https://pythonschoolkvs.wordpress.com/

CLASS-XII

SUBJECT – COMPUTER SCIENCE (083)

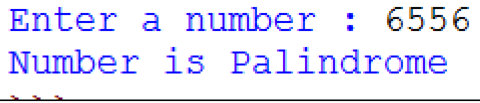
PRACTICAL FILE SOLUTION (SESSION 2020-21)

PRACTICAL NO.	OBJECTIVE & SOLUTION
1.	Write a program to check a number whether it is palindrome or not.

SOURCE CODE:

```
num=int(input("Enter a number : "))
n=num
res=0
while num>0:
    rem=num%10
    res=rem+res*10
    num=num//10
if res==n:
    print("Number is Palindrome")
else:
    print("Number is not Palindrome")
```

OUTPUT:



```
Enter a number : 6556
Number is Palindrome
>>>
```

2. Write a program to display ASCII code of a character and vice versa.

SOURCE CODE:

```
var=True
while var:
    choice=int(input("Press-1 to find the ordinal value of a character \nPress-2 to find a character of a value\n"))
    if choice==1:
        ch=input("Enter a character : ")
        print(ord(ch))
    elif choice==2:
        val=int(input("Enter an integer value: "))
        print(chr(val))
    else:
        print("You entered wrong choice")

    print("Do you want to continue? Y/N")
    option=input()
    if option=='y' or option=='Y':
        var=True
    else:
        var=False
```

OUTPUT:

```
Press-1 to find the ordinal value of a character
Press-2 to find a character of a value
1
```

```
Enter a character : a
97
Do you want to continue? Y/N
Y
Press-1 to find the ordinal value of a character
Press-2 to find a character of a value
2
Enter an integer value: 65
A
Do you want to continue? Y/N
```

3. Write a python program to sum the sequence given below. Take the input n from the user. 1+1/1!+1/2!+1/3!+⋯+1/n!

SOURCE CODE:

```
def fact(x):
    j=1
    res=1
    while j<=x:
        res=res*j
        j=j+1
        return res
n=int(input("enter the number : "))
i=1
```

```
          sum=1
          while i<=n:
           f=fact(i)
           sum=sum+1/f
           i+=1
print(sum)
```

OUTPUT:

```
enter the number : 6
7.0
```

4. Write a program to calculate the factorial of an integer using recursion.

SOURCE CODE:

```
def factorial(n):
   if n == 1:
       return n
   else:
       return n*factorial(n-1)

num=int(input("enter the number: "))
if num < 0:
   print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
   print("The factorial of 0 is 1")
else:
   print("The factorial of  ",num,"  is  ", factorial(num))
```

OUTPUT:

```
enter the number: 5
The factorial of   5   is   120
```

5. Write a program to find sum of all elements of a list using recursion.

SOURCE CODE:

```
def Sum_ListEle(LS,n):
    if n==0:
        return 0
    else:
        return LS[n-1]+Sum_ListEle(LS,n-1)

L=eval(input("enter a list of numbers : "))
size=len(L)
total=Sum_ListEle(L,size)
print("Sum of list elements is :", total)
```

OUTPUT:

```
enter a list of numbers : [10,2,5,8,1]
Sum of list elements is : 26
```

6. Write a program to print fibonacci series using recursion.

SOURCE CODE:

```
def fibonacci(n):
    if n<=1:
        return n
    else:
        return(fibonacci(n-1)+fibonacci(n-2))
```

```
num=int(input("How many terms you want to display: "))
for i in range(num):
    print(fibonacci(i)," ", end=" ")
```

OUTPUT:

```
How many terms you want to display: 8
0   1   1   2   3   5   8   13
```

7. Write a program for binary search using recursion.

SOURCE CODE:

```
def Binary_Search(sequence, item, LB, UB):
    if LB>UB:
        return -5                    # return any negative value
    mid=int((LB+UB)/2)
    if item==sequence[mid]:
        return mid
    elif item<sequence[mid]:
        UB=mid-1
        return Binary_Search(sequence, item, LB, UB)
    else:
        LB=mid+1
        return Binary_Search(sequence, item, LB, UB)

L=eval(input("Enter the elements in sorted order: "))
n=len(L)
element=int(input("Enter the element that you want to search :"))
found=Binary_Search(L,element,0,n-1)
```

```
if found>=0:
    print(element, "Found at the index : ",found)
else:
    print("Element not present in the list")
```

OUTPUT:

```
Enter the elements in sorted order: 12,23,35,46,58,69,75,88,99
Enter the element that you want to search :69
69 Found at the index :  5
```

8. Write a recursive python program to test if a string is palindrome or not.

SOURCE CODE:

```
def isStringPalindrome(str):
    if len(str)<=1:
        return True
    else:
        if str[0]==str[-1]:
            return isStringPalindrome(str[1:-1])
        else:
            return False

#__main__

s=input("Enter the string : ")
y=isStringPalindrome(s)

if y==True:
    print("String is Palindrome")
else:
    print("String is Not Palindrome")
```

OUTPUT:

```
Enter the string : madam
String is Palindrome
```

9.

Write a program to read a text file line by line and display each word separated by '#'.

SOURCE CODE:

```
fin=open("D:\\python programs\\Book.txt",'r')
L1=fin.readlines( )
s=' '
for i in range(len(L1)):
    L=L1[i].split( )
    for j in L:
        s=s+j
        s=s+'#'
print(s)
fin.close( )
```

OUTPUT:

```
Text in file:
hello how are you?
python is case-sensitive language.

Output in python shell:
hello#how#are#you?#python#is#case-sensitive#language.#
```

10.	Write a program to count the number of vowels present in a text file.

SOURCE CODE:	fin=open("D:\\python programs\\MyBook.txt",'r')
	str=fin.read( )
	count=0
	for i in str:
	    if i=='a' or i=='e' or i=='i' or i=='o' or i=='u':
	        count=count+1
	
	print(count)

OUTPUT:	9
11.	Write a program to count number of words in a file.

SOURCE CODE:	fin=open("D:\\python programs\\Story.txt",'r')
	str=fin.read( )
	L=str.split()
	count_words=0
	for i in L:
	    count_words=count_words+1
	print(count_words)

OUTPUT:	16
12.	Write a program to count the number of times the occurrence of 'is' word in a text file.

SOURCE CODE:	fin=open("D:\\python programs\\Book.txt",'r')
	str=fin.read( )

	L=str.split( )
	count=0
	for i in L:
	    if i=='is':
	        count=count+1
	print(count)
	fin.close( )

OUTPUT:	3
13.	Write a program to write those lines which have the character 'p' from one text file to another text file.

SOURCE CODE:	fin=open("E:\\book.txt","r")
	fout=open("E:\\story.txt","a")
	s=fin.readlines( )
	for j in s:
	    if 'p' in j:
	        fout.write(j)
	
	fin.close()
	fout.close()

OUTPUT:	**Write contents of book.txt and story.txt
14.	Create a binary file with name and roll number of student and display the data by reading the file.

SOURCE CODE:

```
import pickle
def writedata( ):
    list =[ ]
    while True:
        roll = input("Enter student Roll No:")
        sname  = input("Enter student Name :")
        student = {"roll":roll,"name":sname}
        list.append(student)
        choice= input("Want to add more record(y/n) :")
        if(choice=='n'):
            break

    file = open("student.dat","wb")
    pickle.dump(list,file)
    file.close( )

def readdata( ):
    file = open("student.dat", "rb")
    list = pickle.load(file)
    print(list)
    file.close( )

print("Press-1 to write data and Press-2 to read data")
```

```
choice=int(input( ))
if choice==1:
    writedata( )
elif choice==2:
    readdata( )
else:
    print("You entered invalid value")
```

OUTPUT:

```
Press-1 to write data and Press-2 to read data
1
Enter student Roll No:1201
Enter student Name :Devansh
Want to add more record(y/n) :y
Enter student Roll No:1202
Enter student Name :Divya
Want to add more record(y/n) :n
Press-1 to write data and Press-2 to read data
2
[{'roll': '1201', 'name': 'Devansh'}, {'roll': '1202', 'name': 'Divya'}]
```

15. Write a program to search a record using its roll number and display the name of student. If record not found then display appropriate message.

SOURCE CODE:

```
import pickle
roll = input('Enter roll number that you want to search in binary file :')
file = open("student.dat", "rb")
```

```
list = pickle.load(file)
file.close( )
for x in list:
    if roll in x['roll']:
        print("Name of student is:", x['name'])
        break
else:
    print("Record not found")
```

OUTPUT:

```
Enter roll number that you want to search in binary file :1202
Name of student is: Divya
```

16. Write a program to update the name of student by using its roll number in a binary file.

SOURCE CODE:

```
import pickle
roll = input('Enter roll number whose name you want to update in binary file :')
file = open("student.dat", "rb+")
list = pickle.load(file)
found = 0
lst = [ ]
for x in list:
    if roll in x['roll']:
        found = 1
        x['name'] = input('Enter new name: ')
    lst.append(x)
```

```
if found == 1:
    file.seek(0)
    pickle.dump(lst, file)
    print("Record Updated")
else:
    print('roll number does not exist')

file.close( )
```

OUTPUT:

```
Enter roll number whose name you want to update in binary file :1202
Enter new name: Harish
Record Updated
```

17. Write a program to delete a record from binary file.

SOURCE CODE:

```
import pickle
roll = input('Enter roll number whose record you want to delete:')
file = open("student.dat", "rb+")
list = pickle.load(file)
found = 0
lst = []
for x in list:
    if roll not in x['roll']:
        lst.append(x)
    else:
        found = 1
```

```
if found == 1:
    file.seek(0)
    pickle.dump(lst, file)
    print("Record Deleted ")
else:
    print('Roll Number does not exist')

file.close()
```

OUTPUT:

```
Enter roll number whose record you want to delete:1201
Record Deleted
```

18. Write a program to perform read and write operation with .csv file.

SOURCE CODE:

```
import csv
def readcsv():
    with open('C:\\Users\\ViNi\\Downloads\\data.csv','rt')as f:
        data = csv.reader(f)      #reader function to generate a reader object
        for row in data:
            print(row)



def writecsv( ):
    with open('C:\\Users\\ViNi\\Downloads\\data.csv', mode='a', newline=") as
file:
        writer = csv.writer(file, delimiter=',', quotechar='"')

        #write new record in file
        writer.writerow(['4', 'Devansh', 'Arts', '404'])


print("Press-1 to Read Data and Press-2 to Write data: ")
a=int(input())
if a==1:
    readcsv()
elif a==2:
    writecsv()
else:
    print("Invalid value")
```

OUTPUT:

```
Press-1 to Read Data and Press-2 to Write data:
1
['Roll No.', 'Name of student', 'stream', 'Marks']
['1', 'Anil', 'Arts', '426']
['2', 'Sujata', 'Science', '412']
['3', 'Shivani', 'Commerce', '448']
['4', 'Devansh', 'Arts', '404']
```

19. Write a program to generate random numbers between 1 to 6 and check whether a user won a lottery or not.

SOURCE CODE:

```
import random
n=random.randint(1,6)
guess=int(input("Enter a number between 1 to 6 :"))
if n==guess:
    print("Congratulations, You won the lottery ")
else:
    print("Sorry, Try again, The lucky number was : ", n)
```

OUTPUT:

```
Enter a number between 1 to 6 :  4
Sorry, Try again, The lucky number was :  1
```

20. Write a program to create a library in python and import it in a program.

SOURCE CODE:

```
#Rect.py
class Rectangle:
    def __init__(self):
        print("Rectangle")

    def Area(self, length, width):
        self.l=length
        self.w=width
        print("Area of Rectangle is : ", self.l*self.w)
```

```
#Sq.py
class Square:
    def __init__(self):
        print("Square")
    def Area(self, side):
        self.a=side
        print("Area of square is : ", self.a*self.a)


#Tri.py
class Triangle:
    def __init__(self):
        print("Trinagle")

    def Area(self, base, height):
        self.b=base
        self.h=height
        ar= (1/2)*self.b*self.h
        print("Area of Triangle is : ", ar )
```

```
#main.py
from Shape import Rect
from Shape import Sq
from Shape import Tri



r=Rect.Rectangle( )    #Create an object r for Rectangle class
r.Area(10,20)       # Call the module Area( ) of Rectangle class by passing argument

s=Sq.Square( )      #Create an object s for Square class
s.Area(10)        # Call the module Area( ) of Square class by passing argument

t=Tri.Triangle( )   #Create an object t for Triangle class
t.Area(6,8)      # Call the module Area( ) of Triangle class by passing argument
```

OUTPUT:

```
Rectangle
Area of Rectangle is :  200
Square
Area of square is :  100
Trinagle
Area of Triangle is :  24.0
```

21. Write a program for linear search.

SOURCE CODE:

```
L=eval(input("Enter the elements: "))
n=len(L)
item=eval(input("Enter the element that you want to search : "))
for i in range(n):
    if L[i]==item:
        print("Element found at the position :", i+1)
        break
else:
    print("Element not Found")
```

OUTPUT:

```
Enter the elements: 23,67,44,99,65,33,78,12
Enter the element that you want to search : 33
Element found at the position : 6
```

22. Write a program for bubble sort.

SOURCE CODE:

```
L=eval(input("Enter the elements:"))
n=len(L)
for p in range(0,n-1):
    for i in range(0,n-1):
        if L[i]>L[i+1]:
            t=L[i]
            L[i]=L[i+1]
            L[i+1]=t
print("The sorted list is : ", L)
```

OUTPUT:

```
Enter the elements:[67,13,89,34,65,8,74,19]
The sorted list is :  [8, 13, 19, 34, 65, 67, 74, 89]
```

23. Write a menu based program to perform the operation on stack in python.

SOURCE CODE:

```
class Stack:
    def __init__(self):
        self.items = [ ]

    def isEmpty(self):          # Checks whether the stack is empty or not
        return self.items == [ ]

    def push(self, item):                       #Insert an element
        self.items.append(item)

    def pop(self):                              # Delete an element
        return self.items.pop( )

    def peek(self):                             #Check the value of top
        return self.items[len(self.items)-1]

    def size(self):             # Size of the stack i.e. total no. of elements in stack
        return len(self.items)
```

```
s = Stack( )
print("MENU BASED STACK")
cd=True
while cd:
    print(" 1.  Push ")
    print(" 2.  Pop  ")
    print(" 3.  Display ")
    print(" 4.  Size of Stack ")
    print(" 5.  Value at Top ")

    choice=int(input("Enter your choice (1-5) :  "))

    if choice==1:
        val=input("Enter the element: ")
        s.push(val)
    elif choice==2:
        if s.items==[ ]:
            print("Stack is empty")
        else:
            print("Deleted element is :", s.pop( ))
    elif choice==3:
        print(s.items)
    elif choice==4:
        print("Size of the stack is :", s.size( ))
    elif choice==5:
        print("Value of top element is :", s.peek( ))
    else:
        print("You enetered wrong choice ")

    print("Do you want to continue? Y/N")
    option=input( )
    if option=='y' or option=='Y':
        var=True
    else:
        var=False
```

OUTPUT:

```
MENU BASED STACK
 1.  Push
 2.  Pop
 3.  Display
 4.  Size of Stack
 5.  Value at Top
Enter your choice (1-5) :  1
Enter the element: 45
Do you want to continue? Y/N
y
```

```
 1.  Push
 2.  Pop
 3.  Display
 4.  Size of Stack
 5.  Value at Top
Enter your choice (1-5) :  3
['45']
Do you want to continue? Y/N
y
 1.  Push
 2.  Pop
 3.  Display
 4.  Size of Stack
 5.  Value at Top
```

24. Write a menu based program to perform the operation on queue in python.

SOURCE CODE:

```
class Queue:
    def __init__(Q):
        Q.items = [ ]

    def isEmpty(Q):          # Checks whether the queue is empty or not
        return Q.items == [ ]

    def Enqueue(Q, item):                        #Insert an element
```

```
          Q.items.append(item)
          if len(Q.items)==1:
               front=rear=0
          else:
               rear=len(Q.items)

     def Dequeue(Q):                                     # Delete an element
         return Q.items.pop(0)

     def peek(Q):                                   #Check the value of rear
         return Q.items[len(Q.items)-1]

     def size(Q):                # Size of the queue i.e. total no. of elements in queue
         return len(Q.items)



q = Queue( )
print("MENU BASED QUEUE")
cd=True
while cd:
    print(" 1.  ENQUEUE ")
    print(" 2.  DEQUEUE  ")
    print(" 3.  Display ")
print(" 4.  Size of Queue ")
print(" 5.  Value at rear ")

choice=int(input("Enter your choice (1-5) :  "))

if choice==1:
    val=input("Enter the element: ")
    q.Enqueue(val)
elif choice==2:
    if q.items==[ ]:
        print("Queue is empty")
    else:
        print("Deleted element is :", q.Dequeue( ))
elif choice==3:
    print(q.items)
elif choice==4:
    print("Size of the queue is :", q.size( ))
elif choice==5:
    print("Value of rear element is :", q.peek( ))
else:
    print("You enetered wrong choice ")

print("Do you want to continue? Y/N")
```

```
    option=input( )
    if option=='y' or option=='Y':
        cd=True
    else:
        cd=False
```

OUTPUT:

```
MENU BASED QUEUE
 1.  ENQUEUE
 2.  DEQUEUE
 3.  Display
 4.  Size of Queue
 5.  Value at rear
Enter your choice (1-5) :  1
Enter the element: 10
Do you want to continue? Y/N
y
 1.  ENQUEUE
 2.  DEQUEUE
 3.  Display
 4.  Size of Queue
 5.  Value at rear
Enter your choice (1-5) :  1
Enter the element: 45
```

	Do you want to continue? Y/N y 1. ENQUEUE 2. DEQUEUE 3. Display 4. Size of Queue 5. Value at rear Enter your choice (1-5) : 3 ['10', '45'] Do you want to continue? Y/N y 1. ENQUEUE 2. DEQUEUE 3. Display 4. Size of Queue 5. Value at rear Enter your choice (1-5) : 2 Deleted element is : 10 Do you want to continue? Y/N
25.	Write a program to find the most common words in a file.

SOURCE CODE:	import collections
	fin = open('E:\\email.txt','r')
	a= fin.read()

```
d={ }
L=a.lower().split()

for word in L:
    word = word.replace(".","")
    word = word.replace(",","")
    word = word.replace(":","")
    word = word.replace("\"","")
    word = word.replace("!","")
    word = word.replace("&","")
    word = word.replace("*","")



for k in L:
    key=k
    if key not in d:
        count=L.count(key)
        d[key]=count

n_print = int(input("How many most common words to print: "))
```

	print("\nOK. The {} most common words are as follows\n".format(n_print))
	
	word_counter = collections.Counter(d)
	
	for word, count in word_counter.most_common(n_print):
	    print(word, ": ", count)
	
	fin.close()

OUTPUT:	How many most common words to print: 5 OK. The 5 most common words are as follows the : 505 a : 297 is : 247 in : 231 to : 214
26.	Create a table EMPLOYEE with constraints
SOLUTION	Step-1 Create a database: CREATE DATABASE Bank;

	Step-2 Display the databases SHOW DATABASES; Step-3: Enter into database Use Bank; Step-4: Create the table EMPLOYEE create table Employee(Ecode int primary key,Ename varchar(20) NOT NULL, Dept varchar(15),City varchar(15), sex char(1), DOB date, Salary float(12,2));
27.	Insert data into the table
SOLUTION	insert into Employee values(1001,"Atul","Production","Vadodara","M","1992-10-23",23000.50); Query OK, 1 row affected (0.11 sec) *Note: Insert more rows as per above insert command.*
28.	Add a new column in a table.
SOLUTION	ALTER TABLE EMPLOYEE ADD address varchar(50);
29.	Change the data-type and size of an existing column.
SOLUTION	ALTER TABLE EMPLOYEE MODIFY city char(30);

30.	Write SQL queries using SELECT, FROM, WHERE clause based on EMPLOYEE table.

SOLUTION	1. List the name of female employees in EMPLOYEE table.
	Solution:- SELECT Ename
	FROM EMPLOYEE
	WHERE sex=’F’;
	2. Display the name and department of those employees who work in surat and salary is greater than 25000.
	Solution:- SELECT Ename, Dept
	FROM EMPLOYEE
	WHERE city=’surat’ and salary > 25000;
	3. Display the name of those female employees who work in Mumbai.
	Solution:- SELECT Ename
	FROM EMPLOYEE
	WHERE sex=’F’ and city=’Mumbai’;
	4. Display the name of those employees whose department is marketing or RND.
	Solution:- SELECT Ename
	FROM EMPLOYEE
	WHERE Dept=’marketing’ OR Dept=’RND’;

	5. List the name of employees who are not males.
	Solution:- SELECT Ename, Sex
	FROM EMPLOYEE
	WHERE sex!=’M’;

31.	Queries using DISTINCT, BETWEEN, IN, LIKE, IS NULL, ORDER BY, GROUP BY, HAVING
A.	Display the name of departments. Each department should be displayed once.

SOLUTION	SELECT DISTINCT(Dept)
	FROM EMPLOYEE;

B.	Find the name and salary of those employees whose salary is between 35000 and 40000.

SOLUTION	SELECT Ename, salary
	FROM EMPLOYEE
	WHERE salary BETWEEN 35000 and 40000;

C.	Find the name of those employees who live in guwahati, surat or jaipur city.

SOLUTION	SELECT Ename, city
	FROM EMPLOYEE
	WHERE city IN(‘Guwahati’,’Surat’,’Jaipur’);

D.	Display the name of those employees whose name starts with ‘M’.

SOLUTION	SELECT Ename
	FROM EMPLOYEE
	WHERE Ename LIKE ‘M%’;

E.	List the name of employees not assigned to any department.

SOLUTION	SELECT Ename
	FROM EMPLOYEE
	WHERE Dept IS NULL;

F.	Display the list of employees in descending order of employee code.

SOLUTION	SELECT *
	FROM EMPLOYEE
	ORDER BY ecode DESC;

G.	Find the average salary at each department.

SOLUTION	SELECT Dept, avg(salary)
	FROM EMPLOYEE
	group by Dept;

H.	Find maximum salary of each department and display the name of that department which has maximum salary more than 39000.

SOLUTION	SELECT Dept, max(salary)
	FROM EMPLOYEE
	group by Dept
	HAVING max(salary)>39000;

32.	Queries for Aggregate functions- SUM( ), AVG( ), MIN( ), MAX( ), COUNT( )
	a. Find the average salary of the employees in employee table. Solution:- SELECT avg(salary) FROM EMPLOYEE; **b. Find the minimum salary of a female employee in EMPLOYEE table.** Solution:- SELECT Ename, min(salary) FROM EMPLOYEE WHERE sex=’F’; **c. Find the maximum salary of a male employee in EMPLOYEE table.** Solution:- SELECT Ename, max(salary) FROM EMPLOYEE WHERE sex=’M’; **d. Find the total salary of those employees who work in Guwahati city.** Solution:- SELECT sum(salary) FROM EMPLOYEE WHERE city=’Guwahati’; **e. Find the number of tuples in the EMPLOYEE relation.** Solution:- SELECT count(*) FROM EMPLOYEE;

33.	Write a program to connect Python with MySQL using database connectivity and perform the following operations on data in database: Fetch, Update and delete the data.
A.	**CREATE A TABLE**

SOLUTION	import mysql.connector
	demodb = mysql.connector.connect(host="localhost", user="root",
	passwd="computer", database="EDUCATION")
	democursor=demodb.cursor( )
	democursor.execute("CREATE TABLE STUDENT (admn_no int primary key, sname varchar(30), gender char(1), DOB date, stream varchar(15), marks float(4,2))")

B.	**INSERT THE DATA**

SOLUTION	import mysql.connector
	demodb = mysql.connector.connect(host="localhost", user="root",
	passwd="computer", database="EDUCATION")
	democursor=demodb.cursor( )

	democursor.execute("insert into student values (%s, %s, %s, %s, %s, %s)", (1245, 'Arush', 'M', '2003-10-04', 'science', 67.34)) demodb.commit( )
C.	**FETCH THE DATA**

SOLUTION	import mysql.connector
	demodb = mysql.connector.connect(host="localhost", user="root",
	passwd="computer", database="EDUCATION")
	democursor=demodb.cursor( )
	democursor.execute("select * from student")
	for i in democursor:
	        print(i)

D.	**UPDATE THE RECORD**

SOLUTION	import mysql.connector
	demodb = mysql.connector.connect(host="localhost", user="root",
	passwd="computer", database="EDUCATION")

	democursor=demodb.cursor( )
	democursor.execute("update student set marks=55.68 where admn_no=1356")
	demodb.commit( )

E.	**DELETE THE DATA**

SOLUTION	import mysql.connector
	demodb = mysql.connector.connect(host="localhost", user="root", passwd="computer", database="EDUCATION")
	democursor=demodb.cursor( )
	democursor.execute("delete from student where admn_no=1356")
	demodb.commit( )