```python
import csv         #TO READ AND WRITE TABULAR DATA AS COMMA SEPARATED VALUES
import pickle      #SERIALISING & DE-SERIALISING PYTHON OBJECT STRUCTURES
import os          #HERE USED TO DELETE AND RENAME FILES


print()
print()
print("-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
*-*-*-*-*-*")
print("                                                                WELCOME TO STUDENT DATABASE")
print("-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
*-*-*-*-*-*")
print()
print()

student = ['Roll', 'Name', 'Age','Attendance','Phone','Marks']
database= 'csvdatabase.csv'
bytebase='databasebinary.dat'


def add_student():
    print("---------------------------------")
    print("Add Student Information")
    print("---------------------------------")
    global student#To avoid UnboundLocalError
    global database#To avoid UnboundLocalError

    student_data = []
    studentbinary= {}
    for field in student:
        if field=="Attendance":
            value = input("Enter attendance of last 30days : ")
        else:
            value = input("Enter " + field + ": ")
        studentbinary[field]=value
        student_data.append(value)
    #writing to csv file
    with open(database, "a",newline='') as f:
        writer = csv.writer(f)
        writer.writerows([student_data])
    #writing to binary file
    outfile = open(bytebase, 'ab')
    pickle.dump(studentbinary, outfile)

    print("Data saved successfully")
    input("Press any key to continue")

def view_students():
    global student#To avoid UnboundLocalError
    global database#To avoid UnboundLocalError

    print("                                                      ---STUDENT RECORDS---")

    with open(database, "r") as f:
        reader = csv.reader(f)
        print("\n---------------------------------------------------------------------------------------------------------------------------------------")
        for x in student:
            #to adjust gap between header elements
            if x=="Roll":
                print(x, end='\t   ')
            if x=="Name":
                print(x, end='\t      ')
            if x=="Age":
                print(x, end='\t   ')
            if x=="Attendance":
                print(x, end='\t     ')
            if x=="Phone":
                print(x, end='\t                            ')
            if x=="Marks":
                print(x, end='\t                                         ')
        print("\n-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")
#to adjust gaps in records
        for row in reader:
            for item in row:
                if item==row[0]:
                    print(item, end="   ")
                if item==row[1]:
                    print(item, end="         ")
                if item==row[2]:
                    print(item, end="      ")
                if item==row[3]:
                    print("       ",item, end="                      ")
                if item==row[4]:
                    print(item, end="               ")
                if item==row[5]:
                    print(item, end="                   ")
            print("\n")

    input("Press any key to continue")

def search_student():
    infile = open(bytebase, 'rb')
    found = False
    roll=input('Enter the roll no. you want to search: ')
    while True:
        try:
            stu = pickle.load(infile)
            if stu['Roll'] == roll:
                for field in student:
                    print(field,end='--')
                    print(stu[field])
                found = True
                break
        except EOFError:
            break
    if found==False:
        print('Record not found!!')
    infile.close()

    input("Press any key to continue")

def delete_student():
    global student#To avoid UnboundLocalError
    global database#To avoid UnboundLocalError
    print('\nDELETE RECORD')

    f=open(database, "r")#OPENING CSV FILE
    reader = csv.reader(f)#CSV READER OBJECT

    infile = open(bytebase, 'rb')#Opening Binary File
    outfile = open("temp.dat","wb")#Temp binary file,later to be renamed
    rollno = input('Enter roll number: ')

    while True:
        try:
            stu1=pickle.load(infile)
            if stu1['Roll'] == rollno:
                continue
            else:
                pickle.dump(stu1,outfile)
        except EOFError:
            break
```

```python
    infile.close()
    outfile.close()
    os.remove(bytebase)
    os.rename("temp.dat","databasebinary.dat")
    #for csv
    updated_data = []
    for row in reader:
        if rollno != row[0]:
            updated_data.append(row)
        else:
            print('Student found in record')
            print("DELETED SUCCESSFULLY")
    with open(database, "w",newline='') as f:
        writer = csv.writer(f)
        writer.writerows(updated_data)
        print("Roll no. ", rollno, "deleted successfully")

    input("Press any key to continue")


def update_student():
    global student#To avoid UnboundLocalError
    global database#To avoid UnboundLocalError
    print("\nUPDATE STUDENT'S")
    f=open(database, "r")#Open csv
    reader = csv.reader(f)#CSV reader object
    student_data = []#empty list contains specific record
    updated_data = []#empty list contains all records

    infile = open(bytebase, 'rb')
    outfile = open("temp.dat","wb")
    found = False
    rollno = input('Enter roll number: ')
    while True:
        try:
            stu = pickle.load(infile)
            if stu['Roll'] == rollno:
                for hdr in student:
                    print(hdr,"---",stu[hdr])
                    ans=input('Wants to edit(y/n)? ')
                    if ans in 'yY':
                        new=input("Enter new one : ")
                        stu[hdr] = new
                        student_data.append(new)
                    else:
                        student_data.append(stu[hdr])

                pickle.dump(stu,outfile)
                found = True
            else:
                pickle.dump(stu,outfile)
        except EOFError:
            break
    for row in reader:
        if rollno==row[0]:
            updated_data.append(student_data)
        else:
            updated_data.append(row)

    if found == False:
        print('Record not Found')
    else:
        print('Record updated')

    infile.close()
    outfile.close()
    os.remove("databasebinary.dat")
    os.rename("temp.dat","databasebinary.dat")

    f=open(database, "w",newline='')
    writer = csv.writer(f)
    writer.writerows(updated_data)

    input("Press any key to continue")

def atper():
    global student#To avoid UnboundLocalError
    global database#To avoid UnboundLocalError
    stuperat=[]
    with open(database, "r")as f:
        reader2 = csv.reader(f)
        for row in reader2:
            #if len(row) > 0:
                if 22.5<= float(row[3]):
                    stuperat.append(row[1])
        print()
        print("Students with attendance above 75% are")
        print(stuperat)

def clsavg():
    global student
    global database
    av=0
    avp=0
    count=0
    with open(database, "r",newline='')as f:
        reader3 = csv.reader(f)
        for row in reader3:
            #if len(row) > 0:
                count+=1
                for num in row:
                    if num==row[5]:
                        av=av+float(num)
    avp=(av/count)
    print()
    print("The class average is",avp)


while True:
    print("    (1)   ::::::::::::::::::::::::::   Add New Student                  (:Adds Data to both CSV and Binary File)")
    print("    (2)   ::::::::::::::::::::::::::   View Students                    (:Retrives Data From CSV File:)")
    print("    (3)   ::::::::::::::::::::::::::   Search Student                   (:Retrives Data from Binary file:)")
    print("    (4)   ::::::::::::::::::::::::::   Update Student's data          (:Updates Data in both CSV and Binary File:)")
    print("    (5)   ::::::::::::::::::::::::::   Delete Student                   (:Deletes the record from both CSV and Binary File:)")
    print("    (6)   ::::::::::::::::::::::::::   Students with above 75% attendance")
    print("    (7)   ::::::::::::::::::::::::::   Class average")
    print("    (8)   ::::::::::::::::::::::::::   Quit")
    print()
    choice = input("    +++++++Enter your choice: ")
    if choice == '1':
        add_student()
    elif choice == '2':
        view_students()
    elif choice == '3':
        search_student()
    elif choice == '4':
        update_student()
    elif choice == '5':
        delete_student()
    elif choice=='6':
        atper()
    elif choice=='7':
        clsavg()
    else:
        break

print("-----------------------------------------------------------------------------------------------")
print("                                                   THANK YOU")
print("-----------------------------------------------------------------------------------------------")
```