

LAB ASSIGNMENT - 4

ARZOO KHAN

(20MCA011)

MCA

(Semester - II)



Due Date: June 20 , 2021

—

CSC26: Lab – III (OOP)

—

Dr. S. Zeeshan Hussain

(Associate Professor, D/o Computer Science)

1. Class student contains roll number, name and course as data members and Input_student() and display_student() as member functions. A derived class exam is created from the class student with publicly inherited. The derived class contains mark1 mark2, mark3 as marks of three subjects and input_marks() and display_result() as member functions. Create an array of objects of the exam class and display the result of 5 students.

SOURCE CODE:

```
#include <iostream>
#include <string>
#include <cstring>           // strcmpi()
#include <iomanip>           // setw()
#include <cstdlib>           // system()
#include <conio.h>           // getch()
#include <cmath>             // ceil()
#define BOLDCYAN "\033[1m\033[36m" // Colour Code
#define RESET "\033[0m"         // Reset Colour Code
#define MAX_STUDENTS 15
#define MAX_SUBJECTS 3

using namespace std;

class Student;
class Exam;
void setTenRecords(Exam *);

// Base Class
class Student
{
private:
    string rollNo;
    string name;
    string courseName;
    string dob;
    string mobile;
    string email;
    static int NoOfStudents;

public:
    Student();
    void setStudentRecord(string, string, string, string, string);
```



```

void inputStudentRecord();
void modifyStudentRecord();
string getRollNo() { return rollNo; }
string getStudentName() { return name; }
string getCourseName() { return courseName; }
string getDOB() { return dob; }
string getMobile() { return mobile; }
string getEmail() { return email; }
// static member function
static int get_NoOfStudents() { return NoOfStudents; }
};

// Derived Class
class Exam : public Student
{
private:
    float marks[MAX_SUBJECTS];
    string paper[MAX_SUBJECTS];
    static float maxMarks;
    string getGrade(float);

public:
    static float getMaxMarks() { return maxMarks; }
    Exam();
    void inputData();
    void modifyMarks();
    void setData(string, string, string, string, string, string, float, string,
float, string, float);
    void printResult();
    friend void showStudentRecord(Exam *);
};

// Static Data Member of Exam class
float Exam ::maxMarks = 200;

// Non-Parameterized Constructor
Exam ::Exam()
{
    for (int i = 0; i < MAX_SUBJECTS; i++)
    {
        paper[i] = '\0';
    }
}

```

```

        marks[i] = 0;
    }
}

// Member function of Exam Class
void Exam ::inputData()
{
    inputStudentRecord();

    cout << "\n Enter Exam Details:" << endl
         << "-----" << endl;
    for (int i = 0; i < MAX_SUBJECTS; i++)
    {
        cout << " Name of Paper " << i + 1 << ": ";
        cin >> paper[i];
        cout << " Marks obtained out of " << maxMarks << ": ";
        cin >> marks[i];
    }
    cout << "-----" << endl;
    // Message to the user
    cout << "\n Student details with roll no. "
         << getRollNo()
         << " has been added successfully!\n\n";
}

```

```

// Member Function of Exam Class
void Exam ::modifyMarks()
{
    int choice;
    chooseAgain: // If user enters a invalid choice
    cout << "\n Choose subject to modify its marks:"
         << "\n 1. " << paper[0]
         << "\n 2. " << paper[1]
         << "\n 3. " << paper[3]
         << "\n 4. All Subject's Marks"
         << "\n\n Enter your choice: ";
    cin >> choice;
    switch (choice)
    {
        case 1:

```

```

    case 2:

    case 3:      cout << "\n Enter Marks Obtained out of 200 in "
<< paper[choice-1] << ": ";
                cin >> marks[choice-1];
                break;

    case 4:      for (int i = 0; i < MAX_SUBJECTS; i++)
                {
                    cout << "\n Enter Marks Obtained out of 200 in "
                        << paper[i] << ": ";
                    cin >> marks[i];
                }
                break;

    default:     cout << "Invalid Choice!";
                system("cls");
                goto chooseAgain;
}

// Message to the user
cout << "\n Student's Marks details with roll no. "
    << getRollNo()
    << " has been modified successfully!\n\n";
}

// Member function of Exam Class
void Exam ::setData(string name, string course, string dob,
                    string mobile, string email,
                    string paper1, float marks1,
                    string paper2, float marks2,
                    string paper3, float marks3)
{
    setStudentRecord(name, course, dob, mobile, email);
    paper[0] = paper1;
    marks[0] = marks1;
    paper[1] = paper2;
    marks[1] = marks2;
    paper[2] = paper3;
    marks[2] = marks3;
}

```

```
// Member function of Exam Class
```

```
void Exam ::printResult()  
{
```

```
    cout << "\n\n\n"  
        << RESET  
        << "=====  
        << "=====  
        << BOLDCYAN  
        << "    Roll No. " << getRollNo() << "    "  
        << RESET  
        << "=====  
        << "=====  
        << "\n\n"  
        << BOLDCYAN;
```

```
// Display Student's Info.
```

```
cout << "    Name          : " << getStudentName() << endl  
    << "    Roll No       : " << getRollNo() << endl  
    << "    Course Name    : " << getCourseName() << endl  
    << "    Date of Birth   : " << getDOB() << endl  
    << "    Mobile No       : " << getMobile() << endl  
    << "    Email           : " << getEmail() << endl  
    << endl  
    << "-----"  
    << "-----"  
    << "-----" << endl  
    << left  
    << setw(13) << "S.No. "  
    << setw(19) << "Paper Name"  
    << setw(19) << "Total Marks"  
    << setw(23) << "Marks Obtained"  
    << setw(19) << "Percentage"  
    << setw(19) << "Grade"  
    << endl  
    << "-----"  
    << "-----"  
    << "-----" << endl;
```

```
float sum_marksObtained = 0, sum_maxMarks = MAX_SUBJECTS * maxMarks;  
for (int i = 0; i < MAX_SUBJECTS; i++)
```

```

    sum_marksObtained += marks[i];

    // Loop to display details of exam for each subject
    for (int i = 0; i < MAX_SUBJECTS; i++)
    {
        cout << left
            << setw(13) << i + 1
            << setw(19) << paper[i]
            << setw(19) << (int)ceil(maxMarks)
            << setw(24) << (int)ceil(marks[i])
            << (marks[i] / maxMarks * 100) << " %" << setw(12)
            << " "
            << setw(19) << getGrade(marks[i] / maxMarks * 100)
            << endl;
    }

    // Display the overall result of the student.

    cout << "-----"
        << "-----"
        << "-----" << endl
        << left << setw(32) << "Total"
        << setw(19) << (int)ceil(sum_maxMarks)
        << setw(24) << (int)ceil(sum_marksObtained)
        << fixed << setprecision(2)
        << (sum_marksObtained / sum_maxMarks * 100) << " %" << setw(12)
        << " "
        << setw(18) << getGrade(sum_marksObtained / sum_maxMarks * 100)
        << endl
        << "-----"
        << "-----"
        << "-----" << endl
        << endl;
}

// Member function of Exam class to get grade value by passing
// percentage value as argument
string Exam ::getGrade(float percentage)
{
    if (percentage >= 90.0)
        return ("A++");

```

```

else if (percentage < 90.0 && percentage >= 80.0)
    return ("A+");
else if (percentage < 80.0 && percentage >= 70.0)
    return ("A");
else if (percentage < 70.0 && percentage >= 60.0)
    return ("B");
else if (percentage < 60.0 && percentage >= 50.0)
    return ("C");
else if (percentage < 50.0 && percentage >= 40.0)
    return ("D");
else
    return ("F");
}

// Data member of Student class
int Student::NoOfStudents = 0;

// Non-Parameterized Constructor of Student Class
Student ::Student()
{
    rollNo = "\0";
    name = '\0';
    courseName = '\0';
    dob = '\0';
    mobile = '\0';
    email = '\0';
}

// Member function of Student Class
void Student ::setStudentRecord(string name, string courseName,
                                string dob, string mobile, string email)
{
    // Each time a record is set for a new student, the
    // NoOfStudent (Static Data Member) is incremented.
    NoOfStudents++;

    this->name = name;
    this->courseName = courseName;
    this->dob = dob;
    this->mobile = mobile;
    this->email = email;
}

```



```

// Initializing the Student Roll No.
this->rollNo = "STUD";
if (NoOfStudents <= 9)
{
    this->rollNo += "00";
    this->rollNo += to_string(NoOfStudents);
}
else if (NoOfStudents > 9 && NoOfStudents <= 99)
{
    this->rollNo += '0';
    this->rollNo += to_string(NoOfStudents);
}
else
{
    this->rollNo += to_string(NoOfStudents);
}
}

// Member function of Student Class
void Student ::inputStudentRecord()
{
    // Input Student Details from the user.

    cout << "\n Enter Student Details:" << endl
         << "-----" << endl;

    cout << " Name: ";
    cin.ignore(1, '\n');
    getline(cin, name, '\n');

    cout << " Course Name: ";
    getline(cin, courseName);

    cout << " Date Of Birth (DD-MM-YYYY): ";
    getline(cin, dob);

    cout << " Mobile No: ";
    cin >> mobile;

    cout << " Email: ";

```

```

cin >> email;

cout << "-----" << endl;

// Each time a record is set for a new student, the
// NoOfStudent (Static Data Member) is incremented.
NoOfStudents++;

// Initializing the Student Roll No.
this->rollNo = "STUD";
if (NoOfStudents <= 9)
{
    this->rollNo += "00";
    this->rollNo += to_string(NoOfStudents);
}
else if (NoOfStudents > 9 && NoOfStudents <= 99)
{
    this->rollNo += '0';
    this->rollNo += to_string(NoOfStudents);
}
else
{
    this->rollNo += to_string(NoOfStudents);
}
}

// Member function of Student Class
void Student ::modifyStudentRecord()
{
    int choice;
    chooseAgain: // If user enters a invalid choice
    cout << "\n Choose a field to modify:"
        << "\n 1. Name"
        << "\n 2. Date Of Birth"
        << "\n 3. Contact No"
        << "\n 4. Email"
        << "\n 5. All Details"
        << "\n\n Enter your choice: ";
    cin >> choice;
    switch (choice)
    {

```

```

case 1:      cout << "\n Enter Name: ";
             cin >> name;
             break;

case 2:      cout << "\n Enter Date Of Birth (DD-MM-YYYY): ";
             cin >> dob;
             break;

case 3:      cout << "\n Enter Contact No: ";
             cin >> mobile;
             break;

case 4:      cout << "\n Enter Email Address: ";
             cin >> email;
             break;

case 5:      this->inputStudentRecord();
             break;

default:     cout << "Invalid Choice!";
             system("cls");
             goto chooseAgain;
}

```

// Message to the user

```

cout << "\n Student details with roll no. "
     << rollNo
     << " has been modified successfully!\n\n";

```

```

}

```

// Friend Function of Student Class

```

void showStudentRecord(Exam *stud)
{

```

*// Calling Static member function to get the No. of Students currently
enrolled in the course.*

```

int studentCount = Student ::get_NoOfStudents();

```

```

int choice, s_no;

```

```

string rollNo, courseName;

```

```

bool found; // to keep a record, if the required details are found or not

```

```

while (true)
{
    rollNo = '\0';
    courseName = '\0';
    cout << "\n 1. Details of students for a perticular Course"
        << "\n 2. Details of student by entering Roll No."
        << "\n 3. Show details of all the students"
        << "\n 4. Main menu"
        << "\n 5. Exit the program"
        << "\n\n Enter your choice: ";
    cin >> choice;

    if (choice == 1)
    {
        cout << "\n Enter Course Name: ";
        cin.ignore(1, '\n');
        getline(cin, courseName, '\n');
    }
    if (choice == 2)
    {
        cout << "\n Enter Roll No: ";
        cin.ignore(1, '\n');
        getline(cin, rollNo, '\n');
    }

    if (choice != 4 && choice != 5)
        cout << endl
            << BOLDCYAN
            << endl
            << right
            << setw(62) << "STUDENT DATA"
            << endl
            << "-----"
            << "-----"
            << "-----" << endl
            << left << " "
            << setw(8) << "S.No."
            << setw(15) << "Roll No"
            << setw(15) << "Name"
            << setw(20) << "Course"
            << setw(20) << "DOB"

```

```

<< setw(20) << "Mobile"
<< setw(30) << "Email"
<< endl
<< "-----"
<< "-----"
<< "-----" << endl
<< RESET;

```

```

switch (choice)
{

```

```

case 1:    s_no = 1;
           found = false;
           for (int i = 0; i < studentCount; i++)
           {
               if (courseName == stud[i].getCourseName())
               {
                   cout << BOLDCYAN
                       << "  "
                       << setw(8) << s_no
                       << setw(15) << stud[i].getRollNo()
                       << setw(15) << stud[i].getStudentName()
                       << setw(20) << stud[i].getCourseName()
                       << setw(20) << stud[i].getDOB()
                       << setw(20) << stud[i].getMobile()
                       << setw(30) << stud[i].getEmail() << endl
                       << RESET;

                   s_no++;
                   found = true;
               }
           }

```

```

// If value of found is false then it means no data
// has been found with Course Name entered by the
// user otherwise it is found.
if (found == false)
    cout << "\n No Data Found!\n\n";
break;

```



```

case 2:    s_no = 1;
           found = false;
           for (int i = 0; i < studentCount; i++)
           {
               if (rollNo == stud[i].getRollNo())
               {
                   cout << BOLDCYAN
                       << "  "
                       << setw(8) << s_no
                       << setw(15) << stud[i].getRollNo()
                       << setw(15) << stud[i].getStudentName()
                       << setw(20) << stud[i].getCourseName()
                       << setw(20) << stud[i].getDOB()
                       << setw(20) << stud[i].getMobile()
                       << setw(30) << stud[i].getEmail() << endl
                       << RESET;

                   s_no++;
                   found = true;
                   break;
               }
           }

           // If value of found is false then it means no data has
           // been found with Roll No entered by the user,
           // otherwise it is found.
           if (!found)
               cout << "\n No Data Found!\n\n";
           break;

case 3:    if (studentCount == 0)
           cout << "\n No Data Found!\n\n";
           else
           {
               // This loop will display details of all the students
               for (int i = 0; i < studentCount; i++)
               {
                   cout << BOLDCYAN
                       << "  "
                       << setw(8) << i + 1
                       << setw(15) << stud[i].getRollNo()

```

```

        << setw(15) << stud[i].getStudentName()
        << setw(20) << stud[i].getCourseName()
        << setw(20) << stud[i].getDOB()
        << setw(20) << stud[i].getMobile()
        << setw(30) << stud[i].getEmail() << endl
        << RESET;
    }
}
break;

case 4:    system("cls");
          return;

case 5:    system("cls");
          exit(0);

default:   cout << "\n Invalid Choice!"
           << "\n\n Press any key to continue...";
           getch();
           cin.ignore(1, '\n');
           system("cls");
}

cout << BOLDCYAN
     << "-----"
     << "-----"
     << "-----" << endl
     << endl
     << RESET;
cout << " Press any key to continue...";
getch();
system("cls");
}
}

// Driver Code
int main(void)
{
    system("cls");

```

```

Exam stud[MAX_STUDENTS];
setTenRecords(stud);

int choice;
char Exit;
bool found;
int studentCount; // to get current count of students in records
string rollNo,courseName; /* to take input from user and search
                           for that in student records */

while (true)
{
    // List of functions this program can perform
    // (Display in output console)
    cout << "\n 1. Show Result of a Student (By Entering Roll No)"
        << "\n 2. Show Result of all Students (By Entering Course Name)"
        << "\n 3. Show Result of all Students "
        << "\n 4. Show Details of Students"
        << "\n 5. Add Student Details"
        << "\n 6. Modify Student Details"
        << "\n 7. Modify Student Marks"
        << "\n 8. Exit" << endl;

    // Input User's Choice
    cout << "\n Enter your choice: ";
    cin >> choice;

    system("cls");

    found = false;
    switch (choice)
    {

    case 1: { // Calling Static member function to get the No. of
              // Students currently enrolled in the course.
              studentCount = Student ::get_NoOfStudents();

              // Taking Roll No. as input from the user whose
              // details he/she wants to display.
              cout << "\n Enter Roll No of the Student: ";

```

```

cin >> rollNo;

int i;
// Searching the Roll No. in records entered by user.
for (i = 0; i < studentCount; i++)
{
    if (rollNo == stud[i].getRollNo())
    {
        // If roll no. is found then the result for
        // that will be printed
        cout << BOLDCYAN;
        stud[i].printResult(); // Calling Member Function
        cout << RESET;
        found = true;
        break;
    }
}

// If found variable have the value false then it means
// no student has been found with roll no. entered by
// the user, otherwise it is found.
if (!found)
    cout << "\n No student exist in records with roll no "
        << rollNo << "\n\n";

cout << " Press any key to continue...";
getch();
system("cls");
break;
}

case 2: { // Calling Static member function to get the No. of
         // Students currently enrolled in the course.
         studentCount = Student ::get_NoOfStudents();

         // Taking Course Name as input from the user whose
         // result he/she wants to display.
         cout << "\n Enter Course Name: ";
         cin.ignore(1, '\n');
         getline(cin, courseName, '\n');

```

```

bool found = false;
int i;
// Searching the Course Name in records entered by user
for (i = 0; i < studentCount; i++)
{
    if (courseName == stud[i].getCourseName())
    {
        // If Course Name is found then the result
        // for that will be printed
        cout << BOLDCYAN;
        stud[i].printResult(); // Calling Member Function
        cout << RESET;

        found = true;
    }
}

// If value of found is false then it means no data
// has been found with Course Name entered by the
// user, otherwise it is found.
if (!found)
    cout << "\n No Data Found!\n\n";

cout << " Press any key to continue...";
getch();
system("cls");
break;
}

```

```

case 3: // Calling Static member function to get the No. of
        // Students currently enrolled in the course.
        studentCount = Student ::get_NoOfStudents();

        // This loop will Print the Result of all Students
        for (int i = 0; i < studentCount; i++)
        {
            cout << BOLDCYAN;
            stud[i].printResult(); // Calling Member Function
            cout << RESET;
        }

```



```

    }
    cout << " Press any key to continue...";
    getch();
    system("cls");
    break;

case 4:    // Calling friend function to display records of all
           // the students.
           showStudentRecord(stud);
           break;

case 5:    // Calling Static member function to get the No. of
           // Students currently enrolled in the course.
           studentCount = Student ::get_NoOfStudents();

           // Showing the number of students currently
           // enrolled in the course
           cout << "\n Total number of Students currently in all Courses::"
                << studentCount << endl;

           // Calling member function to input the details of the
           // student from user.
           stud[studentCount].inputData();

           cout << " Press any key to continue...";
           getch();
           system("cls");
           break;

case 6:    // Calling Static member function to get the No. of
           // Students currently enrolled in the course.
           studentCount = Student ::get_NoOfStudents();
           cout << "Enter Roll No: ";
           cin.ignore(1, '\n');
           getline(cin, rollNo, '\n');

```

```

for (int i = 0; i < studentCount; i++)
{
    if (rollNo == stud[i].getRollNo())
    {
        stud[i].modifyStudentRecord();
        found = true;
        break;
    }
}
// If found variable have the value false then it means
// no student has been found with roll no. entered by
// the user, otherwise it is found.
if (!found)
    cout << "\n No student exist in records with roll no "
        << rollNo << "\n\n";

cout << " Press any key to continue...";
getch();
system("cls");
break;

```

case 7: *// Calling Static member function to get the No. of*
// Students currently enrolled in the course.
studentCount = Student ::get_NoOfStudents();

```

cout << "Enter Roll No: ";
cin.ignore(1, '\n');
getline(cin, rollNo, '\n');
for (int i = 0; i < studentCount; i++)
{
    if (rollNo == stud[i].getRollNo())
    {
        stud[i].modifyMarks();
        found = true;
        break;
    }
}
// If found variable have the value false then it means
// no student has been found with roll no. entered by
// the user, otherwise it is found.
if (!found)

```

```

        cout << "\n No student exist in record with roll no "
              << rollNo << "\n\n";

        cout << " Press any key to continue...";
        getch();
        system("cls");
        break;

    case 8:        system("cls");
                  return 0;

    default:      cout << "\n Invalid Choice!"
                  << "\n\n Press any key to continue...";
                  getch();
                  system("cls");
            }
        }
    }

void setTenRecords(Exam *stud)
{
    // Initializing Ten Students Records

    stud[0].setData("Aleena", "MCA", "25-02-1996",
                    "9785658456", "aleena@gmail.com",
                    "OOP", 157, "ADS", 159, "SE", 147);

    stud[1].setData("Akanksha", "MCA", "01-11-1999",
                    "9867565867", "akanksha@gmail.com",
                    "OOP", 159, "ADS", 151, "SE", 167);

    stud[2].setData("Rahul", "MCA", "12-08-1998",
                    "6787868787", "rahul@gmail.com",
                    "OOP", 134, "ADS", 132, "SE", 148);

    stud[3].setData("Karan", "BSc", "28-01-1999",
                    "8985675586", "karan@gmail.com",
                    "Physics", 191, "Chemistry", 163,
                    "Mathematics", 183);

    stud[4].setData("Pallavi", "BSc", "28-05-1998",

```

```

        "8776765755", "pallavi@gmail.com",
        "Physics", 157, "Chemistry", 110,
        "Mathematics", 179);

stud[5].setData("Pragati", "BSc", "23-03-1998",
        "9867654554", "pragati@gmail.com",
        "Physics", 157, "Chemistry", 159,
        "Mathematics", 147);

stud[6].setData("Farhaz", "MCA", "30-04-1997",
        "8944567454", "farhaz@gmail.com",
        "OOP", 157, "ADS", 159, "SE", 147);

stud[7].setData("Arman", "BSc", "31-01-2000",
        "7896768776", "arman@gmail.com",
        "Physics", 157, "Chemistry", 159,
        "Mathematics", 147);

stud[8].setData("Vishal", "BSc", "20-09-1999",
        "6785677657", "vishal@gmail.com",
        "Physics", 157, "Chemistry", 159,
        "Mathematics", 147);

stud[9].setData("Anjali", "BSc", "19-05-1995",
        "7698699786", "anjali@gmail.com",
        "Physics", 157, "Chemistry", 159,
        "Mathematics", 147);
}

```

OUTPUT

```

C:\Arzoo\JMI_MCA\Sem-2\OOP\A4\Q1.exe

1. Show Result of a Student (By Entering Roll No)
2. Show Result of all Students (By Entering Course Name)
3. Show Result of all Students
4. Show Details of Students
5. Add Student Details
6. Modify Student Details
7. Modify Student Marks
8. Exit

Enter your choice: 1_

```

Enter Roll No of the Student: STUD004

Roll No. STUD004

Name : Karan
Roll No : STUD004
Course Name : BSc
Date of Birth : 28-01-1999
Mobile No : 8985675586
Email : karan@gmail.com

S.No.	Paper Name	Total Marks	Marks Obtained	Percentage	Grade
1	Physics	200	191	95.5 %	A++
2	Chemistry	200	163	81.5 %	A+
3	Mathematics	200	183	91.5 %	A++
Total		600	537	89.50 %	A+

Press any key to continue...

1. Show Result of a Student (By Entering Roll No)
2. Show Result of all Students (By Entering Course Name)
3. Show Result of all Students
4. Show Details of Students
5. Add Student Details
6. Modify Student Details
7. Modify Student Marks
8. Exit

Enter your choice: 4

1. Details of students for a particular Course
2. Details of student by entering Roll No.
3. Show details of all the students
4. Main menu
5. Exit the program

Enter your choice: 1

Enter Course Name: MCA

STUDENT DATA

S.No.	Roll No	Name	Course	DOB	Mobile	Email
1	STUD001	Aleena	MCA	25-02-1996	9785658456	aleena@gmail.com
2	STUD002	Akanksha	MCA	01-11-1999	9867565867	akanksha@gmail.com
3	STUD003	Rahul	MCA	12-08-1998	6787868787	rahul@gmail.com
4	STUD007	Farhaz	MCA	30-04-1997	8944567454	farhaz@gmail.com

Press any key to continue...

1. Details of students for a perticular Course
2. Details of student by entering Roll No.
3. Show details of all the students
4. Main menu
5. Exit the program

Enter your choice: 3

STUDENT DATA

S.No.	Roll No	Name	Course	DOB	Mobile	Email
1	STUD001	Aleena	MCA	25-02-1996	9785658456	aleena@gmail.com
2	STUD002	Akanksha	MCA	01-11-1999	9867565867	akanksha@gmail.com
3	STUD003	Rahul	MCA	12-08-1998	6787868787	rahul@gmail.com
4	STUD004	Karan	BSc	28-01-1999	8985675586	karan@gmail.com
5	STUD005	Pallavi	BSc	28-05-1998	8776765755	pallavi@gmail.com
6	STUD006	Pragati	BSc	23-03-1998	9867654554	pragati@gmail.com
7	STUD007	Farhaz	MCA	30-04-1997	8944567454	farhaz@gmail.com
8	STUD008	Arman	BSc	31-01-2000	7896768776	arman@gmail.com
9	STUD009	Vishal	BSc	20-09-1999	6785677657	vishal@gmail.com
10	STUD010	Anjali	BSc	19-05-1995	7698699786	anjali@gmail.com

Press any key to continue...

1. Details of students for a perticular Course
2. Details of student by entering Roll No.
3. Show details of all the students
4. Main menu
5. Exit the program

Enter your choice: 4_

1. Show Result of a Student (By Entering Roll No)
2. Show Result of all Students (By Entering Course Name)
3. Show Result of all Students
4. Show Details of Students
5. Add Student Details
6. Modify Student Details
7. Modify Student Marks
8. Exit

Enter your choice: 8

Process returned 0 (0x0) execution time : 695.810 s

Press any key to continue.

2. Result of a student is dependent on his/her examination marks and extracurricular marks. Create four classes Student, Examination, Extracurricular and Result. The data members and methods of different classes are given below:

Student Class

data members:

name

member func:

get_details()

display_details()

Examination Class

data members:

test1, test2

member func:

cal_average()

display_average()

Extracurricular Class

data members:

painting, music

member func:

get_score()

display_total()

Result Class

data members:

total

member func:

cal_total()

comment()

Class Examination and Extracurricular are inherited from Student and Result is multiply inherited from Examination and Extracurricular. Write a program to generate the results for N students.

SOURCE CODE:

```
#include <iostream>
#include <string>
#include <cstdlib>           // srand(), rand()
#include <iomanip>           // setw()
#include <cstdlib>           // system()
#include <conio.h>           // getch()
#include <cmath>             // ceil()
#define BOLDCYAN "\033[1m\033[36m" // Colour Code
#define BOLDGREEN "\033[1m\033[32m" // Colour Code
#define BOLDRED "\033[1m\033[31m"  // Colour Code
#define RESET "\033[0m"          // Reset Colour Code
#define MAX_STUDENTS 15
#define MAX_SUBJECTS 3
#define MAX_ACTIVITY 2
#define MAX_MARKS 150
#define MAX_SCORE 10

using namespace std;

/***** CLASSES AND FUNCTIONS DECLARATION *****/

class Student;
class Examination;
class Extracurricular;
class Result;
void line();
```

```

string getGrade(float);
void setTenRecords(Result *);
void addRecord(Result *);
void modifyRecord(Result *);
void viewRecord(Result *);
void viewResult(Result *);

/***** STUDENT CLASS *****/

// Base Class
class Student
{
protected:
    string rollNo;
    string name;
    string course;
    string dob;
    string mobile;
    string email;
    static int NoOfStudents;

public:
    Student()
    {
        rollNo = '\0';
        name = '\0';
        dob = '\0';
        mobile = '\0';
        email = '\0';
    }
    string get_rollNo()
    {
        return rollNo;
    }
    string get_courseName()
    {
        return course;
    }
    static int get_NoOfStudents()
    {
        return NoOfStudents;
    }
}

```

```

static void studentAttributes()
{
    cout << "\n\n" << right
        << setw(62) << "STUDENT DATA" << endl
        << "-----"
        << "-----"
        << "-----" << endl << left
        << setw(10) << "S.No."
        << setw(18) << "Roll No"
        << setw(25) << "Name"
        << setw(25) << "Course"
        << setw(20) << "DOB"
        << setw(20) << "Mobile"
        << setw(30) << "Email" << endl
        << "-----"
        << "-----"
        << "-----" << endl;
}

void setStudentRecord(string name, string dob, string mobile,
                     string email, string course)
{
    this->name = name;
    this->dob = dob;
    this->mobile = mobile;
    this->email = email;
    this->course = course;

    // Each time a record is set for a new student, the
    // NoOfStudent (Static Data Member) is incremented.
    NoOfStudents++;

    // Initializing the Student Roll No.
    this->rollNo = "STUD";
    if (NoOfStudents <= 9)
    {
        this->rollNo += "00";
        this->rollNo += to_string(NoOfStudents);
    }
    else if (NoOfStudents > 9 && NoOfStudents <= 99)
    {
        this->rollNo += '0';
        this->rollNo += to_string(NoOfStudents);
    }
}

```

```

    }
    else
    {
        this->rollNo += to_string(NoOfStudents);
    }
}
void inputStudentRecord()
{
    // Input Student Details from the user.

    cout << "\n Enter Student Details:" << endl
         << "-----" << endl;

    cout << " Name: ";
    cin.ignore(1, '\n');
    getline(cin, name, '\n');

    cout << " Course: ";
    getline(cin, course, '\n');

    cout << " Date Of Birth (DD-MM-YYYY): ";
    getline(cin, dob);

    cout << " Mobile No: ";
    cin >> mobile;

    cout << " Email: ";
    cin >> email;

    cout << "-----" << endl;

    // Each time a record is set for a new student, the
    // NoOfStudent (Static Data Member) is incremented.
    NoOfStudents++;

    // Initializing the Student Roll No.
    this->rollNo = "STUD";
    if (NoOfStudents <= 9)
    {
        this->rollNo += "00";
        this->rollNo += to_string(NoOfStudents);
    }
}

```



```

else if (NoOfStudents > 9 && NoOfStudents <= 99)
{
    this->rollNo += '0';
    this->rollNo += to_string(NoOfStudents);
}
else
{
    this->rollNo += to_string(NoOfStudents);
}

// Message to the user
cout << "\n Student details with roll no. "
    << rollNo << " has been added successfully!";
}
void showStudentRecord(int s_no)
{
    cout << left
        << setw(10) << s_no
        << setw(18) << rollNo
        << setw(25) << name
        << setw(25) << course
        << setw(20) << dob
        << setw(20) << mobile
        << setw(30) << email << endl;
}
void modifyStudentRecord()
{
    // Displaying current details
    cout << "\n CURRENT DETAILS:";
    studentAttributes();
    this->showStudentRecord(1);
    line();
    cout << "\n\n";

    int choice;

chooseAgain: // If user enters an invalid choice

    cout << "\n Choose a field to modify:"
        << "\n 1. Name"
        << "\n 2. Date Of Birth"
        << "\n 3. Contact No"

```

```

        << "\n 4. Email"
        << "\n 5. All Details"
        << "\n\n Enter your choice: ";
cin >> choice;
switch (choice)
{
    case 1:      cout << "\n Enter Name: ";
                  cin.ignore(1, '\n');
                  getline(cin, name, '\n');
                  break;

    case 2:      cout << "\n Enter Date Of Birth (DD-MM-YYYY): ";
                  cin >> dob;
                  break;

    case 3:      cout << "\n Enter Contact No: ";
                  cin >> mobile;
                  break;

    case 4:      cout << "\n Enter Email Address: ";
                  cin >> email;
                  break;

    case 5:      cout << "\n Enter Name: ";
                  cin.ignore(1, '\n');
                  getline(cin, name, '\n');
                  cout << "\n Enter Date Of Birth (DD-MM-YYYY): ";
                  cin >> dob;
                  cout << "\n Enter Contact No: ";
                  cin >> mobile;
                  cout << "\n Enter Email Address: ";
                  cin >> email;
                  break;

    default:     cout << "\n Invalid Choice!"
                  << "\n\n Press any key to continue...";
                  system("cls");
                  goto chooseAgain;
}

// Message to the user
cout << "\n Student details with roll no. " << rollNo

```

```

        << " has been modified successfully!\n";

        // Displaying Modified details
        cout << "\n MODIFIED DETAILS:";
        studentAttributes();
        this->showStudentRecord(1);
        line();
    }
};

int Student ::NoOfStudents = 0;

/***** EXAMINATION CLASS *****/

// Derived Class (Single Inheritance)
class Examination : virtual public Student
{
protected:
    float *marks;
    string *paper;
public:
    Examination()
    {
        marks = new float [MAX_SUBJECTS];
        paper = new string [MAX_SUBJECTS];
        for (int i = 0; i < MAX_SUBJECTS; i++)
        {
            marks[i] = 0;
            paper[i] = "\0";
        }
    }
    ~Examination()
    {
        delete [] marks;
        delete [] paper;
    }
    string get_paperName()
    {
        return paper[0];
    }
    void setMarks(float *marks, string *paper)
    {

```

```

    for (int i = 0; i < MAX_SUBJECTS; i++)
    {
        this->marks[i] = marks[i];
        this->paper[i] = paper[i];
    }
}

void inputMarks()
{
    cout << BOLDCYAN << "\n Student Name: " << name
        << "\n\n" << RESET;
    for (int i = 0; i < MAX_SUBJECTS; i++)
    {
        cout << " Enter Paper Name: ";
        cin.ignore(1, '\n');
        getline(cin, paper[i], '\n');
        cout << " Marks Obtained in " << paper[i]
            << " out of " << MAX_MARKS << " : ";
        cin >> marks[i];
        cout << endl;
    }

    // Message to the user
    cout << "\n Student's Examination details with roll no. "
        << rollNo << " has been added successfully!";
}

void modifyMarks()
{
    int choice, i;

chooseAgain: // If user enters an invalid choice

    cout << BOLDCYAN << " \n Student Name: "
        << name << "\n\n"
        << RESET;

    // Displaying list of papers
    cout << " Modify marks for: \n";
    for (i = 0; i < MAX_SUBJECTS; i++)
    {
        cout << " " << (i + 1) << ". " << paper[i] << endl;
    }
    cout << " " << (i + 1) << ". All";
}

```

```

cout << "\n\n Enter your choice: ";
cin >> choice;
cout << endl;

if (choice == i + 1)
{
    for (int j = 0; j < MAX_SUBJECTS; j++)
    {
        cout << " Enter Score in " << paper[j]
            << " out of " << MAX_MARKS << "\t: ";
        cin >> marks[j];
    }
}
else if (choice >= 1 || choice <= MAX_SUBJECTS)
{
    cout << " Enter Score in " << paper[choice - 1]
        << " out of " << MAX_MARKS << " : ";
    cin >> marks[choice - 1];
}
else
{
    cout << "\n Invalid Choice!"
        << "\n\n Press any key to continue...";
    getch();
    system("cls");
    goto chooseAgain;
}

// Message to the user
cout << "\n Student's Examination marks with roll no. "
    << rollNo << " has been modified successfully!";
}

void printMarks()
{
    cout << "\n EXAMINATION:\n";
    cout << left
        << "-----"
        << "-----"
        << "-----" << endl
        << setw(10) << "S.No."
        << setw(20) << "Paper"
        << setw(20) << "Max Marks"

```

```

    << setw(20) << "Obtained Marks"
    << setw(20) << "Percentage"
    << setw(20) << "Grade"
    << endl
    << "-----"
    << "-----"
    << "-----" << endl;

```

// Calculating the total marks

```

float sum_marksObtained = 0, sum_maxMarks = MAX_SUBJECTS * MAX_MARKS;
for (int i = 0; i < MAX_SUBJECTS; i++)
    sum_marksObtained += marks[i];

```

// Loop to display details of exam for each subject

```

for (int i = 0; i < MAX_SUBJECTS; i++)
{
    cout << left << fixed
        << setprecision(2)
        << setw(10) << i + 1
        << setw(20) << paper[i]
        << setw(20) << MAX_MARKS
        << setw(20) << (int)ceil(marks[i])
        << (marks[i] / MAX_MARKS * 100) << " %" << setw(13) << " "
        << setw(10) << getGrade(marks[i] / MAX_MARKS * 100)
        << endl;
}
cout << "-----"
    << "-----"
    << "-----" << endl;
}
};

```

*/***** EXTRACURRICULAR CLASS *****/*

// Derived Class (Single Inheritance)

```

class Extracurricular : virtual public Student
{
protected:
    float *score;
    string *activity;

public:

```

```

Extracurricular()
{
    score = new float[MAX_ACTIVITY];
    activity = new string[MAX_ACTIVITY];
    for (int i = 0; i < MAX_ACTIVITY; i++)
    {
        score[i] = 0;
        activity[i] = "\0";
    }
}
~Extracurricular()
{
    delete[] score;
    delete[] activity;
}
string get_activityName()
{
    return activity[0];
}
void setScore(float *score, string *activity)
{
    for (int i = 0; i < MAX_ACTIVITY; i++)
    {
        this->score[i] = score[i];
        this->activity[i] = activity[i];
    }
}
void inputScore()
{
    cout << BOLDCYAN << "\n Student Name: " << name << "\n\n"
        << RESET;
    for (int i = 0; i < MAX_ACTIVITY; i++)
    {
        cout << " Enter Activity Name: ";
        cin.ignore(1, '\n');
        getline(cin, activity[i], '\n');
        cout << " Score in " << activity[i] << " out of " << MAX_SCORE << " : ";
        cin >> score[i];
        cout << endl;
    }

    // Message to the user

```



```

        cout << "\n Student's Extracurricular details with roll no. "
            << rollNo << " has been added successfully!";
    }
    void modifyScore()
    {
        int choice, i;

chooseAgain: // If user enters an invalid choice

        cout << BOLDCYAN << " \n Student Name: "
            << name << "\n\n"
            << RESET;

// Displaying the list of activities
        cout << " Modify Score for: \n";
        for (i = 0; i < MAX_ACTIVITY; i++)
        {
            cout << " " << (i + 1) << ". " << activity[i] << endl;
        }
        cout << " " << (i + 1) << ". All";
        cout << "\n\n Enter your choice: ";
        cin >> choice;
        cout << endl;

        if (choice == i + 1)
        {
            for (int j = 0; j < MAX_ACTIVITY; j++)
            {
                cout << " Enter Score in " << activity[j]
                    << " out of " << MAX_SCORE << " : ";
                cin >> score[j];
            }
        }
        else if (choice >= 1 || choice <= MAX_ACTIVITY)
        {
            cout << " Enter Score in " << activity[choice - 1]
                << " out of " << MAX_SCORE << " : ";
            cin >> score[choice - 1];
        }
        else
        {
            cout << "\n Invalid Choice!"

```

```

        << "\n\n Press any key to continue...";
    getch();
    system("cls");
    goto chooseAgain;
}

// Message to the user
cout << "\n Student's Extracurricular scores with roll no. "
    << rollNo << " has been modified successfully!";
}

void printScore()
{
    cout << "\n EXTRACURRICULAR:\n";
    cout << left
        << "-----"
        << "-----"
        << "-----" << endl
        << setw(10) << "S.No."
        << setw(20) << "Activity"
        << setw(20) << "Max Score"
        << setw(20) << "Score"
        << setw(20) << "Percentage"
        << setw(20) << "Grade"
        << endl
        << "-----"
        << "-----"
        << "-----" << endl;

    // Calculating the total score
    float sum_score = 0, sum_maxScore = MAX_SCORE * MAX_ACTIVITY;
    for (int i = 0; i < MAX_ACTIVITY; i++)
        sum_score += score[i];

    // Loop to display details of exam for each subject
    for (int i = 0; i < MAX_ACTIVITY; i++)
    {
        cout << left << fixed
            << setprecision(2)
            << setw(10) << i + 1
            << setw(20) << activity[i]
            << setw(20) << MAX_SCORE
            << setw(20) << (int)ceil(score[i])

```

```

        << (score[i] / MAX_SCORE * 100) << " %" << setw(13) << " "
        << setw(10) << getGrade(score[i] / MAX_SCORE * 100)
        << endl;
    }
    cout << "-----"
        << "-----"
        << "-----" << endl;
}
};

/***** RESULT CLASS *****/

// Derived Class (Multiple Inheritance)
class Result : public Examination, public Extracurricular
{
private:
    float total;
public:
    Result(): total(0) {}
    string comment(string grade)
    {
        if (grade == "A++") return "PASS";
        else if (grade == "A+") return "PASS";
        else if (grade == "A") return "PASS";
        else if (grade == "B") return "PASS";
        else if (grade == "C") return "PASS";
        else if (grade == "D") return "PASS";
        else return "FAIL";
    }
}

void result()
{
    // Display Student's Info.
    cout << "\n STUDENT INFO:\n"
        << "-----"
        << "\n  Name          :  " << name
        << "\n  Roll No       :  " << rollNo
        << "\n  Course Name    :  " << course
        << "\n  Date of Birth  :  " << dob
        << "\n  Mobile No      :  " << mobile
        << "\n  Email          :  " << email
        << "\n-----"

```

```

        << endl;

printMarks(); // Displays Examination Marks
printScore(); // Displays Extracurricular Scores

// Calculating over all result
for (int i = 0; i < MAX_SUBJECTS; i++) total += marks[i];
for (int i = 0; i < MAX_ACTIVITY; i++) total += score[i];
float percent = (total / (MAX_MARKS * MAX_SUBJECTS + MAX_SCORE * MAX_ACTIVITY)) * 100;

cout << "\n OVERALL RESULT:\n"
    << "-----"
    << "\n Marks      : " << (int)ceil(total) << "/"
    << (MAX_MARKS * MAX_SUBJECTS + MAX_SCORE * MAX_ACTIVITY)
    << "\n Percentage : " << (int)ceil(percent) << "%"
    << "\n Grade      : " << getGrade(percent)
    << "\n Result      : "
    << ((comment(getGrade(percent)) == "PASS") ? BOLDGREEN : BOLDRED)
    << comment(getGrade(percent)) << RESET << "\n"
    << "-----\n";
    }
};

/***** DRIVER CODE *****/

int main(void)
{
    Result *studResult = new Result [MAX_STUDENTS];

    setTenRecords(studResult);

    int choice;

    while (true)
    {
        system("cls");

        cout << "\n 1. Add Record"
            << "\n 2. Modify Record"
            << "\n 3. View Record"
            << "\n 4. View Result"
            << "\n 5. Exit"

```

```

        << "\n\n Enter your choice: ";
    cin >> choice;

    switch(choice)
    {
        case 1:      addRecord(studResult);
                     break;

        case 2:      modifyRecord(studResult);
                     break;

        case 3:      viewRecord(studResult);
                     break;

        case 4:      viewResult(studResult);
                     break;

        case 5:      system("cls");
                     return 0;

        default:     cout << "\n Invalid Choice! \n"
                       << "\n Press any key to continue...";
                       getch();
    }
}

}

/*****

void addRecord(Result* studResult)
{
    int choice;
    int studentCount;
    string rollNo;
    bool found;

    while(true)
    {
        system("cls");
        studentCount = Student ::get_NoOfStudents();
        found = false;

```

```

// Menu
cout << "\n 1. Add a new Student"
    << "\n 2. Add Student's Examination Details"
    << "\n 3. Add Student's Extracurricular Details"
    << "\n 4. Return to main menu"
    << "\n 5. Exit"
    << "\n\n Enter your choice: ";
cin >> choice;

system("cls");

switch (choice)
{
    case 1:    studResult[studentCount].inputStudentRecord();
               break;

    case 2:    cout << " Enter Roll No: ";
               cin >> rollNo;

               for (int i = 0; i < studentCount; i++)
               {
                   if (rollNo == studResult[i].get_rollNo())
                   {
                       studResult[i].inputMarks();
                       found = true;
                   }
               }
               if (!found)
                   cout << "\n No Student exist in record with roll no. "
                       << rollNo;
               break;

    case 3:    cout << " Enter Roll No: ";
               cin >> rollNo;

               for (int i = 0; i < studentCount; i++)
               {
                   if (rollNo == studResult[i].get_rollNo())
                   {
                       studResult[i].inputScore();
                       found = true;
                   }
               }

```

```

        }
        if (!found)
            cout << "\n No Student exist in record with roll no. "
                << rollNo;
        break;

        case 4:    return;

        case 5:    system("cls");
                    exit(0);

        default:   cout << "\n Invalid Choice!";
    }
    cout << "\n\n Press any key to continue...";
    getch();
}
}

```

```

void modifyRecord(Result* studResult)
{
    int choice;
    int studentCount;
    string rollNo;
    bool found;

    while(true)
    {
        system("cls");
        studentCount = Student ::get_NoOfStudents();
        found = false;

        // Menu
        cout << "\n 1. Modify Student Record"
            << "\n 2. Modify Student's Examination Marks"
            << "\n 3. Modify Student's Extracurricular Score"
            << "\n 4. Return to main menu"
            << "\n 5. Exit"
            << "\n\n Enter your choice: ";
        cin >> choice;

        system("cls");
    }
}

```



```

switch (choice)
{
case 1:

case 2:

case 3:      cout << " Enter Roll No: ";
             cin >> rollNo;

             for (int i = 0; i < studentCount; i++)
             {
                 if (rollNo == studResult[i].get_rollNo())
                 {
                     found = true;

                     switch (choice)
                     {
                     case 1: studResult[i].modifyStudentRecord();
                             break;

                     case 2: if (studResult[i].get_paperName() == "\\0")
                             cout << "\\n No Data available to modify!\\n"
                                 << "\\n (No Record for exams was "
                                 << "added for roll no. "
                                 << rollNo
                                 << ". Try adding some data first)";
                             else
                                 studResult[i].modifyMarks();
                             break;

                     case 3: if (studResult[i].get_paperName() == "\\0")
                             cout << "\\n No Data available to modify!\\n"
                                 << "\\n (No Record for exams was "
                                 << "added for roll no. "
                                 << rollNo
                                 << ". Try adding some data first)";
                             else
                                 studResult[i].modifyScore();
                             break;
                     }
                 }
             }
}

```

```

        if (!found)
            cout << "\n No Student exist in record with roll no. "
                << rollNo;
        break;

    case 4:        return;

    case 5:        system("cls");
                    exit(0);

    default:       cout << "\n Invalid Choice!";
    }

    cout << "\n\n Press any key to continue...";
    getch();
}
}

```

```

void viewRecord(Result* studResult)
{

```

```

    // Calling Static member function to get the No. of Students currently
    // enrolled in the course.

```

```

    int studentCount = Student ::get_NoOfStudents();

```

```

    int choice, s_no;

```

```

    string rollNo, courseName;

```

```

    bool found; // to keep a record, if the required details are found or not

```

```

    while (true)
    {

```

```

        system("cls");

```

```

        found = false;

```

```

        rollNo = "\0";

```

```

        courseName = "\0";

```

```

        // Menu

```

```

        cout << "\n 1. Details of students for a perticular Course"

```

```

            << "\n 2. Details of student by entering Roll No."

```

```

            << "\n 3. Show details of all the students"

```

```

            << "\n 4. Return to main menu"

```

```

            << "\n 5. Exit"

```

```

        << "\n\n Enter your choice: ";
cin >> choice;

system("cls");

switch (choice)
{

case 1:      cout << "\n Enter Course Name: ";
              cin.ignore(1, '\n');
              getline(cin, courseName, '\n');

              s_no = 1;
              found = false;

              system("cls");

              Student ::studentAttributes();
              for (int i = 0; i < studentCount; i++)
              {
                  if (courseName == studResult[i].get_courseName())
                  {
                      studResult[i].showStudentRecord(s_no);
                      s_no++;
                      found = true;
                  }
              }
              line();

              // If value of found is false then it means no data has been found
              // with Course Name entered by the user, otherwise it is found.
              if (found == false)
                  cout << "\n No Data Found!\n\n";
              break;

case 2:      cout << "\n Enter Roll No: ";
              cin >> rollNo;

              s_no = 1;
              found = false;

```

```

system("cls");

Student ::studentAttributes();
for (int i = 0; i < studentCount; i++)
{
    if (rollNo == studResult[i].get_rollNo())
    {
        studResult[i].showStudentRecord(s_no);
        s_no++;
        found = true;
        break;
    }
}
line();

// If value of found is false then it means no data has been found
// with Roll No entered by the user, otherwise it is found.
if (!found)
    cout << "\n No Data Found!\n\n";
break;

case 3:    if (studentCount == 0)
            cout << "\n No Data Found!\n\n";
        else
        {
            system("cls");

            Student ::studentAttributes();
            // This loop will display details of all the students.
            for (int i = 0; i < studentCount; i++)
            {
                studResult[i].showStudentRecord(i+1);
            }
            line();
        }
        break;

case 4:    return;

case 5:    system("cls");
            exit(0);

```

```

        default:    cout << "\n Invalid Choice!";
    }
    cout << "\n Press any key to continue...";
    getch();
}
}

void viewResult(Result* studResult)
{
    int choice, s_no;
    int studentCount;
    string rollNo, course;
    bool found;

    while (true)
    {
        system("cls");
        studentCount = Student ::get_NoOfStudents();
        found = false;

        cout << "\n 1. Print Result of a Student (By Entering Roll No)"
             << "\n 2. Print Result of Students of a perticular course"
             << "\n 3. Print Result of All Students"
             << "\n 4. Return to main menu"
             << "\n 5. Exit"
             << "\n\n Enter your choice: ";
        cin >> choice;

        system("cls");

        switch (choice)
        {
            case 1:    cout << " Enter Roll No: ";
                      cin >> rollNo;

                      system("cls");
                      for (int i = 0; i < studentCount; i++)
                      {
                          if (rollNo == studResult[i].get_rollNo())

```

```

        {
            studResult[i].result();
            found = true;
            break;
        }
    }

    if (!found)
        cout << " No student exist in record with roll no. "
              << rollNo;
    break;

case 2:    cout << " Enter Course Name: ";
           cin.ignore(1, '\n');
           getline(cin, course, '\n');

           system("cls");
           for (int i = 0; i < studentCount; i++)
           {
               if (course == studResult[i].get_courseName())
               {
                   cout << "\n\n\n"
                        << "===== "
                        << "===== "
                        << "    Roll No. "
                        << studResult[i].get_rollNo()
                        << "    "
                        << "===== "
                        << "===== "
                        << "\n\n";
                   studResult[i].result();
                   found = true;
               }
           }
           if (!found) cout << "\n No Data Found!";
           break;

case 3:    for (int i = 0; i < studentCount; i++)
           {
               cout << "\n\n\n"
                    << "===== "
                    << "===== "

```

```

        << "    Roll No. "
        << studResult[i].get_rollNo()
        << "    "
        << "===== "
        << "===== "
        << "\n\n";
        studResult[i].result();
    }
    break;

    case 4:    return;

    case 5:    system("cls");
              exit(0);

    default:   cout << " Invalid Choice!";

}

cout << "\n\n Press any key to continue...";
getch();
}
}

string getGrade(float percentage)
{
    if (percentage >= 90.0)
        return ("A++");
    else if (percentage < 90.0 && percentage >= 80.0)
        return ("A+");
    else if (percentage < 80.0 && percentage >= 70.0)
        return ("A");
    else if (percentage < 70.0 && percentage >= 60.0)
        return ("B");
    else if (percentage < 60.0 && percentage >= 50.0)
        return ("C");
    else if (percentage < 50.0 && percentage >= 40.0)
        return ("D");
    else
        return ("F");
}

```



```

void setTenRecords(Result *studResult)
{
    // Student's Record (10 Students)

    studResult[0].setStudentRecord("Pragati Yadav", "23-03-1998", "9867654554",
                                    "pragati@gmail.com", "Bachelor of Sci");

    studResult[1].setStudentRecord("Farhaz Beg", "30-04-1997", "8944567454",
                                    "farhaz@gmail.com", "Bachelor of Sci");

    studResult[2].setStudentRecord("Arman Khan", "31-01-2000", "7896768776",
                                    "arman@gmail.com", "Bachelor of Sci");

    studResult[3].setStudentRecord("Vishal Singh", "20-09-1999", "6785677657",
                                    "vishal@gmail.com", "Bachelor of Sci");

    studResult[4].setStudentRecord("Anjali Yadav", "19-05-1995", "7698699786",
                                    "anjali@gmail.com", "Bachelor of Sci");

    studResult[5].setStudentRecord("Aleena Farooqui", "25-02-1996", "9785658456",
                                    "aleena@gmail.com", "Bachelor of Sci");

    studResult[6].setStudentRecord("Akanksha Verma", "01-11-1999", "9867565867",
                                    "akanksha@gmail.com", "Bachelor of Sci");

    studResult[7].setStudentRecord("Rahul Rawat", "12-08-1998", "6787868787",
                                    "rahul@gmail.com", "Bachelor of Sci");

    studResult[8].setStudentRecord("Karan Singh", "28-01-1999", "8985675586",
                                    "karan@gmail.com", "Bachelor of Sci");

    // Examination and Extracurricular Record

    string paper[] = {"Physics", "Chemistry", "Mathematics"};
    string activity[] = {"Music", "Painting"};
    float marks[MAX_SUBJECTS], score[MAX_ACTIVITY];

    srand(time(NULL));

    for (int i = 0; i < Student ::get_NoOfStudents(); i++)
    {
        for (int j = 0; j < MAX_SUBJECTS; j++)

```

```

        marks[j] = (rand() % 100) + 61;

    for (int j = 0; j < MAX_ACTIVITY; j++)
        score[j] = (rand() % 5) + 5;

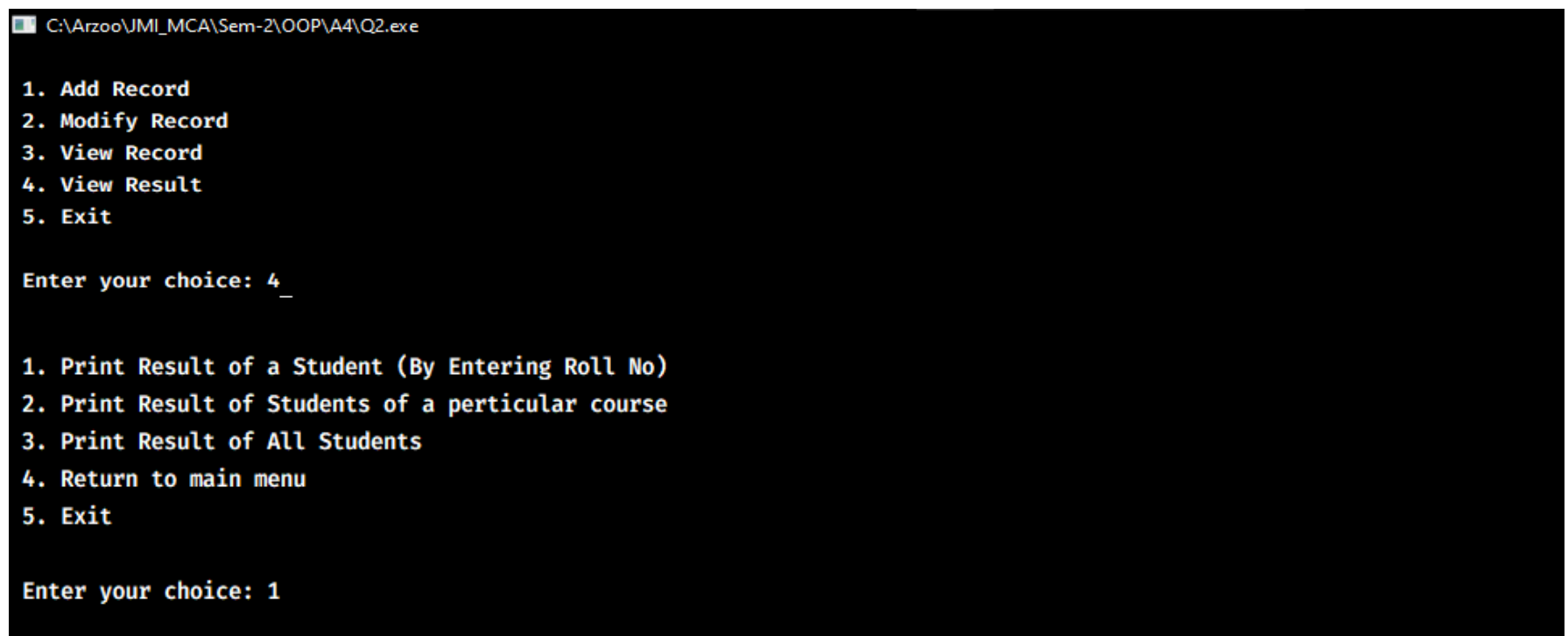
    studResult[i].setMarks(marks, paper);
    studResult[i].setScore(score, activity);
}

studResult[9].setStudentRecord("Pallavi Singh", "28-05-1998", "8776765755",
                                "pallavi@gmail.com", "Bachelor of Sci");
}

void line()
{
    cout << "-----"
         << "-----"
         << "-----" << endl;
}

```

OUTPUT:



The screenshot shows a Windows command prompt window with the title "C:\Arzoo\JMI_MCA\Sem-2\OOP\A4\Q2.exe". The program displays a menu with five options: 1. Add Record, 2. Modify Record, 3. View Record, 4. View Result, and 5. Exit. The user enters '4' and presses Enter. The program then displays a second menu with five options: 1. Print Result of a Student (By Entering Roll No), 2. Print Result of Students of a perticular course, 3. Print Result of All Students, 4. Return to main menu, and 5. Exit. The user enters '1' and presses Enter.

```

C:\Arzoo\JMI_MCA\Sem-2\OOP\A4\Q2.exe

1. Add Record
2. Modify Record
3. View Record
4. View Result
5. Exit

Enter your choice: 4_

1. Print Result of a Student (By Entering Roll No)
2. Print Result of Students of a perticular course
3. Print Result of All Students
4. Return to main menu
5. Exit

Enter your choice: 1

```

Enter Roll No: STUD009

STUDENT INFO:

Name : Karan Singh
Roll No : STUD009
Course Name : Bachelor of Sci
Date of Birth : 28-01-1999
Mobile No : 8985675586
Email : karan@gmail.com

EXAMINATION:

S.No.	Paper	Max Marks	Obtained Marks	Percentage	Grade
1	Physics	150	127	84.67 %	A+
2	Chemistry	150	77	51.33 %	C
3	Mathematics	150	96	64.00 %	B

EXTRACURRICULAR:

S.No.	Activity	Max Score	Score	Percentage	Grade
1	Music	10	7	70 %	A
2	Painting	10	5	50 %	C

OVERALL RESULT:

Marks : 312/470
Percentage : 67%
Grade : B
Result : **PASS**

Press any key to continue...

1. Print Result of a Student (By Entering Roll No)
2. Print Result of Students of a perticular course
3. Print Result of All Students
4. Return to main menu
5. Exit

Enter your choice: 5

3. Class polygon contains data members- width and height and public method set_value() to assign values to width and height. Classes Rectangle and Triangle are inherited from polygon class. Both the classes contain public method calculate_area() to calculate the area of Rectangle and Triangle. Use base class pointer to access the derived class object and show the area calculated. Write a suitable program to illustrate virtual functions.

SOURCE CODE:

```
// Dynamic Allocation and Dynamic Polymorphism
#include <conio.h>
#include <iostream>
using namespace std;

// Abstract Base Class
class Polygon
{
protected:
    double width, height;

public:
    Polygon(double w, double h) : width(w), height(h) {}
    // virtual double area(void) { return 0; } // Virtual Function
    virtual double area(void) = 0; // Pure Virtual Function
};

// Derived Class
class Rectangle : public Polygon
{
public:
    Rectangle(double w, double h) : Polygon(w, h) {}
    double area()
    {
        return width * height;
    }
};

// Derived Class
class Triangle : public Polygon
{
public:
    Triangle(double w, double h) : Polygon(w, h) {}
    double area()
```

```

    {
        return width * height / 2;
    }
};

// Driver Code
int main()
{
    double width, height;
    cout << "\n Enter height and width: ";
    cin >> height >> width;

    Polygon *poly1 = new Rectangle(width, height);
    Polygon *poly2 = new Triangle(width, height);

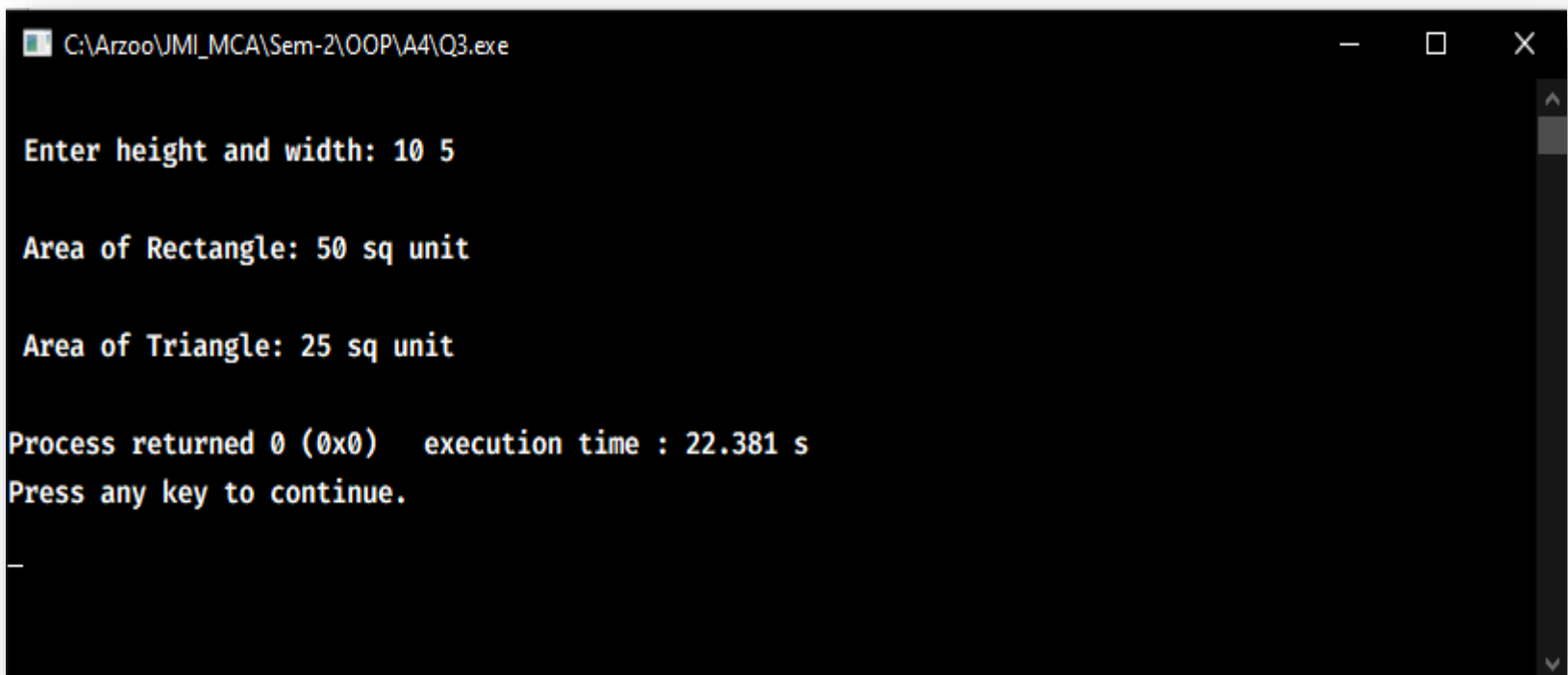
    cout << "\n Area of Rectangle: " << poly1->area() << " sq unit" << endl;
    cout << "\n Area of Triangle: " << poly2->area() << " sq unit" << endl;

    delete poly1;
    delete poly2;

    getch();
    return 0;
}

```

OUTPUT:



```

C:\Arzoo\JMI_MCA\Sem-2\OOP\A4\Q3.exe

Enter height and width: 10 5

Area of Rectangle: 50 sq unit

Area of Triangle: 25 sq unit

Process returned 0 (0x0)   execution time : 22.381 s
Press any key to continue.

```

4. Write a program with Student as abstract class and create derive classes- Engineering, Medicine and Science from base class Student. Create the objects of the derived classes and process them and access them using array of pointers of type base class Student. Include the relevant data members, constructors/destructors and member functions in each of the above classes.

SOURCE CODE:

```
#include <iostream>
#include <string>
#include <cstdio>
#include <cstring>           // strcmpi()
#include <iomanip>           // setw()
#include <cstdlib>           // system()
#include <conio.h>           // getch()
#include <cmath>             // ceil()
#define BOLDCYAN "\033[1m\033[36m" // Colour Code
#define RESET "\033[0m"         // Reset Colour Code
#define MAX_STUDENTS 15

using namespace std;

/***** Classes and functions declaration *****/

class Student;
class Engineering;
class Medicine;
class Science;
void attributes();
void line();
void initializeRecords(Student*, Student*, Student*);

/***** Abstract Base Class *****/

class Student
{
protected:
    string rollNo;
    string name;
    string dob;
    string mobile;
    string email;
```

```

public:
    Student()
    {
        rollNo = '\0';
        name = '\0';
        dob = '\0';
        mobile = '\0';
        email = '\0';
    }
    /*****
    The assignment of function with 0 indicates the pure virtual function.
    It doesnot allow the creation of object of class Student but pointer
    to Student can be created. The student is now Abstract class. To make
    a derive class concrete the member functions of Student should be
    overridden in derived classes. The base class is made abstract because
    it is most generic class so that we do not need its object.
    *****/
    virtual string get_rollNo() = 0;
    virtual void set_data(string, string, string, string) = 0;
    virtual void input_data() = 0;
    virtual void show_data(int s_no) = 0;
    virtual void modify_data() = 0;
};

```

*/****** Engineering Class (Derived) *****/*

```

class Engineering : public Student
{
private:
    static string dept;
    static int NoOfStudents;

public:
    string get_rollNo()
    {
        return rollNo;
    }
    static int get_NoOfStudents()
    {
        return NoOfStudents;
    }
}

```

```

}
void set_data(string name, string dob, string mobile, string email)
{
    this->name = name;
    this->dob = dob;
    this->mobile = mobile;
    this->email = email;

    // Each time a record is set for a new student, the
    // NoOfStudent (Static Data Member) is increamented.
    NoOfStudents++;

    // Initializing the Student Roll No.
    this->rollNo = "ENG";
    if (NoOfStudents <= 9)
    {
        this->rollNo += "00";
        this->rollNo += to_string(NoOfStudents);
    }
    else if (NoOfStudents > 9 && NoOfStudents <= 99)
    {
        this->rollNo += '0';
        this->rollNo += to_string(NoOfStudents);
    }
    else
    {
        this->rollNo += to_string(NoOfStudents);
    }
}
void input_data()
{
    // Input Student Details from the user.

    cout << "\n Enter Student Details:" << endl
         << "-----" << endl;

    cout << " Name: ";
    cin.ignore(1, '\n');
    getline(cin, name, '\n');

    cout << " Date Of Birth (DD-MM-YYYY): ";

```



```

getline(cin, dob);

cout << " Mobile No: ";
cin >> mobile;

cout << " Email: ";
cin >> email;

cout << "-----" << endl;

// Each time a record is set for a new student, the
// NoOfStudent (Static Data Member) is incremented.
NoOfStudents++;

// Initializing the Student Roll No.
this->rollNo = "ENG";
if (NoOfStudents <= 9)
{
    this->rollNo += "00";
    this->rollNo += to_string(NoOfStudents);
}
else if (NoOfStudents > 9 && NoOfStudents <= 99)
{
    this->rollNo += '0';
    this->rollNo += to_string(NoOfStudents);
}
else
{
    this->rollNo += to_string(NoOfStudents);
}

// Message to the user
cout << "\n Student details with roll no. "
    << rollNo
    << " has been added successfully!\n\n";
}
void show_data(int s_no)
{
    cout << BOLD CYAN
        << " "
        << setw(8) << s_no

```

```

        << setw(15) << rollNo
        << setw(25) << name
        << setw(20) << dept
        << setw(20) << dob
        << setw(20) << mobile
        << setw(30) << email << endl
        << RESET;
    }
    void modify_data()
    {

        cout << "\n CURRENT DETAILS: \n";
        attributes();
        this->show_data(1);
        line();
        cout << "\n\n";

        int choice;
        chooseAgain: // If user enters an invalid choice
        cout << "\n Choose a field to modify:"
            << "\n 1. Name"
            << "\n 2. Date Of Birth"
            << "\n 3. Contact No"
            << "\n 4. Email"
            << "\n 5. All Details"
            << "\n\n Enter your choice: ";
        cin >> choice;
        switch (choice)
        {
            case 1:      cout << "\n Enter Name: ";
                        cin >> name;
                        break;

            case 2:      cout << "\n Enter Date Of Birth (DD-MM-YYYY): ";
                        cin >> dob;
                        break;

            case 3:      cout << "\n Enter Contact No: ";
                        cin >> mobile;
                        break;

```

```

        case 4:      cout << "\n Enter Email Address: ";
                     cin >> email;
                     break;

        case 5:      this->input_data();
                     break;

        default:     cout << "Invalid Choice!";
                     system("cls");
                     goto chooseAgain;
    }

    // Message to the user
    cout << "\n Student details with roll no. "
         << rollNo
         << " has been modified successfully!\n\n";

    cout << "\n MODIFIED DETAILS: \n";
    attributes();
    this->show_data(1);
    line();
    cout << "\n\n";
}

};

/***** Medicine Class (Derived) *****/

class Medicine : public Student
{
private:
    static string dept;
    static int NoOfStudents;

public:
    string get_rollNo()
    {
        return rollNo;
    }
    static int get_NoOfStudents()
    {
        return NoOfStudents;
    }
};

```

```

}
void set_data(string name, string dob, string mobile, string email)
{
    this->name = name;
    this->dob = dob;
    this->mobile = mobile;
    this->email = email;

    // Each time a record is set for a new student, the
    // NoOfStudent (Static Data Member) is incremented.
    NoOfStudents++;

    // Initializing the Student Roll No.
    this->rollNo = "MED";
    if (NoOfStudents <= 9)
    {
        this->rollNo += "00";
        this->rollNo += to_string(NoOfStudents);
    }
    else if (NoOfStudents > 9 && NoOfStudents <= 99)
    {
        this->rollNo += '0';
        this->rollNo += to_string(NoOfStudents);
    }
    else
    {
        this->rollNo += to_string(NoOfStudents);
    }
}
void input_data()
{
    // Input Student Details from the user.

    cout << "\n Enter Student Details:" << endl
         << "-----" << endl;

    cout << " Name: ";
    cin.ignore(1, '\n');
    getline(cin, name, '\n');

    cout << " Date Of Birth (DD-MM-YYYY): ";

```

```

getline(cin, dob);

cout << " Mobile No: ";
cin >> mobile;

cout << " Email: ";
cin >> email;

cout << "-----" << endl;

// Each time a record is set for a new student, the
// NoOfStudent (Static Data Member) is incremented.
NoOfStudents++;

// Initializing the Student Roll No.
this->rollNo = "MED";
if (NoOfStudents <= 9)
{
    this->rollNo += "00";
    this->rollNo += to_string(NoOfStudents);
}
else if (NoOfStudents > 9 && NoOfStudents <= 99)
{
    this->rollNo += '0';
    this->rollNo += to_string(NoOfStudents);
}
else
{
    this->rollNo += to_string(NoOfStudents);
}

// Message to the user
cout << "\n Student details with roll no. "
    << rollNo
    << " has been added successfully!\n\n";
}
void show_data(int s_no)
{
    cout << BOLD CYAN
        << " "
        << setw(8) << s_no

```

```

        << setw(15) << rollNo
        << setw(25) << name
        << setw(20) << dept
        << setw(20) << dob
        << setw(20) << mobile
        << setw(30) << email << endl
        << RESET;
    }
    void modify_data()
    {

        cout << "\n CURRENT DETAILS: \n";
        attributes();
        this->show_data(1);
        line();
        cout << "\n\n";

        int choice;
        chooseAgain: // If user enters a invalid choice
        cout << "\n Choose a field to modify:"
            << "\n 1. Name"
            << "\n 2. Date Of Birth"
            << "\n 3. Contact No"
            << "\n 4. Email"
            << "\n 5. All Details"
            << "\n\n Enter your choice: ";
        cin >> choice;
        switch (choice)
        {
            case 1:      cout << "\n Enter Name: ";
                        cin >> name;
                        break;

            case 2:      cout << "\n Enter Date Of Birth (DD-MM-YYYY): ";
                        cin >> dob;
                        break;

            case 3:      cout << "\n Enter Contact No: ";
                        cin >> mobile;
                        break;

```

```

        case 4:      cout << "\n Enter Email Address: ";
                     cin >> email;
                     break;

        case 5:      this->input_data();
                     break;

        default:     cout << "Invalid Choice!";
                     system("cls");
                     goto chooseAgain;
    }

    // Message to the user
    cout << "\n Student details with roll no. "
         << rollNo
         << " has been modified successfully!\n\n";

    cout << "\n MODIFIED DETAILS: \n";
    attributes();
    this->show_data(1);
    line();
    cout << "\n\n";
}
};

/***** Science Class (Derived) *****/

class Science : public Student
{
private:
    static string dept;
    static int NoOfStudents;

public:
    string get_rollNo()
    {
        return rollNo;
    }
    static int get_NoOfStudents()
    {
        return NoOfStudents;
    }
};

```

```

}
void set_data(string name, string dob, string mobile, string email)
{
    this->name = name;
    this->dob = dob;
    this->mobile = mobile;
    this->email = email;

    // Each time a record is set for a new student, the
    // NoOfStudent (Static Data Member) is incremented.
    NoOfStudents++;

    // Initializing the Student Roll No.
    this->rollNo = "SCI";
    if (NoOfStudents <= 9)
    {
        this->rollNo += "00";
        this->rollNo += to_string(NoOfStudents);
    }
    else if (NoOfStudents > 9 && NoOfStudents <= 99)
    {
        this->rollNo += '0';
        this->rollNo += to_string(NoOfStudents);
    }
    else
    {
        this->rollNo += to_string(NoOfStudents);
    }
}
void input_data()
{
    // Input Student Details from the user.

    cout << "\n Enter Student Details:" << endl
         << "-----" << endl;

    cout << " Name: ";
    cin.ignore(1, '\n');
    getline(cin, name, '\n');

    cout << " Date Of Birth (DD-MM-YYYY): ";

```



```

getline(cin, dob);

cout << " Mobile No: ";
cin >> mobile;

cout << " Email: ";
cin >> email;

cout << "-----" << endl;

// Each time a record is set for a new student, the
// NoOfStudent (Static Data Member) is incremented.
NoOfStudents++;

// Initializing the Student Roll No.
this->rollNo = "SCI";
if (NoOfStudents <= 9)
{
    this->rollNo += "00";
    this->rollNo += to_string(NoOfStudents);
}
else if (NoOfStudents > 9 && NoOfStudents <= 99)
{
    this->rollNo += '0';
    this->rollNo += to_string(NoOfStudents);
}
else
{
    this->rollNo += to_string(NoOfStudents);
}

// Message to the user
cout << "\n Student details with roll no. "
    << rollNo
    << " has been added successfully!\n\n";
}
void show_data(int s_no)
{
    cout << BOLD CYAN
        << " "
        << setw(8) << s_no

```

```

        << setw(15) << rollNo
        << setw(25) << name
        << setw(20) << dept
        << setw(20) << dob
        << setw(20) << mobile
        << setw(30) << email << endl
        << RESET;
    }
    void modify_data()
    {

        cout << "\n CURRENT DETAILS: \n";
        attributes();
        this->show_data(1);
        line();
        cout << "\n\n";

        int choice;
        chooseAgain: // If user enters a invalid choice
        cout << "\n Choose a field to modify:"
            << "\n 1. Name"
            << "\n 2. Date Of Birth"
            << "\n 3. Contact No"
            << "\n 4. Email"
            << "\n 5. All Details"
            << "\n\n Enter your choice: ";
        cin >> choice;
        switch (choice)
        {
            case 1:      cout << "\n Enter Name: ";
                        cin >> name;
                        break;

            case 2:      cout << "\n Enter Date Of Birth (DD-MM-YYYY): ";
                        cin >> dob;
                        break;

            case 3:      cout << "\n Enter Contact No: ";
                        cin >> mobile;
                        break;

```

```

        case 4:      cout << "\n Enter Email Address: ";
                     cin >> email;
                     break;

        case 5:      cout << "\n Enter Name: ";
                     cin >> name;
                     cout << "\n Enter Date Of Birth (DD-MM-YYYY): ";
                     cin >> dob;
                     cout << "\n Enter Contact No: ";
                     cin >> mobile;
                     cout << "\n Enter Email Address: ";
                     cin >> email;
                     break;

        default:     cout << "Invalid Choice!";
                     system("cls");
                     goto chooseAgain;
    }

```

// Message to the user

```

cout << "\n Student details with roll no. "
      << rollNo
      << " has been modified successfully!\n\n";

```

```

cout << "\n MODIFIED DETAILS: \n";
attributes();
this->show_data(1);
line();
cout << "\n\n";

```

```

    }
};

```

*/****** Static Data Member Definition *****/*

```

string Engineering ::dept = "Engineering";
int Engineering ::NoOfStudents = 0;
string Medicine ::dept = "Medicine";
int Medicine ::NoOfStudents = 0;
string Science ::dept = "Science";
int Science ::NoOfStudents = 0;

```

*/****** Driver Code *****/*

```
int main()
{
    Student *stud[3];
    stud[0] = new Engineering[MAX_STUDENTS];
    stud[1] = new Medicine[MAX_STUDENTS];
    stud[2] = new Science[MAX_STUDENTS];

    initializeRecords(stud[0], stud[1], stud[2]);

    int choice, s_no;
    int studentCount[3];
    string rollNo;
    bool found;

    while (true)
    {
        studentCount[0] = Engineering ::get_NoOfStudents();
        studentCount[1] = Medicine ::get_NoOfStudents();
        studentCount[2] = Science ::get_NoOfStudents();
        system("cls");
        cout << "\n 1. Add a Student"
              << "\n 2. Search a Student's Details by Roll No"
              << "\n 3. Show Student Details"
              << "\n 4. Modify Student Details"
              << "\n 5. Exit" << endl
              << "\n Enter your choice: ";
        cin >> choice;

        found = false;
        switch (choice)
        {
            case 1:      system("cls");
                        cout << "\n Choose department: "
                              << "\n 1. Engineering"
                              << "\n 2. Medicine"
                              << "\n 3. Science"
                              << "\n\n Enter your choice: ";
                        cin >> choice;
                        if (choice >= 1 && choice <= 3)
```

```

    {
        stud[choice-1][studentCount[choice-1]].input_data();
    }
    else cout << "\n Invalid Choice!";

    cout << " Press any key to continue...";
    getch();
    break;

case 2:
    system("cls");
    cout << "\n A. Choose a department first where you"
        << " want to search for a student record:"
        << "\n      1. Engineering"
        << "\n      2. Medicine"
        << "\n      3. Science"
        << "\n B. Directly Search with Roll No"
        << "\n\n Enter your choice: ";
    scanf("%d", &choice);

    if (choice >= 1 && choice <= 3)
    {
        cout << "\n Enter RollNo: ";
        cin.ignore(1, '\n');
        cin >> rollNo;

        for (int i=0; i < studentCount[choice-1]; i++)
        {
            if (rollNo==stud[choice-1][i].get_rollNo())
            {
                attributes();
                stud[choice - 1][i].show_data(1);
                line();
                found = true;
                break;
            }
        }
    }
    else if (choice == 98 || choice == 66)
    {
        cout << "\n Enter RollNo: ";
        cin >> rollNo;
    }
}

```

```

        for (int dept = 0; dept < 3; dept++)
        {
            for (int i=0; i < studentCount[dept]; i++)
            {
                if (rollNo==stud[dept][i].get_rollNo())
                {
                    attributes();
                    stud[dept][i].show_data(1);
                    line();
                    found = true;
                    break;
                }
            }
        }
    }
    else cout << "\n Invalid Choice!";

    if (!found)
        cout << "\n No Student exists in records with roll no. "
            << rollNo << "\n";

    cout << "\n Press any key to continue...";
    getch();
    break;

case 3: while (true)
    {
        system("cls");
        cout << "\n Choose department: "
            << "\n 1. Engineering"
            << "\n 2. Medicine"
            << "\n 3. Science"
            << "\n 4. All departments"
            << "\n 5. Return to Main"
            << "\n\n Enter your choice: ";
        cin >> choice;
        system("cls");
        switch (choice)
        {
            case 1:

```

```

case 2:

case 3:  if (studentCount[choice - 1] == 0)
        cout << "\n No Data Found!\n";
        else
        {
            attributes();
            for (int i=0; i < studentCount[choice - 1]; i++)
            {
                stud[choice - 1][i].show_data(i + 1);
            }
            line();
        }
        break;

case 4:  if (studentCount[0]+studentCount[1]+studentCount[2]==0)
        cout << "\n No Data Found!\n";
        else
        {
            s_no = 1;
            attributes();
            for (int dept = 0; dept < 3; dept++)
            {
                for (int i=0; i < studentCount[dept]; i++)
                {
                    stud[dept][i].show_data(s_no++);
                }
            }
            line();
        }
        break;

case 5:  break;

default: cout << "Invalid Choice!";
}

if (choice == 5) break;
cout << "\n Press any key to continue...";
getch();
}
break;

```

```

case 4:    system("cls");
           cout << "\n Choose department: "
               << "\n 1. Engineering"
               << "\n 2. Medicine"
               << "\n 3. Science"
               << "\n\n Enter your choice: ";
           cin >> choice;

           if (choice >= 1 && choice <= 3)
           {
               cout << "\n Enter RollNo: ";
               cin >> rollNo;

               for (int i = 0; i < studentCount[choice-1]; i++)
               {
                   if (rollNo==stud[choice-1][i].get_rollNo())
                   {
                       stud[choice - 1][i].modify_data();
                       found = true;
                       break;
                   }
               }
           }
           else cout << "\n Invalid Choice!";

           if (!found)
               cout << "\n No Student Exist with roll no. "
                   << rollNo << "\n";

           cout << " \n Press any key to continue...";
           getch();
           break;

case 5:    system("cls");
           return 0;

default:   cout << "\n Invalid Choice!"
           << "\n\n Press any key to continue...";
           getch();
}

```



```
}
```

```
void initializeRecords(Student *eng, Student *med, Student *sci)
{
    eng[0].set_data("Aleena Farooqui", "11-09-1999",
                    "8743695698", "aleenafarooqui@gmail.com");
    eng[1].set_data("Vishal Singh", "07-08-1999",
                    "9563695458", "vishalsingh@gmail.com");
    eng[2].set_data("Anam Mansoori", "05-09-1999",
                    "7533695698", "anammansoori@gmail.com");
    eng[3].set_data("Akanksha Verma", "11-10-1999",
                    "8453565664", "akankshaverma@gmail.com");
    eng[4].set_data("Anjali Yadav", "19-01-1999",
                    "6547746445", "anjaliyadav@gmail.com");
    eng[5].set_data("Abhay Singh", "23-02-1999",
                    "8967575757", "abhaysingh@gmail.com");

    med[0].set_data("Ayush Pandey", "12-03-1999",
                    "9866564634", "ayushpandey@gmail.com");
    med[1].set_data("Aasmeen Khan", "19-05-1999",
                    "8747556546", "aasmeenkhan@gmail.com");
    med[2].set_data("Atul Saini", "26-04-1999",
                    "8346767677", "atulsaini@gmail.com");
    med[3].set_data("Anjum Nisha", "05-01-1999",
                    "8546767767", "anjumnisha@gmail.com");
    med[4].set_data("Sadiya Aslam", "16-02-1999",
                    "7868565675", "sadiyaaslam@gmail.com");
    med[5].set_data("Ifrah Kamal", "03-08-1999",
                    "7546754656", "ifrahkamal@gmail.com");

    sci[0].set_data("Seerat Ul Ain", "18-11-1999",
                    "6785456575", "seeratulain@gmail.com");
    sci[1].set_data("Shahid Ul Islam", "12-09-1999",
                    "8357665756", "shahidulislam@gmail.com");
    sci[2].set_data("Shah Afnan Ansari", "11-03-1999",
                    "9765754646", "shahafnanansari@gmail.com");
    sci[3].set_data("Huda", "28-06-1999",
                    "7545464464", "huda@gmail.com");
    sci[4].set_data("Siraj Ul Arfin", "26-03-1999",
                    "9767677577", "sirajularfin@gmail.com");
    sci[5].set_data("Ravi Ranjan Ojha", "19-05-1999",
```

```

        "9007854664", "raviranjanojha@gmail.com");
    }

    // Function to print the attributes of the student data
    void attributes()
    {
        cout << endl
            << BOLDCYAN
            << endl
            << right
            << setw(62) << "STUDENT DATA"
            << endl
            << "-----"
            << "-----"
            << "-----" << endl
            << left << "    "
            << setw(8) << "S.No."
            << setw(15) << "Roll No"
            << setw(25) << "Name"
            << setw(20) << "Department"
            << setw(20) << "DOB"
            << setw(20) << "Mobile"
            << setw(30) << "Email"
            << endl
            << "-----"
            << "-----"
            << "-----" << endl
            << RESET;
    }

    // Function to create a line
    void line()
    {
        cout << BOLDCYAN
            << "-----"
            << "-----"
            << "-----" << endl
            << RESET;
    }

```

OUTPUT:

C:\Arzoo\JMI_MCA\Sem-2\OOP\A4\Q4.exe

1. Add a Student
2. Search a Student's Details by Roll No
3. Show Student Details
4. Modify Student Details
5. Exit

Enter your choice: 3

Choose department:

1. Engineering
2. Medicine
3. Science
4. All departments
5. Return to Main

Enter your choice: 1

STUDENT DATA

S.No.	Roll No	Name	Department	DOB	Mobile	Email
1	ENG001	Aleena Farooqui	Engineering	11-09-1999	8743695698	aleenafarooqui@gmail.com
2	ENG002	Vishal Singh	Engineering	07-08-1999	9563695458	vishalsingh@gmail.com
3	ENG003	Anam Mansoori	Engineering	05-09-1999	7533695698	anammansoori@gmail.com
4	ENG004	Akanksha Verma	Engineering	11-10-1999	8453565664	akankshaverma@gmail.com
5	ENG005	Anjali Yadav	Engineering	19-01-1999	6547746445	anjaliyadav@gmail.com
6	ENG006	Abhay Singh	Engineering	23-02-1999	8967575757	abhaysingh@gmail.com

Press any key to continue..._

Choose department:

1. Engineering
2. Medicine
3. Science
4. All departments
5. Return to Main

Enter your choice: 2

STUDENT DATA

S.No.	Roll No	Name	Department	DOB	Mobile	Email
1	MED001	Ayush Pandey	Medicine	12-03-1999	9866564634	ayushpandey@gmail.com
2	MED002	Aasmeen Khan	Medicine	19-05-1999	8747556546	aasmeenkhan@gmail.com
3	MED003	Atul Saini	Medicine	26-04-1999	8346767677	atulsaini@gmail.com
4	MED004	Anjum Nisha	Medicine	05-01-1999	8546767767	anjumnisha@gmail.com
5	MED005	Sadiya Aslam	Medicine	16-02-1999	7868565675	sadiyaaslam@gmail.com
6	MED006	Ifrah Kamal	Medicine	03-08-1999	7546754656	ifrahkamal@gmail.com

Press any key to continue...

Choose department:

1. Engineering
2. Medicine
3. Science
4. All departments
5. Return to Main

Enter your choice: 3

STUDENT DATA

S.No.	Roll No	Name	Department	DOB	Mobile	Email
1	SCI001	Seerat Ul Ain	Science	18-11-1999	6785456575	seeratulain@gmail.com
2	SCI002	Shahid Ul Islam	Science	12-09-1999	8357665756	shahidulislam@gmail.com
3	SCI003	Shah Afnan Ansari	Science	11-03-1999	9765754646	shahafnanansari@gmail.com
4	SCI004	Huda	Science	28-06-1999	7545464464	huda@gmail.com
5	SCI005	Siraj Ul Arfin	Science	26-03-1999	9767677577	sirajularfin@gmail.com
6	SCI006	Ravi Ranjan Ojha	Science	19-05-1999	9007854664	raviranjanojha@gmail.com

Press any key to continue...

Choose department:

1. Engineering
2. Medicine
3. Science
4. All departments
5. Return to Main

Enter your choice: 4

STUDENT DATA

S.No.	Roll No	Name	Department	DOB	Mobile	Email
1	ENG001	Aleena Farooqui	Engineering	11-09-1999	8743695698	aleenafarooqui@gmail.com
2	ENG002	Vishal Singh	Engineering	07-08-1999	9563695458	vishalsingh@gmail.com
3	ENG003	Anam Mansoori	Engineering	05-09-1999	7533695698	anamasoori@gmail.com
4	ENG004	Akanksha Verma	Engineering	11-10-1999	8453565664	akankshaverma@gmail.com
5	ENG005	Anjali Yadav	Engineering	19-01-1999	6547746445	amjaliyadav@gmail.com
6	ENG006	Abhay Singh	Engineering	23-02-1999	8967575757	abhaysingh@gmail.com
7	MED001	Ayush Pandey	Medicine	12-03-1999	9866564634	ayushpandey@gmail.com
8	MED002	Aasmeen Khan	Medicine	19-05-1999	8747556546	aasmeenkhan@gmail.com
9	MED003	Atul Saini	Medicine	26-04-1999	8346767677	atulsaini@gmail.com
10	MED004	Anjum Nisha	Medicine	05-01-1999	8546767767	anjumnisha@gmail.com
11	MED005	Sadiya Aslam	Medicine	16-02-1999	7868565675	sadiyaaslam@gmail.com
12	MED006	Ifrah Kamal	Medicine	03-08-1999	7546754656	ifrahkamal@gmail.com
13	SCI001	Seerat Ul Ain	Science	18-11-1999	6785456575	seeratulain@gmail.com
14	SCI002	Shahid Ul Islam	Science	12-09-1999	8357665756	shahidulislam@gmail.com
15	SCI003	Shah Afnan Ansari	Science	11-03-1999	9765754646	shahafnanansari@gmail.com
16	SCI004	Huda	Science	28-06-1999	7545464464	huda@gmail.com
17	SCI005	Siraj Ul Arfin	Science	26-03-1999	9767677577	sirajularfin@gmail.com
18	SCI006	Ravi Ranjan Ojha	Science	19-05-1999	9007854664	raviranjanojha@gmail.com

Press any key to continue..._

Choose department:

1. Engineering
2. Medicine
3. Science
4. All departments
5. Return to Main

Enter your choice: 5_

1. Add a Student
2. Search a Student's Details by Roll No
3. Show Student Details
4. Modify Student Details
5. Exit

Enter your choice: 2_

A. Choose a department first where you want to search for a student record:

1. Engineering
2. Medicine
3. Science

B. Directly Search with Roll No

Enter your choice: B

Enter RollNo: MED005

STUDENT DATA

S.No.	Roll No	Name	Department	DOB	Mobile	Email
1	MED005	Sadiya Aslam	Medicine	16-02-1999	7868565675	sadiyaaslam@gmail.com

Press any key to continue...

1. Add a Student
2. Search a Student's Details by Roll No
3. Show Student Details
4. Modify Student Details
5. Exit

Enter your choice: 5
