INDEX

Sr. No.	Name of the Practical	Date	Grade	Co Mapping	Prof. In-charge Sign
1	WAP to demonstrate Identifiers, Reserve words & constants in Python.				
2	WAP to Interpret Data Types in Python.				
3	WAP to Evaluate Different Arithmetic Operations Using				
4	WAP to Find Factorial of Given Number using Control Flow				
5	WAP to Design Function to Find Max and Min of two, three				
6	WAP to Find Power of Given Numbers by Lambda Expression.				
7	WAP to Create Class and Objectives for various Arithmetic				
8	WAP to perform CRUD operations with MySQL Database.				
9	WAP to Understand Python List Comprehension with suitable				
10	WAP to Demonstrate Inheritance concept with suitable example.				
11	WAP to Analyze various Exceptions.				
12	WAP to Design Numeric value to Word Converter.				

Practical No.: 1

Name of the Practical: - Write a program to demonstrate Identifiers, Reserve words & constants in Python.

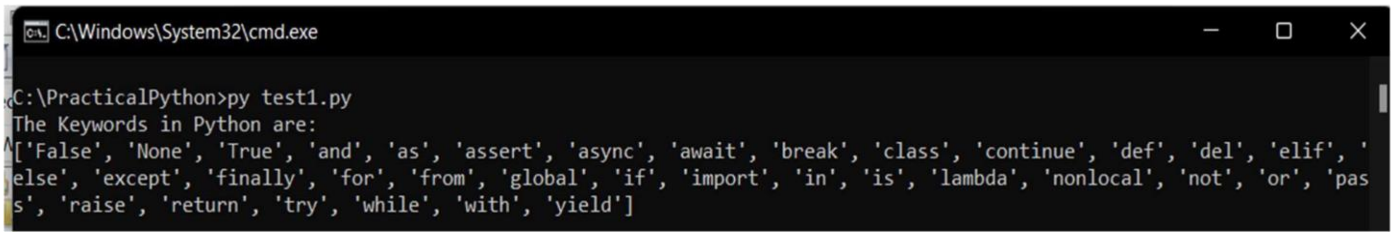
Software Required: - Python 3.10.4, Editplus 5.5

Program:

A) Importing Keywords: -

```
import keyword
print("The Keywords in Python are: ")
print(keyword.kwlist)
```

Output: -

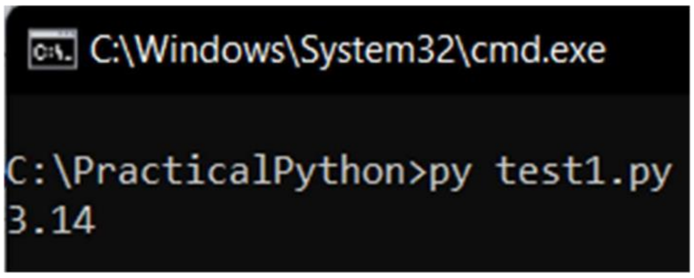


```
C:\PracticalPython>py test1.py
The Keywords in Python are:
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', '
else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pas
s', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

B) Constant: -

```
PI = 3.14
print(PI)
```

Output: -

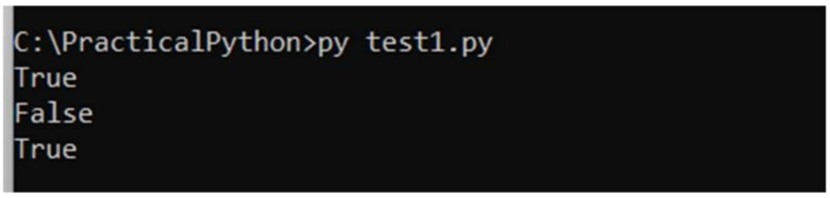


```
C:\PracticalPython>py test1.py
3.14
```

C) Identifier: -

```
print("aBc".isidentifier())
print("@s".isidentifier())
print("ff34_".isidentifier())
```

Output :-



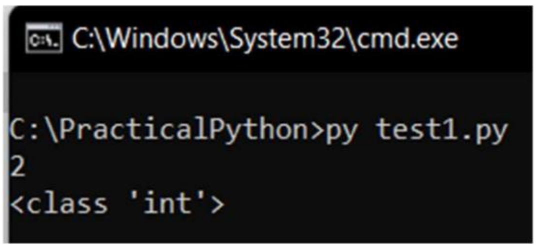
```
C:\PracticalPython>py test1.py
True
False
True
```

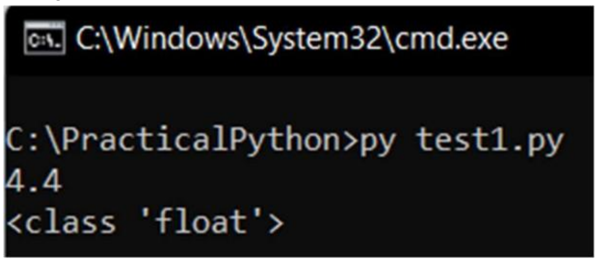
Practical No.: 2

Name of Practical: - Write a program to Interpret Data Types in Python.

Software Required: - Python 3.10.4, Editplus 5.5

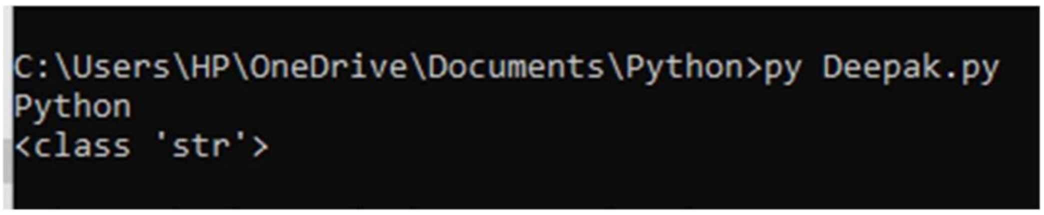
Programs: -

1) **Integer (int):-**
   Input: -
   ```
   x = 2
   print(x)
   print(type(x))
   ```
   Output: -
   
   ```
   C:\Windows\System32\cmd.exe

   C:\PracticalPython>py test1.py
   2
   <class 'int'>
   ```

2) **Float (float): -**
   Input: -
   ```
   x = 4.4
   print(x)
   print(type(x))
   ```
   Output: -
   
   ```
   C:\Windows\System32\cmd.exe

   C:\PracticalPython>py test1.py
   4.4
   <class 'float'>
   ```

3) **String (str): -**
   Input: -
   ```
   x = 'Python'
   print(x)
   print(type(x))
   ```

Output: -



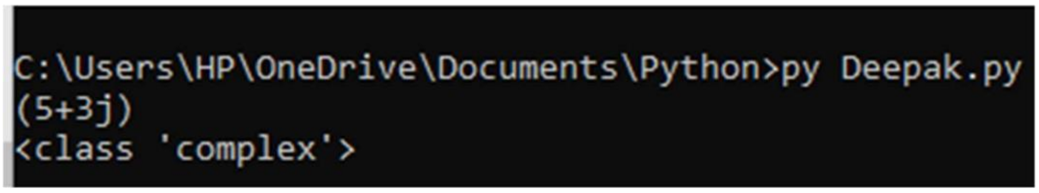
```
C:\Users\HP\OneDrive\Documents\Python>py Deepak.py
Python
<class 'str'>
```

4) **Complex (complex): -**

Input: -

```
x = 5+3j
print(x)
print(type(x))
```

Output: -



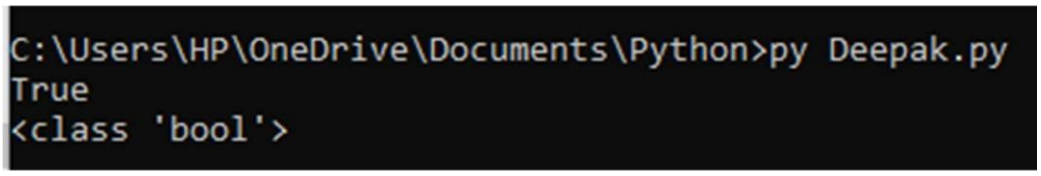
```
C:\Users\HP\OneDrive\Documents\Python>py Deepak.py
(5+3j)
<class 'complex'>
```

5) **Boolean (bool): -**

Input: -

```
x = True
print(x)
print(type(x))
```

Output: -



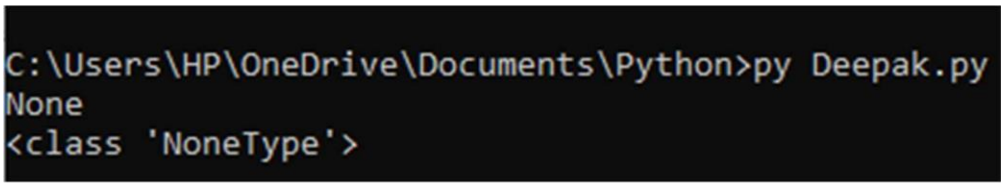
```
C:\Users\HP\OneDrive\Documents\Python>py Deepak.py
True
<class 'bool'>
```

6) **None (NoneType): -**

Input: -

```
x = None
print(x)
print(type(x))
```

Output: -



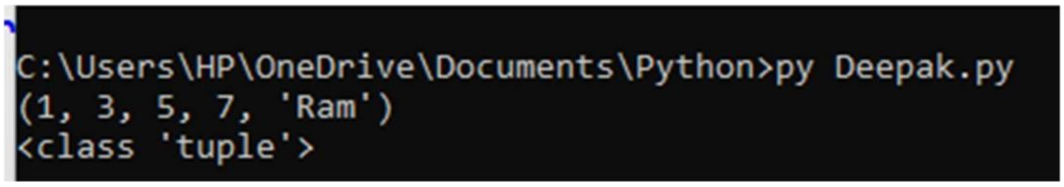
```
C:\Users\HP\OneDrive\Documents\Python>py Deepak.py
None
<class 'NoneType'>
```

7) **Tuple (tuple): -**

Input: -

```
T = (1, 3, 5, 7, ’Ram’)
 print(T)
 print(type(T))
```

Output: -



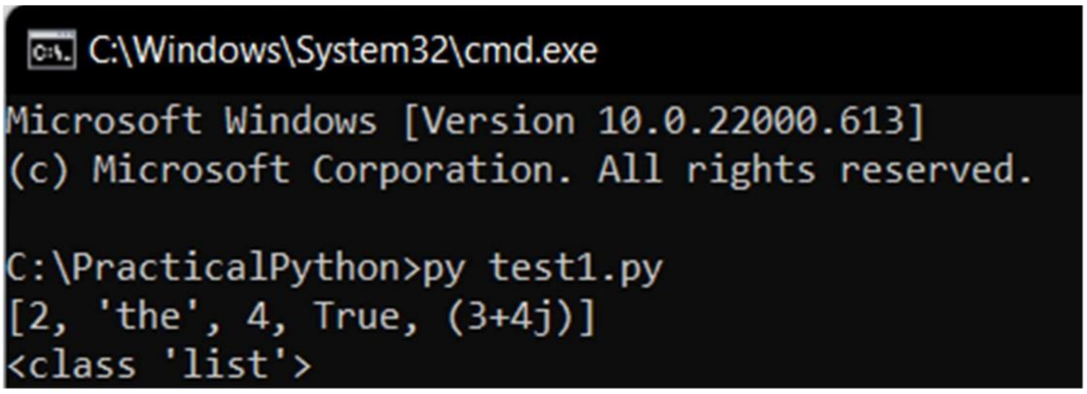
```
C:\Users\HP\OneDrive\Documents\Python>py Deepak.py
(1, 3, 5, 7, 'Ram')
<class 'tuple'>
```

8) **List (list): -**

Input: -

```
L = [2,’the’,4,True,3+4j]
print(L)
print(type(L))
```

Output:-



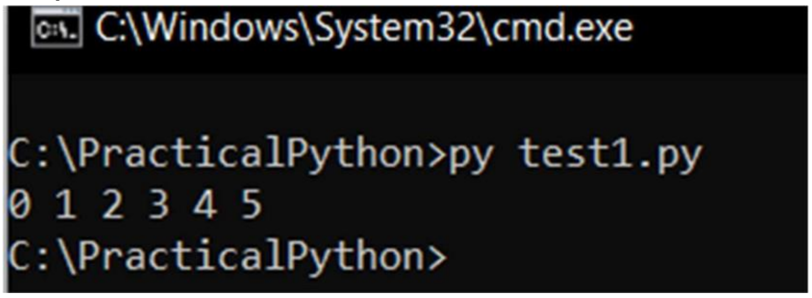
```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22000.613]
(c) Microsoft Corporation. All rights reserved.

C:\PracticalPython>py test1.py
[2, 'the', 4, True, (3+4j)]
<class 'list'>
```

9) **Range (range): -**

Input: -

```
x = range(6)
for n in x:
     print(n,end=” ”)
```

Output: -



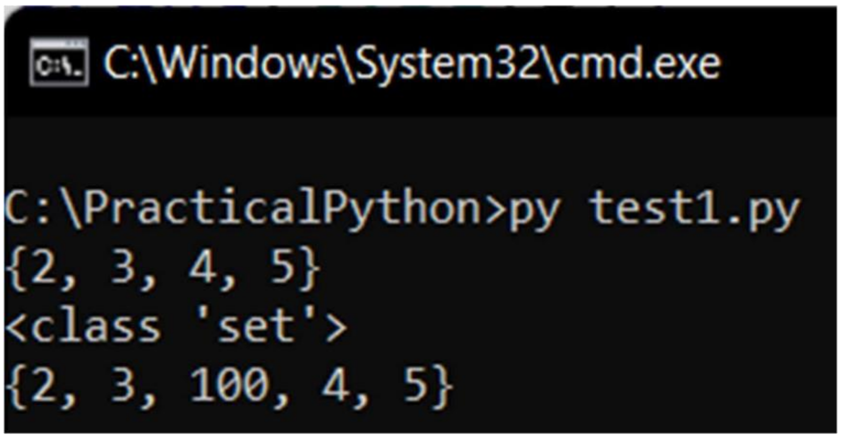
```
C:\Windows\System32\cmd.exe

C:\PracticalPython>py test1.py
0 1 2 3 4 5
C:\PracticalPython>
```

10) **Set(set): -**

Input: -

```
s = {2,3,4,5,5,5,4,3,2}
print(s)
print(type(s))
s.add(100)
print(s)
```

Output: -



C:\Windows\System32\cmd.exe

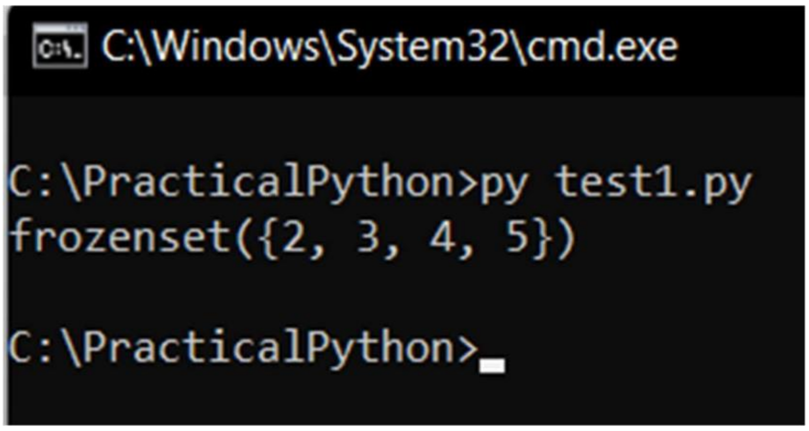
```
C:\PracticalPython>py test1.py
{2, 3, 4, 5}
<class 'set'>
{2, 3, 100, 4, 5}
```

11) **Frozen Set (frozenset): -**

Input:-

```
s = {2,3,4,5}
fs = frozenset(s)
print(fs)
```

Output: -



C:\Windows\System32\cmd.exe

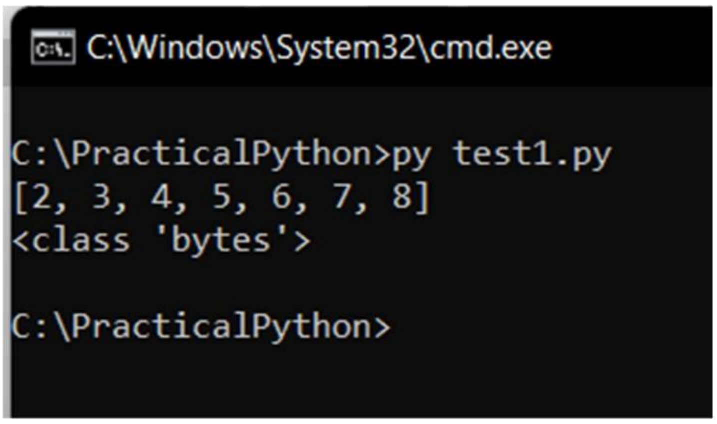
```
C:\PracticalPython>py test1.py
frozenset({2, 3, 4, 5})

C:\PracticalPython>_
```

12) **Bytes:-**

Input : -

```
l=[2,3,4,5,6,7,8]
b=bytes(l)
print(l)
print(type(b))
```

Output:-



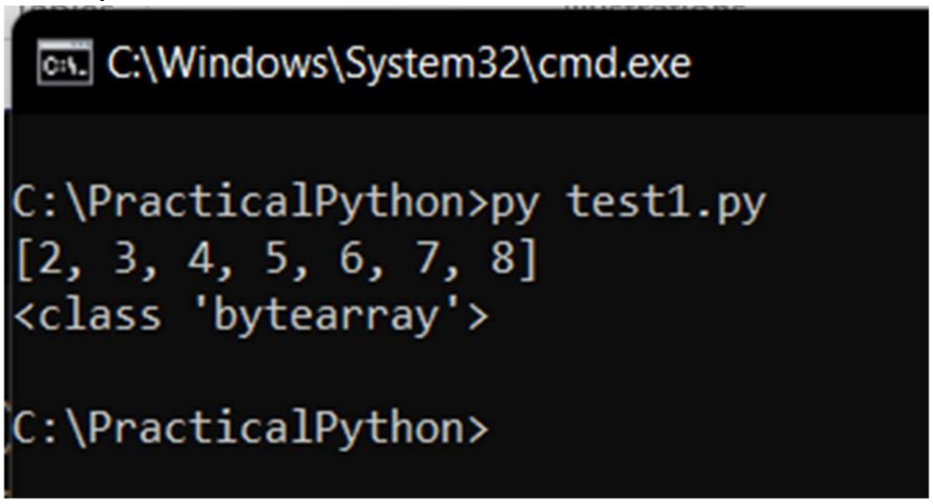
```
C:\Windows\System32\cmd.exe

C:\PracticalPython>py test1.py
[2, 3, 4, 5, 6, 7, 8]
<class 'bytes'>

C:\PracticalPython>
```

13)Bytesarray:-

Input :-

```
l=[2,3,4,5,6,7,8]
ba=bytearray(l)
print(l)
print(type(ba))
```

Output:-



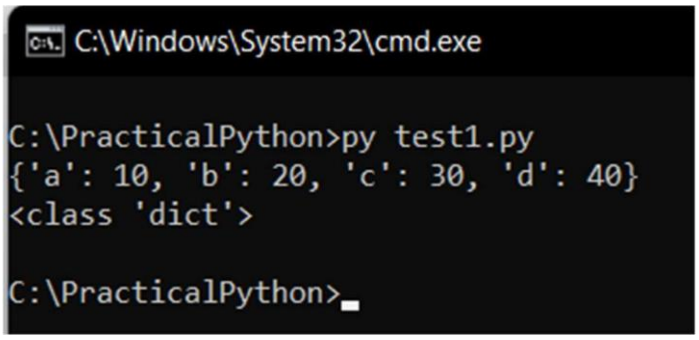
```
C:\Windows\System32\cmd.exe

C:\PracticalPython>py test1.py
[2, 3, 4, 5, 6, 7, 8]
<class 'bytearray'>

C:\PracticalPython>
```

14)Dictionary:-

Input:-

```
d={'a':10,'b':20,'c':30,'d':40}
print(d)
print(type(d))
```

Output:-



```
C:\Windows\System32\cmd.exe

C:\PracticalPython>py test1.py
{'a': 10, 'b': 20, 'c': 30, 'd': 40}
<class 'dict'>

C:\PracticalPython>
```

Practical No.:- 3

Name of the Practical:-WAP to Evaluate Different Arithmetic Operations Using Operators

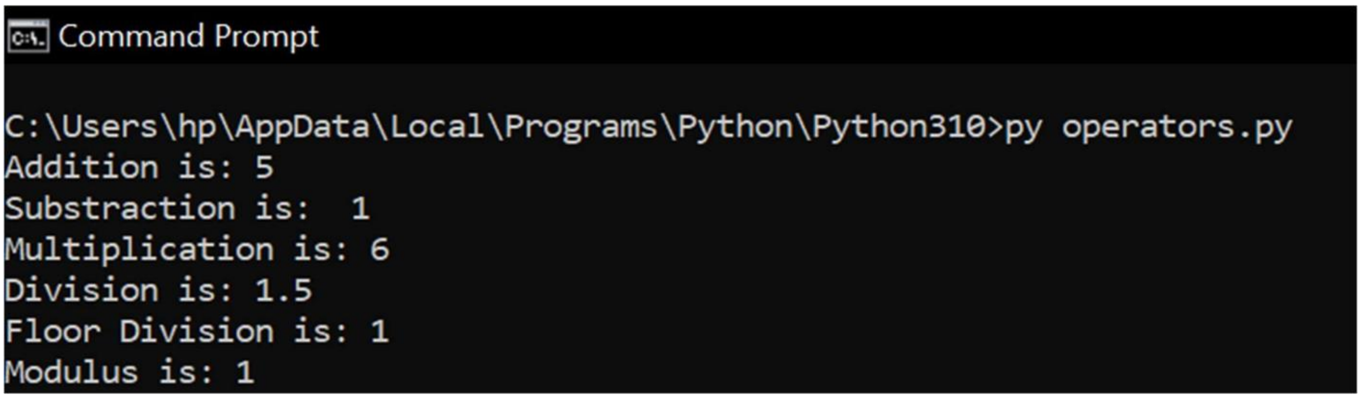
Software Required:- Python 3.10, Edit Plus

Syntax :-
```
val1+val2
val1-val2
val1*val2
val1/val2
val1//val2
val1%val2
val1**val2
```

Program :-
```
val1=3
val2=2
#using the addition operator
Add=val1+val2
print('Addition is:',Add)
#using the substraction operator
Sub=val1-val2
print('Substraction is: ',Sub)
#using the multiplication operator
Mul=val1*val2
print('Multiplication is:',Mul)
#using the division operator
Div=val1/val2
print('Division is:',Div)
#using the floor division operator
Fdiv=val1//val2
print('Floor Division is:',Fdiv)
#using the modulus operator
Mod=val1%val2
print('Modulus is:',Mod)
#using the exponentiation operator
```

```
Exp=val1**val2

print('Exponent is:',Exp)
```

Output:-



Command Prompt

```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py operators.py
Addition is: 5
Substraction is:  1
Multiplication is: 6
Division is: 1.5
Floor Division is: 1
Modulus is: 1
```

Practical No.:- 4

Name of the Practical:-WAP to Find Factorial of Given Number using Control Flow Statement

Software Required:-Python 3.10, Edit Plus

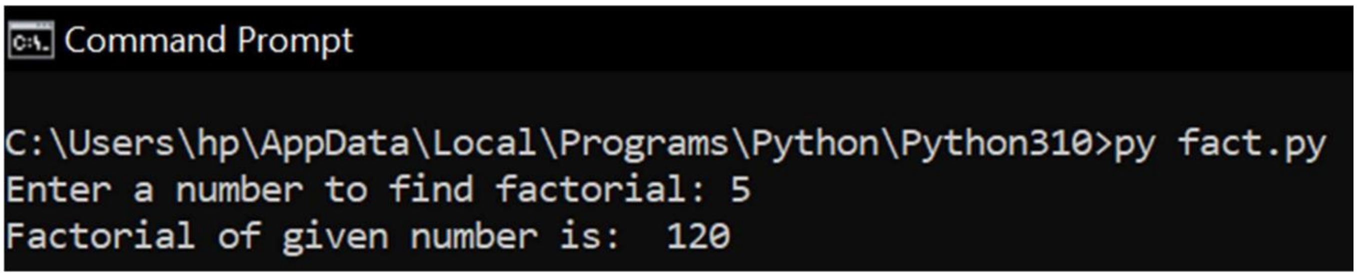
Syntax :-

```
for iterator_var in sequence:
    statements
        OR
while expression:
    statements
```

Program:-

```
n=int(input('Enter a number to find factorial: '))
factorial=1
while n>0:
        factorial=factorial*num
        n=n-1
print('Factorial of given number is: ',factorial)
                              OR
n=int(input('Enter a number to find factorial: '))
factorial=1
for i in range(1,num+1):
        factorial=factorial*num
        num=num-1
print('Factorial of given number is: ',factorial)
```

Output:



```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py fact.py
Enter a number to find factorial: 5
Factorial of given number is:  120
```

Practical No.:- 5

Name of the Practical:-WAP to Design Function to Find Max and Min of two, three numbers

Software Required:-Python 3.10, Edit Plus

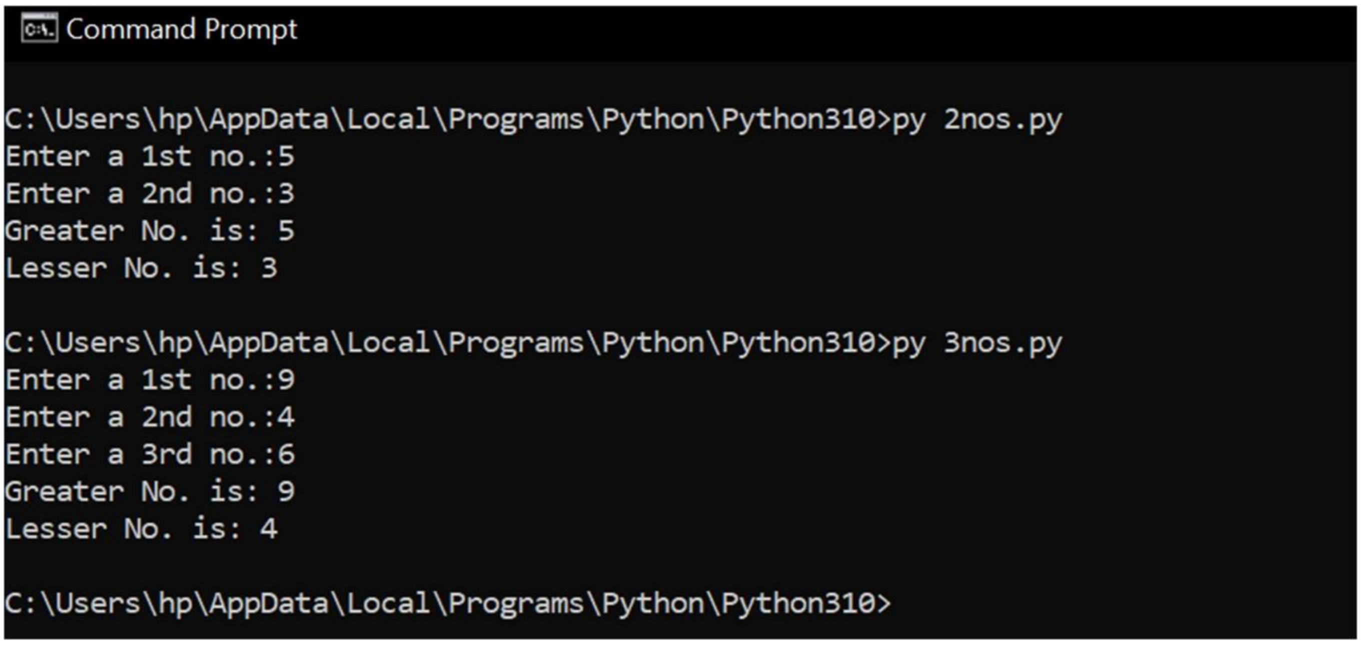
Syntax :-def function_name(parameters):

```
        Statements
```

Program:- #for two numbers

```
def max(a,b):
   m=a   if   a>b   else   b
   print('Greater No. is:',m)
def min(a,b):
   m=a   if   a<b   else   b
   print('Lesser No. is:',m)
n1=int(input('Enter a 1st no.:'))
n2=int(input('Enter a 2nd no.:'))
max(n1,n2)
min(n1,n2)
 #for three numbers
def max(a,b,c):
   m=a if a>b and a>c else b if b>c else c
   print('Greater No. is:',m)
def min(a,b,c):
   m=a if a<b and a<c else b if b<c else c
   print('Lesser No. is:',m)
n1=int(input('Enter a 1st no.:'))
n2=int(input('Enter a 2nd no.:'))
n3=int(input('Enter a 3rd no.:'))
max(n1,n2,n3)
min(n1,n2,n3)
```

Output:-



Command Prompt

```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py 2nos.py
Enter a 1st no.:5
Enter a 2nd no.:3
Greater No. is: 5
Lesser No. is: 3

C:\Users\hp\AppData\Local\Programs\Python\Python310>py 3nos.py
Enter a 1st no.:9
Enter a 2nd no.:4
Enter a 3rd no.:6
Greater No. is: 9
Lesser No. is: 4

C:\Users\hp\AppData\Local\Programs\Python\Python310>
```

Practical No.:- 6

Name of the Practical:-WAP to Find Power of Given Numbers by Lambda Expression.

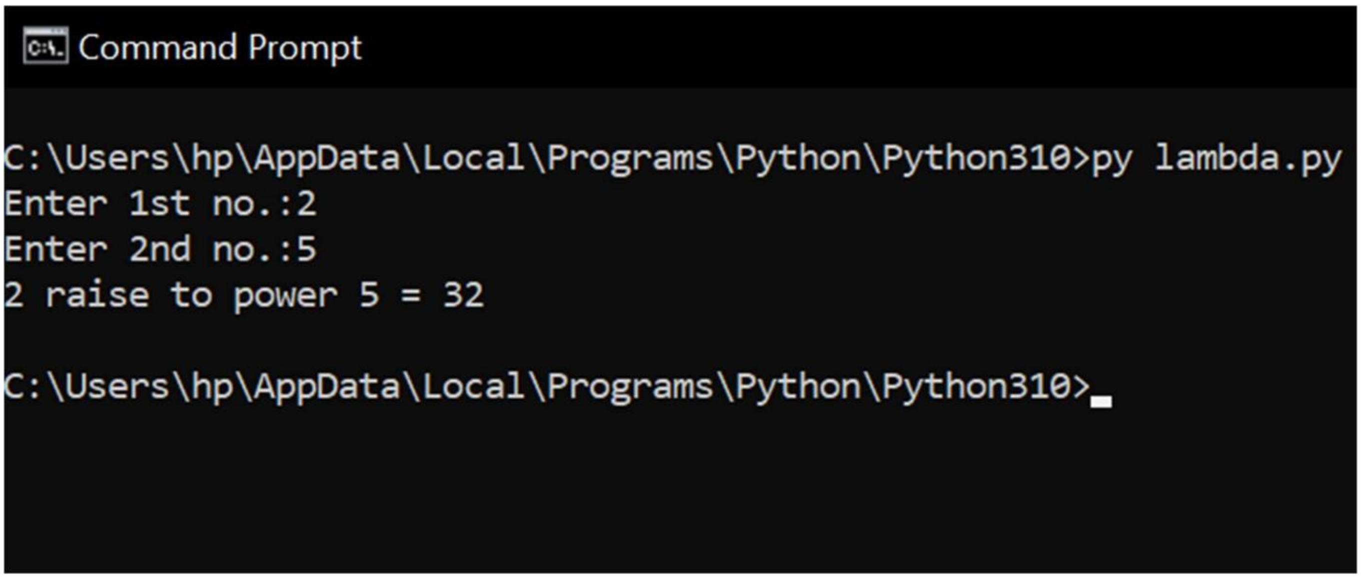
Software Required:-Python 3.10, Edit Plus

Syntax :-lambda arguments:expression

Program:-

```
p=lambda x,y: x**y
x=int(input('Enter 1st no.:'))
y=int(input('Enter 2nd no.:'))
print(x,'raise to power',y,'=',p(x,y))
```

Output:-



```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py lambda.py
Enter 1st no.:2
Enter 2nd no.:5
2 raise to power 5 = 32

C:\Users\hp\AppData\Local\Programs\Python\Python310>
```

Practical No.:- 7

Name of the Practical:-WAP to Create Class and Objectives for various Arithmetic Operations.

Software Required:-Python 3.10, Edit Plus

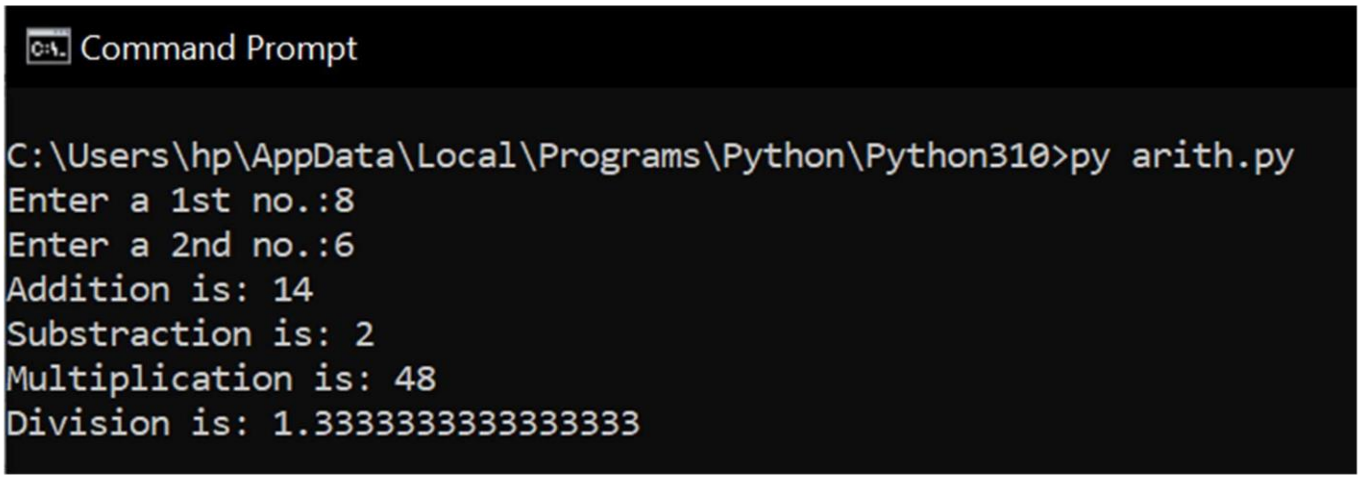
Syntax :- class ClassName:-

```
    Statements
```

Program:-class Arith:

```
    def __init_(self,n1,n2):
        self.n1=n1
        self.n2=n2
    def add(self):
       print('Addition is:',self.n1+self.n2)
    def sub(self):
       print('Substraction is:',self.n1-self.n2)
    def mul(self):
       print('Multiplication is:',self.n1*self.n2)
    def div(self):
       print('Division is:',self.n1/self.n2)
n1=int(input('Enter a 1st no.:'))
n2=int(input('Enter a 2nd no.:'))
a=Arith(n1,n2)
a.add()
a.sub()
a.mul()
a.div()
```

Output:-



```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py arith.py
Enter a 1st no.:8
Enter a 2nd no.:6
Addition is: 14
Substraction is: 2
Multiplication is: 48
Division is: 1.3333333333333333
```

Practical No.: 8

Name of the Practical: W.A.P. to perform CRUD operations with MySQL Database.

Software Required: Python 3.10.2, Edit Plus, Command Prompt.

Syntax: show databases;

C- Create Database (create database (name);)

R- Read Database (display table;)

U- Update Database

D- Delete Database (drop database (name);) (desc ;)

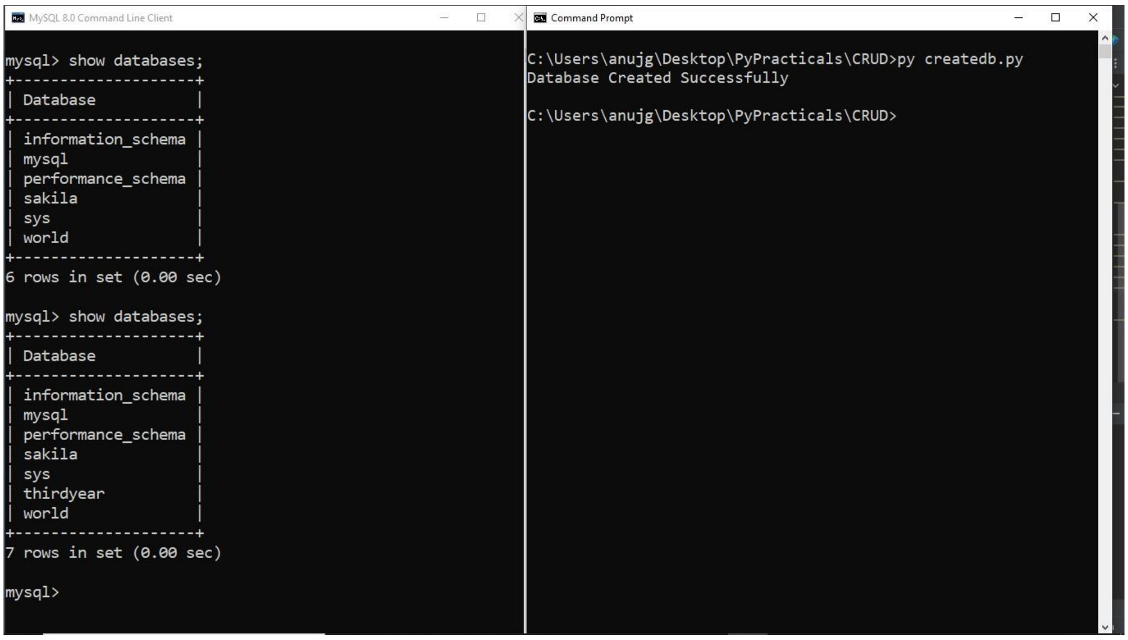
Program:

A) CRUD operations with MySQL

1) Create database:

```
#show databases;
#create database thirdyear;
#drop database thirdyear;
#use thirdyear;
#show tables;
#desc student;
#SELECT * FROM student;

#CREATE DATABASE

#WAP to create table in mysql database: thridyear
try:
      import mysql.connector #Importing Connectorpackage
      mysqldb=mysql.connector.connect(host="localhost",port=3306,user="root",
      password="root")#established connection
      mycursor=mysqldb.cursor()#cursor() method createa cursor object
      mycursor.execute("create database thirdyear")#Execute SQL Query to
      create a database
      mysqldb.close()#Connection Close
except:
        print('Database Not Created')
finally:
         print('Database Created Successfully')
```

Output:


MySQL 8.0 Command Line Client

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| sys                |
| world              |
+--------------------+
6 rows in set (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| sys                |
| thirdyear          |
| world              |
+--------------------+
7 rows in set (0.00 sec)

mysql>
```

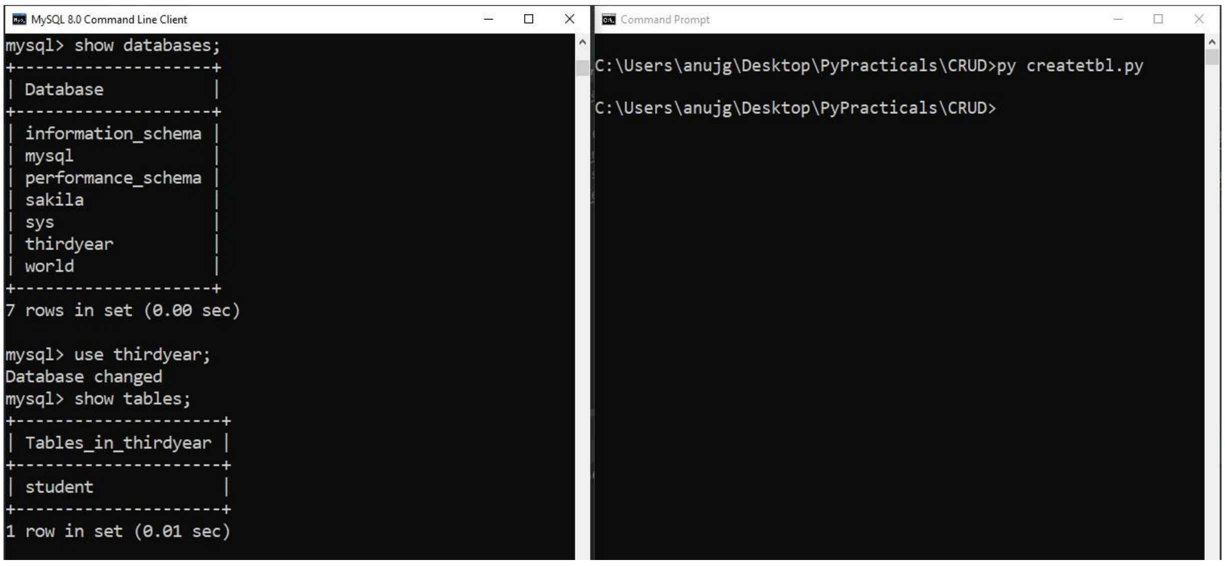
Command Prompt

```
C:\Users\anujg\Desktop\PyPracticals\CRUD>py createdb.py
Database Created Successfully

C:\Users\anujg\Desktop\PyPracticals\CRUD>
```

2) Create Table:

```
#Create a table into dbpython database

import mysql.connector
mysqldb=mysql.connector.connect(host="localhost",user="root",
password="root",database="thirdyear")#established connection between your
database
mycursor=mysqldb.cursor()#cursor() method create a cursor object
mycursor.execute("create table student(roll INT,nameVARCHAR(255), marks
INT)")#Execute SQL Query to create a table into your database
mysqldb.close()#Connection Close
```

Output:


MySQL 8.0 Command Line Client

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| sys                |
| thirdyear          |
| world              |
+--------------------+
7 rows in set (0.00 sec)

mysql> use thirdyear;
Database changed
mysql> show tables;
+---------------------+
| Tables_in_thirdyear |
+---------------------+
| student             |
+---------------------+
1 row in set (0.01 sec)
```

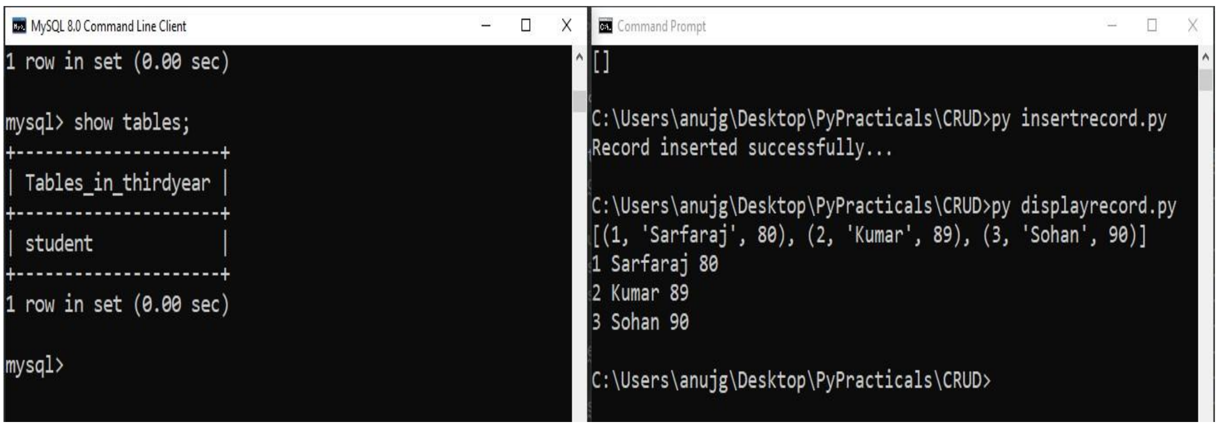
Command Prompt

```
C:\Users\anujg\Desktop\PyPracticals\CRUD>py createtbl.py

C:\Users\anujg\Desktop\PyPracticals\CRUD>
```

3) Insert Record:

```
import mysql.connector

mysqldb=mysql.connector.connect(host="localhost",user="root",
password="root", database="thirdyear")#established connection between your
database
mycursor=mysqldb.cursor()#cursor() method create a cursor object
try:
    #Execute SQL Query to insert record mycursor.execute("insert into student
    values(1,'Sarfaraj',80),(2,'Kumar',89),(3,'Sohan',90)")
    mysqldb.commit() # Commit is used for yourchanges in the
    database
     print('Record inserted successfully...')
except:
    # rollback used for if any errormysqldb.rollback()
mysqldb.close()#Connection Close
```

Output:



```
1 row in set (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_thirdyear |
+--------------------+
| student            |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

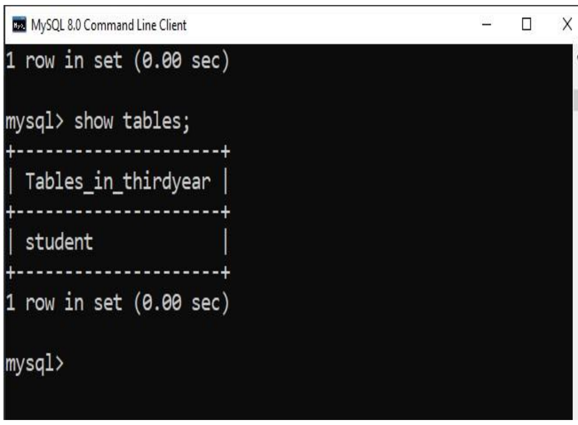
```
[]

C:\Users\anujg\Desktop\PyPracticals\CRUD>py insertrecord.py
Record inserted successfully...

C:\Users\anujg\Desktop\PyPracticals\CRUD>py displayrecord.py
[(1, 'Sarfaraj', 80), (2, 'Kumar', 89), (3, 'Sohan', 90)]
1 Sarfaraj 80
2 Kumar 89
3 Sohan 90

C:\Users\anujg\Desktop\PyPracticals\CRUD>
```

4) Display Record:

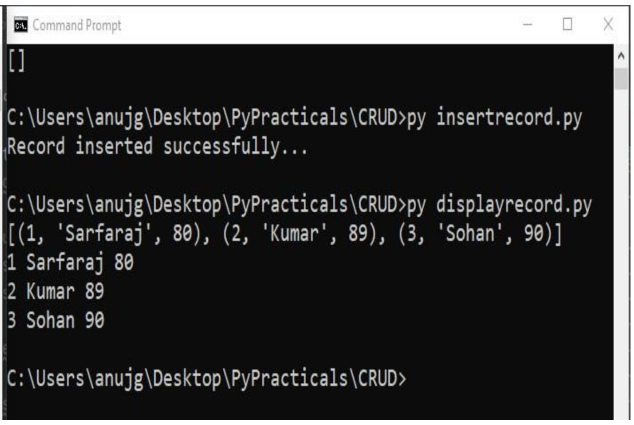
```
import mysql.connector

 mysqldb=mysql.connector.connect(host="localhost",user="root",
password="root", database="thirdyear")#established connection between your
database
mycursor=mysqldb.cursor()#cursor() method create a cursor object
try:
    mycursor.execute("select * from student")#ExecuteSQL Query to select all
    record
    result=mycursor.fetchall() #fetches all the rowsin a result set
    print(result) for i in result:
    roll=i[0] name=i[1]
    marks=i[2]
```

```
        print(roll,name,marks)
    except:
        print('Error:Unable to fetch data.')
    mysqldb.close()#Connection Close
```

Output:



```
1 row in set (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_thirdyear |
+--------------------+
| student            |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

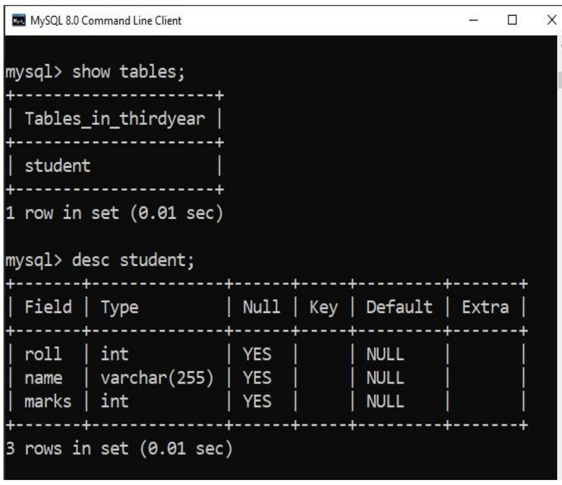


```
[]

C:\Users\anujg\Desktop\PyPracticals\CRUD>py insertrecord.py
Record inserted successfully...

C:\Users\anujg\Desktop\PyPracticals\CRUD>py displayrecord.py
[(1, 'Sarfaraj', 80), (2, 'Kumar', 89), (3, 'Sohan', 90)]
1 Sarfaraj 80
2 Kumar 89
3 Sohan 90

C:\Users\anujg\Desktop\PyPracticals\CRUD>
```

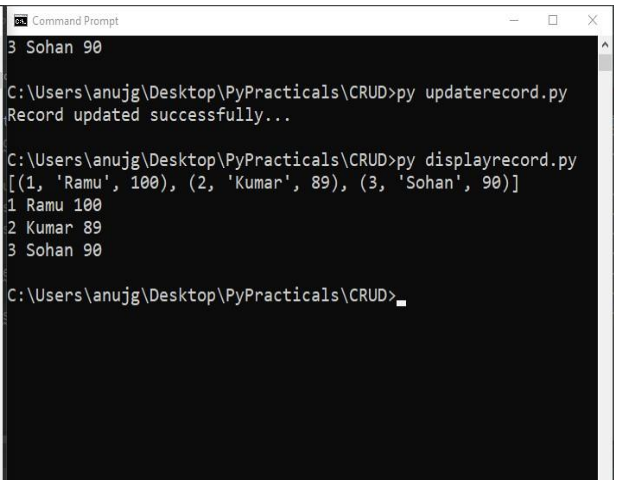
5) Update Record:

```
    import mysql.connector

    mysqldb=mysql.connector.connect(host="localhost",user="root",
    password="root",database="thirdyear")#established connection between your
    database
    mycursor=mysqldb.cursor()#cursor() method create a cursor object
    try:
        mycursor.execute("UPDATE student SET name='Ramu',marks=100 WHERE
         roll=1")#Execute SQL Query to updaterecord
        mysqldb.commit() # Commit is used for yourchanges in the
        database
        print('Record updated successfully...')
    except:
      # rollback used for if any errormysqldb.rollback()
    mysqldb.close()#Connection Close
```

Output:



```
mysql> show tables;
+--------------------+
| Tables_in_thirdyear |
+--------------------+
| student            |
+--------------------+
1 row in set (0.01 sec)

mysql> desc student;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| roll  | int          | YES  |     | NULL    |       |
| name  | varchar(255) | YES  |     | NULL    |       |
| marks | int          | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

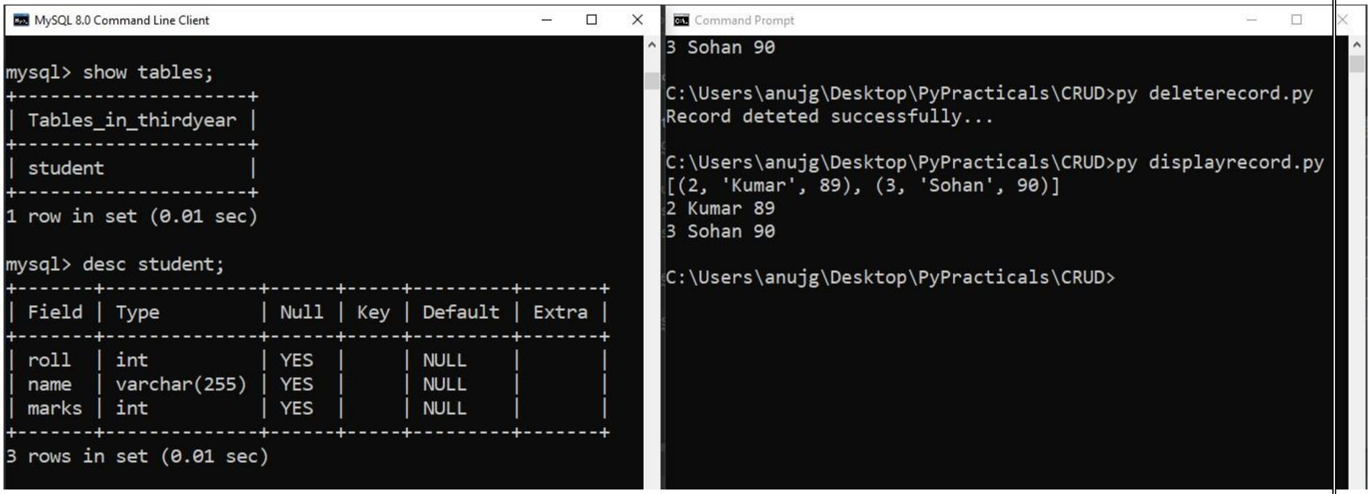


```
3 Sohan 90

C:\Users\anujg\Desktop\PyPracticals\CRUD>py updaterecord.py
Record updated successfully...

C:\Users\anujg\Desktop\PyPracticals\CRUD>py displayrecord.py
[(1, 'Ramu', 100), (2, 'Kumar', 89), (3, 'Sohan', 90)]
1 Ramu 100
2 Kumar 89
3 Sohan 90

C:\Users\anujg\Desktop\PyPracticals\CRUD>
```

6) Delete Record:

```
import mysql.connector

 mysqldb=mysql.connector.connect(host="localhost",user="root",
password="root", database="thirdyear")#established connection between your
database
mycursor=mysqldb.cursor()#cursor() method create a cursor object
try:
        mycursor.execute("DELETE FROM student WHEREroll=1")#Execute
        SQL Query to detete a record
        mysqldb.commit() # Commit is used for yourchanges in the
database
        print('Record deteted successfully...')
except:
     # rollback used for if any errormysqldb.rollback()
mysqldb.close()#Connection Close
```

Output:



MySQL 8.0 Command Line Client

```
mysql> show tables;
+---------------------+
| Tables_in_thirdyear |
+---------------------+
| student             |
+---------------------+
1 row in set (0.01 sec)

mysql> desc student;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| roll  | int          | YES  |     | NULL    |       |
| name  | varchar(255) | YES  |     | NULL    |       |
| marks | int          | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

Command Prompt

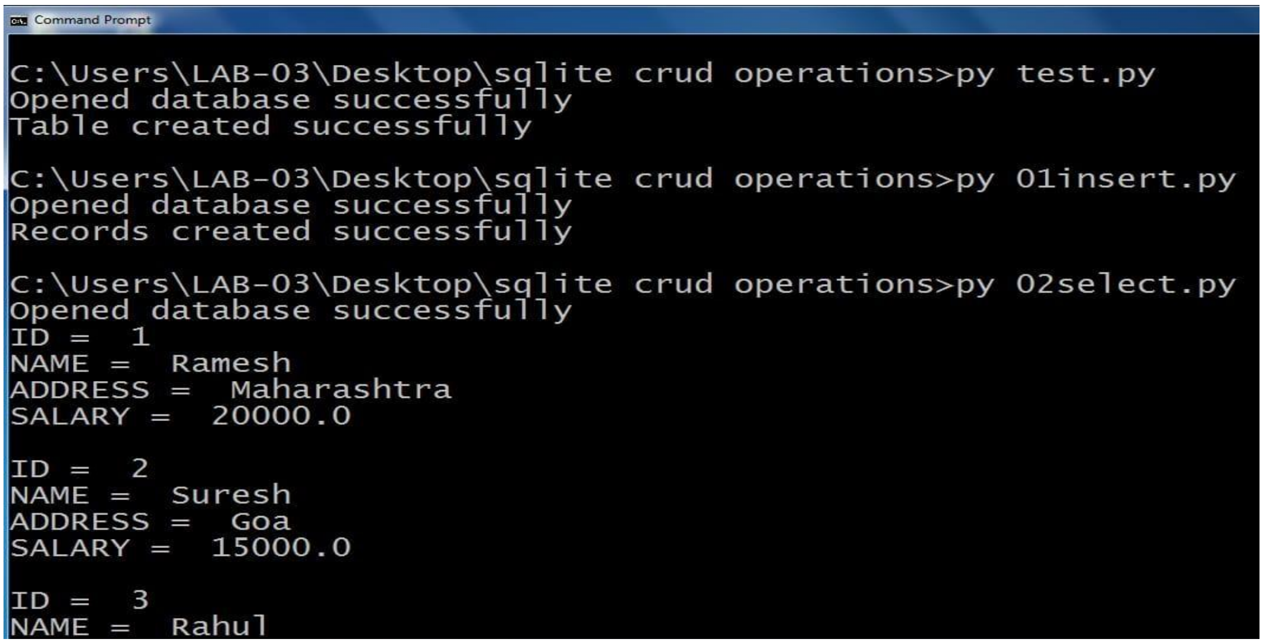
```
3 Sohan 90

C:\Users\anujg\Desktop\PyPracticals\CRUD>py deleterecord.py
Record deteted successfully...

C:\Users\anujg\Desktop\PyPracticals\CRUD>py displayrecord.py
[(2, 'Kumar', 89), (3, 'Sohan', 90)]
2 Kumar 89
3 Sohan 90

C:\Users\anujg\Desktop\PyPracticals\CRUD>
```

B) <u>CRUD operations using SQLite</u>

1) Create Database:

```
import sqlite3

conn = sqlite3.connect('test2.db') print("Opened database

successfully")

conn.execute('''CREATE TABLE COMPANY
(ID INT PRIMARY KEY                         NOT NULL,
NAME                              TEXT      NOT NULL,
AGE                               INT       NOT NULL,
ADDRESS                           CHAR(50),
SALARY                            REAL);''')
print("Table created successfully")

conn.close()
```

Output:

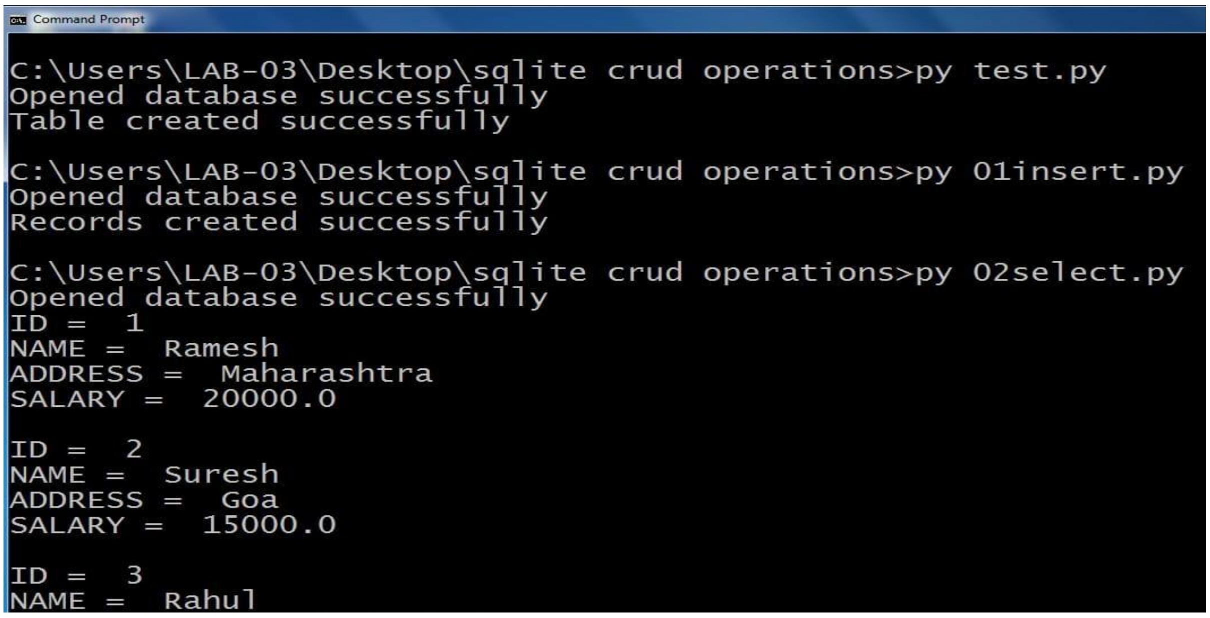


```
C:\Users\LAB-03\Desktop\sqlite crud operations>py test.py
Opened database successfully
Table created successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>py 01insert.py
Opened database successfully
Records created successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>py 02select.py
Opened database successfully
ID =  1
NAME =  Ramesh
ADDRESS =  Maharashtra
SALARY =  20000.0

ID =  2
NAME =  Suresh
ADDRESS =  Goa
SALARY =  15000.0

ID =  3
NAME =  Rahul
```

2) Insert Record:

```
import sqlite3

conn = sqlite3.connect('test2.db') print("Opened database
successfully")

conn.execute("INSERT INTO COMPANY
(ID,NAME, AGE,ADDRESS,SALARY) \
VALUES (1, 'Ramesh', 32, 'Maharashtra',
20000.00 )");

conn.execute("INSERT INTO COMPANY
(ID,NAME,AGE,ADDRESS,SALARY) \
VALUES (2, 'Suresh', 25, 'Goa', 15000.00 )");

conn.execute("INSERT INTO COMPANY
(ID,NAME,AGE,ADDRESS,SALARY) \
VALUES (3, 'Rahul', 23, 'MP', 20000.00 )");

conn.execute("INSERT INTO COMPANY
(ID,NAME,AGE,ADDRESS,SALARY)\
VALUES (4, 'Sachin', 25, 'Gujrat ', 65000.00)");

conn.commit()
print("Records created successfully")conn.close()
```

Output:

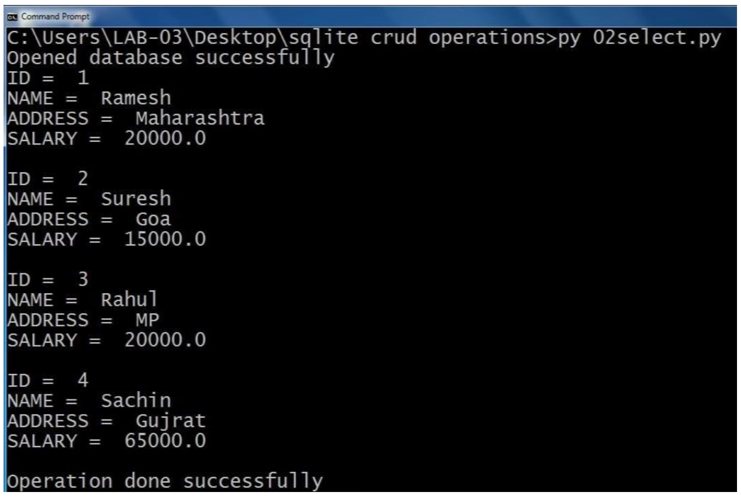


```
Command Prompt

C:\Users\LAB-03\Desktop\sqlite crud operations>py test.py
Opened database successfully
Table created successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>py 01insert.py
Opened database successfully
Records created successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>py 02select.py
Opened database successfully
ID =  1
NAME =  Ramesh
ADDRESS =  Maharashtra
SALARY =  20000.0

ID =  2
NAME =  Suresh
ADDRESS =  Goa
SALARY =  15000.0

ID =  3
NAME =  Rahul
```

3) Select Record:

```
import sqlite3

conn = sqlite3.connect('test2.db') print("Opened database
successfully")

cursor = conn.execute("SELECT id, name, address,salary from COMPANY")
for row in cursor: print("ID = ", row[0])
print("NAME = ", row[1]) print("ADDRESS = ", row[2])
print("SALARY = ", row[3], "\n")

print("Operation done successfully")conn.close()
```

Output:

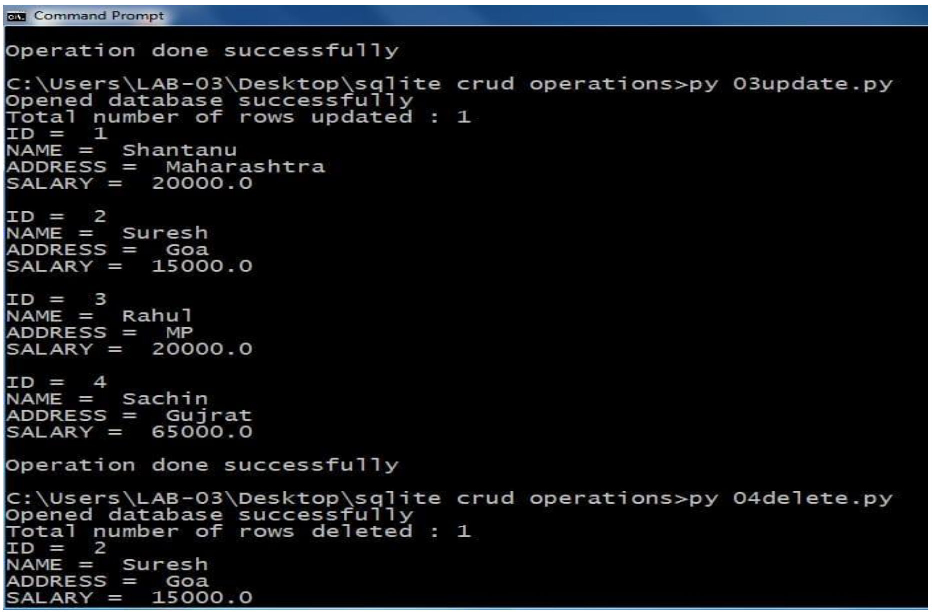


```
Command Prompt
C:\Users\LAB-03\Desktop\sqlite crud operations>py 02select.py
Opened database successfully
ID =  1
NAME =  Ramesh
ADDRESS =  Maharashtra
SALARY =  20000.0

ID =  2
NAME =  Suresh
ADDRESS =  Goa
SALARY =  15000.0

ID =  3
NAME =  Rahul
ADDRESS =  MP
SALARY =  20000.0

ID =  4
NAME =  Sachin
ADDRESS =  Gujrat
SALARY =  65000.0

Operation done successfully
```

4) Update Record:

```
import sqlite3

conn = sqlite3.connect('test2.db') print("Opened database
successfully")

conn.execute("UPDATE COMPANY set NAME = 67000.00where ID = 1")
conn.commit()
print("Total number of rows updated :",conn.total_changes)

cursor = conn.execute("SELECT id, name, address,salary from COMPANY")
for row in cursor: print("ID = ", row[0])
print("NAME = ", row[1]) print("ADDRESS = ", row[2])
print("SALARY = ", row[3], "\n")

print("Operation done successfully")conn.close()
```

Output:

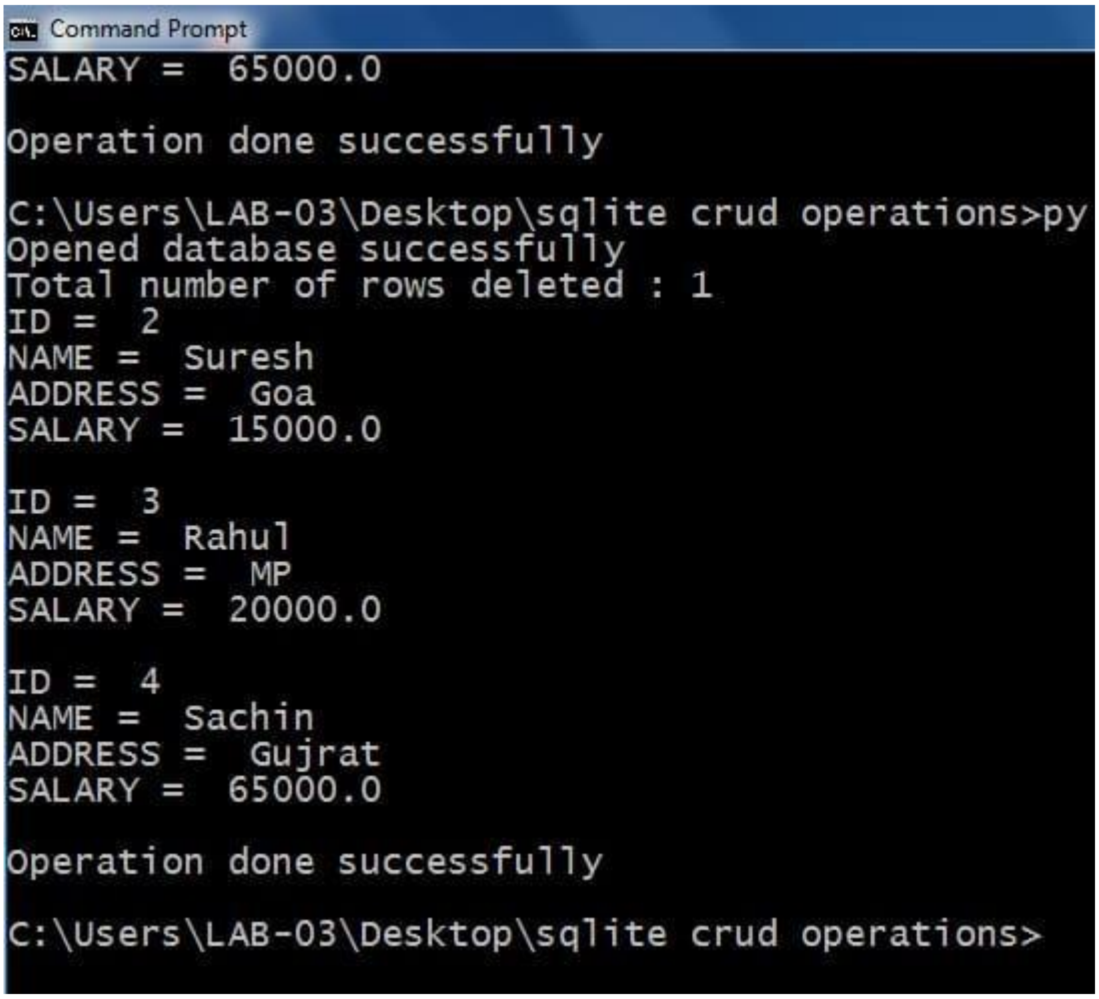


```
Operation done successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>py 03update.py
Opened database successfully
Total number of rows updated : 1
ID =  1
NAME =  Shantanu
ADDRESS =  Maharashtra
SALARY =  20000.0

ID =  2
NAME =  Suresh
ADDRESS =  Goa
SALARY =  15000.0

ID =  3
NAME =  Rahul
ADDRESS =  MP
SALARY =  20000.0

ID =  4
NAME =  Sachin
ADDRESS =  Gujrat
SALARY =  65000.0

Operation done successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>py 04delete.py
Opened database successfully
Total number of rows deleted : 1
ID =  2
NAME =  Suresh
ADDRESS =  Goa
SALARY =  15000.0
```

5) Delete Record:

```
import sqlite3

conn = sqlite3.connect('test2.db') print("Opened database
successfully");

conn.execute("DELETE from COMPANY where ID = 1;")
conn.commit()
print("Total number of rows deleted :",conn.total_changes)

cursor = conn.execute("SELECT id, name, address,salary from COMPANY")
for row in cursor: print("ID = ", row[0])
print("NAME = ", row[1]) print("ADDRESS = ", row[2]) print("SALARY = ",
row[3], "\n")

print("Operation done successfully")conn.close()
```

Output:



```
SALARY =  65000.0

Operation done successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>py
Opened database successfully
Total number of rows deleted : 1
ID =  2
NAME =  Suresh
ADDRESS =  Goa
SALARY =  15000.0

ID =  3
NAME =  Rahul
ADDRESS =  MP
SALARY =  20000.0

ID =  4
NAME =  Sachin
ADDRESS =  Gujrat
SALARY =  65000.0

Operation done successfully

C:\Users\LAB-03\Desktop\sqlite crud operations>
```

Practical No.:- 9

Name of the Practical:-WAP to Understand Python List Comprehension with suitable example.

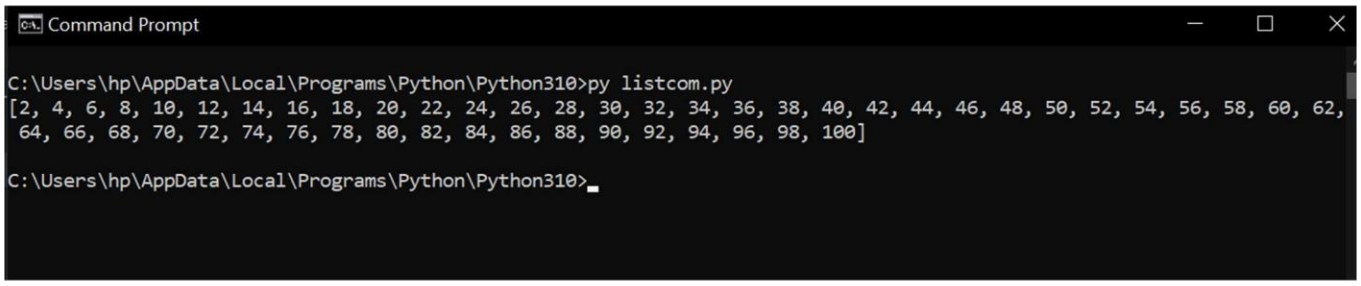
Software Required:-Python 3.10, Edit Plus

Syntax :-list=[element expression for element if condition]

Program:-#Creating list of even no.from 1 to 100 using list comprehensive

```
list=[x for x in range(0,101) if x%2==0]
print(list)
```

Output:-



```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py listcom.py
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62,
 64, 66, 68, 70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100]

C:\Users\hp\AppData\Local\Programs\Python\Python310>
```

Practical No.: 10

Name of the Practical:-WAP to Demonstrate Inheritance concept with suitable example.

Software Required:-Python 3.10, Edit Plus

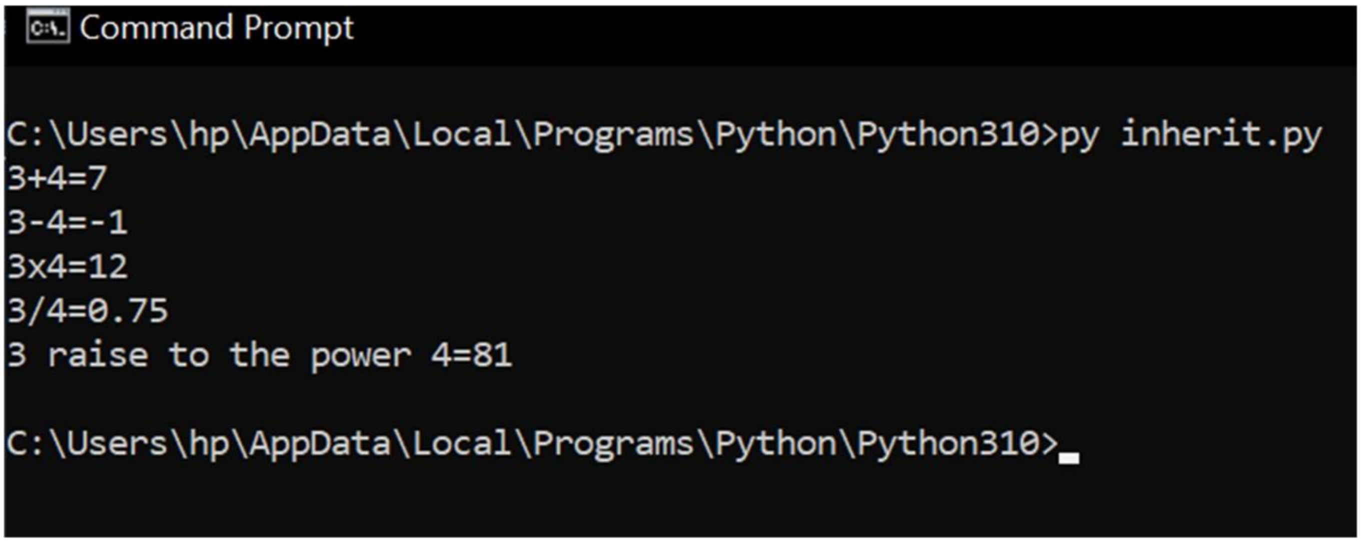
Syntax :-class a:

```
        Statements
      class b(a):
          Statements
```

Program:class Convension:

```
        def __init_(self,n1,n2):
              self.n1=n1
              self.n2=n2
          def add(self):
               print(f'{self.n1}+{self.n2}={self.n1+self.n2}')\
          def sub(self):
               print(f'{self.n1}-{self.n2}={self.n1-self.n2}')
          def mul(self):
               print(f'{self.n1}x{self.n2}={self.n1*self.n2}')
          def div(self):
               print(f'{self.n1}/{self.n2}={self.n1/self.n2}')
        class Scientific:
          def___init_(self,n1,n2):
              self.n1=n1
              self.n2=n2
            def pow(self):
               print(f'{self.n1} raise to the power {self.n2}={self.n1**self.n2}')
          s=Scienfic(3,4)
          s.add()
          s.sub()
          s.mul()
          s.div()
          s.pow()
```

Output:-



```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py inherit.py
3+4=7
3-4=-1
3x4=12
3/4=0.75
3 raise to the power 4=81

C:\Users\hp\AppData\Local\Programs\Python\Python310>
```

Practical No.: 11

Name of the Practical:-WAP to Analyze various Exceptions.

Software Required:-Python 3.10, Edit Plus

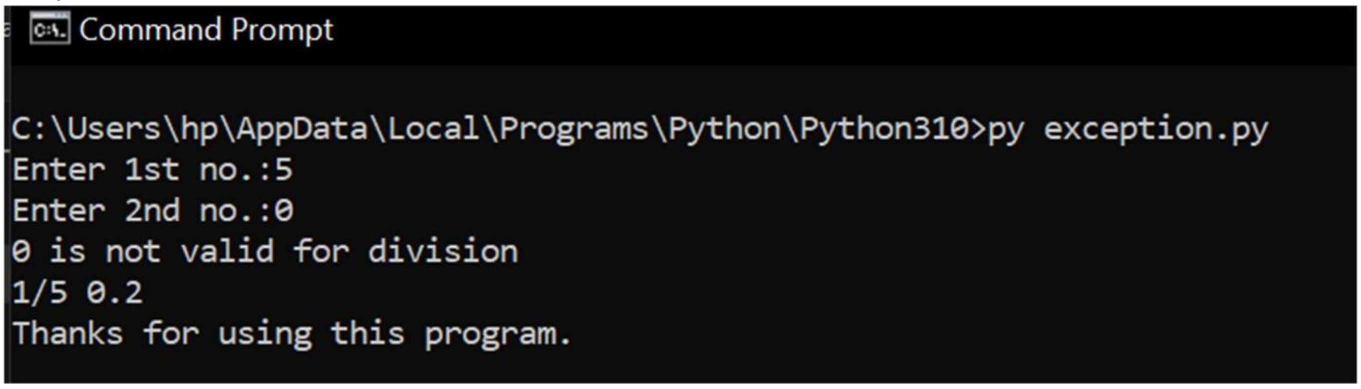
Syntax :-try:

```
try:
    Statements
except:
    Statements
```

Program:--#Exception Handling

```
try:
    n1=int(input('Enter 1st no.:'))
    n2=int(input('Enter 2nd no.:'))
    div=n1/n2
    print(f'{n1}/{n2}={div}')
except ZeroDivisionError:
    print(f'{n2} is not valid for division ')
finally:
    try:
        print(f'1/{n1}',1/n1)
    except ZeroDivisionError:
        print('Use valid no. for Inverse')
    finally:
        print('Thanks for using this program.')
```

Output:-



```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py exception.py
Enter 1st no.:5
Enter 2nd no.:0
0 is not valid for division
1/5 0.2
Thanks for using this program.
```

Practical No.: 12

Name of the Practical:-WAP to Design Numeric value to Word Converter.

Software Required:-Python 3.10, Edit Plus

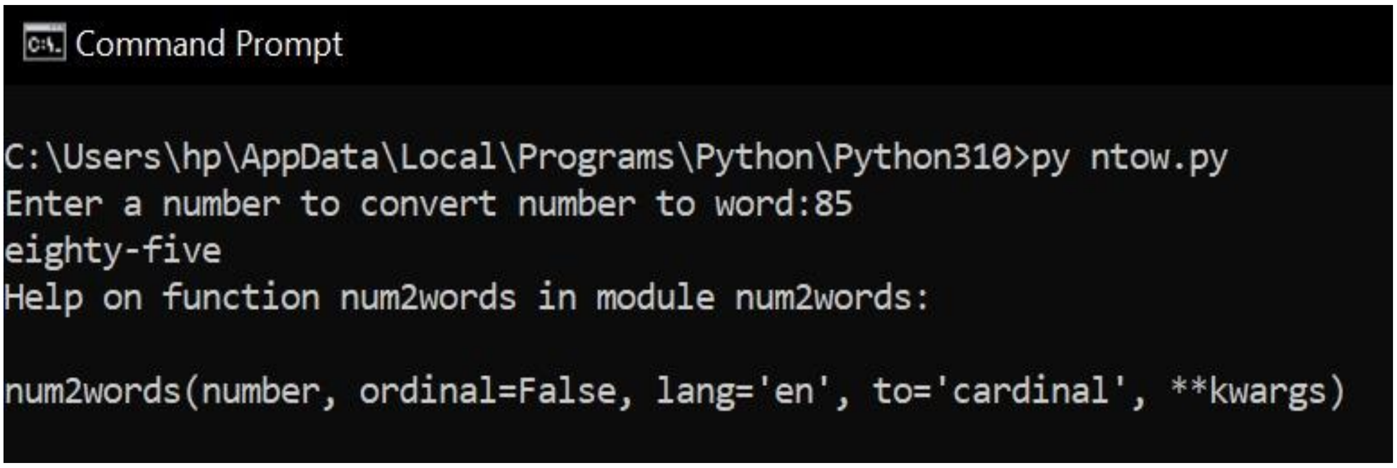
Syntax :-from num2words import num2words

```
Print(num2words(number))
```

Program: #word to numeric converter

```
from num2words import num2words
n=int(input('Enter a number to convert number to word:'))
print(num2words(n))

help(num2words)

dir(num2words)
```

Output:-



```
C:\Users\hp\AppData\Local\Programs\Python\Python310>py ntow.py
Enter a number to convert number to word:85
eighty-five
Help on function num2words in module num2words:

num2words(number, ordinal=False, lang='en', to='cardinal', **kwargs)
```