Module

--------

Module in python used to group set of function which can be use as a unit. Simple .py file also work as a module. That can be use by using import statement.

There diff. mechan. to import....

Example---

MyModule.py which contains function

-----------------------------------

#mymodule.py

def sum(a,b):

c=a+b

print('Sum..',c)

def hello():

print('hello')

print('function')

def square(x):

y=x\*x

return y

--------------

Using of module

--------------------------

1. import modulename

#useone.py

example

import mymodule

mymodule.sum(3,2)

mymodule.hello()

print(mymodule.square(40))

2. from modulename import function/s name

#usetwo.py

from mymodule import sum,square

sum(8,2)

print(square(8))

3. from modulename import \*

#usethree.py

sum(65,3)

hello()

print(square(9))

4. alias to the module by using 'as' keyword

this alias will be use to call functions.

#usefour.py

import mymodule as mp

mp.sum(8,3)

print(mp.square(7))

mp.hello()

-------------------------------

time module-- for time related information

import time

t=time.time()

#no of ticks

print(t)

#complete date/time info

d=time.localtime(t)

print(d)

#format date/time

r=time.asctime(d)

print(r)

--------------------

Calendar module for print calendar for a month

import calendar

c=calendar.month(2021, 8)

print(c)

------

module datetime - which also use for date/time information

---------------------

import datetime

d=datetime.datetime.now()

print(d)

----------------

import datetime

d=datetime.datetime.now()

print(d)

print(d.year)

print(d.month)

print(d.date())

print(d.hour)

print(d.minute)

print(d.second)

--------------------

Object Oriented programming

------------------------------

Class and object

---------------------

Object--- Any real world entity is an object. Each

---------

have set of attributes and behaviour.

Example--

CRETA - object

color,model,eng. cap. (att)

speed() gears() brake()

Class... Logical entity which is group of similar objects..

SUV-- CRETA,CRV.....

Class-- Objects

-----------------------

Features-- Abstraction,Encapsulation,Inheritance,Polymorphism

--------------------------------------------------

Classes in python

------------------

we use class keyword

example

class First:

rollno=0

name=''

To use class we create object

------------------------------

objectname=classname()

then use . operator to call data/functions

class Employee:

empno=0

name=''

obj=Employee()

obj.empno=101

obj.name='Rajeev'

print(obj.empno)

print(obj.name)

---------------------------

functions in class--

--------------------

each function in python class must have 'self' keyword.. self keyword is used to refer current instance members..

class Employee:

empno=0

name=''

def get(self):

self.empno=101

self.name='Rajeev'

def show(self):

print(self.empno)

print(self.name)

obj=Employee()

obj.get()

obj.show()

--------------

class Employee:

empno=0

salary=0

grade=''

def get(self,a,b):

self.empno=a

self.salary=b

def check(self):

if self.salary>=30000:

self.grade='A'

else:

self.grade='B'

def show(self):

self.check()

print(self.empno)

print(self.grade)

obj=Employee()

obj.get(103,31000)

obj.show()

----------------

Python have some inbuilt function of oop..

these are..

hasattr(object,attribute) - true if attr. found

getattr(object,att) to get att. value

setattr(object,att,newvalue)- to change value of attr.

delattr(object,att.) to delete attribute...

class Employee:

empno=101

salary=21000

grade='A'

obj=Employee()

print(hasattr(obj, 'empno')) #true

print(getattr(obj, 'salary'))

setattr(obj, 'salary', 45000)

print(getattr(obj, 'salary'))

delattr(obj,'empno')

print(hasattr(obj, 'empno')) #error

--------------

Constructor and destructor in class

-----------------------------------

Constructor is spl function which get call automatic when we create object of class. It is used to initialize the instance values. In python the name of constructor must be \_\_init\_\_.

(2 underscores).

class Employee:

empno=0

salary=0

grade=''

def \_\_init\_\_(self):

self.empno=101

self.salary=34000

self.grade='A'

print('Const. called')

def show(self):

print(self.empno)

print(self.salary)

print(self.grade)

obj=Employee() #\_\_init\_\_ calls

obj.show()

-----------------

class Employee:

empno=0

salary=0

grade=''

#parametriezed

def \_\_init\_\_(self,a,b,c):

self.empno=a

self.salary=b

self.grade=c

print('Const. called')

def show(self):

print(self.empno)

print(self.salary)

print(self.grade)

obj=Employee(101,43000,'B')

#\_\_init\_\_ calls

obj.show()

---------------

destructor----deallocate memory/free memory. Calls at the end. The name must be \_\_del\_\_.

class Employee:

empno=0

salary=0

grade=''

#parametriezed

def \_\_init\_\_(self,a,b,c):

self.empno=a

self.salary=b

self.grade=c

print('Const. called')

def show(self):

print(self.empno)

print(self.salary)

print(self.grade)

def \_\_del\_\_(self):

print('Bye...')

obj=Employee(101,43000,'B')

obj.show()

obj1=Employee(102,41000,'A')

obj1.show()

--------------

Inheritance---Reusing the data/function of a class in other class. Main class is parent class and new class called child class.

Types are

------------

Single

MultiLevel

Multiple

to link a class we use

child -class classname(parentclass name):

Single

class A:

def hello(self):

print('hello')

print('parent')

#B is child of A

class B(A):

def sum(self,a,b):

print('sum.',(a+b))

obj=B()

obj.sum(32,54)

obj.hello()

------------------

Multi-Level

class A:

def hello(self):

print('hello')

print('parent')

#B is child of A

class B(A):

def sum(self,a,b):

print('sum.',(a+b))

#C is child of B

class C(B):

def Hi(self):

print('HI C..')

obj=C()

obj.hello()

obj.Hi()

obj.sum(12,3)

---------------------------

Multiple--

class A:

def hello(self):

print('hello')

print('parent')

class B:

def sum(self,a,b):

print('sum.',(a+b))

#C is child of A and B

class C(A,B):

def Hi(self):

print('HI C..')

obj=C()

obj.hello()

obj.Hi()

obj.sum(12,3)

-----------------

We can have \_\_str\_\_ function that help to return string representation for the object.

class A:

def hello(self):

print('hello')

print('parent')

def \_\_str\_\_(self):

return "My A Class.."

obj=A()

print(obj)

------------

To make any member hidden...

to do this prefix \_\_ (two underscore)

class A:

#hidden

\_\_name='Rajeev'

obj=A()

print(obj.\_\_name) #error

-----------------

Command line values--

It is possible to run program from command prompt and we can pass the value as input from command line. These are called command line values.

The values are given by space separation. We need to sys module to use these values. The argv list used to get the values first index 0 is file name itself, All values are in string format. So we need to convert to int/float for numeric operations.

import sys

print('Hi Dear..')

print(sys.argv[1])

print('File..',sys.argv[0])

to run form command prompt

python First.py AKASH

------------------------

Q.

python First.py 12 + 4

import sys

a=int(sys.argv[1])

b=int(sys.argv[3])

if sys.argv[2]=='+':

print(a+b)

if sys.argv[2]=='-':

print(a-b)

if sys.argv[2]=='\*':

print(a\*b)

if sys.argv[2]=='/':

print(a/b)

--------------------------

Exception Handling in python

----------------------------

Exception-- Exception are run time situation that terminate the program abnormally.When exception the control goes to interpretator which handle it.

Exception handling is a mechanism where programmer writes a block of statement which helps to control all run time stiuations.

A proper exception handling helps to achieve robustness of code---

Python have strong exception handling mechanism...

Where it contains various classes to handle exception.

Uncaught exception----Exception which are not be handled...

----------------------

a=int(input('Enter no..'))

b=int(input('Enter no..'))

c=a/b

print('ans..',c)

-----------------

Python have many inbuit classes for exception handling..

Main classes are..

Exception-- Base class for all exception

ZeroDivisionError- when we do divide by 0

OverFlowError

NameError

ValueError

ArithmeticError base class for all number exception..

ImportError-- When import statement fails..

KeyError- when key is present in dictionary

IOError

TypeError

IndexError--

Exception Handling in Python Main keywords or block

----------------

try -- the block of statements which are supposed to raise an exception will be the part of try block. If we use try block then your code understand the you are going to handle by using except .. we can also have finally...

except-- the block of statement which are responsible to handle exception. The except block use particular type of exception class or even base class Exception to handle this.

Note - One try can have multiple except as well.

try:

a=int(input('Enter no..'))

b=int(input('Enter no..'))

c=a/b

print('ans..',c)

except ZeroDivisionError:

print('Check 2nd no..')

---------------

Note One try can have multiple except

Q. replicate the multiple exception scenario in 1 code. example command line values ..

exception..

IndexError ValueError ZeroDivisionError

python Calc.py 12 3 (fine)

python Calc.py (index error)

python Calc.py a b (Value)

python Calc.py 12 0 (ZeroDivisionError)

-----------------------

import sys

try:

a=int(sys.argv[1])

b=int(sys.argv[2])

c=a/b

print(c)

except IndexError:

print('Check Value..')

except ValueError:

print('Non numeric.')

except ZeroDivisionError:

print('Check Second number')

--------------------

except- Use Exception class for all kinds of exception.

keyword 'as' it can be use with exception class to get complete infor for exception object.

try:

a=int(input('Enter no..'))

b=int(input('Enter no..'))

c=a/b

print('ans..',c)

except Exception as ex:

print('Any issue.',ex)

--------------------

Note- We can except with exception class also (not recommended)--

try:

a=int(input('Enter no..'))

b=int(input('Enter no..'))

c=a/b

print('ans..',c)

except:

print('Any issue.')

----------------

finally-- must do operation whether exception comes or not. some example are clean up, close data connection,close network connection...

try:

a=int(input('Enter no..'))

b=int(input('Enter no..'))

c=a/b

print('ans..',c)

except:

print('Any issue.')

finally:

print('The Ends..')

-------------------------

raise keyword-- this is used to explicitly(forceful) raise and exception , it must be used with in try and must be handle with except block.

try:

a=int(input('Enter no..'))

if a<10:

raise Exception()

else:

print('Ok')

except:

print('Any issue.')

finally:

print('The Ends..')

-------------

Custom Exception Class (User Defined Exception Class)

To do so we create a class that is child of Exception class. Inside this we can create a function. That can be used in except block (for message or any other info.)

class MyException(Exception):

def showmessage(self):

print('My Issue')

try:

a=int(input('Number..'))

if a<10:

raise MyException()

else:

print('OK')

except MyException as ex:

ex.showmessage()

-------------------------

assert-- it use in testing (unit testing) assert got fails when a condition false. It raise assertionError..

a=int(input('number.'))

assert a<10

We can give customized message in case of failure..

a=int(input('number.'))

assert a<10,"Check this"

-----------------------

File Handling in python

-----------------------------

File is collection of data on disk.

Steps in file handling are...

1. File is open by open function.. We get the object of the file...

syntax open(filename,mode)

Mode-- r -read

w - write

a - append

example

obj=open('AA.txt','w')

obj can be used for writing the data in file.

2. After opening the file

We perform

Read or Write operation

write(value)--- string data

read(length) -- reading the data

3. close the file by close function.

------------------------------------

empno='E0001'

name='Amit'

city='Delhi'

obj=open('Data.txt','w')

obj.write(empno+'\n')

obj.write(name+'\n')

obj.write(city+'\n')

print('Data Saved..')

obj.close()

-----------------------

empno=int(input('Enter empno.'))

name=input('Enter Name.')

city=input('Enter City.')

fn=input('Enter FileName.')

obj=open(fn,'w')

obj.write(str(empno)+'\n')

obj.write(name+'\n')

obj.write(city+'\n')

print('Data Saved..')

obj.close()

---------------

empno=int(input('Enter empno.'))

name=input('Enter Name.')

city=input('Enter City.')

fn=input('Enter FileName.')

#append data

obj=open(fn,'a')

obj.write(str(empno)+'\n')

obj.write(name+'\n')

obj.write(city+'\n')

print('Data Saved..')

obj.close()

------------------

Reading of the data

----------------------

x=input('enter file to read')

f=open(x,'r')

#no. of letters to read

st=f.read(2000)

print(st)

f.close()

-----------------

Q.. Make a file handling utility which read the data from a file and transfer to new file.

(copy / paste utility)..This will run on 1st of every month.

Q.. We have a file on disk create a utility that counts no. of words,no. of digits,no. of lower letters,no. of upper letters.

x=input('Enter file name')

f=open(x,'r')

st=f.read(5000)

c=0

d=0

e=0

fd=0

for r in st:

if r.isdigit():

c=c+1

if r.isupper():

d=d+1

if r.islower():

e=e+1

if r.isspace():

fd=fd+1

print('Digits..',c)

print('Upper..',d)

print('lower..',e)

print('Space..',fd)

f.close()

----------------

Positioning of the file pointer

-------------------------------

two main functions

---------------------

tell() return the pointer position

seek(offset,from) from 0 begin,1 current,2 end

offset -- location

f=open('EmpData.txt','r')

st=f.read(150)

print(st)

pos=f.tell()

print('Position..',pos)

f.close()

----------

Main attribute of File objects are

closed - return true if file is closed

mode- return mode of file

name - name of file.

f=open('EmpData.txt','r')

print(f.closed)

print(f.mode)

print(f.name)

f.close()

--------------

os module-- is spl module , helps to work with file and folder system of current environment.

import os

#current working directory

print(os.getcwd())

--------------

import os

os.mkdir('newpython')

# d:\\....\\

print('folder created')

----------------------

import os

os.rmdir('newpython')

# d:\\....\\

print('folder removed')

-----------

Rename the file...

import os

os.rename(oldfilename,newfilename)

print('Name changed..')

--------

import os

os.remove('NData.txt')

print('File Removed..')

-----------------

To check existance of a file or folder...

import os

#True

print(os.path.exists('EmpData.txt'))

------------

import os

#nt - windows

print(os.name)

---------------

Database connectivity with python

---------------------------------

MySQL-- as backend database...

database fis;

create table employee (empno int primary key,ename varchar(20),ecity varchar(20),salary int);

-----------

pymysql is one of the main module, used to connect python with database..

install this...

open anaconda prompt

pip install pymysql (pip utility- to install external library)

------------------------------

to verify this.. create a new program

import pymysql

(if no error is coming-- success)..

Steps in connectivity of data base

-------------------------------------

1. import pymysql

2. get connection by using

pymysql.connect(host='localhost',

user='root',

password='...',

db='Databasename')

3. Cursor- object which use to perform data operation for this we use syntax..

dataconnection.cursor() it return cursor object..

4. You can use string variable for sql query

5. execute the statement by using cursor object

cursorobject.execute(sql query)

6. data is returned by fetchone(1 record or fecthall(>1 record) function..

7. it return a list where element is on index position..

import pymysql

#step 1

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

sql="select version()"

#step3

cur.execute(sql)

data=cur.fetchone()

print(data)

d.close()

---------------

import pymysql

#step 1

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

sql="select \* from employee"

#step3

cur.execute(sql)

data=cur.fetchall()

for res in data:

print('Empno..',res[0])

print('Name..',res[1])

print('City..',res[2])

print('Salary..',res[3])

d.close()

------------

import pymysql

#step 1

try:

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

sql="insert into employee values(109,'Ravi','Pune',31000)"

#step3

cur.execute(sql)

print('Record saved.')

d.commit()

except Exception as ex:

print(ex)

d.close()

--------------------------

import pymysql

#step 1

try:

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

sql="update employee set salary=70000 where empno=102"

#step3

cur.execute(sql)

print('Record Updated.')

d.commit()

except Exception as ex:

print(ex)

d.close()

----------------

import pymysql

#step 1

try:

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

sql="create table test(empno int,month varchar(20))"

#step3

cur.execute(sql)

print('Table Created..')

d.commit()

except Exception as ex:

print(ex)

d.close()

---------------

import pymysql

#step 1

try:

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

sql="drop table test"

#step3

cur.execute(sql)

print('Table Deleted..')

d.commit()

except Exception as ex:

print(ex)

d.close()

---------------------

Dynamic value pass in sql queries....

We take the help of %s.. String %d .. int..

to pass values...

import pymysql

#step 1

try:

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

a=int(input('Enter Empno.'))

b=input('Enter Name..')

c=input('Enter City..')

dd=int(input('Enter Salary.'))

sql="insert into employee values(%d,'%s','%s',%d)"%(a,b,c,dd)

#step3

cur.execute(sql)

print('Record saved.')

d.commit()

except Exception as ex:

print(ex)

d.close()

--------------

Q.. Input Empno we will check in table employee it is present we will show message other wise it will insert..

import pymysql

#step 1

try:

d=pymysql.connect(host='localhost',

user='root',password='root',

db='fis')

#step 2

cur=d.cursor()

a=int(input('Enter Empno.'))

sql="delete from employee where empno=%d"%(a)

#step3

cur.execute(sql)

print('Record deleted.')

d.commit()

except Exception as ex:

print(ex)

d.close()

---------------------------------