test:
  pass


a=test()


type(test())
```

```
__main__.test
```

```
print(type(test()))
```

```
<class '__main__.test'>
```

```
class pwskills:
  def welcome_msg(self):    # we have used self which help to bind welcome_msg to pwskills class
    print('welcome to pwskills')

rohan=pwskills()
rohan.welcome_msg()
```

```
welcome to pwskills
```

```
Rakesh=pwskills()
Rakesh.welcome_msg()
```

```
welcome to pwskills
```

```
class pwskills1:
  def __init__(self,Phone_number,Email_id,student_id):   # __init__ it is constructor (construter always take data by the help of __init__ we
    self.Phone_number=Phone_number
    self.Email_id=Email_id
    self.student_id=student_id

  def return_student_details(self):
    return self.student_id,self.Phone_number,self.Email_id
    pass


rohan=pwskills1(9891139045,'rohan@gmail.com',101)


rohan.return_student_details()
```

```
(101, 9891139045, 'rohan@gmail.com')
```

```
Rakesh=pwskills1(9891139045,'rakesh@gmail.com',102)
Rakesh.return_student_details()
```

```
(102, 9891139045, 'rakesh@gmail.com')
```

```
Rakesh.Phone_number
```

```
9891139045
```

```
class pwskills2:
  def __init__(self,Phone_number,Email_id,student_id):   # __init__ it is constructor (construter always take data)
    self.Phone_number1=Phone_number
    self.Email_id1=Email_id
    self.student_id1=student_id
```

```
  def return_student_details(self):
    return self.student_id1,self.Phone_number1,self.Email_id1
    pass


Rakesh=pwskills2(9891139045,'rakesh@gmail.com',102)
Rakesh.return_student_details()
```

```
(102, 9891139045, 'rakesh@gmail.com')
```

```
class pwskills3:
  def __init__(sudh,Phone_number,Email_id,student_id):   # __init__ it is constructor (construter always take data)
    sudh.Phone_number1=Phone_number
    sudh.Email_id1=Email_id
    sudh.student_id1=student_id

  def return_student_details(sudh):
    return sudh.student_id1,sudh.Phone_number1,sudh.Email_id1
    pass


Rakesh=pwskills3(9891139045,'rakesh@gmail.com',102)
Rakesh.return_student_details()
```

```
(102, 9891139045, 'rakesh@gmail.com')
```

from above we can find conclusion is this self is not a fix parameter which help the function to bind with class we can pass any variable in the first position in constructor which perform same function

```
my_dic={"a":1,"b":2,"c":3}
my_set=set(my_dic.keys())


my_dic.keys()
```

```
dict_keys(['a', 'b', 'c'])
```

```
a=1
print(a)
```

```
1
```

Polymorphism

polymorphism is property which tell that there a one entity they have different different behaviour in different different circumstance that property we call polymorphism

```
def test(a,b):
  return a+b



print(test(3,4))
print("ash","kumar")
print(test([3,4,5,5],[45,78,2]))
```

```
7
ash kumar
[3, 4, 5, 5, 45, 78, 2]
```

from above we can see that generally i have make test function for adding a two number but when i use this function for concatinate two string that this is also take place .so from there we can understand the property of ploymorphishm at different different time function have behave different operation

```
class data_science:
  def syllabus(self):
    print("this is my syllabus for data science master")
```

```
class web_dev:
  def syllabus(self):
    print("this is my syllabus for web dev")


def class_parcer(class_obj):
  for i in class_obj:
    i.syllabus()


a=data_science()


b=web_dev()


Class_obj=[data_science,web_dev]


class_parcer(class_obj)
```

```
     this is my syllabus for data science master
     this is my syllabus for web dev
```

Encapsulation

encapsultion is method which prevent the user to understant inside data of class *italicized text*

Double-click (or enter) to edit

```
class test:
  def __init__(self,a,b):
    self.a=a
    self.b=b


t=test(9,8)


t.a=345


t.a
```

```
     345
```

from above you can see that i have assign A value is 9 but another user come and they assign A value 345 this create a problem to resolve this problem we use the encupsulation

```
class Car:
  def __init__(self,year,make,model,speed):
    self.__year=year
    self.__make=make
    self.__model=model
    self.__speed=0
  def set_speed(self,speed):
    self.__speed=0 if speed <0 else speed
  def get_speed(self):
    return self.__speed


c=Car(2021,"toyota","innova",12)


c._Car__year
```

```
     2021
```

```
c.set_speed(-3245)


c.get_speed()
```

```
     0
```

```
class bank_account:
  def __init__(self,balance):
    self.__balance=balance

  def deposite(self,amount):
    self.__balance=self.__balance+amount
  def withdaw(self,amount):
    if self.__balance>=amount:
      self.__balance=self.__balance-amount
      return True
    else:
      return False
  def get_balance(self):
    return self.__balance


ash=bank_account(1000)


ash.get_balance()
```

```
1000
```

```
ash.deposite(10000)


ash.get_balance()
```

```
11000
```

```
ash.withdaw(8000)
```

```
True
```

```
ash.get_balance()
```

```
3000
```

```
ash.withdaw(6000)
```

```
False
```

inheritance

inheritance is the property in which the one entity propety also transfer to another entity

```
class test:
  def test_math(self):
    return "this is my first class"


class child_test(test):
  pass


child_test_obj=child_test()


child_test_obj.test_math()
```

```
'this is my first class'
```

look above class test property will transfer into child_test class

Multiple level inheritance

multilevel inheritance is the property where one class property is transfer to another class and that class property is transfer to another class for example

class a propery transfer to class b and class b property transfer in class c so we can say that in class c property of class a such type inheritance we called multilevel inheritance

Example gradfather -->father-->child

```
class class1:
  def test_class(self):
    return "this is a math from class1"


class class2(class1):
  def test_class1(self):
    return "this is new class"



class class3(class2):
  pass


obj=class3()


obj.test_class()
```

```
'this is a math from class1'
```

```
obj.test_class1()
```

```
'this is new class'
```

multiple inheritance

multiple inheritance is that if we excess from one class more then two class we call it multiple inheritance

class 3 have class 2 and class 1 property both mix so we call class 3 as multiple class but in the multilevel class class c have excess class 1 property by the help of class 2

example mom and dad ----> child

```
class class1:
  def test_class1(self):
    return 'aur dude'



class class2:
  def test_class2(self):
    return 'hi guys'


class class3(class1,class2):
  pass


obj1=class3()


obj1.test_class1()
obj1.test_class2()
```

```
'hi guys'
```

Decorater

Decorater is help to enhance the beautiness of the function

```
def test():
  print("this is start of my fun")
  print("this is my fun to test")
  print(4+5)
  print("this is the end of fun")
```

```
test()
```

```
     this is start of my fun
     this is my fun to test
     9
     this is the end of fun
```

```
def deco(func):
  def inner_dec():
    print("this is the start of my fun")
    func()
    print("this is the end of my fun")
  return inner_dec
```

```
@deco
def test1():
  print(6+7)
```

```
test1()
```

```
     this is the start of my fun
     13
     this is the end of my fun
```

form above we can see that by the help of deco function we can print all the value of function deco function and test1

```
# how many time is take by the function for execute

import time
def time_test(func):
  def timer_test_inner():
    start=time.time()
    func()
    end=time.time()
    print(end-start)
  return timer_test_inner
```

```
@time_test
def test2():
  print(45+78)
```

```
test2()
```

```
     123
     6.008148193359375e-05
```

class method

```
class pwskills:
  def __init__(self,name,email):
    self.name=name
    self.email=email
  def student_details(self):
    return self.name,self.email
```

```
pw=pwskills("ash",'ash@gmail.com')
```

```
pw.name
```

```
     'ash'
```

```
pw.email
```

```
     'ash@gmail.com'
```

```
class pwskills1:
  def __init__(self,name,email):
```

```
    self.name=name
    self.email=email

  @classmethod
  def details(cls,name,email):
    return cls(name,email)
  def students_details(self):
    print(self.name,self.email)


pw1=pwskills1.details("sudh","sudh@gmail.com")


pw1.name
```

```
'sudh'
```

from above we can see that we can print the class data without creating of object on that time we just need to mentioned classmethod class method at all

```
class pwskills4:
  mobile_num=912345
  def __init__(self,name,email):
    self.name=name
    self.email=email

  @classmethod
  def change_number(cls,mobile):
    pwskills2.mobile_num=mobile
  @classmethod
  def details(cls,name,email):
    return cls(name,email)
  def students_details(self):
    print(self.name,self.email,pwskills2.mobile_num)


# for deleted the change_no from the pwskills4 class
del pwskills4.change_number
```

static method

```
class pwskills1:
  def student_details(self,name,mail_id,number):
    print(name,mail_id,number)

  @staticmethod
  def mentor_class(list_mentor):
    print(list_mentor)
  def mentor(self,mentor_list):
    print(mentor_list)


pwskills1.mentor_class(['sudh',"krish"])
```

```
['sudh', 'krish']
```

special magic and Dunder method

```
a=100


a+5
```

```
105
```

mechanism of above addition program is take place by the help of mazic no or dunder function for opperation generally behind magic function or dunder function' **add** ' is used by the help of this magic and dunder function we used it

```
# Example of above mechanism
a.__add__(5)
```

```
105
```

```
dir(int)
```

```
['__abs__',
 '__add__',
 '__and__',
 '__bool__',
 '__ceil__',
 '__class__',
 '__delattr__',
 '__dir__',
 '__divmod__',
 '__doc__',
 '__eq__',
 '__float__',
 '__floor__',
 '__floordiv__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getnewargs__',
 '__gt__',
 '__hash__',
 '__index__',
 '__init__',
 '__init_subclass__',
 '__int__',
 '__invert__',
 '__le__',
 '__lshift__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__neg__',
 '__new__',
 '__or__',
 '__pos__',
 '__pow__',
 '__radd__',
 '__rand__',
 '__rdivmod__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rfloordiv__',
 '__rlshift__',
 '__rmod__',
 '__rmul__',
 '__ror__',
 '__round__',
 '__rpow__',
 '__rrshift__',
 '__rshift__',
 '__rsub__',
 '__rtruediv__',
 '__rxor__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__sub__',
```

Properties of Decorater function

Getters,setter and Deletes

```
class pwskills:
  def __init__(self,course_price,course_name):
    self.__course_price=course_price
    self.course_name=course_name

    # if we want to provide access rights to user for the access above value without call init so on that time we use properties function of

    @property
    def course_price_access(self):
```

```
        return self.__course_price

    # if we want to provide rights to user for the modify so on that time we use setter function which help
    # user to modify the set value as per his requirement

    @course_price_access.setter
    def course_price_set(self,price):
      if price<=3500:
        pass
      else:
        self.__course_price=price


     # if we want to provide a Role wrigths to user for the delet the price so on that time we use deleter
    @course_price_access.deleter
    def delete_course_price(self):
      del self.__course_price

pw=pwskills(3500,'data science master')



pw.course_price
```

```
    3500
```

```
pw.course_name
```

```
    'data science master'
```

```
pw.course_price_set=4500
```

```
del pw.delete_course_price
```

working with the file (11-Feb-23)

file operation

```
file=open()  #open is the inbuit function which to create file and  perform different operation in the file


f=open("text.txt","w")
```

```
pwd
```

```
    '/content'
```

```
# for the wright in the .txt file
f.write("this is my first file to write")


# note in the file you will not get wright data until you does not close the file
f.close()



# when we use w in the file function it always truncte the data from the file and then it insert the new data if we want our old data as well
# on that time we use 'a' inside of file
f=open("text.txt",'a')  # a mode help to store old data as well as new data


f.write("I am Data scientist")
```

```
    19
```

```
f.close()
```

```
# if we want only read data so on that time we use read mode
f=open("text.txt","r")
f.read()
```

```
'this is my first file to writeI am Data scientist'
```

```
f=open("text.txt",'a')
f.write('An open file format is a file format for storing digital data, defined by an openly published specification usually maintained by a
```

```
252
```

```
f.close()
```

```
f=open('text.txt','r')
f.read()
```

```
'this is my first file to writeI am Data scientistAn open file format is a file format for storing digital data, defined by an openly p
ublished specification usually maintained by a standards organization, and which can be used and implemented by anyone. Open file forma
t is licensed with open license '
```

```
# if we want to read data line by line so on that time we follow this readline function with seek method because in the seek method we fix th
# if we want to read data from the 15 pointer so on that time we can use seek=15 so on
f.seek(0)
f.readline()
```

```
'this is my first file to writeI am Data scientistAn open file format is a file format for storing digital data, defined by an openly p
ublished specification usually maintained by a standards organization, and which can be used and implemented by anyone. Open file forma
t is licensed with open license '
```

```
# if we want to find the size of text file so on that time we follow below

import os
os.path.getsize("text.txt")
```

```
301
```

```
# for the delet the file
os.remove("text.txt")
```

```
# rename the file
os.rename("text.txt","ash.txt")
```

```
# create copy of file
import shutil
shutil.copy("ash.txt","copy_ash.txt")
```

```
'copy_ash.txt'
```

```
with open("ash.txt","r") as f:
  print(f.read())
```

```
this is my first file to writeI am Data scientistAn open file format is a file format for storing digital data, defined by an openly put
```

```
# if we have data in the key value pair so on that time we use json method which help to wright such file
data={
    "name":"sudh",

    "mail_id" : "sudh@gmail.com",
    "phone number": 989123334,
    "subject":["data science","big data","data analytics"]

}
```

```
import json
with open("data.json","w") as f:
  json.dump(data,f)
```

```
# for the read file
with open("data.json","r") as f:
  data1=json.load(f)
```

```
data['subject'][1]
```

```
'big data'
```

Read and right of Csv file

```
data=[["name","email_id","phone_no"],
      ['Raj','raj@gmail.com',989555],
      ['ash','ash@gmail.com',896544],
      ['raku','raku@gmail.com',977655]]
```

```
# for the insert the csv data
import csv
with open("data.csv",'w') as f:
  writer =csv.writer(f)
  for i in data:
    writer.writerow(i)
```

```
# for read the csv data
with open("data.csv",'r') as f:
  read_data=csv.reader(f)
  for i in read_data:
    print(i)
```

```
['name', 'email_id', 'phone_no']
['Raj', 'raj@gmail.com', '989555']
['ash', 'ash@gmail.com', '896544']
['raku', 'raku@gmail.com', '977655']
```

for read wright of binary data

what is binary data ? binary data are data which is present in the 0 and 1 form

Example of binary file - when we open video ,photo data on that time we find such kind of data

> Indented block

```
# for create the binary file
with open('test4.bin',"wb") as f:
  f.write(b"\x01\x02\04")
```

```
# read binary
with open("test4.bin","rb") as f:
  print(f.read())
```

```
b'\x01\x02\x04'
```

buffer read rights

when we have big file and we want to read that big file on that time we used buffer if we talk about buffer buffer is method which help us to read rights big file in the small chunk

```
# for creating buffer file
import io
with open("test1.txt",'wb') as f:
  file =io.BufferedWriter(f)
  file.write(b"this is my first file hen we have big file and we want to read that big file on that time we used buffer if we talk about buff
  file.write(b"this is my second file")
  # for close the file in them we use flush
  file.flush
```

```
with open("test1.txt",'rb') as f:
  file=io.BufferedReader(f)
```

```
data=file.read(10)
print(data)
```

```
b''
```

Logging module

Loggin module is the module in which we create file and store all futuristic data in the file

use of loggin module

lOggin model is the model which help to investigate the code for future like at which place which code are crash or fail so many thing

and logging concept is come because if we print data in the console it remove when we print again another data so we have no record of privious data of console so in that scenario we use logging module where we store each point of console data for future investigation

Note on the Production level we not use the print method we always use loggin model for the print any thing in the console

inside the logging we have multiple level

1.NOTSET

2.DEBUG

3.INFO

4.WARNING

5.ERROR

6.CRITICAL

Note- level is always follow hirachy if we set logging as info then we can get all data of info and below info we can not store data above info example (Notset DeBUG etc)

```
print("this is my print")
```

```
this is my print
```

```
# if we not want print above thing in the console so on that time we loggin module which store all information in the new file
import logging
logging.basicConfig(filename='test.log',level=logging.INFO)
logging.info("log this line of execution")
logging.info("this is my print")
```

```
logging.debug("this is critical task")  # we can not got this data in the log file because it is above on info and we set logging as info
logging.warning('this is warning msg')  # we can  get this data in the log file because it is below on info and we set logging as info
logging.error("this is my error")       # we can  get this data in the log file because it is below on info and we set logging as info
```

```
WARNING:root:this is warning msg
ERROR:root:this is my error
```

```
# if we want to access time of data as well
logging.basicConfig(filename='text.log',level=logging.DEBUG,format='%(asctime)s%(message)s')
logging.info("this is my info logging")
logging.error("this is my error message")
logging.critical("this is my critical")
```

```
ERROR:root:this is my error message
CRITICAL:root:this is my critical
```

```
# for close above logging
logging.shutdown()
```

```
l=[1,3,3,4,5,[1,3,3,4],'ash','raj']
```

```
l1=[]
l2=[]
for i in l:
  logging.info('we are iterating through list and our local var is i')
  if type(i)==list:
```

```
    logging.info("i am inside if statement and i am trying to check list type")
    for j in i:
      logging.info("i am in another for loop for list inside list element")

      if type(j)==int or type(j)==float:
        logging.info("i am insude if statemet")
        l1.append(j)
  elif type(i)==int or type(i)==float:
    l1.append(i)
  else:
    if type(i)==str:
      l2.append(i)
```

module and packages:

aim of moule and packages is that where we will excess code from file 1 or file 2 which is store at different location mean(we interlink the two file toghther for access the code

Package- model modul- file which store code

```
# for interconnect to different file

import os,sys
from os.path import dirname,join,abspath
sys.path.insert(0,abspath(join(dirname(__file__),'..')))
```

```
# from floder_name import file_name
from payment import payment_details # where payment is the folder and payment_detail is the file name where we right code
def course():
  print("this is my course details")
payment_details.payment()
```

12-Feb-23

(Exception Handling with Try-Except)

in the programming if the one code is crash so on that time below of that code is not excuted so on the production this thing create big problem because if one code not excute below that all code will be not exceute and whole program crash so to resolve this problem what we do we use exception handling in the program

so if we know at one place we got error so on that place we use Exception handling with try

```
a/0
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-deacdc4ccabc> in <module>
----> 1 a/0

NameError: name 'a' is not defined

SEARCH STACK OVERFLOW
```

```
# example
f=open("text.txt","r")  # we know that this file is not in the our system so on that time we get error so in that scenario we are not able to
print('hi dude')
```

```
---------------------------------------------------------------------------
```

```
# we resolve above problem
try :
  f=open("test.txt","r")
except Exception as e:
  print("this is my except block ",e)   # look above we can resolve above problem by using this
a=4+5
a
```

```
this is my except block  [Errno 2] No such file or directory: 'test.txt'
9
```

else block-

like try block we have another else block which is only excuted when our try block succesfully Excuted

```
try:
  f=open("test.txt","r")
  f.write("write into my file")
except Exception as e:
  print("this is my Except block",e)
else:
  f.close()
  print("this will be excuted once your try will excuted without error")
```

```
this is my Except block [Errno 2] No such file or directory: 'test.txt'
```

```
try:
  f=open("test3.txt","r")
  f.write("write something ")
finally:
  print("finally will execute itself in any situation")
```

```
finally will execute itself in any situation
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-5-b244c677798a> in <module>
      1 try:
----> 2   f=open("test3.txt","r")
      3   f.write("write something ")
      4 finally:
      5   print("finally will execute itself in any situation")

FileNotFoundError: [Errno 2] No such file or directory: 'test3.txt'
```

SEARCH STACK OVERFLOW

above you can see that try excution will fail but finally excute sucesfully

Custom Exception Handling

```
age=int(input("enter your age"))
```

```
enter your age-78
```

custome message help to look if we pass negative no in the age object so on that time age is store succefully and there is no error message becuase acc to code logic it true

so resolve this proble we can use custome exception handling

```
class validation(Exception):
  def __init__(self,msg):
    self.msg=msg


def validation1(age):
  if age<0:
    raise validation("enter age negative")
  elif age>200:
    raise validation("enter age to much high")
```

```
  else:
    print("age is valid")

try:
  age=int(input("enter your age"))
  validation1(age)
except validation as e:
  print(e)
```

```
enter your age-90
enter age negative
```

List of general use Exception

```
-45
```

```
-45
```

Multiprocessing

```
# for the message sending

import multiprocessing
def sender(conn,msg):
  for i in msg:
    conn.send(i)
  conn.close()

def recieve(conn):
  while True:
    try:
      msg=conn.recv()
    except Exception as e:
      print(e)
      break
    print(msg)
if __name__=='__main__':
  msg =['my name is sudh','this is my msg to student','i am talking class for multiprocessing ']
  parent_con,child_con=multiprocessing.Pipe()
  m1=multiprocessing.Process(target=sender,args=(child_con,msg))
  m2=multiprocessing.Process(target=recieve,args=(parent_con,))
  m1.start()
  m2.start()
  m1.join()
  child_con.close()
  m2.join()
  parent_con.close()
```

```
my name is sudh
this is my msg to student
i am talking class for multiprocessing
```

working sql with python

```
! pip install Mysql
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting Mysql
  Downloading mysql-0.0.3-py3-none-any.whl (1.2 kB)
Collecting mysqlclient
  Downloading mysqlclient-2.1.1.tar.gz (88 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 88.1/88.1 KB 2.6 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: mysqlclient
  Building wheel for mysqlclient (setup.py) ... done
  Created wheel for mysqlclient: filename=mysqlclient-2.1.1-cp38-cp38-linux_x86_64.whl size=109198 sha256=7a7c1fae60b538a1d88d7ab6d877e7
  Stored in directory: /root/.cache/pip/wheels/5b/e1/84/a6185eaec318899f59a32d393af7729a0719cd93695d71f9a1
Successfully built mysqlclient
```

```
Installing collected packages: mysqlclient, Mysql
Successfully installed Mysql-0.0.3 mysqlclient-2.1.1
```

```
!pip install mysql.connector
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting mysql.connector
  Downloading mysql-connector-2.2.9.tar.gz (11.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.9/11.9 MB 71.7 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: mysql.connector
  Building wheel for mysql.connector (setup.py) ... done
  Created wheel for mysql.connector: filename=mysql_connector-2.2.9-cp38-cp38-linux_x86_64.whl size=247965 sha256=41e70c842d6d9aeac298dt
  Stored in directory: /root/.cache/pip/wheels/57/e4/98/5feafb5c393dd2540e44b064a6f95832990d543e5b4f53ea8f
Successfully built mysql.connector
Installing collected packages: mysql.connector
Successfully installed mysql.connector-2.2.9
```

```
!pi
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
ERROR: Could not find a version that satisfies the requirement mysql.connector.connect (from versions: none)
ERROR: No matching distribution found for mysql.connector.connect
```

```
import mysql.connector
mydb=mysql.connector.connect(
    host="localhost",
    user="abc",
    password="password"
)
print(mydb)
mycursor=mydb.cursor()
mycursor.execute("show Database")
for x in mycursor:
  print(x)
```

```
---------------------------------------------------------------------------
ConnectionRefusedError                    Traceback (most recent call last)
/usr/local/lib/python3.8/dist-packages/mysql/connector/network.py in open_connection(self)
    508             self.sock.settimeout(self._connection_timeout)
--> 509             self.sock.connect(sockaddr)
    510         except IOError as err:

ConnectionRefusedError: [Errno 111] Connection refused

During handling of the above exception, another exception occurred:

InterfaceError                            Traceback (most recent call last)
                                          5 frames
/usr/local/lib/python3.8/dist-packages/mysql/connector/network.py in open_connection(self)
    509             self.sock.connect(sockaddr)
    510         except IOError as err:
--> 511             raise errors.InterfaceError(
    512                 errno=2003, values=(self.get_address(), _strioerror(err)))
    513         except Exception as err:

InterfaceError: 2003: Can't connect to MySQL server on 'localhost:3306' (111 Connection refused)

SEARCH STACK OVERFLOW
```

```
# for the connect the database
import mysql.connector
mydb=mysql.connector.connect(
    host="localhost",
    user="abc",
    password="password"
)

mycursor=mydb.cursor()
mycursor.execute("CREATE DATABASE test1")
mydb.close()
```

```
---------------------------------------------------------------------------
ConnectionRefusedError                    Traceback (most recent call last)
/usr/local/lib/python3.8/dist-packages/mysql/connector/network.py in open_connection(self)
    508             self.sock.settimeout(self._connection_timeout)
--> 509             self.sock.connect(sockaddr)
    510         except IOError as err:

ConnectionRefusedError: [Errno 111] Connection refused

During handling of the above exception, another exception occurred:

InterfaceError                            Traceback (most recent call last)
                                          5 frames
/usr/local/lib/python3.8/dist-packages/mysql/connector/network.py in open_connection(self)
    509             self.sock.connect(sockaddr)
    510         except IOError as err:
--> 511             raise errors.InterfaceError(
    512                 errno=2003, values=(self.get_address(), _strioerror(err)))
    513         except Exception as err:

InterfaceError: 2003: Can't connect to MySQL server on 'localhost:3306' (111 Connection refused)
```

```
# create table

import mysql.connector
mydb=mysql.connector.connect(
    host="localhost",
    user="abc",
    password="password"
)

mycursor=mydb.cursor()
mycursor.execute("CREATE DATABASE test2.test_table(c1,INT,c2 VARCHAR(50")
mydb.close()


# for the insert the data in the table
import mysql.connector
mydb=mysql.connector.connect(
    host="localhost",
    user="abc",
    password="password"
)

mycursor=mydb.cursor()
mycursor.execute("insert into test2.test_table values(122,'Ash')")
mydb.commit()
mydb.close()


# for the get the data from the table
import mysql.connector
mydb=mysql.connector.connect(
    host="localhost",
    user="abc",
    password="password"
)

mycursor=mydb.cursor()
mycursor.execute("select * from test2.test_table")
for i in mycursor.fetchall()
mydb.close()
```


0s completed at 12:37 AM