`x1 User De 3-06-24 fined Functions

Taylor's Series

1. Write a menu driven program using user defined functions to:

- To calculate the value of sin(x) using its Taylor's series expansion up to n terms.

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots,$$

- To generate random integer numbers within a given range.
- To generate n terms of the series 3, 12, 27, 48, 75, 108,
- To generate n terms of the series 8, 7, 11, 12, 14, 17, 17, 22,

Program

```
import math
import random

def fact(k):
   if k<=1:
      return 1
   else:
      return k*fact(k-1)
--
def sine_series():
   step=int(input("How many terms : "))
   x=int(input("Enter the value of x :"))
   sum=0
   for i in range(step+1):
      sum+=(math.pow(-1,i)*math.pow(x,2*i+1))/fact(2*i+1)
   print("The result of sin",'(', x, ')', "is :", sum)

def lottery():

   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)

def series1():
    n=int(input("How many numbers"))
    i=3
    j=1
    while j<=n:
        print(i*pow(j,2),end=' ')
        j+=1

def series2():

    i=8 n=int(input("Enter no.of terms:"))
    if n%2!=0:
        print("No of terms should be even ")
        n+=1
        print("New value of n is :",n)


    j=7
    t=2
    while t<=n:
        print(i,j,end=' ')
        i+=3
        j+=5
        t+=2

while True:
    print("Main Menu")
    print("1.Sine Series")
    print("2.Random Number Generation")
    print("3.Series1")
    print("4.Series2")
    print()

    ch=int(input("Enter the choice:"))
    if ch==1:
        print("Sum of the sine series")
```

```
        sine_series()
        print()
    elif ch==2:
        print("Random number generation")
        lottery()
        print()
    elif ch==3:
        print("Printing series1")
        series1()
        print()
    elif ch==4:
        series2()
        print()
    else:
        print("Invalid choice")
        break
```

Output

```
Main Menu
1.Sine Series
2.Random Number Generation
3.Series1
4.Series2

Enter the choice:1
Sum of the sine series
How many terms : 3
Enter the value of x :2
The result of sin ( 2 ) is : 0.9079365079365079

Main Menu
1.Sine Series
2.Random Number Generation

3.Series1
4.Series2

Enter the choice:2
Random number generation
```

Enter a number between 1 to 6 :3
Sorry, Try again, The lucky number was : 1

10-06-24 **User Defined Functions**

Perfect/Armstrong/Mersenne Prime numbers

2. Write a menu driven program in Python using user defined functions that takes a number n as parameter and returns the

- Perfect number
- Armstrong number
- Mersenne prime number

Note

A Perfect number is a positive integer, which is equal to the sum of its divisors.

If a 3 digit number is equal to the sum of the cubes of its each digit, then it is an Armstrong number.

Numbers in the form $2^n - 1$ are called Mersenne numbers.

A number that is divisible only by itself and 1 is a Prime number.

Pcrogram

```
def perfect_no(n):
   s=0
   for i in range(1,n):
      if n%i ==0:
```

```
            s+=i
        if s==n:
            print("The number ",n," is a perfect number")
        else:
            print("The number ",n," is not a perfect number")

def amstrong_no(n):
        s=
        temp=n
        while temp>0:
            digit=temp%10
            s+=digit**3
            temp//= 10
        if n==s:
            print("The number ",n," is an Armstrong number")
        else:
            print("The number ",n," is not an Armstrong number")

    def mersenne_prime(n):
        for a in range(1,n+1):
            mernum=2**a-1
            mid=int(mernum/2)+1
            for b in range(2,mid):
                if mernum%b==0:
                    print(mernum)
                    break
            else:
                print(mernum,"\t Prime")

    while True:
        print("Main Menu")
        print("1.Perfect Number Checking")
        print("2.Amstrong Number Checking")
        print("3.Mersenne Prime Numbers")
        print("4.Exit")
        print()

        ch=int(input("Enter the choice:"))
        if ch==1:
            print("Perfect number checking....")
```

```
        n=int(input("Enter the number n:"))
        perfect_no(n)
        print()
    elif ch==2:
        print("Armstrong number checking....")
        n=int(input("Enter the number n:"))
        amstrong_no(n)
        print()
    elif ch==3:
        print("Mersenne Prime Number")
        n=int(input("Enter the number n:"))
        mersenne_prime(n)
        print()
    else:
        print("Invalid choice")
        break
```

Output

```
Main Menu
1.Perfect Number Checking
2.Amstrong Number Checking
3.Mersenne Prime Numbers
```

```
4.Exit

Enter the choice:3
Mersenne Prime Number
Enter the number n:10
1       Prime
3       Prime
7       Prime
15
31      Prime
63
127     Prime
255
511
1023
```

24-06-24 **User Defined Functions**

Lists-Maximum/Minimum/Sum of the elements

3. Write a menu driven program in Python using user defined functions that takes a list as parameter and returns the

- Maximum
- Minimum
- Sum of the elements

Note : Do not use built in functions to find the maximum and minimum.

Program

```
lst=eval(input("Enter a list"))
l=len(lst)
```

```
def maximum(lst):
    max=lst[0]
    for i in range(l):
        if lst[i]> max:
            max=lst[i]
    return max

def minimum(lst):
    min=lst[0]
+
    for i in range(l):
        if lst[i]< min:
            min=lst[i]
    return min

def sum_of_elements(lst):
    s=0
    for i in range(l):
        s+=lst[i]
    return s

def main():

    while True:
        print("Main Menu")
        print("1.To find the Maximum")
        print("2.To find the Minimum")
        print("3.To find the Sum of the elements")

        ch=int(input("Enter the choice:"))
        if ch==1:
            print("Finding the maximum")
            ma=maximum(lst)
            print("Maximum value is",ma)
            print()
        elif ch==2:
            print("Finding the minimum")
            mi=minimum(lst)
            print("Minimum value is",mi)
```

```
            print()
        elif ch==3:
            print("Sum of the elements")
            s=sum_of_elements(lst)
            print("Sum=",s)
            print()
        else:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

```
Enter a list1,2,3,4,5
Main Menu
1.To find the Maximum
2.To find the Minimum
3.To find the Sum of the elements
Enter the choice:1
Finding the maximu
Maximum value is 5

Main Menu
1.To find the Maximum
2.To find the Minimum
3.To find the Sum of the elements
Enter the choice:2
Finding the minimum
Minimum value is 1

Main Menu
1.To find the Maximum
2.To find the Minimum
3.To find the Sum of the elements
Enter the choice:3
Sum of the elements
Sum= 15

Main Menu
1.To find the Maximum
2.To find the Minimum
3.To find the Sum of the elements
Enter the choice:4
Invalid choice
```

01-07-24 **User Defined Functions**

Strings-Palindrome check/Count occurrences of a given character /Reverse

4. Write a menu driven program in Python using user defined functions to take a string as input and

- Check if it is palindrome
- Count number of occurrences of a given character
- Reverse it

Program

```
def palindrome(s):z
    l=len(s)
    p=l-1
    index=0
    while index<p:
        if(s[index]==s[p]):
            index+=1
            p-=1
        else:
            print("String is not a palindrome")
            break
```

```
    else:
        print("String is a palindrome")

def count_for_char(s):
    character=input("Enter any character to search for:")
    count = 0
    for i in s:
        if i == character:
            count = count + 1
    return count

def reverse(s):
  str = ""
  for i in s:
    str = i + str
  return str

def main():
    while True:
        print("String Operations")
        print("1.Palindrome Check")
        print("2.Counting for occurrence of a char")
        print("3.Reversing a string")
        print("4.Quit")

        ch=int(input("Enter the choice"))
                if ch==1:
                    s=input("Enter the string:")
                    palindrome(s)
                elif ch==2:
                    s=input("Enter the string:")
                    c=count_for_char(s)
                    print ("Count of character in string is : "+str(c))
                elif ch==3:
                    s=input("Enter the string:")
                    rev_string=reverse(s)
                    print("Reversed string is :",rev_string)
                else:
                    print("Invalid choice")
                    break
```

```
if __name__ == '__main__':
    main()
```

Output

```
String Operations
1.Palindrome Check
2.Counting for occurrence of a char
3.Reversing a string
4.Quit

Enter the choice3
Enter the string:I am a student
Reversed string is : tneduts a ma I

String Operations
1.Palindrome Check
2.Counting for occurrence of a char
3.Reversing a string
4.Quit

Enter the choice2
Enter the string:madam
Enter any character to search for:a
Count of character in string is : 2
```

01-07-24 **User Defined Functions**

Strings-Replacing the string at an index / Words starting with vowel

5. Write a menu driven program in Python using user defined functions to take a string as input and

- Get an index from user and replace the string at that index with user given value.
- Count and display the occurrences of words starting with a vowel in the given string.

Program

```
def Convert(string):
    li = list(string.split(" "))
    return li

def vowcount(string):
    c=0
    word=string.split()
    for i in word:
        if i[0] in 'aeiou' or i[0] in 'AEIOU':
            c+=1
            print(i)
```

```
    return c

# Driver code
while True:
    print("String Operations")
    print("1.replacing a string at an index")
    print("2.Occurrences of words starting with vowels")
    print("3.Quit")

    ch=input("Enter the choice:")
    if ch=='1':
        str1=input("Enter any string:")
        li=Convert(str1)
        n=len(li)
        index=int(input("Enter index:"))
        x=input("Enter the word to be replaced with:")
        for i in range(n):
            if index==i:
                li[i]=x
        print(li)
        str2=' '.join(li)
        print(str2)
    elif ch=='2':
        s=input("Enter the string:")
        c=vowcount(s)
        print("No.of words starting with a vowel:",c)
    else:
        print("Invalid choice")
        break
```

Output

```
String Operations

1.replacing a string at an index
2.Occurrences of words starting with vowels
3.Quit
Enter the choice:1
Enter any string:I am a student
Enter index:2
```

```
Enter the word to be replaced with:AM
['I', 'am', 'AM', 'student']
I am AM student

String Operations

1.replacing a string at an index
2.Occurrences of words starting with vowels
3.Quit
Enter the choice:2
Enter the string:I am a student
I
am
a
No.of words starting with a vowel: 3

String Operations

1.replacing a string at an index
2.Occurrences of words starting with vowels
3.Quit
Enter the choice:3
Invalid choice
```

08-07-24 **Text Files**

Creation/UC/LC/Digits/no.of characters

6. Write a menu driven program in Python using function to a read a text file and

- Count number of characters
- Count number of upper case characters
- Count number of lower case characters
- Count number of digits

Program

```
#file creation
f=open("count.txt","w")
f.write("#We are writing-\n")
f.write("data to a 1\n")
f.write("text file $\n")
print("Data written to the file successfully\n")
f.close()

def Count_CHAR():
   ccount=0
   f=open("count.txt",'r')
```

```
    if f.mode=='r':
        str=f.read().replace(" ","")
        print(str)
        ccount=len(str)

        print("No.of Characters excluding white spaces:",ccount)
    f.close()

def Count_UC():
    upper=0
    f=open("count.txt",'r')
    if f.mode=='r':

        str=f.read()
        print(str)

        l=len(str)
        for i in range(l):
            if str[i]>= chr(65) and str[i]<= chr(90):
                upper+=1

        print("No.of upper case letters :",upper)
    f.close()

def Count_LC():
    lower=0
    f=open("count.txt",'r')
    if f.mode=='r':
        str=f.read()
        print(str)

        l=len(str)
        for i in range(l):
            if str[i]>= chr(97) and str[i]<= chr(122):
                lower+=1

        print("No.of lower case letters :",lower)
    f.close()

def Count_DG():
```

```
    digits=0
    f=open("count.txt",'r')
    if f.mode=='r':
        str=f.read()
        print(str)

        l=len(str)
        for i in range(l):
            if str[i]>= chr(48) and str[i]<= chr(57):
                digits+=1

        print("No.of digits :",digits)
    f.close()

def main():
    while True:
        print("File Operations")
        print("1.Counting number of characters excluding white spaces")
        print("2.Counting UC letters")
        print("3.Counting LC letters")
        print("4.Counting Digits ")
        print("5.Quit")

        ch=int(input("Enter the choice:"))
        if ch==1:
            Count_CHAR()
        elif ch==2:
            Count_UC()
        elif ch==3:
            Count_LC()
        elif ch==4:
            Count_DG()
        elif ch==5:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

Data written to the file successfully

File Operations
1.Counting number of characters excluding white spaces
2.Counting UC letters
3.Counting LC letters
4.Counting Digits
5.Quit
Enter the choice:1
#Wearewriting-
datatoa1
textfile$

No.of Characters excluding white spaces: 34

File Operations
1.Counting number of characters excluding white spaces
2.Counting UC letters
3.Counting LC letters
4.Counting Digits
5.Quit
Enter the choice:2
#We are writing-
data to a 1
text file $

No.of upper case letters : 1

08-07-24 **Text Files**

Counting Words/Vowels/Special Characters/Lines

7. Write a menu driven program in Python using function to a read a text file and

- Count number of words
- Count number of vowels
- Count number of lines
- Count number of special characters

Program

```
#file creation
f=open("count.txt","w")
f.write("#We are writing-\n")
f.write("data to a 1\n")
f.write("text file $\n")
print("Data written to the file successfully\n")
f.close()

def Count_Words():
```

```
    f=open("count.txt",'r')
    str=f.read()
    print(str)
    L=str.split()
    count_words=0
    for i in L:
        count_words=count_words+1
    print("No.of words:",count_words)
    f.close()

def Count_Lines():
    f=open("count.txt",'r')
    str=f.readlines()
    print(str)
    count_line=0
    for i in str:
        count_line=count_line+1
    print("No.of lines:",count_line)
    f.close( )

def Count_Vowels():
    f=open("count.txt",'r')
    str=f.read()
    print(str)
    count=0
    for i in str:
        if i in 'aeiou' or i in 'AEIOU':
            count=count+1
    print("No.of vowels:",count)
    f.close( )

def Count_SC():
    f=open("count.txt",'r')
    str=f.read()
    print(str)
    sc=0
    l=len(str)
    for i in range(l):
        if not str[i].isalnum() and not str[i].isspace():
            sc += 1
```

```
    print("No.of special characters:",sc)
    f.close()

def main():
    while True:
        print("File Operations")
        print("1.Counting the number of words")
        print("2.Counting the number of lines")
        print("3.Counting the number of vowels")
        print("4.Counting the number of special characters")
        print("5.Quit")

        ch=int(input("Enter the choice:"))
        if ch==1:
            Count_Words()
        elif ch==2:
            Count_Lines()
        elif ch==3:
            Count_Vowels()
        elif ch==4:
            Count_SC()
        elif ch==5:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

```
Data written to the file successfully

File Operations
1.Counting the number of words
2.Counting the number of lines
3.Counting the number of vowels
4.Counting the number of special characters
5.Quit
Enter the choice:1
```

```
#We are writing-
data to a 1
text file $

No.of words: 10

File Operations
1.Counting the number of words
2.Counting the number of lines
3.Counting the number of vowels
4.Counting the number of special characters
5.Quit
Enter the choice:2
['#We are writing-\n', 'data to a 1\n', 'text file $\n']
No.of lines: 3

File Operations
1.Counting the number of words
2.Counting the number of lines
3.Counting the number of vowels
4.Counting the number of special characters
5.Quit
Enter the choice:3
#We are writing-
data to a 1
text file $

No.of vowels: 12
```

15-07-24 **Text Files**

Counting longest word/words starting with UC letters/number of occurrences of a word

8. Write a menu driven program in Python using function to a read a text file and

- Find the longest word
- Find the words starting with UC letters
- Count number of occurrences of a word

Program

```
#file creation
f=open("Book.txt","w")
f.write("We are writing\n")
f.write("data to a \n")
f.write("text file \n")
f.write("Welcome to the world of Programmers\n")
f.close()

f=open("Book.txt",'r')
```

```
str=f.read()
print(str)

def longest_word(filename):
    with open(filename,'r') as infile:
        words=infile.read()
        L=words.split()
        print(words)
        print()
    max_len=len(max(L))
    for word in L:
        if len(word)==max_len:
            print(word)
    infile.close()

def UCWord():
    f=open("Book.txt",'r')
    str=f.read()
    L=str.split()
    print(str)
    print()
    for word in L:
        if word[0].isupper():
            print(word)
    f.close()

def count_word():
    f=open("Book.txt",'r')
    str=f.read()
    L=str.split()
    print(str)
    print()
    count=0
    ch=input("Enter the word to search for:")
    for i in L:
        if i==ch:
            count=count+1
    print(ch,"occurs",count,"times")
    f.close()
```

```
def main():
    while True:
        print("File Operations")
        print("1.Longest word in the file")
        print("2.Words starting with UC letters")
        print("3.Counting for occurrence of a word")
        print("4.Quit")
        print()

        ch=int(input("Enter the choice:"))
        if ch==1:
            longest_word('Book.txt')
            print()
        elif ch==2:
            UCWord()
            print()
        elif ch==3:
            count_word()
            print()
        elif ch==4:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

```
We are writing
data to a
text file
Welcome to the world of Programmers

File Operations
1.Longest word in the file
2.Words starting with UC letters
3.Counting for occurrence of a word
4.Quit
```

```
Enter the choice:3
We are writing
data to a
text file
Welcome to the world of Programmers


Enter the word to search for:to
to occurs 2 times

File Operations
1.Longest word in the file
2.Words starting with UC letters
3.Counting for occurrence of a word
4.Quit
```

15-07-24 **Text Files**

Display each word separated by a # / Remove all the lines that contain the character 'a' / Printing words in reverse order

9. Write a menu driven program in Python using function to a read a text file and

- Display each word separated by a #
- Remove all the lines that contain the character 'a' in that file and write other lines into file.
- Print only word starting with 'I' in reverse order

Program

```
#file creation
f=open("file.txt","w")
f.write("We are writing\n")
f.write("data to a\n")
f.write("text file \n")
f.write("Iam an Indian\n")
print("Data written to the file successfully\n")
f.close()
```

```
def sep_hash():

    myfile=open("file.txt","r")
    line=" "
    while line:
        line=myfile.readline()
        for word in line.split():
            print(word,end='#')
            print()
    myfile.close()

def remove_a_lines():
    f1=open("file.txt","r")
    f2=open("file_new.txt","w")
    line=" "
    while line:
        line=f1.readline()
        if 'a' not in line:
            f2.write(line)
    f1.close()
    f2.close()
    f2=open("file_new.txt","r")
    line=" "
    while line:
        line=f2.readline()
        print(line)
    f2.close()

def revtext():
    f=open("file.txt","r")
    s=" "
    while True:
        d=f.readline()
        if not d:
            break
        else:
            m=d.split()
            for i in m:
                if i[0]=='i' or i[0]=='I':
                    s+=" "+(i[::-1])
```

```
            else:
                s+=" "+i
        print(s)
        s=" "

def main():
    while True:
        print("File Operations")
        print("1.Separate using#")
        print("2.Removing a")
        print("3.Reversing word")
        print("4.Quit")

        ch=int(input("Enter the choice:"))
        if ch==1:
            sep_hash()
            print()
        elif ch==2:
            remove_a_lines()
            print()
        elif ch==3:
            revtext()
            print()
        elif ch==4:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

```
Data written to the file successfully

File Operations
1.Separate using#
2.Removing a
3.Reversing word
4.Quit
Enter the choice:1
We#
```

```
are#
writing#
data#
to#
a#
text#
file#
Iam#
an#
Indian#

File Operations
1.Separate using#
2.Removing a
3.Reversing word
4.Quit
Enter the choice:2
text file

File Operations
1.Separate using#
2.Removing a
3.Reversing word
4.Quit
Enter the choice:3
We are writing
data to a
text file
maI an naidnI
```

22-07-24 **Binary Files**

Creation/Display/Searching using Lists

10. Write a menu driven program in Python using Pickle library and create a binary file with following structure:

    - Admission number
    - Student name
    - Age

- Display the contents of the binary file

- Display the student whose age is above user given value
- Search a student by admission number given by user

Program

```
import pickle
def create_file():

    record=[]
    while True:
        admno=int(input("Enter the admission number:"))
        name=input("Enter name:")
        age=int(input("Enter age:"))
        data=[admno,name,age]
        record.append(data)
```

```
        ch=input("Want to enter more records?")
        if ch.upper()=='N':
            break

    f=open("Student.dat","wb")
    pickle.dump(record,f)
    print("record added")
    f.close()

def disp_file():
    f=open("Student.dat","rb")
    stud_rec=pickle.load(f) # to read the object from the binary file
    print("Contents of the file are:")
    #reading the fields
    for i in stud_rec:
        admno=i[0]
        name=i[1]
        age=i[2]
        print(admno,name,age)
    f.close()

def search_admno():
    f=open("Student.dat","rb")
    stud_rec=pickle.load(f)
    found=0
    admno=int(input("Enter the admission number to search for:"))

    for i in stud_rec:
        if i[0]==admno:
            print("Search Successful",i[1],"found")
            found=1
            break
    if found==0:
        print("Record not found")
    f.close()

def search_age():
    f=open("Student.dat","rb")
    stud_rec=pickle.load(f)
    found=0
```

```
    age=int(input("Enter the age to search for:"))

    for i in stud_rec:
        if i[2]==age:
            print(i[1])
            found=1
    if found==0:
        print("Record not found")
    f.close()

def main():
    while True:
        print("File Operations")
        print("1.File Creation")
        print("2.File Display")
        print("3.Search for an admission number")
        print("4.Search for age")
        print("5.Quit")
        print()

        ch=int(input("Enter the choice:"))
        if ch==1:
            create_file()
            print()
        elif ch==2:
            disp_file()
            print()
        elif ch==3:
            search_admno()
            print()
        elif ch==4:
            search_age()
        elif ch==5:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

File Operations
1.File Creation
2.File Display
3.Search for an admission number
4.Search for age
5.Quit

Enter the choice:2
Contents of the file are:
1 Anil 15
2 Akash 16

File Operations
1.File Creation
2.File Display
3.Search for an admission number
4.Search for age
5.Quit

Enter the choice:4
Enter the age to search for:15
Anil

22-07-24 **Binary Files**

Append/Search/Update using Dictionaries

11. Write a menu driven program in Python using Pickle library and create a binary file with following structure using a dictionary.

    - Travelid
    - From
    - To

- Append data to the file

- Display the contents of the binary file
- Searching based on Travelid
- Update a record based on Travelid

`Q`

`

Program

```
#Accepting data for a dictionary
def insertrec():
    Travelid=int(input("Enter the travel id"))
    From=input("From:")
    To=input("To:")
    #Creating the dictionary
    rec={"Travelid":Travelid,"From":From,"To":To}
    #Writing to the dictionary
```

```
    f=open("Travel.dat","ab")
    pickle.dump(rec,f)
    f.close()

#Reading the records
def readrec():
    f=open("Travel.dat","rb")
    while True:
        try:
            rec=pickle.load(f)
            print("Travel id:",rec['Travelid'])
            print("From:",rec['From'])
            print("To:",rec['To'])
        except EOFError:
            break
    f.close()

#Searching for a record based on travel id
def search_tid(tid):
    f=open("Travel.dat","rb")
    flag=False
    while True:
        try:
            rec=pickle.load(f)
            if rec['Travelid']==tid:
                print("Travelid:",rec['Travelid'])
                print("From:",rec['From'])
                print("To:",rec['To'])
                flag=True
        except EOFError:
            break
    if flag==False:
        print("No record found")
    f.close()

#Modification of tid
def update_tid(tid,ntid):
    f=open("Travel.dat","rb")
    reclist=[]
    flag=False
```

```
    while True:
        try:
            rec=pickle.load(f)
            reclist.append(rec)
        except EOFError:
            break
    f.close()
    for i in range(len(reclist)):
        if reclist[i]['Travelid']==tid:
            reclist[i]['Travelid']=ntid
            flag=True
    f=open("Travel.dat",'wb')
    for x in reclist:
        pickle.dump(x,f)
    if flag==False:
        print("Record not found")
    f.close()

#__main__
import pickle
while True:
    print("Main Menu")
    print("1.Insert Record")
    print("2.Display Record")
    print("3.Search Record")
    print("4.Update Record")
    print("0.Exit")
    print("Enter your choice:")
    ch=int(input("Enter choice:"))
    if ch==0:
        break
    elif ch==1:
        insertrec()
    elif ch==2:
        readrec()
    elif ch==3:
        tid=int(input("Enter the travel id to search:"))
        search_tid(tid)
    elif ch==4:
        tid=int(input("Enter a travel id to update:"))
```

```
ntid=int(input("Enter the new travel id:"))
update_tid(tid,ntid)
```

Output

```
Main Menu
1.Insert Record
2.Display Record
3.Search Record
4.Update Record
0.Exit

Enter your choice:
Enter choice:3
Enter the travel id to search:10

Travelid: 10
From: Chennai
To: Delhi
```

29-07-24 **Binary Files**

Append/Search/Delete using Dictionaries

12. Write a menu driven program in Python using Pickle library and create a binary file with following structure using a dictionary.

    - Travelid
    - From
    - To

- Append data to the file
- Display the contents of the binary file
- Searching based on Travelid
- Delete a record based on Travelid

Program

```
#Accepting data for a dictionary
def insertrec():
    Travelid=int(input("Enter the travel id"))
    From=input("From:")
    To=input("To:")
    #Creating the dictionary
    rec={"Travelid":Travelid,"From":From,"To":To}
    #Writing to the dictionary
```

```
    f=open("Travel.dat","ab")
    pickle.dump(rec,f)
    f.close()

#Reading the records
def readrec():
    f=open("Travel.dat","rb")
    while True:
        try:
            rec=pickle.load(f)
            print("Travel id:",rec['Travelid'])
            print("From:",rec['From'])
            print("To:",rec['To'])
        except EOFError:
            break
    f.close()

#Searching for a record based on travel id
def search_tid(tid):
    f=open("Travel.dat","rb")
    flag=False
    while True:
        try:
            rec=pickle.load(f)
            if rec['Travelid']==tid:
                print("Travelid:",rec['Travelid'])
                print("From:",rec['From'])
                print("To:",rec['To'])
                flag=True
        except EOFError:
            break
    if flag==False:
        print("No record found")
    f.close()

def deleterec(tid):
    f=open("Travel.dat","rb")
    reclist=[]
    flag=False
    while True:
```

```
        try:
            rec=pickle.load(f)
            reclist.append(rec)
        except EOFError:
            break
    f.close()
    f=open("Travel.dat","wb")
    for x in reclist:
        if x['Travelid']==tid:
            flag=True
            continue
        pickle.dump(x,f)

    if flag==False:
        print("Record not found")
    f.close()
#__main__
import pickle
while True:
    print("1.Insert Record")
    print("2.Display Record")
    print("3.Search Record")
    print("4.Delete Record")
    print("0.Exit")
    print("Enter your choice:")
    ch=int(input("Enter choice:"))
    if ch==0:
        break
    elif ch==1:
        insertrec()
    elif ch==2:
        readrec()
    elif ch==3:
        tid=int(input("Enter the travel id to search:"))
        search_tid(tid)
    elif ch==4:
        tid=int(input("Enter a travel id to delete:"))
        deleterec(tid)
```

Output

1.Insert Record
2.Display Record
3.Search Record
4.Delete Record
0.Exit
Enter your choice:
Enter choice:2
Travel id: 12
From: Kerala
To: Chennai

1.Insert Record
2.Display Record
3.Search Record
4.Delete Record
0.Exit
Enter your choice:
Enter choice:3
Enter the travel id to search:10
No record found

1.Insert Record
2.Display Record
3.Search Record
4.Delete Record
0.Exit
Enter your choice:
Enter choice:4
Enter a travel id to delete:11
Record not found

29-07-24 **CSV Files**

Calculate Total/Percentage/Search

13. Write a menu driven program in Python to create a CSV file with following data

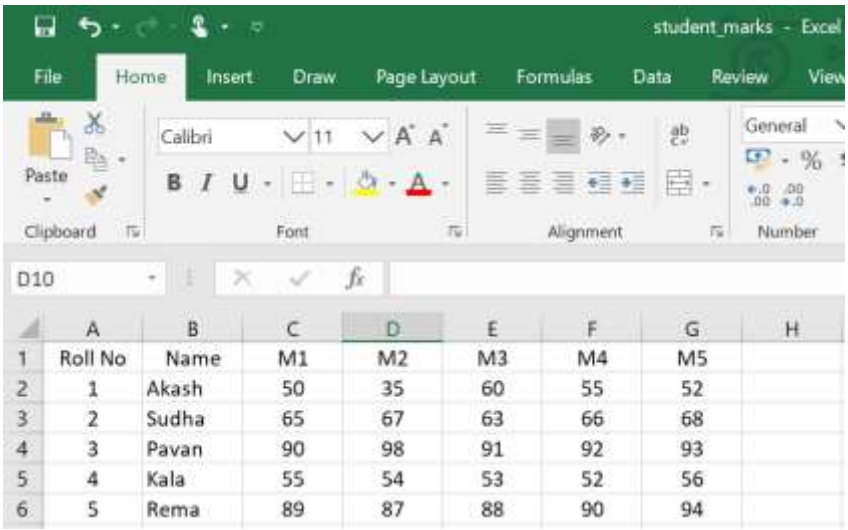
- Roll no
- Name of student
- Mark in Sub1

- Mark in sub2
- Mark in sub3
- Mark in sub4
- Mark in sub5

Perform following operations on the CSV file after reading it.

- Calculate total and percentage for each student.
- Display the name of student if in any subject marks are greater than 80% (Assume marks are out of 100)

Sample Excel file



	A	B	C	D	E	F	G
1	Roll No	Name	M1	M2	M3	M4	M5
2	1	Akash	50	35	60	55	52
3	2	Sudha	65	67	63	66	68
4	3	Pavan	90	98	91	92	93
5	4	Kala	55	54	53	52	56
6	5	Rema	89	87	88	90	94

<u>Program</u>

```
import csv
def search_rollno():
   f=open("C:\\Users\\Udhaya
Khumari\\AppData\\Local\\Programs\\Python\\Python36-
32\\student_marks.csv",'r')
   csv_reader=csv.reader(f)
   found=0
   rno=input("Enter the roll number to be searched for:")
```

```
    for row in csv_reader:
        if row[0] == rno:
            found=1
            print(row)
    if found==0:
        print("Record not found")
    f.close()

def calc_perc():
    f=open(r"C:\Users\Udhaya
Khumari\AppData\Local\Programs\Python\Python36-
32\student_marks.csv",'r')
    csv_reader=csv.reader(f)
    s=0
    for row in csv_reader:
        if csv_reader.line_num==1:
            continue
        s=0
        s=s+int(row[2])+int(row[3])+int(row[4])+int(row[5])+int(row[6])
        perc=s/5
        print("Sum=",s," Percentage=",perc)
    f.close()

def disp_name():
    f=open(r"C:\Users\Udhaya
Khumari\AppData\Local\Programs\Python\Python36-
32\student_marks.csv",'r')
    csv_reader=csv.reader(f)
    found=0
    s=0
    for row in csv_reader:
        if csv_reader.line_num==1:
            continue
        s=0
        s=s+int(row[2])+int(row[3])+int(row[4])+int(row[5])+int(row[6])
        perc=s/5
        if perc>80:
            print("Name:",row[1])
            found=1
    if not found:
```

```
        print("No student has got above 80 percentage")
    f.close()

def main():
    while True:
        print("Main Menu")
        print("1.Search for a roll number")
        print("2.Calculate percentage")
        print("3.Display names")
        print("4.Exit")
        ch=int(input("Enter choice:"))

        if ch==1:
            search_rollno()
        elif ch==2:
            calc_perc()
        elif ch==3:
            disp_name()
        elif ch==4:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

```
Main Menu
1.Search for a roll number
2.Calculate percentage
3.Display names
4.Exit
Enter choice:1
Enter the roll number to be searched for:3
['3', 'Pavan', '90', '98', '91', '92', '93']
```

```
Main Menu
1.Search for a roll number
2.Calculate percentage
3.Display names
4.Exit
Enter choice:2
Sum= 252  Percentage= 50.4
Sum= 329  Percentage= 65.8
Sum= 464  Percentage= 92.8
Sum= 270  Percentage= 54.0
Sum= 448  Percentage= 89.6
Main Menu
1.Search for a roll number
2.Calculate percentage
3.Display names
4.Exit
Enter choice:3
Name: Pavan
Name: Rema
Main Menu
1.Search for a roll number
2.Calculate percentage
3.Display names
4.Exit
Enter choice:4
Invalid choice
```

05-08-24 **CSV Files**

Creation/Counting number of records

14. Write a menu driven program to write data onto a CSV file

- Using writerow() method (Games1.csv)
- Using writerows() method (Games2.csv)
- To count the exact number of records present in the csv file Games1.csv excluding the header.

Games1 and Games2 records:

Gcode	Gamename	Number	Prizemoney	Scheduledate
101	Kabadi	2	5000	23-Jan-07
102	Badminton	2	12000	12-Dec-13
103	Table Tennis	4	8000	14-Feb-14
105	Chess	2	9000	01-Jan-15
108	Table Tennis	4	25000	19-Mar-14

Program

```
import csv
def using_writerow():
    #field names
    fields=['Gcode','Gamename','Number','Prizemoney','Scheduledate']
    #data rows of csv files
    rows=[
        ['101','Kabadi','2','5000','23-Jan-2007'],
        ['102','Badminton','2','12000','12-Dec-2013'],
```

```
        ['103','Table Tennis','4','8000','14-Feb-2014'],
        ['105','Chess','2','9000','01-Jan-2015'],
        ['108','Table Tennis','4','25000','19-Mar-2014']]

    fname="D:/Games1.csv"
    with open(fname,'w',newline='') as f:
        #by default, newline is '\r\n'
        #creating a csv writer object
        csv_w=csv.writer(f,delimiter=',')
        #writing the fields once
        csv_w.writerow(fields)
        #writing all the rows in one go
        csv_w.writerows(rows)
        print("All rows written in one go")

def using_writerows():
    #field names
    fields=['Gcode','Gamename','Number','Prizemoney','Scheduledate']
    #data rows of csv files
    rows=[
        ['101','Kabadi','2','5000','23-Jan-2007'],
        ['102','Badminton','2','12000','12-Dec-2013'],
        ['103','Table Tennis','4','8000','14-Feb-2014'],
        ['105','Chess','2','9000','01-Jan-2015'],
        ['108','Table Tennis','4','25000','19-Mar-2014']]

    fname="D:/Games2.csv"
    with open(fname,'w',newline='') as f:
        #by default, newline is '\r\n'
        #creating a csv writer object
        csv_w=csv.writer(f,delimiter=',')
        #writing the fields once
        csv_w.writerow(fields)
        for i in rows:
            #writing the data rowwise
            csv_w.writerow(i)
        print("File created")

def no_of_rows():
    f=open("D:/Games1.csv",'r')
```

```
    csv_reader=csv.reader(f)
    value=0
    for row in csv_reader:
        if csv_reader.line_num == 1: # skip first row
            continue
        print(row)
        value=value+1
    print("\n No.of records :",value)
    f.close()

def main():
    while True:
        print("Main Menu")
        print("1.Writing using writerow")
        print("2.Writing using writerows")
        print("3.Counting the no.of lines")
        print("4.Exit")
        print()

        ch=int(input("Enter choice:"))

        if ch==1:
            using_writerow()
            print()
        elif ch==2:
            using_writerows()
            print()
        elif ch==3:
            no_of_rows()
            print()
        elif ch==4:
            print("Invalid choice")
            break

if __name__ == '__main__':
    main()
```

Output

```
Main Menu
```

```
1.Writing using writerow
2.Writing using writerows
3.Counting the no.of lines
4.Exit

Enter choice:1
All rows written in one go

Main Menu
1.Writing using writerow
2.Writing using writerows
3.Counting the no.of lines
4.Exit

Enter choice:2
File created

Main Menu
1.Writing using writerow
2.Writing using writerows
3.Counting the no.of lines
4.Exit
Enter choice:3
['101', 'Kabadi', '2', '5000', '23-Jan-2007']
['102', 'Badminton', '2', '12000', '12-Dec-2013']
['103', 'Table Tennis', '4', '8000', '14-Feb-2014']
['105', 'Chess', '2', '9000', '01-Jan-2015']
['108', 'Table Tennis', '4', '25000', '19-Mar-2014']

 No.of records : 5
```

05-08-24 **CSV Files**

Add/Modify/Delete/Search/Display

15. Write a menu driven program to write data onto a CSV file with the following data:

- Rollno of the student
- Name of student

- Marks of the student

Perform following operations on the CSV file:

- Add a new Record
- Modify Existing Record
- Delete Existing Record
- Search a Record
- List all Records

Program

```
import os
import csv
def addrecord():
    print("Add a new Record")
    print("================")
    f=open('students.csv','a',newline='\r\n')
    s=csv.writer(f)
```

```
    rollno=int(input('Enter rollno='))
    name=input('Enter name=')
    marks=float(input('Enter marks='))
    rec=[rollno,name,marks]
    s.writerow(rec)
    f.close()
    print("Record Saved")
    input("Press any key to continue..")

def modifyrecord():
    print("Modify a Record")
    print("================")
    f=open('students.csv','r',newline='\r\n')
    f1=open('temp.csv','w',newline='\r\n')
    f1=open('temp.csv','a',newline='\r\n')
    r=input('Enter rollno you want to modify')
    s=csv.reader(f)
    s1=csv.writer(f1)
    for rec in s:
        if rec[0]==r:
            print("Rollno=",rec[0])
            print("Name=",rec[1])
            print("Marks=",rec[2])
            choice=input("Do you want to modify this record(y/n)")
            if choice=='y' or choice=='Y':
                rollno=int(input('Enter New rollno='))
                name=input('Enter new name=')
                marks=float(input('Enter new marks='))
                rec=[rollno,name,marks]
                s1.writerow(rec)
                print("Record Modified")
            else:
                s1.writerow(rec)
        else:
            s1.writerow(rec)
    f.close()
    f1.close()
    os.remove("students.csv")
    os.rename("temp.csv","students.csv")
```

```
    input("Press any key to continue..")

def deleterecord():
    f=open('students.csv','r',newline='\r\n')
    f1=open('temp.csv','w',newline='\r\n')
    f1=open('temp.csv','a',newline='\r\n')
    r=input('Enter rollno you want to delete')
    s=csv.reader(f)
    s1=csv.writer(f1)
    for rec in s:
        if rec[0]==r:
            print("Rollno=",rec[0])
            print("Name=",rec[1])
            print("Marks=",rec[2])
            choice=input("Do you want to delete this record(y/n)")
            if choice=='y' or choice=='Y':
                pass
                print("Record Deleted")
            else:
                s1.writerow(rec)
        else:
             s1.writerow(rec)
    f.close()
    f1.close()
    os.remove("students.csv")
    os.rename("temp.csv","students.csv")

    input("Press any key to continue..")

def search():
    print("Search a Record")
    print("====================")
    f=open('students.csv','r',newline='\r\n') #Remove new line character
from output
    r=input('Enter rollno you want to search')
    s=csv.reader(f)
    for rec in s:
        if rec[0]==r:
            print("Rollno=",rec[0])
            print("Name=",rec[1])
```

```
            print("Marks=",rec[2])
    f.close()
    input("Press any key to continue..")

def viewall():
    print("List of All Records")
    print("===================")
    f=open('students.csv','r',newline='\r\n') #Remove new line character
from output
    s=csv.reader(f)
    i=1
    for rec in s:
        print(rec[0],end="\t\t")
        print(rec[1],end="\t\t")
        print(rec[2])
        i+=1
    f.close()
    input("Press any key to continue..")

def mainmenu():
    choice=0
    while choice!=6:
        print("\n")
        print("Main Menu")
        print("==========")
        print("1. Add a new Record")
        print("2. Modify Existing Record")
        print("3. Delete Existing Record")
        print("4. Search a Record")
        print("5. List all Records")
        print("6.Exit")
        choice=int(input('Enter your choice'))
        if choice==1:
            addrecord()
        elif choice==2:
            modifyrecord()
        elif choice==3:
            deleterecord()
        elif choice==4:
            search()
```

```
        elif choice==5:
            viewall()
        elif choice==6:
            print("Software Terminated")
            break
mainmenu()
```

Output

```
Main Menu
==========
1. Add a new Record
2. Modify Existing Record
3. Delete Existing Record
4. Search a Record
5. List all Records
6.Exit
```

```
Enter your choice5
List of All Records
===================
5          Sanu       300.0
2          Denny      100.0
1          Anil       200.0
Press any key to continue..


Main Menu
==========
1. Add a new Record
2. Modify Existing Record
3. Delete Existing Record
4. Search a Record
5. List all Records
6.Exit
Enter your choice3
Enter rollno you want to delete1
Rollno= 1
Name= Anil
Marks= 200.0
Do you want to delete this record(y/n)y
Record Deleted
Press any key to continue..
```

12-08-24 **Data Structures**

Stack Operations

Push and Pop

16. Write a menu driven program to perform basic stack operations PUSH and POP using lists and display the elements of the stack. Also check for the conditions of stack empty and peek.

Program

```
#Implementation of a list as a stack
def isEmpty(stk):
    if stk==[]:
        return True
    else:
        return False
```

```
def push(stk,item):
    stk.append(item)
    top=len(stk)-1

def pop(stk):
    if isEmpty(stk):
        return "Underflow"
    else:
        item=stk.pop()
        if len(stk)==0:
            top=None
        else:
            top=len(stk)-1
        return item

def Peek(stk):
    if isEmpty(stk):
        return "Underflow"
    else:
        top=len(stk)-1
        return stk[top]

def display(stk):
    if isEmpty(stk):
        print("Stack empty")
    else:
        top=len(stk)-1
        print(stk[top],"<-top")
        for a in range(top-1,-1,-1):
            print(stk[a])

#__main__
Stack=[]
top=None
while True:
    print("Stack operations")
    print("1.Push")
    print("2.Pop")
    print("3.Peek")
    print("4.Display stack")
```

```
print("5.Exit")
ch=int(input("Enter ch 1-5:"))
if ch==1:
    item=int(input("Enter item:"))
    push(Stack,item)
elif ch==2:
    item=pop(Stack)
    if item=="Underflow":
        print("Underflow!Stack empty")
    else:
        print("Popped item is",item)
elif ch==3:
    item=Peek(Stack)
    if item=="Underflow":
        print("Underflow!Stack is empty!")
    else:
        print("Topmost item is:",item)
elif ch==4:
    display(Stack)
elif ch==5:
    break
else:
    print("Invalid choice")
```

Output

```
Stack operations
1.Push
2.Pop
3.Peek
4.Display stack
5.Exit
Enter ch 1-5:1
```

```
Enter item:1
Stack operations
1.Push
2.Pop
3.Peek
4.Display stack
5.Exit
Enter ch 1-5:1
Enter item:2
Stack operations
1.Push
2.Pop
3.Peek
4.Display stack
5.Exit
Enter ch 1-5:4
2 <-top
1
Stack operations
1.Push
2.Pop
3.Peek
4.Display stack
5.Exit
Enter ch 1-5:3
Topmost item is: 2
```

12-08-24 **Stack Operations**

Reversing the stack

17. Write a menu driven program using functions to reverse a string using stack in Python.

Program

```
def reverse(str):
    stack = []
    reversed_string = ""
    for ch in str:
        stack.append(ch)
    while len(stack):
        reversed_string += stack.pop()
```

```
    return reversed_string

while True:
    print("Stack operations")
    print("1.Reverse stack")
    print("2.Exit")
    ch=int(input("Enter ch 1-2:"))
    if ch==1:
        s=input("Enter the string to be reversed:")
        print(reverse(s))
    elif ch==2:
        break
    else:
        print("Invalid choice")
```

Output

```
Stack operations
1.Reverse stack
2.Exit
Enter ch 1-2:1
Enter the string to be reversed:nohtyP ot emocleW
Reversed String is:
Welcome to Python
Stack operations
1.Reverse stack
2.Exit
Enter ch 1-2:2
```

19-08-24 **SQL - Table-1**

18. Write SQL queries with outputs based on the tables ***Personal*** and ***Job***:

Personal

Empno	***Nam4e***	***Dobirth***	***Nativeplace***	***Hobby***
123	Amit	1965-01-23	Delhi	Music
127	Manoj	1976-12-12	Mumbai	Writing

124	Abhai	1975-08-11	Allahabad	Music
125	Vinod	1977-04-04	Delhi	Sports
128	Abhay	1974-03-10	Mumbai	Gardening
129	Ramesh	1981-10-28	Pune	Sports

Job

Sno	*Area*	*App_date*	*Salary*	*Retd_date*	*Dept*
123	Agra	2006-01-25	5000	2026-01-25	Marketing
127	Mathura	2006-12-22	6000	2026-12-22	Finance
124	Agra	2007-08-19	5500	2027-08-19	Marketing
125	Delhi	2004-04-14	8500	2018-04-14	Sales
128	Pune	2008-03-13	7500	2028-03-13	Sales

(i) To display the name and native place if the name is starting with V .

Select name,nativeplace from personal where name like 'V%';

Output:

```
NAME     NATIVEPLACE
--------  --------------------
Vinod    Delhi
```

(ii) To display the name and salary if the hobby is Music or Sports.(Use IN operator)

Select name, salary from personal , job where hobby IN('Music','Sports') and personal.empno=job.sno;

Output:

```
NAME     SALARY
--------  ------------

Amit     5000
Abhai    5500
Vinod    8500
```

(iii) To display the name and hobby if the employee number is 123.

Select name,hobby from personal where empno=123;

Output:

```
NAME    HOBBY
--------  ----------
Amit      Music
```

(iv) To display the name and area if the native place is Allahabad.

Select name,area from personal ,job where empno=sno and nativeplace='Allahabad';

Output:

```
NAME    AREA
--------  --------
Abhai     Agra
```

(v) To list all the different hobbies.

Select distinct hobby from personal;

Output:

```
DISTINCT HOBBY
------------- -----------
Music
Writing
Sports
Gardening
```

19-08-24 **SQL Table-2**

19. Write SQL queries with outputs based on the tables ***Faculty*** and ***Courses***:

Faculty

F_id	***Fname***	***Lname***	***Hiredate***	***Salary***
102	Amit	Mishra	1998-10-12	12000

103	Nitin	Vyas	1994-12-24	8000
104	Rakshit	Soni	2001-05-18	14000
105	Rashmi	Malhotra	2004-09-11	11000
106	Suleka	Srivatsava	2006-06-05	10000
107	Niranjan	Kumar	1996-08-26	16000

Courses

C_id	***F_id***	***Cname***	***Fees***
C21	102	Grid Computing	40000
C22	106	System Design	16000
C23	104	Computer Security	8000
C24	106	Human Biology	15000
C25	102	Computer Network	20000
C26	105	Visual Basic	6000
C27	107	Dreamweaver	4000

i) To display the last name and fees if the first name is Amit.

Select lname,fees from faculty f,courses c where f.f_id=c.f_id and fname='Amit';

Output:

```
LNAME FEES
---------- -------
Mishra    40000
```

ii) To display the first name and hire date if the salary is greater than or equal to 14000.

Select fname,hiredate from faculty where salary >=14000;

Output:

```
FNAME HIREDATE
---------- ---------------
Rakshit   2001-05-18
Niranjan  1996-08-26
```

iii) To display the first name and last name if the course name is Computer Security.

Select fname,lname from faculty f,courses c where cname='Computer Security' and f.f_id = c.f_id;

Output:

```
FNAME  LNAME
----------  ----------
Rakshit   Soni
```

iv) To display the details if the hire date is 12th October 1998.

Select * from faculty where hiredate = '1998-10-12';

Output:

```
F_ID     FNAME      LNAME      HIREDATE SALARY
------   ----------  ----------  --------------- -----------
 102     Amit       Mishra     1998-10-12  12000
```

v) To count the different course names from the table courses.

Select count(distinct cname) from courses;

Output:

```
COUNT(DISTINCT CNAME)
-------------------------------------
7
```

26-08-24 **SQL Table-3**

20. Write SQL queries with outputs based on the tables S***choolbus*** and ***Driver_details***:

Schoolbus

RtNo	***Area_covered***	***Capacity***	***No_of_students***	***Distance***	***Transporter***	***Charges***

1	Vasant Kunj	100	100	10	Shivam Tra	100000
2	Rohini	80	80	10	Anand Tra	55000
3	Saket	50	50	30	Bhalla Co	95000
4	Yamuna Vihar	120	120	35	Speed Tr	10000
5	Saket	100	100	20	Raj Tr	80000
6	Janak Puri	40	40	10	Yadav Co	60000

Driver_details

RtNo	***Dname***	***Bcode***
1	Ahmad	1001
5	Hussain	1008
3	Ravinder	1001
1	Raghveer	1003

i) To count the different bus codes.

Select count(distinct bcode) from driver_details;

Output:

COUNT(DISTINCT BCODE)

3

ii) To create a view of area covered,transporter and driver name if capacity is 50. Display the contents of the view table.

Create view myview as select area_covered,transporter,dname from Schoolbus s,driver_details d where s.rt_no=d.rt_no and capacity=50;

Select * from myview;

Output:

```
AREA_COVERED          TRANSPORTER  DNAME
----------------------  -------------------  -----------
Saket                 Bhalla Co    Ravinder
```

iii) To display area covered and transporter if number of students is more than 100.

Select area_covered,transporter from schoolbus where no_of_students >100;

Output:

```
AREA_COVERED          TRANSPORTER
```

---------- ----------
Yamuna Vihar Speed Tr

iv) To find the maximum charges from school bus table.

Select max(charges) from schoolbus;

Output:

MAX(CHARGES)

100000

v) To select the area covered and number of students if distance is 20.

Select area_covered,No_of_students from schoolbus where distance=20;

Output:

AREA_COVERED CAPACITY
---------- ----------
Saket 100

26-08-24 **SQL Table-4**

21. Write SQL queries with outputs based on the tables ***Employee*** and ***Incharge***:

Employee

Eno	Name	Basic	Department	Dateofapp	Age	Sex

1	Karan	8000	Personnel	1997-03-27	35	M
2	Divakar	9500	Computer	1998-01-20	34	M
3	Divya	7300	Accounts	1997-02-19	34	F
4	Arun	8350	Personnel	1995-01-01	33	M
5	Sabina	9500	Accounts	1996-01-12	36	F
6	John	7400	Finance	1997-02-24	36	M
7	Robert	8250	Personnel	1997-02-20	39	M
8	Rubina	9450	Maintenance	1998-02-22	37	F
9	Vikas	7500	Computer	1994-01-13	41	M
10	Mohan	9300	Maintenance	1998-02-19	37	M

Incharge

Dept	**Head**
Personnel	Rahul
Computer	Satyam
Accounts	Nath
Finance	Ganesh
Maintenance	Jacob

i) To arrange the table in ascending order of department for female employees.

Select * from employee where sex='F' order by department;

Output:

ENO	NAME	BASIC	DEPARTMENT	DATEOFAPP	AGE	SEX
3	Divya	7300	Accounts	1997-02-19	34	F
5	Sabina	9500	Accounts	1996-01-12	36	F
8	Rubina	9450	Maintenance	1998-02-22	37	F

ii) To count the number of male employees.

Select count(*) from employee where sex='M';

Output:

COUNT(*)

7

iii) To display the total number of employees for each department.

Select department,count(*) from employee group by department;

Output:

DEPARTMENT	COUNT(*)
Accounts	2
Computer	2
Finance	1
Maintenance	2
Personnel	3

iv) Display the details of name , dateofapp and the department head if the head is ‘Jacob’.

Select name,dateofapp,head from employee,incharge

where department=dept and head='Jacob';

Output:

NAME	DATEOFAPP	HEAD
Rubina	1998-02-22	Jacob
Mohan	1998-02-19	Jacob

v) To delete the rows if the department is Computer.

Delete from employee where department='Computer';

Output:

Query OK, 2 rows affected (0.01 sec)

02-09-24 **SQL Table-5**

22. Write SQL queries with outputs based on the tables ***Staff*** and ***Salary***:

Staff

ID	NAME	DEPT	SEX	EXPERIENCE

101	Siddharth	SALES	M	12
104	Raghav	FINANCE	M	5
107	Naman	RESEARCH	M	10
114	Nupur	SALES	F	3
109	Janvi	FINANCE	F	9
105	Rama	RESEARCH	M	10
117	James	SALES	F	3
111	Binoy	FINANCE	F	12
130	Samuel	SALES	M	15

Salary

ID	BASIC	ALLOWANCE	COMMISSION(%)
101	12000	1000	3
104	23000	2300	5
107	32000	4000	5
114	12000	5200	10
109	42000	1700	20
105	18900	1690	3
130	21700	2600	30

i) Display NAME of all staff who are in “SALES” having more than 10 years’ experience from the table STAFF.

Select name from staff where dept='SALES' and experience >10;

Output:

NAME

Siddharth
Samuel

ii) To display the maximum experience department wise.

Select dept,max(experience) from staff group by dept;

Output:

DEPT	MAX(EXPERIENCE)
FINANCE	12
RESEARCH	10
SALES	15

iii) To create a view called myview of name ,department and allowance if the department name starts with 'F'. Display the content of the view table.

Create view myview as select name,dept,allowance from staff,salary where staff.id=salary.id and dept like 'F%';

Select * from myview;

Output:

Query OK, 0 rows affected (0.00 sec)

NAME	DEPT	ALLOWANCE
Raghav	FINANCE	2300
Janvi	FINANCE	1700

iv) To insert a new row into staff table.

Insert into staff values(102,'Rasul','SALES','M',3);

Output:

Query OK, 1 row affected (0.02 sec)

v) To increase the allowance by 1000 for all staff.

Update salary set allowance =allowance+1000;

Output:

Query OK, 7 rows affected (0.00 sec)
Rows matched: 7 Changed: 7 Warnings: 0

02-09-24 **Interfacing Python with SQL**

23. Create the following student4 table and insert data as given below:

Field	Type	Null	Key	Default	Extra
Rollno	int(3)	NO	PRI	NULL	
Name	varchar(15)	YES		NULL	
age	int(11)	YES		NULL	
city	char(8)	YES		NULL	
t_marks	int(11)	YES		NULL	

Rollno	Name	age	city	t_marks
2	Pooja	21	Chail	390
3	Radhika	18	Shimla	388
4	Sonia	24	Goa	300
5	Vinay	25	Pune	410
10	Shaurya	15	Delhi	345

Implement the following SQL commands on the student4 table:

i) Display all the databases.
ii) Display the tables existing in the database.
iii) Display all the records in the table.

Program

```
import mysql.connector
def menu():
    while True:
        print("Main Menu")
        print("1. Display Databases")
        print("2. Create Table")
        print("3. Display Tables")
        print("4. Insert Records")
        print("5. Display Records")
        print("6.Exiting")
        ch=int(input("Enter the choice:"))
        if ch==1:
            show_databases()
        elif ch==2:
            create_table()
        elif ch==3:
            disp_tables()
```

```
        elif ch==4:
            insert_rec()
        elif ch==5:
            disp_records()
        elif ch==6:
            print("Exiting...")
            break
    else:
        print("Wrong input")


def show_databases():


        mydb=mysql.connector.connect(host="localhost",user="root",pa
        sswd="sql123")
    mycursor=mydb.cursor()
    mycursor.execute("show databases")
    print("Existing Databases")
    for x in mycursor:
        print(x)

def create_table():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("create table student4(Rollno int(3) not null
          unique\
                                      primary key,\
                                       Name varchar(15),age integer,city char(8),\
                                       t_marks integer)")
        print("Table created")
    except:
        print("Table cannot be created")

def disp_tables():
```

```
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("show tables")
        for x in mycursor:
            print(x)
    except:
        print(" Tables cannot be displayed")


def insert_rec():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("insert into student4 values('2','Pooja',21,'Chail',390)")
        mycursor.execute("insert into student4 values('3','Radhika',18,'Shimla',388)")
        mycursor.execute("insert into student4 values('4','Sonia',24,'Goa',300)")
        mycursor.execute("insert into student4 values('5','Vinay',25,'Pune',410)")
        mycursor.execute("insert into student4 values('10','Shaurya',15,'Delhi',345)")
        mydb.commit()
        print("Records inserted")
    except:
        print("Records cannot be inserted")
def disp_records():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("select * from student4")
        myrecords=mycursor.fetchall()
        if mycursor.rowcount>0:
            print("Records are:")
```

```
            for x in myrecords:
                print(x)
        else:
            print("No records")
    except:
        print("Records cannot be displayed")

menu()
```

Output

```
Main Menu
1. Display Databases
2. Create Table
```

3. Display Tables
4. Insert Records
5. Display Records
6.Exiting
Enter the choice:2
Table created
Main Menu
1. Display Databases
2. Create Table
3. Display Tables
4. Insert Records
5. Display Records
6.Exiting
Enter the choice:4
Records inserted
Main Menu
1. Display Databases
2. Create Table
3. Display Tables
4. Insert Records
5. Display Records
6.Exiting
Enter the choice:5
Records are:
(2, 'Pooja', 21, 'Chail', 390)
(3, 'Radhika', 18, 'Shimla', 388)
(4, 'Sonia', 24, 'Goa', 300)
(5, 'Vinay', 25, 'Pune', 410)
(10, 'Shaurya', 15, 'Delhi', 345)

09-09-24 **Interfacing Python with SQL**

24. Create the following student table and insert data as given below:

STUDENT

Field	Type	Null	Key	Default	Extra
Rollno	int(3)	NO	PRI	NULL	
Name	varchar(15)	YES		NULL	
Gender	char(1)	YES		NULL	
Marks	int(4)	YES		NULL	
DOB	date	YES		NULL	
Mobile_no	varchar(12)	YES		NULL	
Stream	varchar(15)	YES		NULL	

Rollno	Name	Gender	Marks	DOB	Mobile_no	Stream
1	Raj Kumar	M	93	2017-12-10	9586774748	Science
2	Deep Singh	M	98	2018-10-12	8988886577	Commerce
3	Ankit Sharma	M	76	2019-09-29	NULL	Science
4	Radhika Gupta	F	78	2018-08-09	9818675444	Humanities
5	Payal Goel	F	82	2019-11-17	9845639990	Vocational
6	Diksha Sharma	F	80	2017-05-19	9897666650	Humanities
7	Gurpreet Kaur	F	NULL	2019-04-20	7560875609	Science
8	Akshay Dureja	M	90	2020-05-23	9660567890	Commerce
9	Shreya Anand	F	70	2018-11-24	NULL	Vocational
10	Prateek Mittal	M	75	2019-12-05	9999967543	Science

Implement the following SQL commands on the student table:

i) Display all the records.
ii) Display the records if name starts with ‘R’ or having ‘y’ as the 3rd character.
iii) Delete the rows if the stream is ‘Vocational’ or ‘Science’ (use IN operator).
iv) Add a new column called sports_played.

```
import mysql.connector
def menu():
    while True:
        print("1. Create Table")
        print("2. Insert Records")
        print("3. Display Records")
        print("4. Using like")
        print("5. Delete Records")
        print("6. Alter Table")
        print("7. Exiting")
        ch=int(input("Enter the choice:"))
        if ch==1:
            create_table()
```

```
        elif ch==2:
            insert_rec()
        elif ch==3:
            disp_records()
        elif ch==4:
            using_like()
        elif ch==5:
            del_records()
        elif ch==6:
            alter_table()
        elif ch==7:
            print("Exiting...")
            break
    else:
        print("Wrong input")

def create_table():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("create table if not exists student(Rollno int(3) not
null unique\
                                      primary key,\
                                       Name varchar(15),Gender char(1),Marks int(4),\
                                       DOB date, Mobile_no varchar(12),Stream
varchar(15))")

        print("Table created")
    except:
        print("Table cannot be created")

def insert_rec():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
```

```
        mycursor=mydb.cursor()

        mycursor.execute("insert into student values(1,'Raj Kumar','M',93,'2017-12-
        10',9586774748,'Science')")
        mycursor.execute("insert into student values(2,'Deep Singh','M',98,'2018-10-
        12',8988886577,'Commerce')")
        mycursor.execute("insert into student values(3,'Ankit Sharma','M',76,'2019-09-
        29',NULL,'Science')")
        mycursor.execute("insert into student values(4,'Radhika Gupta','F',78,'2018-08-
        09',9818675444,'Humanities')")
        mycursor.execute("insert into student values(5,'Payal Goel','F',82,'2019-11-
        17',9845639990,'Vocational')")
        mycursor.execute("insert into student values(6,'Diksha Sharma','F',80,'2017-05-
        19',9897666650,'Humanities')")
        mycursor.execute("insert into student values(7,'Gurpreet Kaur','F',NULL,'2019-04-
        20',7560875609,'Science')")
        mycursor.execute("insert into student values(8,'Akshay Dureja','M',90,'2020-05-
        23',9660567890,'Commerce')")
        mycursor.execute("insert into student values(9,'Shreya Anand','F',70,'2018-11-
        24',NULL,'Vocational')")
        mycursor.execute("insert into student values(10,'Prateek Mittal','M',75,'2019-12-
        05',9999967543,'Science')")

        mydb.commit()
        print("Records inserted")
    except:
        print("Records cannot be inserted")

def disp_records():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("select * from student")
        myrecords=mycursor.fetchall()
        if mycursor.rowcount>0:
            print("Records are:")
            for x in myrecords:
                print(x)
        else:
            print("No records")
    except:
```

```
        print("Records cannot be displayed")

def using_like():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("select * from student where name like 'R%' or
name\
                         like '__y%'")
        myrecords=mycursor.fetchall()
        print("Records are:")
        for x in myrecords:
            print(x)
    except:
        print("Records cannot be displayed")

def del_records():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("delete from student where stream
in('Vocational','Science')")
        mydb.commit()
        myrecords=mycursor.fetchall()
        print("Records deleted:")
        for x in myrecords:
            print(x)
    except:
        print("Records already deleted")

def alter_table():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
```

```
                                passwd="sql123",\
                                database="school")
        mycursor=mydb.cursor()
        mycursor.execute("alter table student add sports_played
        varchar(10)”)

        print(“Table Altered”)
    except:
        print(“No Column Added”)

menu()
```

Output

1. Create Table
2. Insert Records

3. Display Records
4. Using like
5. Delete Records
6. Alter Table
7. Exiting
Enter the choice:1
Table created
1. Create Table
2. Insert Records
3. Display Records
4. Using like
5. Delete Records
6. Alter Table
7. Exiting
Enter the choice:2
Records inserted
1. Create Table
2. Insert Records
3. Display Records
4. Using like
5. Delete Records
6. Alter Table
7. Exiting
Enter the choice:4
Records are:
(1, 'Raj Kumar', 'M', 93, datetime.date(2017, 12, 10), '9586774748', 'Science')
(4, 'Radhika Gupta', 'F', 78, datetime.date(2018, 8, 9), '9818675444', 'Humanities')
(5, 'Payal Goel', 'F', 82, datetime.date(2019, 11, 17), '9845639990', 'Vocational')

16-09-24 **Interfacing Python with SQL**

25. Create the following emp table and insert data as given below:

Field	Type	Null	Key	Default	Extra
empno	int(4)	NO	PRI	NULL	
Name	varchar(15)	YES		NULL	
job	varchar(12)	YES		NULL	
mgr	int(4)	YES		NULL	
hiredate	date	YES		NULL	
sal	float(8,2)	YES		NULL	
comm	float(8,2)	YES		NULL	
deptno	int(2)	YES		NULL	

empno	Name	job	mgr	hiredate	sal	comm	deptno
8251	Seth	Salesman	8698	2019-10-16	1250.00	500.00	30
8369	Smith	Clerk	8902	2018-12-15	800.00	NULL	20
8499	Anya	Salesman	8698	2019-11-16	1600.00	500.00	30
8566	Mahadevan	Manager	8839	2020-05-18	2985.00	NULL	20
8654	Momin	Salesman	8698	2016-09-20	1250.00	500.00	30
8698	Bina	Manager	8839	2018-12-19	2850.00	NULL	10
8839	Amir	President	NULL	2018-06-17	5000.00	0.00	30
8888	Scott	Analyst	8566	2019-05-16	3000.00	NULL	10
8902	Fakir	Analyst	8566	2018-08-18	1500.00	NULL	20
8934	Mita	Clerk	8882	2017-11-19	1300.00	NULL	30

Implement the following SQL commands on the emp table:

i) Display all the records.
ii) Display the sum of salaries for each job type.
iii) Change the commission to 600 for all Salesman.
iv) Create a view of emp table consisting of employee name, hire date and department number if hire date is after 18th December, 2015.

Program

```
import mysql.connector
def menu():
    while True:
        print("1. Create Table")
        print("2. Insert Records")
        print("3. Display Records")
        print("4. Using group by")
        print("5. Update Records")
        print("6. View Table")
        print("7. Exiting")
```

```
        ch=int(input("Enter the choice:"))
        if ch==1:
            create_table()
        elif ch==2:
            insert_rec()
        elif ch==3:
            disp_records()
        elif ch==4:
            using_groupby()
        elif ch==5:
            up_records()
        elif ch==6:
            view_table()
        elif ch==7:
            print("Exiting...")
            break
    else:
        print("Wrong input")

def create_table():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("create table if not exists emp(empno int(4) not
null unique\
                                     primary key,\
                                      Name varchar(15),job varchar(12),mgr int(4),\
                                      hiredate date, sal float(8,2),comm float(8,2),\
                                      deptno int(2))")

        print("Table created")
    except:
        print("Table cannot be created")
def insert_rec():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
```

```
                        passwd="sql123",\
                        database="school")
    mycursor=mydb.cursor()

    mycursor.execute("insert into emp values(8369,'Smith','Clerk',8902,'2018-12-15',800,NULL,20)")
    mycursor.execute("insert into emp values(8499,'Anya','Salesman',8698,'2019-11-16',1600,300,30)")
    mycursor.execute("insert into emp values(8251,'Seth','Salesman',8698,'2019-10-16',1250,500,30)")
    mycursor.execute("insert into emp values(8566,'Mahadevan','Manager',8839,'2020-05-18',2985,NULL,20)")
    mycursor.execute("insert into emp values(8654,'Momin','Salesman',8698,'2016-09-20',1250,1400,30)")
    mycursor.execute("insert into emp values(8698,'Bina','Manager',8839,'2018-12-19',2850,NULL,10)")
    mycursor.execute("insert into emp values(8888,'Scott','Analyst',8566,'2019-05-16',3000,NULL,10)")
    mycursor.execute("insert into emp values(8839,'Amir','President',NULL,'2018-06-17',5000,0,30)")
    mycursor.execute("insert into emp values(8902,'Fakir','Analyst',8566,'2018-08-18',1500,NULL,20)")
    mycursor.execute("insert into emp values(8934,'Mita','Clerk',8882,'2017-11-19',1300,NULL,30)")
    mydb.commit()
    print("Records inserted")
  except:
    print("Records cannot be inserted")

def disp_records():
  try:
    mydb=mysql.connector.connect(host="localhost",\
                        user="root",\
                        passwd="sql123",\
                        database="school")
    mycursor=mydb.cursor()
    mycursor.execute("select * from emp")
    myrecords=mycursor.fetchall()
    if mycursor.rowcount>0:
      print("Records are:")
      for x in myrecords:
        print(x)
    else:
      print("No records")
```

```
    except:
        print("Records cannot be displayed")

def using_groupby():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("select job,sum(sal) from emp group by job\
                         having sum(sal)>3000")

        myrecords=mycursor.fetchall()
        print("Records are:")
        for x in myrecords:
            print(x)
    except:
        print("Records cannot be displayed")

def up_records():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="school")
        mycursor=mydb.cursor()
        mycursor.execute("update emp set comm=600\
                         where job='Salesman'")
        mydb.commit()
        myrecords=mycursor.fetchall()
        print("Records updated")
        for x in myrecords:
            print(x)
    except:
        print("Records already updated")
def view_table():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
```

```
                                passwd="sql123",\
                                database="school")
        mycursor=mydb.cursor()
        mycursor.execute("create view myview as select name,hiredate,\
                        deptno from emp where hiredate > '2018-12-15'")
        mycursor1=mydb.cursor()
        mycursor1.execute("select * from myview")

        print("View table created")
        myrecords=mycursor1.fetchall()
        print("Records are:")
        for x in myrecords:
            print(x)
    except:
        print("No view created")

menu()
```

Output

1. Create Table
2. Insert Records

```
3. Display Records
4. Using group by
5. Update Records
6. View Table
7. Exiting
Enter the choice:1
Table created
1. Create Table
2. Insert Records
3. Display Records
4. Using group by
5. Update Records
6. View Table
7. Exiting
Enter the choice:2
Records inserted
Enter the choice:4
Records are:
('Analyst', 4500.0)
('Manager', 5835.0)
('President', 5000.0)
('Salesman', 4100.0)
```

16-09-24 **Interfacing Python with SQL**

26. Create the following sports table and insert data interactively as given below:

Field	Type	Null	Key	Default	Extra
studentno	int(2)	NO	PRI	NULL	
class	int(2)	YES		NULL	
sname	varchar(10)	YES		NULL	
game1	varchar(15)	YES		NULL	
grade1	char(1)	YES		NULL	
game2	varchar(15)	YES		NULL	
grade2	char(1)	YES		NULL	

studentno	class	sname	game1	grade1	game2	grade2
10	7	Sameer	Cricket	B	Swimming	A
11	8	Sujith	Tennis	A	Skating	C
12	7	Kamal	Swimming	B	Football	B
13	7	Venna	Tennis	C	Tennis	A
14	9	Archana	Basketball	A	Cricket	A
15	10	Arpit	Cricket	A	Athletics	C

Implement the following SQL commands on the sports table:

i) Display all the records.
ii) Update game1 for student number 13 to Throwball. The game1 value and the student number are entered through keyboard. (Use parameterized query).
iii) Display the details of the table in descending order of class.
iv) Truncate the table.

Program

```
import mysql.connector
def menu():
    while True:
        print("1. Create Table")
        print("2. Insert Records")
        print("3. Display Records")
        print("4. Update Records")
        print("5. Using order by")
        print("6. Truncate Table")
        print("7. Exiting")
        ch=int(input("Enter the choice:"))
```

```
        if ch==1:
            create_table()
        elif ch==2:
            insert_rec()
        elif ch==3:
            disp_records()
        elif ch==4:
            up_rec()
        elif ch==5:
            using_orderby()
        elif ch==6:
            truncate_table()
        elif ch==7:
            print("Exiting...")
            break
    else:
        print("Wrong input")

def create_table():
        try:
            mydb=mysql.connector.connect(host="localhost",\
                                         user="root",\
                                         passwd="sql123",\
                                         database="exam")
            mycursor=mydb.cursor()
            mycursor.execute("create table if not exists sports(studentno int(2)
                                 not null unique primary key,\
                                 class int(2),sname varchar(10),\
                                 game1 varchar(15), grade1 char(1),game2
varchar(15),grade2 char(1))")

            print("Table created")
        except:
            print("Table cannot be created")

def insert_rec():
    mydb=mysql.connector.connect(host="localhost",\
                                 user="root",\
                                 passwd="sql123",\
                                 database="exam")
    mycursor=mydb.cursor()
```

```
    sql="insert into
sports(studentno,class,sname,game1,grade1,game2,grade2)
values(%s,%s,%s,%s,%s,%s,%s)"
    print('\nPLEASE PROVIDE THE REQUIRED INFORMATION\n')
    no=int(input('\nENTER THE STUDENT NUMBER:'))
    c=int(input('\nENTER THE CLASS:'))
    nm=input('\nENTER THE NAME:')
    g1=input('\nENTER GAME1:')
    gr1=input('\nENTER GRADE1:')
    g2=input('\nENTER GAME2:')
    gr2=input('\nENTER GRADE2:')
    value=(no,c,nm,g1,gr1,g2,gr2)
    try:
        mycursor.execute(sql,value)
        print(no,' ADDED SUCCESSFULLY TO STOCK TABLE')
        mydb.commit()
    except:
        print('UNABLE TO INSERT!!!!!')

def disp_records():
    try:
        mydb=mysql.connector.connect(host="localhost",\
                                     user="root",\
                                     passwd="sql123",\
                                     database="exam")
        mycursor=mydb.cursor()
        mycursor.execute("select * from sports")
        myrecords=mycursor.fetchall()
        if mycursor.rowcount>0:
            print("Records are:")
            for x in myrecords:
                print(x)
        else:
            print("No records")
    except:
        print("Records cannot be displayed")

def up_rec():
    mydb=mysql.connector.connect(host="localhost",\
                                 user="root",\
                                 passwd="sql123",\
                                 database="exam")
    mycursor=mydb.cursor()
```

```
    sql="update sports set game1= '{}' where
            studentno={}".format('Throwball',13)
    try:
        mycursor.execute(sql)
        mydb.commit()
        print(' RECORD UPDATED SUCCESSFULLY')
    except:
        print('UNABLE TO UPDATE THE GAME!!!!')


def using_orderby():
    mydb=mysql.connector.connect(host="localhost",\
                                 user="root",\
                                 passwd="sql123",\
                                 database="exam")
    mycursor=mydb.cursor()

    sql="select * from sports order by class desc"
    try:
        mycursor.execute(sql)
        print(' RECORDS ORDERED SUCCESSFULLY')
        myrecords=mycursor.fetchall()
        for x in myrecords:
            print(x)
    except:
        print('UNABLE TO ORDER!!!!')

def truncate_table():
    mydb=mysql.connector.connect(host="localhost",\
                                 user="root",\
                                 passwd="sql123",\
                                 database="exam")
    mycursor=mydb.cursor()

    sql="truncate table sports"

    try:
        mycursor.execute(sql)
        mydb.commit()
        print('RECORDS DELETED SUCCESSFULLY')
    except:
        mydb.rollback()
```

```
        print('UNABLE TO DELETE RECORDs!!!')

    menu()
```

Output

```
1. Create Table
2. Insert Records
3. Display Records
4. Update Records
5. Using order by
6. Truncate Table
7. Exiting
Enter the choice:1
Table created
1. Create Table
2. Insert Records
3. Display Records
4. Update Records
5. Using order by
6. Truncate Table
7. Exiting
Enter the choice:3
Records are:
(13, 7, 'Venna', 'Throwball', 'C', 'Tennis', 'A')
1. Create Table
2. Insert Records
3. Display Records
4. Update Records
5. Using order by
6. Truncate Table
7. Exiting
Enter the choice:6
RECORDS DELETED SUCCESSFULLY
1. Create Table
2. Insert Records
3. Display Records
4. Update Records
5. Using order by
6. Truncate Table
7. Exiting
```