Student Management System

Version 1.0 (2020-2021)

Computer Science (083) Project

Developed By

S. Shruti

XII-K

Delhi Public School, R.K.Puram, New Delhi

www.dpsrkp.net

Index

Certificate

This is to certify that the Student Management System Computer Science project is developed by S. Shruti under my supervision in the session 2020-2021.

The work done by her is original.

____

Computer Science Teacher

Date: _______________

Acknowledgement

I would like to express my sincere gratitude to all those people who have been instrumental in bringing this project to completion. First and foremost, I wish to thank my parents for their constant moral support and encouragement.

I would like to extend my heartfelt thanks to my Computer teacher Mr. Mohitendra Dey for providing an informative and experiential platform to learn coding and Computer Science. Without his expertise and guidance, I would have not been able to complete my project successfully. His inexhaustible source of inspiration and supervision were crucial to develop my project work.

I feel obliged to be a part of this great school, DPS, R.K.Puram where I have been always given the chance to grow in an environment of diverse opportunities. The teachers here have helped me to expand my knowledge base and improve myself as a person. I have had an enriching experience so far, with tons of new things to learn and explore. I am grateful for all that I have been blessed with.

References : Class Notes

Introduction

Keeping in view of growing requirements of effective and efficient solutions for various problems, with my knowledge of Python and data files (pickled files), in this project, I have tried to provide a simple solution for the **School Management System**.

My project has the complete management solutions to take care of the measures required to process all the essential details- from attendance to assessment, crucial for the smooth functioning of the School Management System.

I have divided the program into small user defined functions to take care of all future upgrades and expansion. The following is the list of Functions along with their description.

Functions pertaining to the **Admin Login System**:

Sno	Function Name	Description
1	CreateStudentList()	To create a record containing the personal details of each student.
2	AddStudent()	To add a new student to the record and his/her corresponding details.
3	ViewDetails()	To display a list of all the students along with the particulars entered.
4	ModifyDetails()	To change any of the particulars of a student.
5	DeleteStudentRecord()	To delete a specific student's entire record of details.
6	Male_Female()	To display the list of male and female students along with their respective counts.
7	SearchStudent()	To search a particular student on the basis of admission no.
8	SortAlphabeticalOrder()	To arrange the list of students in an alphabetical order.
9	GenerateAttendance()	To mark the attendance of a specific student by the admin.
10	GenerateResult()	To enter the marks for each subject and declare the result of a specific student.
11	GenerateReport()	To create a collective class report containing all the academic details of each student. Topper of the class, no. of candidates passed and failed, student with highest attendance and the grading scheme is also displayed.
13	RankList()	To allot ranks to the students based on their academic performance.

Functions pertaining to the **Student (Service) Login System**:

Sno	Function Name	Description
1	AddYourself()	To add a student to the existing record created by the admin.
2	ViewYourDetails()	To enable the student to view his/her details on the basis of the admission no. assigned to them.
3	ModifyDetails()	To change any of the particulars of the student.
4	DeleteStudentRecord()	To delete a specific student's entire record of details.
5	ViewAttendance()	To display the attendance of the student to him/her as entered by the admin.
6	ViewResult()	To display the result of the student to him/her as entered by the admin.

Functions pertaining to the **User-driven Menu**:

Sno	Function Name	Description
1	AdminLoginMenu()	To display the menu for the admin login system.
2	StudentLoginMenu()	To display the menu for the student(service) login system.
3	FinalMenu()	To display the final menu for toggling between the admin and service login.

Binary Files with structure used in the program:

File Name: STUDENT.DAT

Sno	Field Name	Data Type	Description
1	Admission No.	Alphanumeric	Auto-generated sequential admission no. assigned to each new student in the record.
2	Name	String character	As entered by the student/admin.
3	Date of Birth	String character	As entered by the student/admin.
4	Class	Integer type	As entered by the student/admin.
5	Section	String character	As entered by the student/admin.
6	Age	Integer type	As entered by the student/admin.
7	Gender	String character	As entered by the student/admin.
8	Blood Group	String character	As entered by the student/admin.
9	Attendance	Float type	As entered by the admin.
10	Average/Percentage	Float type	Marks are entered by the admin while the overall percentage is calculated by the system and stored.
11	Result/Grade	String character	The grade as per the overall percentage is allotted to each student on the basis of a set grading scheme.

External Modules used in the Program : pickle, statistics

Salient Features of the Project:

- User friendly menu driven options.
- Exceptions used for FileNotFoundError and ValueError to avoid inconvenience to users.
- Data Validation for string and integer data type to avoid ambiguous/wrong/invalid data entry.
- Password to prevent unauthorised users from gaining access to the academic and personal record of each student.
- Auto-generated sequential admission no. allotment to eliminate data redundancy and provide the basis for unique identification of each student.
- Uppercase, lowercase and trailing spaces of data during entry is checked upon to ensure reliability of the code and for the ease of the user.
- Program termination has been avoided through multiple checks and notify users with an appropriate error message to facilitate smooth access of the application.
- Extra facilities for admin has also been provided like- displaying the count, list of male and female students, and initiating the generation of rank list.

Source Code

```
# Project Title  : Student Management System
# Version        : 1.0 (2020-2021)
# Developed By   : S. Shruti
# Guide          : Mr. Mohitendra Dey
# Last Updated On: 2020-11-21

import pickle
import statistics

def CreateStudentList():
    try:
        with open("STUDENT.DAT", "wb") as F:
         A=[];VAL=100
         while True:
            ANO="Q"+str(VAL)
            print("Admission no. assigned to student : "+ANO)
            while True:
             NAME=str(input("Name of student : "))
             if NAME.isalpha() is True:
                     break
             else:
                     print("Please enter only alphabets!")
            DOB=str(input("Date of birth (format:DD-MM-YY) : "))
            CLASS=int(input("Class : "))
            SECTION=str(input("Section : "))
            AGE=int(input("Age : "))
            GENDER=str(input("Gender : "))
            BLDGRP=str(input("Blood Group : "))
            ATTENDANCE=' '
            AVERAGE=' '
            RESULT_GRADE =' '
           A.append([ANO, NAME, DOB, CLASS, SECTION, AGE, GENDER, BLDGRP,
ATTENDANCE, AVERAGE, RESULT_GRADE])
            VAL+=1
            Q=input("Want to add more?(Y/N) ")
            Q=Q.strip()
            if Q in ['n', 'N']:
                break
         pickle.dump(A,F)
    except ValueError:
            print("Please enter the correct data type!")

def AddStudent(): #For admin login
    try:
        with open("STUDENT.DAT", "rb+") as F:
         A=pickle.load(F)
         X=int(len(A)-1)
         VAL=int(A[X][0][1::])+1
         while True:
            ANO="Q"+str(VAL)
            print("Admission no. assigned to student : "+ANO)
            while True:
             NAME=str(input("Name of student : "))
```

```
                if NAME.isalpha() is True:
                        break
                else:
                        print("Please enter only alphabets!")
               DOB=str(input("Date of birth (format:DD-MM-YY) : "))
               CLASS=int(input("Class : "))
               SECTION=str(input("Section : "))
               AGE=int(input("Age : "))
               GENDER=str(input("Gender : "))
               BLDGRP=str(input("Blood Group : "))
               ATTENDANCE=' '
               AVERAGE=' '
               RESULT_GRADE =' '
                  A.append([ANO, NAME, DOB, CLASS, SECTION, AGE, GENDER, BLDGRP,
ATTENDANCE, AVERAGE, RESULT_GRADE])
               VAL+=1
               Q=input("Want to add more?(Y/N) ")
               Q=Q.strip()
               if Q in ['n', 'N']:
                   break
            F.seek(0)
            pickle.dump(A,F)
    except FileNotFoundError:
        print("File not found! Retry.")
    except ValueError:
        print("Please enter the correct data type!")

def **AddYourself**(): #For service login
    try:
        with open("STUDENT.DAT", "rb+") as F:
            A=pickle.load(F)
            X=int(len(A)-1)
            VAL=int(A[X][0][1::])+1
            ANO="Q"+str(VAL)
            print("Your admission no. : "+ANO)
            while True:
             NAME=str(input("Name of student : "))
             if NAME.isalpha() is True:
                     break
             else:
                     print("Please enter only alphabets!")
            DOB=str(input("Date of birth (format:DD-MM-YY) : "))
            CLASS=int(input("Class : "))
            SECTION=str(input("Section : "))
            AGE=int(input("Age : "))
            GENDER=str(input("Gender : "))
            BLDGRP=str(input("Blood Group : "))
            ATTENDANCE=' '
            AVERAGE=' '
            RESULT_GRADE =' '
              A.append([ANO, NAME, DOB, CLASS, SECTION, AGE, GENDER, BLDGRP,
ATTENDANCE, AVERAGE, RESULT_GRADE])
            F.seek(0)
            pickle.dump(A,F)
    except FileNotFoundError:
        print("File not found! Retry.")
    except ValueError:
        print("Please enter the correct data type!")

def ViewDetails(): #For admin login
    try:
        with open("STUDENT.DAT", "rb") as F:
            A=pickle.load(F)
            for i in A:
                print(i)
    except FileNotFoundError:
        print("File not found! Retry.")

def ViewYourDetails(): #For service login
    try:
        with open("STUDENT.DAT", "rb") as F:
            A=pickle.load(F)
            Q=input("Enter your admission no. to view your details : ")
            for i in A:
                if Q==i[0]:
                        print(i)
                        break
            else:
                    print("Admission no. not found!")
    except FileNotFoundError:
        print("File not found! Retry.")

def ModifyDetails():
    try:
        with open("STUDENT.DAT", "rb+") as F:
            A=pickle.load(F)
              Q=input("Enter student admission no. whose details have to be
modified : ")
            for i in A:
                if Q==i[0]:
                             S=input("Enter which detail has to be modified
(NAME/DOB/CLASS/SECTION/AGE/GENDER/BLDGRP/ATD/AVG/RESULT) : ")
                    if S=="NAME":
                        i[1]=input("Enter new student name : ")
                        print("Record Updated.")
                    elif S=="DOB":
                        i[2]=input("Enter new date of birth : ")
                        print("Record Updated.")
                    elif S=="CLASS":
                        i[3]=int(input("Enter new class : "))
                        print("Record Updated.")
                    elif S=="SECTION":
                        i[4]=input("Enter new section : ")
                        print("Record Updated.")
                    elif S=="AGE":
                        i[5]=int(input("Enter new age : "))
                        print("Record Updated.")
                    elif S=="GENDER":
                        i[6]=input("Enter new gender : ")
                        print("Record Updated.")
                    elif S=="BLDGRP":
                        i[7]=input("Enter new blood group : ")
                        print("Record Updated.")
                    elif S=="ATD":
                        i[8]=int(input("Enter new attendance: "))
                        print("Record Updated.")
                    elif S=="AVG":
                        i[9]=int(input("Enter new percentage : "))
                        print("Record Updated.")
                    elif S=="RESULT":
                        i[10]=input("Enter new result : ")
                        print("Record Updated.")
                    else:
                        print("Invalid Entry!")
                    break
            else:
                print("Admission no. not found!")
            F.seek(0)
            pickle.dump(A,F)
    except FileNotFoundError:
        print("File not found! Retry.")
    except ValueError:
        print("Please enter correct data type!")

def DeleteStudentRecord():
    try:
        with open("STUDENT.DAT", "rb+") as F:
            A=pickle.load(F)
                Q=input("Enter student admission no. whose record has to be
deleted : ")
            for i in A:
                if Q==i[0]:
                    Z=A.index(i)
                    A.pop(Z)
                    print("Student record deleted.")
                    break
            else:
                print("Admission no. not found!")
            F.seek(0)
            pickle.dump(A,F)
    except FileNotFoundError:
        print("File not found! Retry.")

def Male_Female():
    with open("STUDENT.DAT", "rb") as F:
         A=pickle.load(F)
         d=[]; e=[]
         for i in A:
            if i[6] in ["Male", 'MALE']:
                d.append(i[1])
            elif i[6] in ["Female", 'FEMALE']:
                e.append(i[1])
         print(d, "Count of male students: ", len(d))
         print(e, "Count of female students: ", len(e))

def SearchStudent():
    try:
        with open("STUDENT.DAT", "rb") as F:
            A=pickle.load(F)
                Q=input("Enter student admission no. whose record has to be
displayed : ")
```

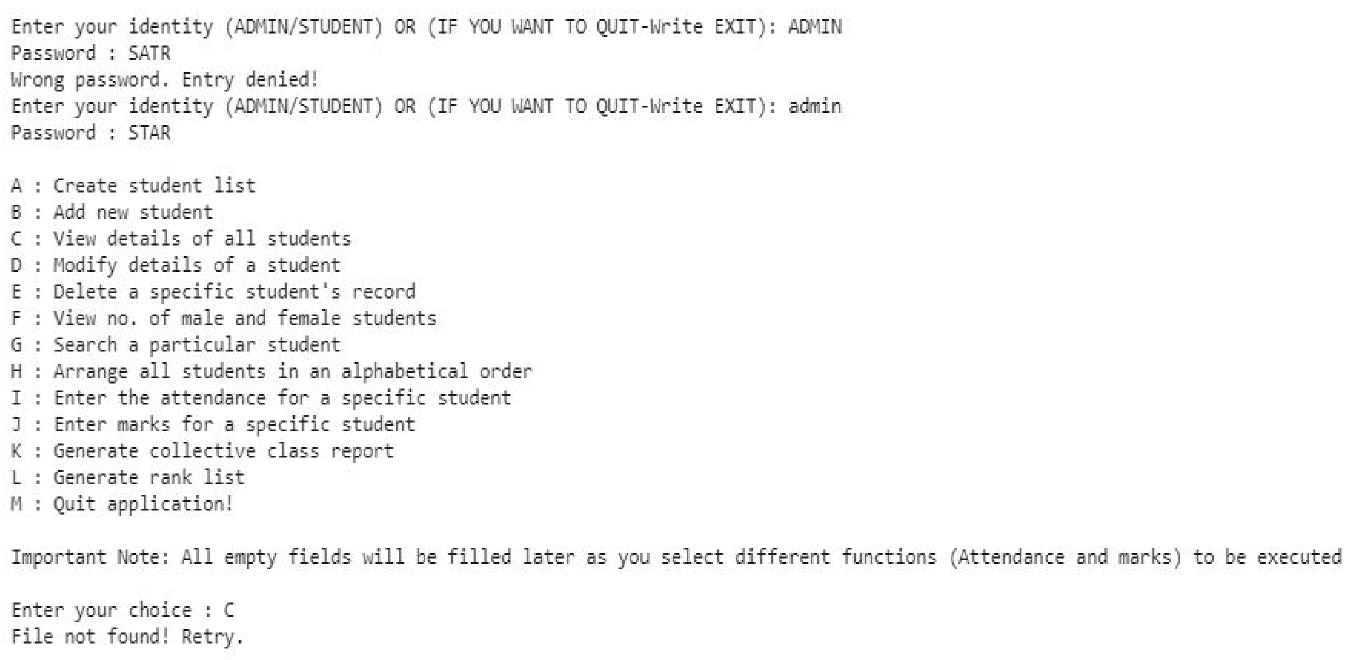
```
            for i in A:
                    if Q==i[0]:
                        print(i)
                        break
            else:
                    print("Admission no. not found!")
    except FileNotFoundError:
        print("File not found! Retry.")

def SortAlphabeticalOrder():
    try:
        with open("STUDENT.DAT", "rb") as F:
            A=pickle.load(F)
            B=[]
            for i in A:
                B.append(i[1])
            B.sort()
            print(B)
    except FileNotFoundError:
        print("File not found! Retry.")

def GenerateAttendance(): #For Admin login
    try:
        with open("STUDENT.DAT", "rb+") as F:
         A=pickle.load(F)
         Q=input("Enter student admission no. to update attendance : ")
         for i in A:
             if Q==i[0]:
                    X=int(input("Enter student's attendance (for the academic
session ie enter the no. of days attended out of 365) : "))
                 ATD=round(((X/365)*100),2)
                 print("Attendance percentage : ", ATD)
                 print("Attendance marked.")
                 i[8]=ATD
                 break
         else:
             print("Admission no. not found!")
         F.seek(0)
         pickle.dump(A,F)
    except FileNotFoundError:
        print("File not found! Retry.")

def ViewAttendance(): #For Service login
    try:
        with open("STUDENT.DAT", "rb") as F:
         A=pickle.load(F)
         Q=input("Enter your admission no. to know your attendance : ")
         for i in A:
             if Q==i[0]:
                if i[8]!=" ":
                 print("Attendance : ", i[8])
                 break
                elif i[8]==" ":
                   print("Your attendance has not been marked. Please inform
your teacher.")
                 break
         else:
                print("Admission no. not found!")
    except FileNotFoundError:
        print("File not found! Retry.")

def GenerateResult(): #For Admin login
     try:
        with open("STUDENT.DAT", "rb+") as F:
         A=pickle.load(F)
         B=[]
         Q=input("Enter student admission no. to update marks in record : ")
         for i in A:
          if Q==i[0]:
            print("Enter student marks in each subject...")
            while True:
                ENG=int(input("English : "))
                MATH=int(input("Maths : "))
                SCI=int(input("Science : "))
                SST=int(input("Social Studies : "))
                COMP=int(input("Computer Science : "))
                TL=int(input("Third Language : "))
                AVG=round(((ENG+MATH+SCI+SST+COMP+TL)/6),2)
                i[9]=AVG
                print("Student's overall percentage is: ", AVG)
                if AVG>=90 and AVG<=100:
                    R='PASS (A)'
                    i[10]=R
                    print("Student grade is : A")
                elif AVG>=80 and AVG<=89:
                    R='PASS (B)'
                    i[10]=R
                    print("Student grade is : B")
                elif AVG>=65 and AVG<=79:
                    R='PASS (C)'
                    i[10]=R
                    print("Student grade is : C")
                elif AVG>=40 and AVG<=64:
                    R='PASS (D)'
                    i[10]=R
                    print("Student grade is : D")
                elif AVG>=0 and AVG<=39:
                    R='FAIL'
                    i[10]=R
                    print("Student grade is : FAIL")
                break
            break
         else:
              print("Admission no. not found!")
         F.seek(0)
         pickle.dump(A,F)
     except FileNotFoundError:
        print("File not found! Retry.")
     except ValueError:
        print("Please enter the correct data type!")

def ViewResult(): #For Service login
    try:
        with open("STUDENT.DAT", "rb") as F:
         A=pickle.load(F)
         Q=input("Enter your admission no. to know your result: ")
         for i in A:
             if Q==i[0]:
                if i[9]!=" ":
                 print("Overall percentage : ", i[9])
                 print("Your Grade : ", i[10])
                 break
                elif i[9]==" ":
                    print("Your result has not been generated. Please inform
your teacher.")
                 break
         else:
             print("Admission no. not found!")
    except FileNotFoundError:
        print("File not found! Retry.")

B=[];C=[];D=[]#For rank list
def GenerateReport(): #For Admin login
    try:
     with open("STUDENT.DAT", "rb+") as F:
         A=pickle.load(F)
          print("Name     Date_of_Birth     Class     Section     Percentage
Result     Attendance")
         global B; global C; global D
         for i in A:
                   print(i[1],"    ",i[2],"           ",i[3],"        ",i[4],"
",i[9],"        ",i[10],"      ",i[8])
         for i in A:
           if i[9] not in B:
             B.append(i[9])
             C.append(i[10])
             D.append(i[8])
         print("The class average is : ", round(statistics.mean(B),2))
         x=max(B)
         for i in A:
            if i[9]==x:
                print("The class topper is : ", i[1])
                break
         PASS=[];FAIL=[]
         for i in A:
             if i[10]=="PASS (A)" or i[10]=="PASS (B)" or i[10]=="PASS (C)" or
i[10]=="PASS (D)":
                PASS.append(i[1])
            elif i[10]=='FAIL':
                FAIL.append(i[1])
         print("No. of students who have passed : ", len(PASS), PASS)
         print("No. of students who have failed : ", len(FAIL), FAIL)
         y=max(D)
         for i in A:
            if i[8]==y:
                   print("Student who has attended the most no. of days : ",
i[1])
                break
         print("")
         print("Grading scheme is as follows:")
         print("A : 100-90")
         print("B : 89-80")
         print("C : 79-65")
         print("D : 64-40")
         print("FAIL : less than 40")
    except FileNotFoundError:
        print("File not found! Retry.")

def RankList():
    global B
    RANK=sorted(B)
    try:
        with open("STUDENT.DAT", "rb+") as F:
         A=pickle.load(F)
         LIST=[]
         for j in RANK:
             for i in A:
                 if i[9]==j:
                     LIST.append(i[1])
        print("Rank list is as follows:")
        print("Rank     Name")
        z=int(len(LIST))
        for k in range(0,z):
            print(k+1,"     ", LIST[z-k-1])
    except FileNotFoundError:
        print("File not found! Retry.")

def AdminLoginMenu():
      while True:
        print("")
        print("A : Create student list")
        print("B : Add new student")
        print("C : View details of all students")
        print("D : Modify details of a student")
        print("E : Delete a specific student's record")
        print("F : View no. of male and female students")
        print("G : Search a particular student")
        print("H : Arrange all students in an alphabetical order")
        print("I : Enter the attendance for a specific student")
        print("J : Enter marks for a specific student")
        print("K : Generate collective class report")
        print("L : Generate rank list")
        print("M : Quit application!")
        print("")
         print("Important Note: All empty fields will be filled later as you
select different functions (Attendance and marks) to be executed")
        print("")
        Q=input("Enter your choice : ")
        Q=Q.strip()
        if Q=="A" or Q=="a":
            CreateStudentList()
        elif Q=="B" or Q=="b":
            AddStudent()
        elif Q=="C" or Q=="c":
            ViewDetails()
        elif Q=="D" or Q=="d":
            ModifyDetails()
        elif Q=="E" or Q=="e":
            DeleteStudentRecord()
        elif Q=="F" or Q=="f":
            Male_Female()
        elif Q=="G" or Q=="g":
            SearchStudent()
        elif Q=="H" or Q=="h":
            SortAlphabeticalOrder()
        elif Q=="I" or Q=="i":
            GenerateAttendance()
        elif Q=="J" or Q=="j":
            GenerateResult()
        elif Q=="K" or Q=="k":
                with open("STUDENT.DAT","rb") as F:
                        A=pickle.load(F)
                        LEN=len(A)
                        count=0
                        for i in A:
                                if i[8]!=" " and i[9]!=" " and i[10]!=" ":
                                        count+=1
                        if count==LEN:
                                GenerateReport()
                        else:
                                print("Please fill the empty entries first")
        elif Q=="L" or Q=="l":
                with open("STUDENT.DAT","rb") as F:
                        A=pickle.load(F)
                        LEN=len(A)
                        count=0
                        for i in A:
                                if i[8]!=" " and i[9]!=" " and i[10]!=" ":
                                        count+=1
                        if count==LEN and B!=[]:
                                RankList()
                        else:
                                      print("Please generate Collective Class
Report first...")
        elif Q=="M" or Q=="m":
            break
        else:
            print("Invalid Entry!")

def StudentLoginMenu():
    while True:
        print("")
        print("A : Enter your personal details")
        print("B : Modify your details")
        print("C : Delete your record")
        print("D : View your details")
        print("E : Know your attendance")
        print("F : Know your result")
        print("G : Quit application!")
        print("")
        Q=input("Enter your choice : ")
        Q=Q.strip()
        if Q=="A" or Q=="a":
            AddYourself()
        elif Q=="B" or Q=="b":
            ModifyDetails()
        elif Q=="C" or Q=="c":
            DeleteStudentRecord()
        elif Q=="D" or Q=="d":
            ViewYourDetails()
        elif Q=="E" or Q=="e":
            ViewAttendance()
        elif Q=="F" or Q=="f":
            ViewResult()
        elif Q=="G" or Q=="g":
            break
        else:
            print("Invalid Entry!")

def FinalMenu():#For admin login and service login
    while True:
           Q=input("Enter your identity (ADMIN/STUDENT) OR (IF YOU WANT TO
QUIT-Write EXIT): ")
        Q=Q.strip()
        if Q in ["ADMIN", "admin"]:
            PSWRD=input("Password : ") #Password="STAR" (sample)
            if PSWRD=="STAR":
                AdminLoginMenu()
            else:
                print("Wrong password. Entry denied!")
        elif Q in ["STUDENT", "student"]:
            StudentLoginMenu()
        elif Q in ["EXIT", "exit"]:
             break
        else:
            print("Invalid Entry!")

FinalMenu()
```

Output Screen

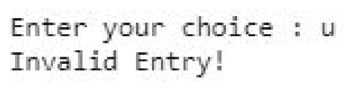
(All Operations)

Operation 1

Admin entry with password check. Admin menu displayed. Option C raises FileNotFoundError which is handled using try-except method.



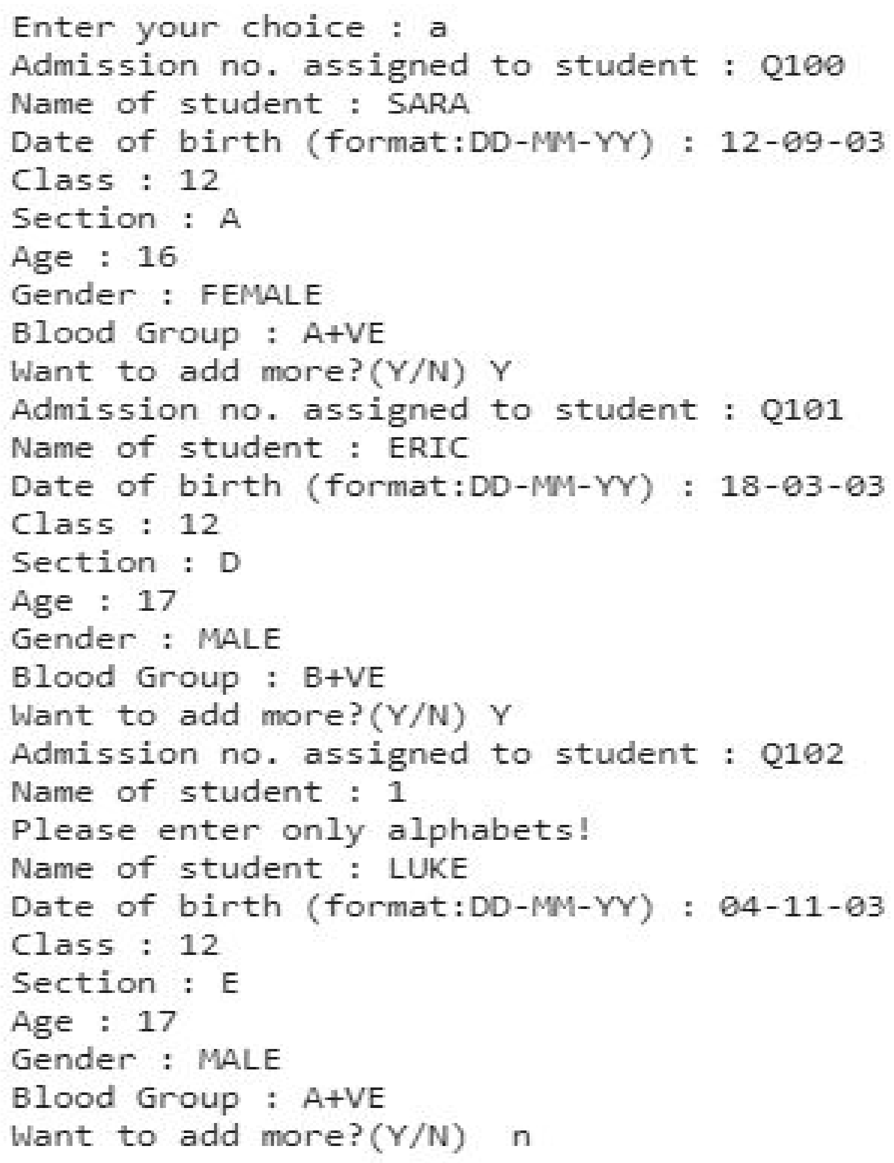
```
Enter your identity (ADMIN/STUDENT) OR (IF YOU WANT TO QUIT-Write EXIT): ADMIN
Password : SATR
Wrong password. Entry denied!
Enter your identity (ADMIN/STUDENT) OR (IF YOU WANT TO QUIT-Write EXIT): admin
Password : STAR

A : Create student list
B : Add new student
C : View details of all students
D : Modify details of a student
E : Delete a specific student's record
F : View no. of male and female students
G : Search a particular student
H : Arrange all students in an alphabetical order
I : Enter the attendance for a specific student
J : Enter marks for a specific student
K : Generate collective class report
L : Generate rank list
M : Quit application!

Important Note: All empty fields will be filled later as you select different functions (Attendance and marks) to be executed

Enter your choice : C
File not found! Retry.
```



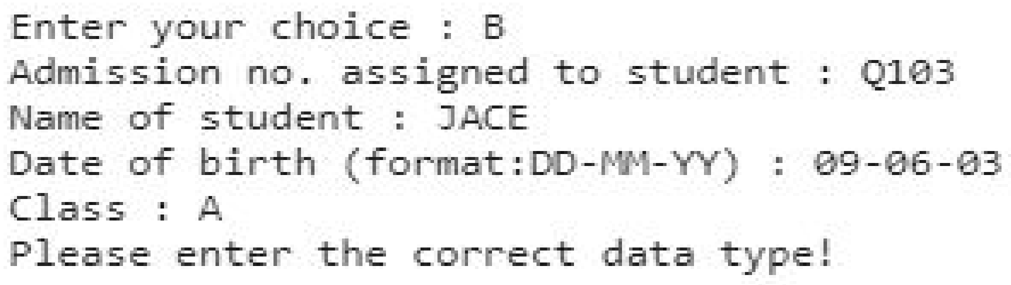
```
Enter your choice : u
Invalid Entry!
```

Operation 2

Data entry with check of string and integer type characters.



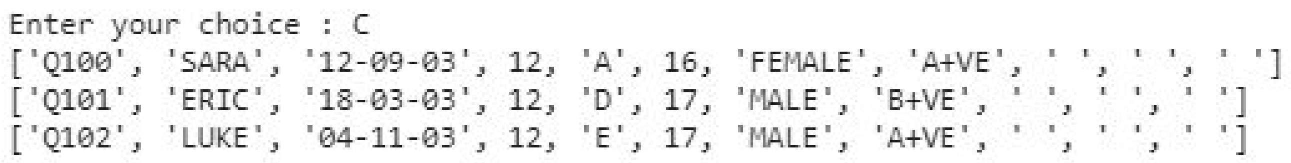
```
Enter your choice : a
Admission no. assigned to student : Q100
Name of student : SARA
Date of birth (format:DD-MM-YY) : 12-09-03
Class : 12
Section : A
Age : 16
Gender : FEMALE
Blood Group : A+VE
Want to add more?(Y/N) Y
Admission no. assigned to student : Q101
Name of student : ERIC
Date of birth (format:DD-MM-YY) : 18-03-03
Class : 12
Section : D
Age : 17
Gender : MALE
Blood Group : B+VE
Want to add more?(Y/N) Y
Admission no. assigned to student : Q102
Name of student : 1
Please enter only alphabets!
Name of student : LUKE
Date of birth (format:DD-MM-YY) : 04-11-03
Class : 12
Section : E
Age : 17
Gender : MALE
Blood Group : A+VE
Want to add more?(Y/N)  n
```



```
Enter your choice : B
Admission no. assigned to student : Q103
Name of student : JACE
Date of birth (format:DD-MM-YY) : 09-06-03
Class : A
Please enter the correct data type!
```

Operation 3

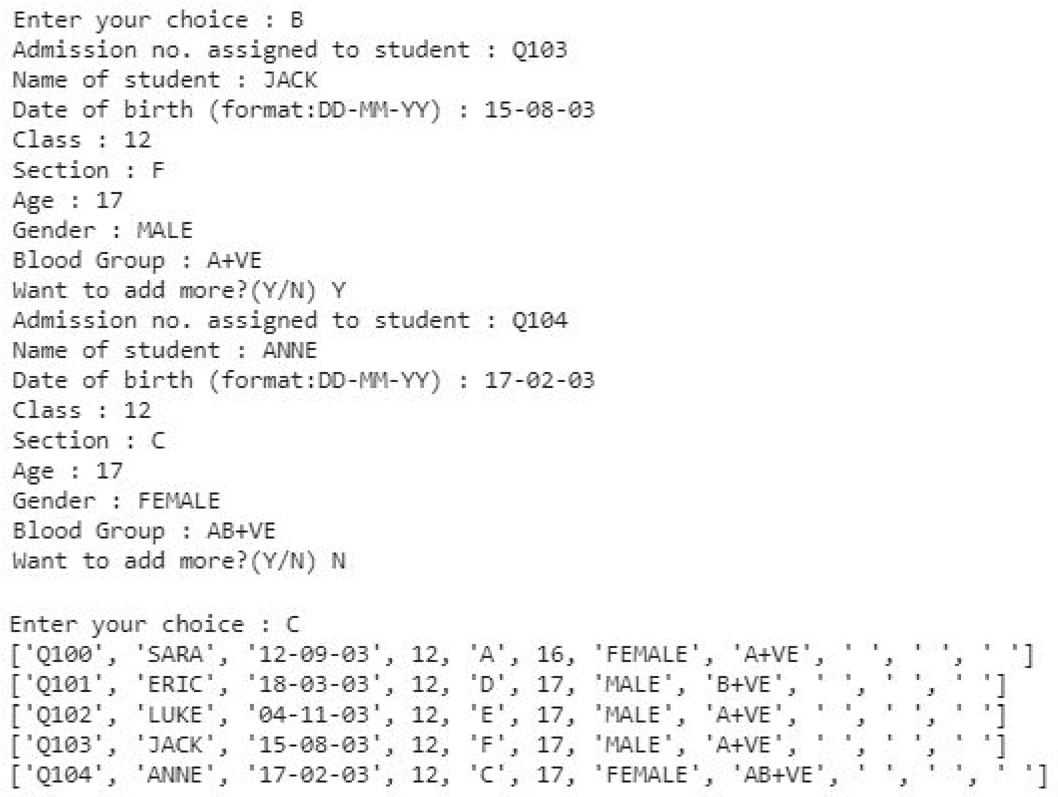
Display of list of students entered by the admin.



```
Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 16, 'FEMALE', 'A+VE', ' ', ' ', ' ']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', ' ', ' ', ' ']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', ' ', ' ', ' ']
```

Operation 4

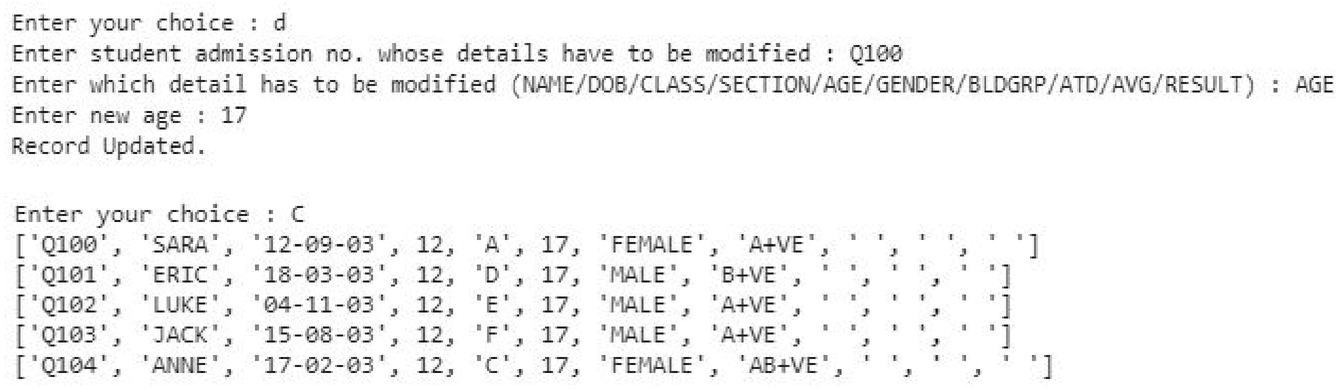
Adding 2 new students to the existing record.



```
Enter your choice : B
Admission no. assigned to student : Q103
Name of student : JACK
Date of birth (format:DD-MM-YY) : 15-08-03
Class : 12
Section : F
Age : 17
Gender : MALE
Blood Group : A+VE
Want to add more?(Y/N) Y
Admission no. assigned to student : Q104
Name of student : ANNE
Date of birth (format:DD-MM-YY) : 17-02-03
Class : 12
Section : C
Age : 17
Gender : FEMALE
Blood Group : AB+VE
Want to add more?(Y/N) N

Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 16, 'FEMALE', 'A+VE', ' ', ' ', ' ']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', ' ', ' ', ' ']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', ' ', ' ', ' ']
['Q103', 'JACK', '15-08-03', 12, 'F', 17, 'MALE', 'A+VE', ' ', ' ', ' ']
['Q104', 'ANNE', '17-02-03', 12, 'C', 17, 'FEMALE', 'AB+VE', ' ', ' ', ' ']
```

Operation 5

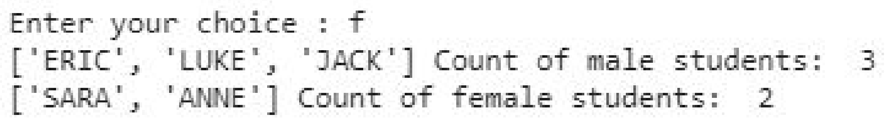
Changing Sara’s age from 16 to 17.



```
Enter your choice : d
Enter student admission no. whose details have to be modified : Q100
Enter which detail has to be modified (NAME/DOB/CLASS/SECTION/AGE/GENDER/BLDGRP/ATD/AVG/RESULT) : AGE
Enter new age : 17
Record Updated.

Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 17, 'FEMALE', 'A+VE', ' ', ' ', ' ']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', ' ', ' ', ' ']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', ' ', ' ', ' ']
['Q103', 'JACK', '15-08-03', 12, 'F', 17, 'MALE', 'A+VE', ' ', ' ', ' ']
['Q104', 'ANNE', '17-02-03', 12, 'C', 17, 'FEMALE', 'AB+VE', ' ', ' ', ' ']
```

Operation 6

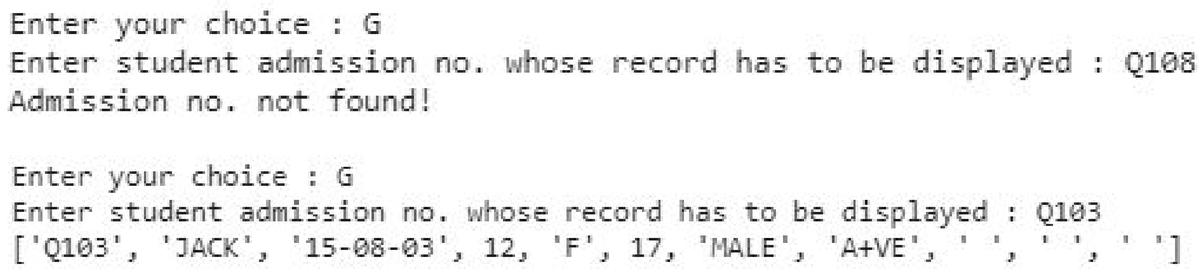
Viewing no. of male and female students.



```
Enter your choice : f
['ERIC', 'LUKE', 'JACK'] Count of male students:  3
['SARA', 'ANNE'] Count of female students:  2
```

Operation 7

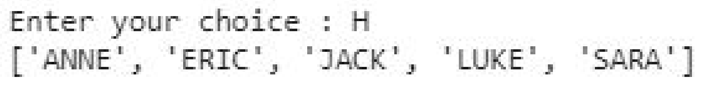
Searching for a particular student.



```
Enter your choice : G
Enter student admission no. whose record has to be displayed : Q108
Admission no. not found!

Enter your choice : G
Enter student admission no. whose record has to be displayed : Q103
['Q103', 'JACK', '15-08-03', 12, 'F', 17, 'MALE', 'A+VE', ' ', ' ', ' ']
```

Operation 8

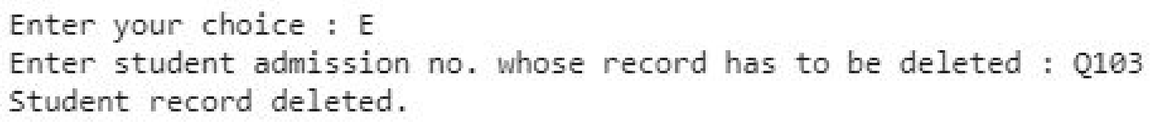
Viewing alphabetical order of student names.



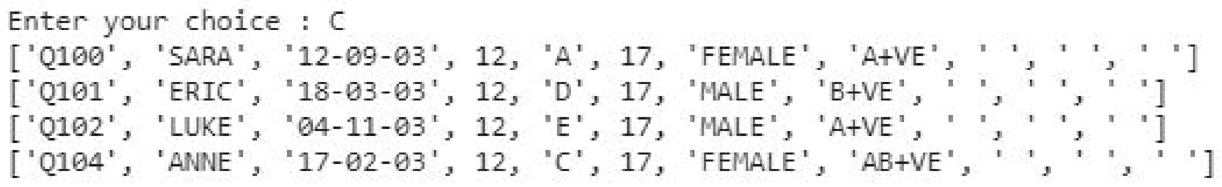
```
Enter your choice : H
['ANNE', 'ERIC', 'JACK', 'LUKE', 'SARA']
```

Operation 9

Deleting a student's record.



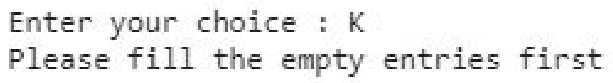
```
Enter your choice : E
Enter student admission no. whose record has to be deleted : Q103
Student record deleted.
```



```
Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 17, 'FEMALE', 'A+VE', ' ', ' ', ' ']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', ' ', ' ', ' ']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', ' ', ' ', ' ']
['Q104', 'ANNE', '17-02-03', 12, 'C', 17, 'FEMALE', 'AB+VE', ' ', ' ', ' ']
```

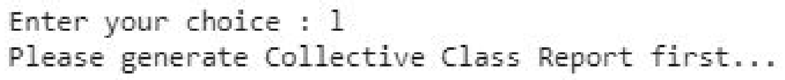
Operation 10

Generating class report without entering marks of students.



```
Enter your choice : K
Please fill the empty entries first
```

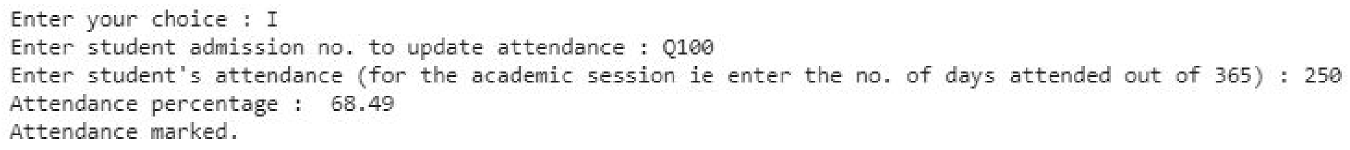
Generating rank list without generating class report.



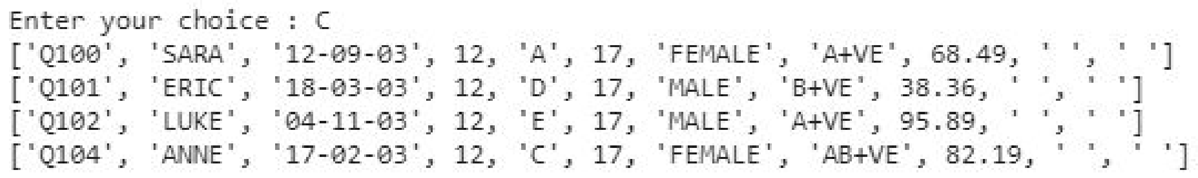
```
Enter your choice : l
Please generate Collective Class Report first...
```

Operation 11

Entering the attendance for a specific student.



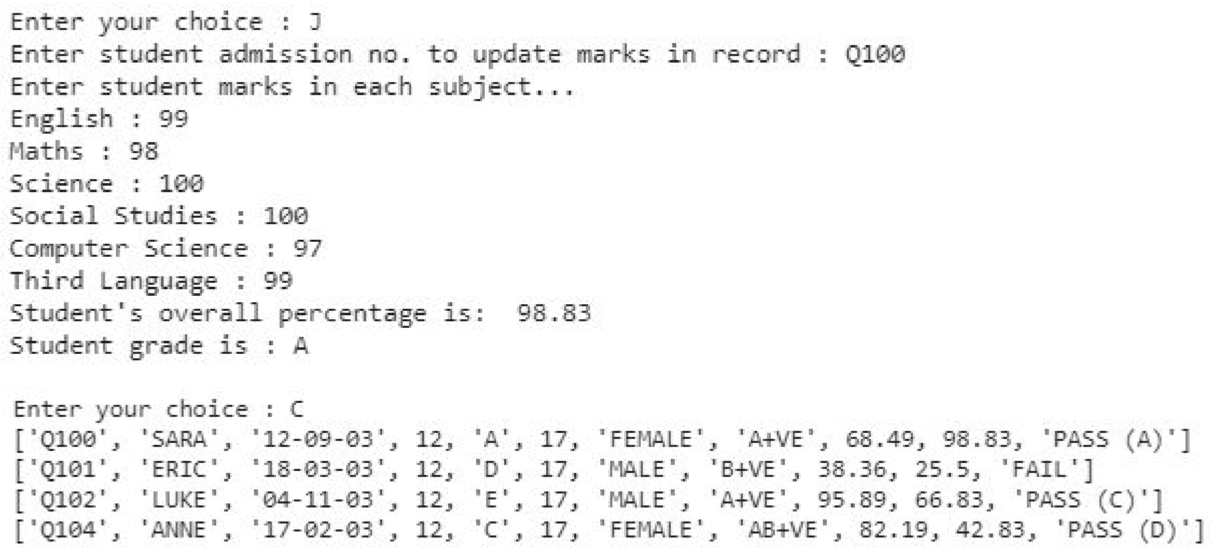
```
Enter your choice : I
Enter student admission no. to update attendance : Q100
Enter student's attendance (for the academic session ie enter the no. of days attended out of 365) : 250
Attendance percentage :  68.49
Attendance marked.
```



```
Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 17, 'FEMALE', 'A+VE', 68.49, ' ', ' ']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', 38.36, ' ', ' ']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', 95.89, ' ', ' ']
['Q104', 'ANNE', '17-02-03', 12, 'C', 17, 'FEMALE', 'AB+VE', 82.19, ' ', ' ']
```

Operation 12

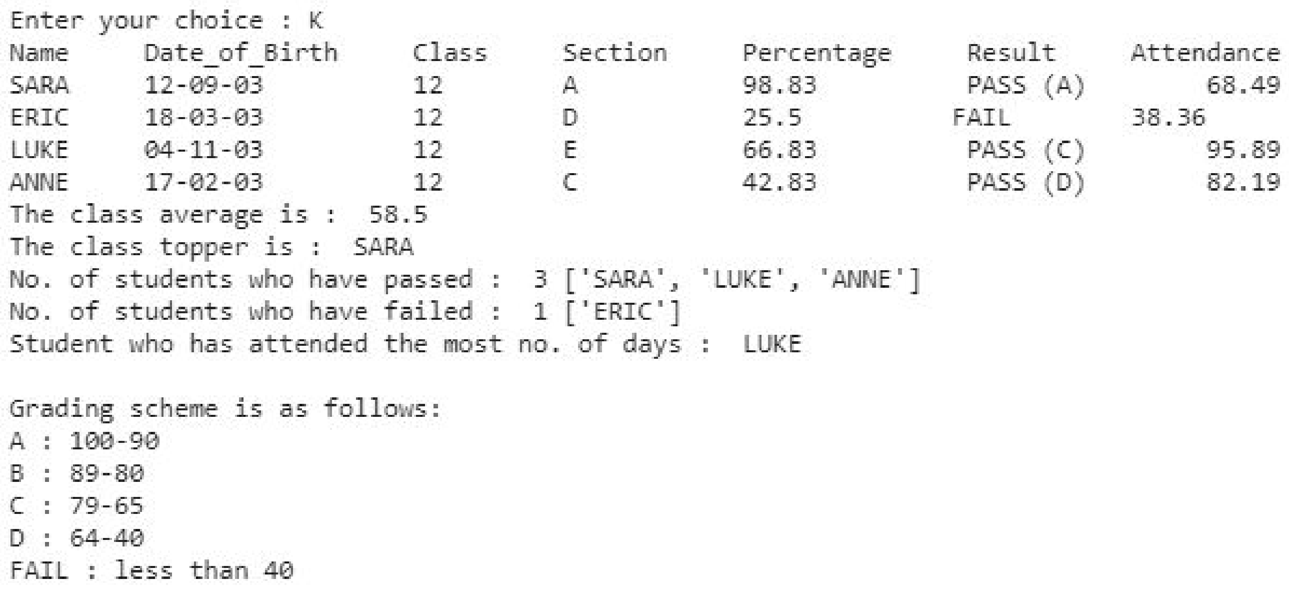
Entering the marks for a specific student.



```
Enter your choice : J
Enter student admission no. to update marks in record : Q100
Enter student marks in each subject...
English : 99
Maths : 98
Science : 100
Social Studies : 100
Computer Science : 97
Third Language : 99
Student's overall percentage is:  98.83
Student grade is : A

Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 17, 'FEMALE', 'A+VE', 68.49, 98.83, 'PASS (A)']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', 38.36, 25.5, 'FAIL']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', 95.89, 66.83, 'PASS (C)']
['Q104', 'ANNE', '17-02-03', 12, 'C', 17, 'FEMALE', 'AB+VE', 82.19, 42.83, 'PASS (D)']
```

Operation 13

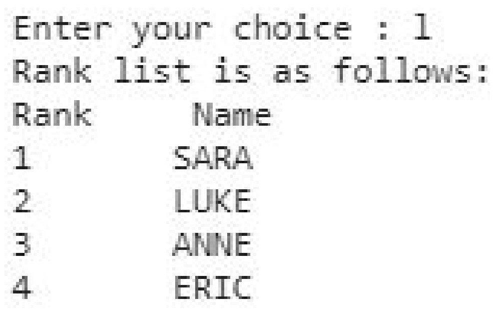
Generating collective class report.



```
Enter your choice : K
Name     Date_of_Birth      Class     Section     Percentage     Result     Attendance
SARA     12-09-03           12        A           98.83          PASS (A)        68.49
ERIC     18-03-03           12        D           25.5          FAIL         38.36
LUKE     04-11-03           12        E           66.83          PASS (C)        95.89
ANNE     17-02-03           12        C           42.83          PASS (D)        82.19
The class average is :  58.5
The class topper is :  SARA
No. of students who have passed :  3 ['SARA', 'LUKE', 'ANNE']
No. of students who have failed :  1 ['ERIC']
Student who has attended the most no. of days :  LUKE

Grading scheme is as follows:
A : 100-90
B : 89-80
C : 79-65
D : 64-40
FAIL : less than 40
```

Operation 14

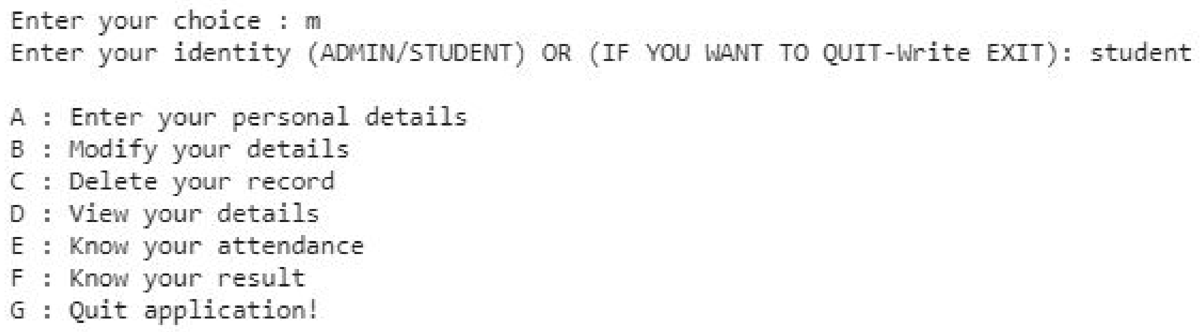
Generating rank list.



```
Enter your choice : l
Rank list is as follows:
Rank      Name
1        SARA
2        LUKE
3        ANNE
4        ERIC
```

Operation 15

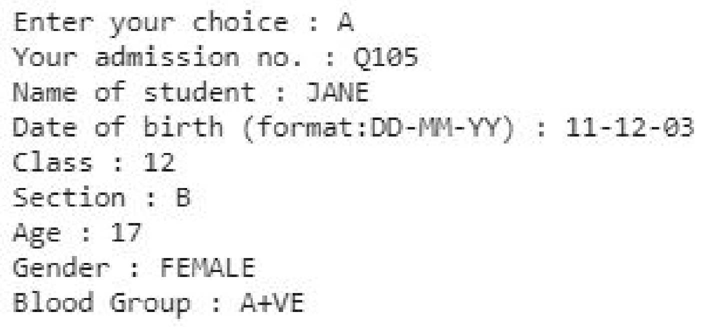
Quitting admin login and using student login system.



```
Enter your choice : m
Enter your identity (ADMIN/STUDENT) OR (IF YOU WANT TO QUIT-Write EXIT): student

A : Enter your personal details
B : Modify your details
C : Delete your record
D : View your details
E : Know your attendance
F : Know your result
G : Quit application!
```

Operation 16

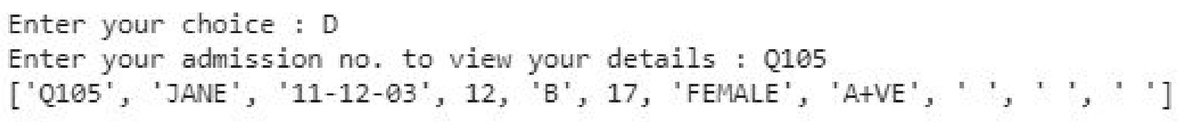
Entering of personal details by a student.



```
Enter your choice : A
Your admission no. : Q105
Name of student : JANE
Date of birth (format:DD-MM-YY) : 11-12-03
Class : 12
Section : B
Age : 17
Gender : FEMALE
Blood Group : A+VE
```

Operation 17

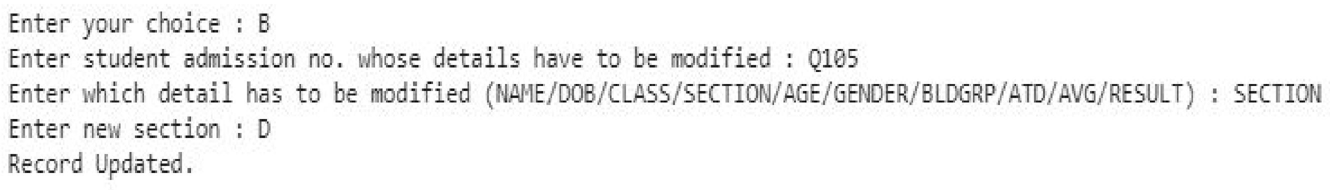
Viewing of personal details.



```
Enter your choice : D
Enter your admission no. to view your details : Q105
['Q105', 'JANE', '11-12-03', 12, 'B', 17, 'FEMALE', 'A+VE', ' ', ' ', ' ']
```

Operation 18

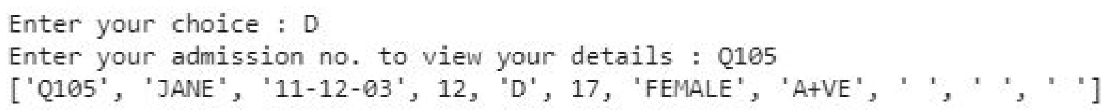
Modifying personal details.



```
Enter your choice : B
Enter student admission no. whose details have to be modified : Q105
Enter which detail has to be modified (NAME/DOB/CLASS/SECTION/AGE/GENDER/BLDGRP/ATD/AVG/RESULT) : SECTION
Enter new section : D
Record Updated.
```

Operation 19

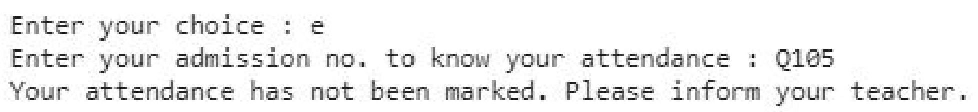
Viewing modified personal details.



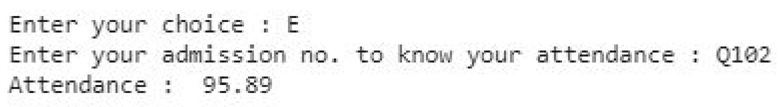
```
Enter your choice : D
Enter your admission no. to view your details : Q105
['Q105', 'JANE', '11-12-03', 12, 'D', 17, 'FEMALE', 'A+VE', ' ', ' ', ' ']
```

Operation 20

Viewing attendance.



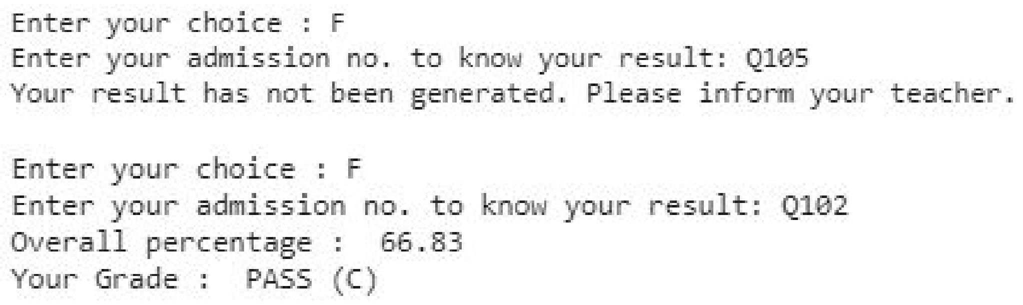
```
Enter your choice : e
Enter your admission no. to know your attendance : Q105
Your attendance has not been marked. Please inform your teacher.
```



```
Enter your choice : E
Enter your admission no. to know your attendance : Q102
Attendance :  95.89
```

Operation 21

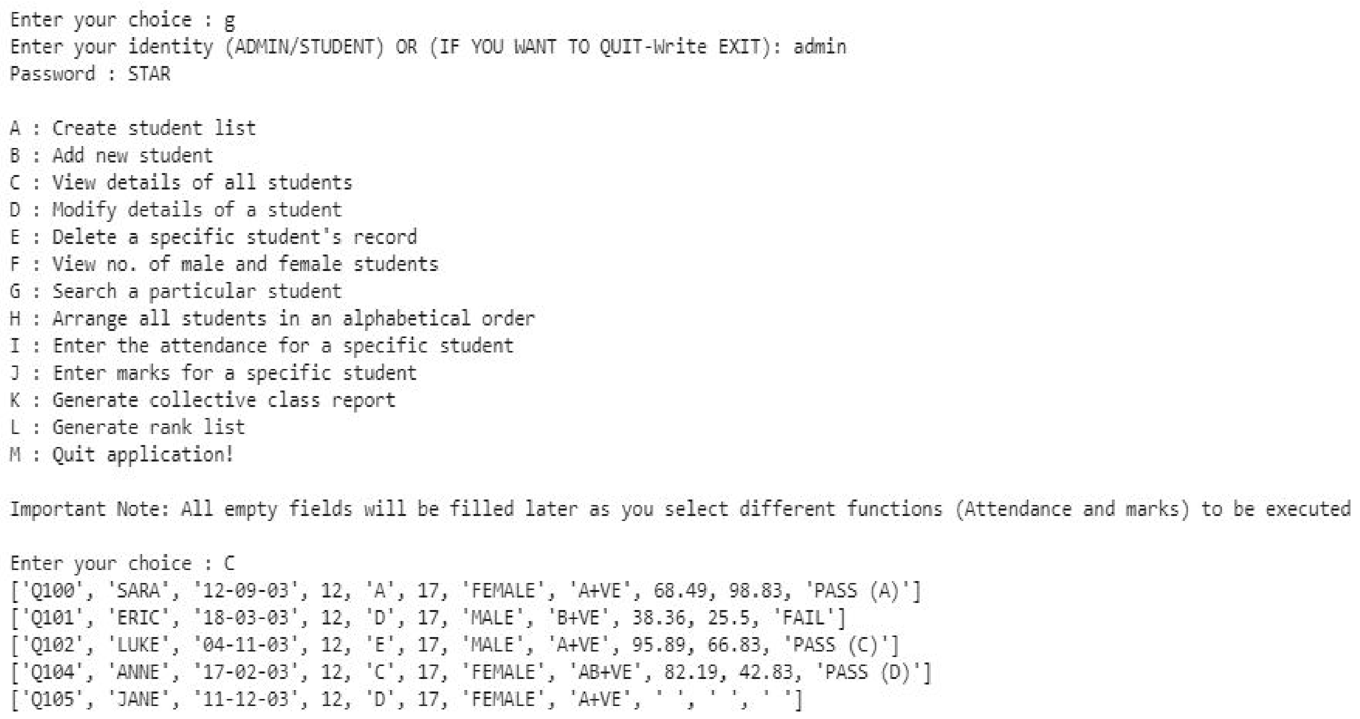
Viewing result.



```
Enter your choice : F
Enter your admission no. to know your result: Q105
Your result has not been generated. Please inform your teacher.

Enter your choice : F
Enter your admission no. to know your result: Q102
Overall percentage :  66.83
Your Grade :  PASS (C)
```

Operation 22

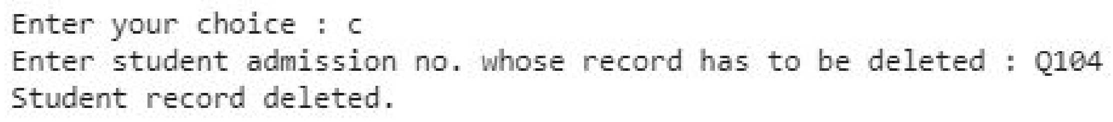
Quitting student login and viewing the entire list through the admin login system.



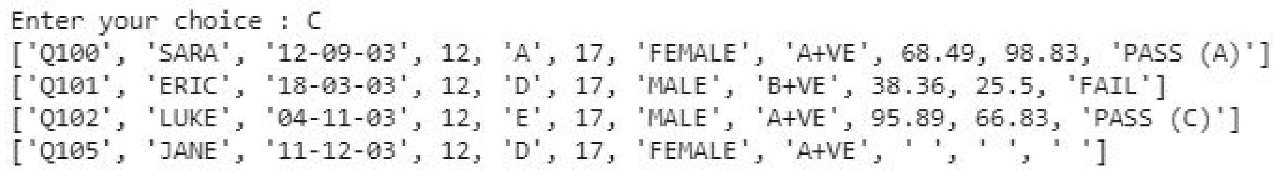
```
Enter your choice : g
Enter your identity (ADMIN/STUDENT) OR (IF YOU WANT TO QUIT-Write EXIT): admin
Password : STAR

A : Create student list
B : Add new student
C : View details of all students
D : Modify details of a student
E : Delete a specific student's record
F : View no. of male and female students
G : Search a particular student
H : Arrange all students in an alphabetical order
I : Enter the attendance for a specific student
J : Enter marks for a specific student
K : Generate collective class report
L : Generate rank list
M : Quit application!

Important Note: All empty fields will be filled later as you select different functions (Attendance and marks) to be executed

Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 17, 'FEMALE', 'A+VE', 68.49, 98.83, 'PASS (A)']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', 38.36, 25.5, 'FAIL']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', 95.89, 66.83, 'PASS (C)']
['Q104', 'ANNE', '17-02-03', 12, 'C', 17, 'FEMALE', 'AB+VE', 82.19, 42.83, 'PASS (D)']
['Q105', 'JANE', '11-12-03', 12, 'D', 17, 'FEMALE', 'A+VE', ' ', ' ', ' ']
```

Operation 23

Deleting record through student login system.



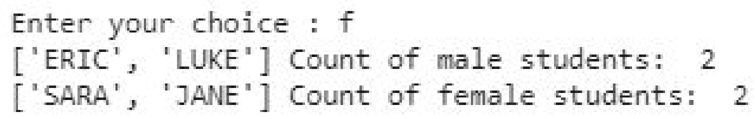
```
Enter your choice : c
Enter student admission no. whose record has to be deleted : Q104
Student record deleted.
```



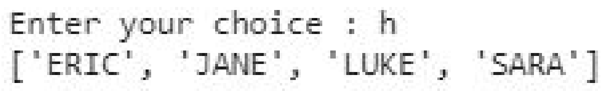
```
Enter your choice : C
['Q100', 'SARA', '12-09-03', 12, 'A', 17, 'FEMALE', 'A+VE', 68.49, 98.83, 'PASS (A)']
['Q101', 'ERIC', '18-03-03', 12, 'D', 17, 'MALE', 'B+VE', 38.36, 25.5, 'FAIL']
['Q102', 'LUKE', '04-11-03', 12, 'E', 17, 'MALE', 'A+VE', 95.89, 66.83, 'PASS (C)']
['Q105', 'JANE', '11-12-03', 12, 'D', 17, 'FEMALE', 'A+VE', ' ', ' ', ' ']
```

Operation 24

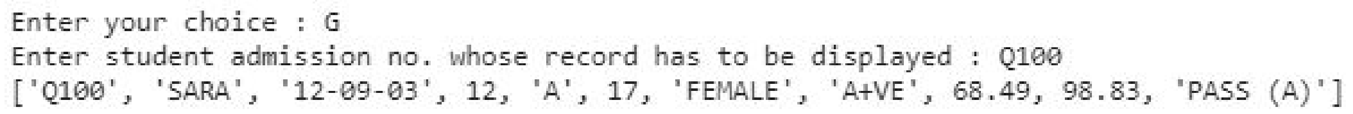
Testing some options with the new student ‘Jane’.



```
Enter your choice : f
['ERIC', 'LUKE'] Count of male students:  2
['SARA', 'JANE'] Count of female students:  2
```



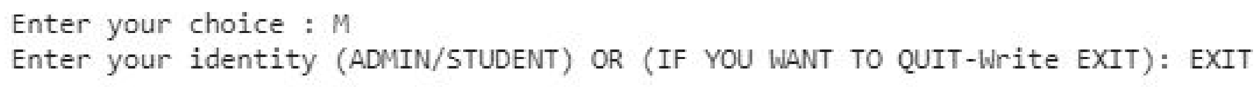
```
Enter your choice : h
['ERIC', 'JANE', 'LUKE', 'SARA']
```



```
Enter your choice : G
Enter student admission no. whose record has to be displayed : Q100
['Q100', 'SARA', '12-09-03', 12, 'A', 17, 'FEMALE', 'A+VE', 68.49, 98.83, 'PASS (A)']
```

Operation 25

Final exit out of application.



```
Enter your choice : M
Enter your identity (ADMIN/STUDENT) OR (IF YOU WANT TO QUIT-Write EXIT): EXIT
```

Hardware & Software Requirement

Hardware Requirement

PC/Laptop/MacBook with
Intel core/i3/i5/i7 or any equivalent
With at least 2 GB RAM
10 MB free space on Hard Disk
LCD/LED

Operating System & Compiler

MS Windows/Ubuntu/MacOS

Python IDLE 3.8.2
OR
colab.research.google.com (gmail account)