ASSIGNMENT FOR DATA STRUCTURES (EC2022E)

BADHON DATTA PROTTOY

ROLL: B230101EC

EC01

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

SET 1

Question no 1:

Code:

```
#include<iostream>
using namespace std;
class comwarhouse{
 private :
 int peripheral_sold;
 int amount_received;
 bool target_achived;
 public :
 void setdata(int ps,int ar,bool ta)
 {
 peripheral_sold=ps;
 amount_received=ar;
 target_achived=ta;
 }
 void showdata()
 {
 cout<<" each peripheral sold is " <<peripheral_sold<<endl;
 cout<<"total amount received is "<< amount_received<<endl;
 cout<<" month's target is achived "<<(target_achived ? "yes" : "no")<<endl;
 }
};
int main (){
  comwarhouse warh1;
   warh1.setdata(2,20000,true);
   warh1.showdata();
   comwarhouse warh2;
   warh2.setdata(3,200000,false);
   warh2.showdata();
   return 0;
  }
```

Output:

```
 each peripheral sold is 2
total amount received is 20000
 month's target is achived yes
 each peripheral sold is 3
total amount received is 200000
 month's target is achived no
```

Question no 2:

Code:

```
#include<iostream>
#include<string>
using namespace std;
class employee{
 private :
 string employe_name;
 int identity_name ;
 int salary;
 int no_leave;
 int leave_lmt=12;
 public:
 void setdata(const string& ename, int id, int sal,int nl )
 {
 employe_name = ename;
 identity_name =id;
 salary =sal;
 no_leave=nl;
 }
 void getincrement(int increment)
 {
 salary += increment;
 cout<< "salary incremented by:"<<increment<<". new salary :"<<salary<<endl;
 }
 void takeLeave(int leaves) {
 if (leaves <= leave_lmt - no_leave) {
 no_leave += leaves;
 cout << "Leave approved. Total leaves taken: " << no_leave << endl;
 } else {
 cout << "Leave request denied. Exceeds leave limit of " << leave_lmt << " days." <<
endl;
 }
 }
 void showdata()
 {
 cout<<"name of the employee :"<< employe_name<<endl;
 cout<<"indentity number of the employee :" <<identity_name<<endl;
 cout<<"salary of the employee:"<<salary<<endl;
 cout<<"number of leave taken by the employee :"<< no_leave<<endl;
 }
};
int main(){
```

```
 employee e1;
 e1.setdata("Ram",230456741,35000,2);
 e1.showdata();
 e1.getincrement(5000);
 e1.takeLeave(2);
 e1.takeLeave(6);
 employee e2;
 e2.setdata("Mohan",230456731,45000,3); // Corrected line
 e2.showdata();
 e2.getincrement(5000);
 e2.takeLeave(3); //how many leave you want
 e2.takeLeave(7);
 return 0 ;
}
```

Output:

```
name of the employee :Ram
indentity number of the employee :230456741
salary of the employee:35000
number of leave taken by the employee :2
salary incremented by:5000. new salary :40000
Leave approved. Total leaves taken: 4
Leave approved. Total leaves taken: 10
name of the employee :Mohan
indentity number of the employee :230456731
salary of the employee:45000
number of leave taken by the employee :3
salary incremented by:5000. new salary :50000
Leave approved. Total leaves taken: 6
Leave request denied. Exceeds leave limit of 12 days.
```

Question No 3:
Code:

```
#include <bits/stdc++.h>
using namespace std;

class PersonalInformation {
public:
    string name;
    string dateOfBirth;
    string bloodGroup;
    double height;
    double weight;
    string contactAddress;
    string telephoneNumber;
```

```
    PersonalInformation(string n, string dob, string bg, double h, double w, string addr, string
phone) {
        this->name = n;
        this->dateOfBirth = dob;
        this->bloodGroup = bg;
        this->height = h;
        this->weight = w;
        this->contactAddress = addr;
        this->telephoneNumber = phone;
    }
};

class Library {
public:
    int accessionNumber;
    string authorName;
    string bookTitle;
    int yearOfPublication;
    double costOfBook;

    Library(int accession, const std::string& author, const std::string& title, int year, double cost)
        : accessionNumber(accession), authorName(author), bookTitle(title),
yearOfPublication(year), costOfBook(cost) {}

    Library(int accession, string& author, string& title, int year, double cost) {
        this->accessionNumber = accession;
        this->bookTitle = title;
        this->yearOfPublication = year;
        this->costOfBook = cost;
    }
};

class Database {
private:
    vector<PersonalInformation> personalInfoTable;
    vector<Library> libraryTable;

public:
    void buildPersonalInfoTable(vector<PersonalInformation>& data) {
        personalInfoTable = data;
    }

    void buildLibraryTable(vector<Library>& data) {
        libraryTable = data;
```

```
    }

    void listPersonalInfoTable() {
        for (auto& person : personalInfoTable) {
            cout << "Name: " << person.name << ", Date of Birth: " << person.dateOfBirth << ",
Blood Group: " << person.bloodGroup << ", Height: " << person.height << ", Weight: " <<
person.weight << ", Contact Address: " << person.contactAddress << ", Telephone Number: " <<
person.telephoneNumber << endl;
        }
    }

    void listLibraryTable() {
        for (auto& book : libraryTable) {
            cout << "Accession Number: " << book.accessionNumber << ", Author: " <<
book.authorName << ", Title: " << book.bookTitle << ", Year of Publication: " <<
book.yearOfPublication << ", Cost: " << book.costOfBook << endl;
        }
    }

    void insertPersonalInfoEntry(const PersonalInformation& newPerson) {
        personalInfoTable.push_back(newPerson);
    }

    void insertLibraryEntry(const Library& newBook) {
        libraryTable.push_back(newBook);
    }

    void editPersonalInfoEntry(const string& nameToEdit, const PersonalInformation&
updatedPerson) {
        auto it = find_if(personalInfoTable.begin(), personalInfoTable.end(), [nameToEdit](const
PersonalInformation& p) { return p.name == nameToEdit; });

        if (it != personalInfoTable.end()) {
            *it = updatedPerson;
        } else {
            cout << "Entry not found for editing in Personal Information table." << endl;
        }
    }

    void editLibraryEntry(int accessionToEdit, Library& updatedBook) {
        auto it = find_if(libraryTable.begin(), libraryTable.end(), [accessionToEdit](const Library&
book) { return book.accessionNumber == accessionToEdit; });

        if (it != libraryTable.end()) {
```

```
            *it = updatedBook;
        } else {
            cout << "Entry not found for editing in Library table." << std::endl;
        }
    }

    void searchAndPrintPersonalInfo(const string& nameToSearch) const {
        auto it = std::find_if(personalInfoTable.begin(), personalInfoTable.end(),
[nameToSearch](const PersonalInformation& p) { return p.name == nameToSearch; });

        if (it != personalInfoTable.end()) {
            cout << "Name: " << it->name << ", Date of Birth: " << it->dateOfBirth << ", Blood Group:
" << it->bloodGroup << ", Height: " << it->height << ", Weight: " << it->weight << ", Contact
Address: " << it->contactAddress << ", Telephone Number: " << it->telephoneNumber << endl;
        } else {
            cout << "Entry not found in Personal Information table." << endl;
        }
    }
    void searchAndPrintLibrary(int accessionToSearch) const {
        auto it = find_if(libraryTable.begin(), libraryTable.end(), [accessionToSearch](const Library&
book) { return book.accessionNumber == accessionToSearch; });

        if (it != libraryTable.end()) {
            cout << "Accession Number: " << it->accessionNumber << ", Author: " <<
it->authorName  << ", Title: "  << it->bookTitle  << ", Year of Publication: "  <<
it->yearOfPublication  << ", Cost: "  << it->costOfBook  << endl;
        } else {
            cout  <<"Entry not found in Library table."<<endl;
        }
    }
    void sortPersonalInfoEntries() {
        sort(personalInfoTable.begin(), personalInfoTable.end(), [](const PersonalInformation& a,
const PersonalInformation& b) {
            return a.name < b.name;
        });
    }

    void sortLibraryEntries() {
        sort(libraryTable.begin(), libraryTable.end(), [](const Library& a, const Library& b) {
            return a.accessionNumber < b.accessionNumber;
        });
    }
};
```

```
int main() {
    vector<PersonalInformation> personalData = {
        {"Aman", "1990-01-15", "A+", 165.0, 55.5, "123 Main St", "555-1234"},
        {"Aditya", "1985-05-20", "B-", 175.5, 70.2, "456 Oak St", "555-5678"},
        {"Aditi", "1992-09-10", "O+", 160.0, 65.0, "789 Elm St", "555-9876"}
    };
     vector<Library> libraryData = {
        {101, "Avinash", "Introduction to C++", 2010, 29.99},
        {102, "Sumit", "Data Structures", 2015, 45.50},
        {103, "Abhay", "HDL", 2018, 55.25}
    };
    Database myDatabase;
    myDatabase.buildPersonalInfoTable(personalData);
    myDatabase.buildLibraryTable(libraryData);

    cout<<"Listing the Personal Information table:"<<endl;
    myDatabase.listPersonalInfoTable();

    cout<<"\nListing the Library table:"<<endl;
    myDatabase.listLibraryTable();

    PersonalInformation newPerson("Ritesh Gupta", "1995-03-08", "AB+", 170.5,
                                  60.0,"543 Pine St","555-8765");
    myDatabase.insertPersonalInfoEntry(newPerson);

    Library newBook(104,"Abhishek Gupta","Advanced C++ Programming",2022,
                    69.99);
    myDatabase.insertLibraryEntry(newBook);

    cout<<"\nListing the Personal Information table after insertion:"<<endl;
    myDatabase.listPersonalInfoTable();

    cout<<"\nListing the Library table after insertion:"<<endl;
    myDatabase.listLibraryTable();

    PersonalInformation updatedPerson("Aditya", "1985-05-20", "B-", 175.5,
                                      72.0,"456 Oak St, Apt 2","555-5678");
    myDatabase.editPersonalInfoEntry("Aditya", updatedPerson);

    Library updatedBook(102,"Sumit","Data Structures and Algorithms (Updated Edition)",2018,
                        49.99);
    myDatabase.editLibraryEntry(102, updatedBook);

    cout<<"\nListing the Personal Information table after editing:"<<std::endl;
```

```
    myDatabase.listPersonalInfoTable();

    cout<<"\nListing the Library table after editing:"<<std::endl;
    myDatabase.listLibraryTable();

    cout<<"\nSearching for and printing an entry in Personal Information table:"<<endl;
    myDatabase.searchAndPrintPersonalInfo("Alice");

    cout<<"\nSearching for and printing an entry in Library table:"<<endl;
    myDatabase.searchAndPrintLibrary(103);

    myDatabase.sortPersonalInfoEntries();

    myDatabase.sortLibraryEntries();

    cout<<"\nListing the Personal Information table after sorting:"<<endl;
    myDatabase.listPersonalInfoTable();

    cout<<"\nListing the Library table after sorting:"<<endl;
    myDatabase.listLibraryTable();

  return 0;
}
```

Output:

```
Listing the Personal Information table:
Name: Aman, Date of Birth: 1990-01-15, Blood Group: A+, Height: 165, Weight: 55.5, Contact Add
ress: 123 Main St, Telephone Number: 555-1234
Name: Aditya, Date of Birth: 1985-05-20, Blood Group: B-, Height: 175.5, Weight: 70.2, Contact
 Address: 456 Oak St, Telephone Number: 555-5678
Name: Aditi, Date of Birth: 1992-09-10, Blood Group: O+, Height: 160, Weight: 65, Contact Addr
ess: 789 Elm St, Telephone Number: 555-9876

Listing the Library table:
Accession Number: 101, Author: Avinash, Title: Introduction to C++, Year of Publication: 2010,
 Cost: 29.99
Accession Number: 102, Author: Sumit, Title: Data Structures, Year of Publication: 2015, Cost:
 45.5
Accession Number: 103, Author: Abhay, Title: HDL, Year of Publication: 2018, Cost: 55.25

Listing the Personal Information table after insertion:
Name: Aman, Date of Birth: 1990-01-15, Blood Group: A+, Height: 165, Weight: 55.5, Contact Add
ress: 123 Main St, Telephone Number: 555-1234
Name: Aditya, Date of Birth: 1985-05-20, Blood Group: B-, Height: 175.5, Weight: 70.2, Contact
 Address: 456 Oak St, Telephone Number: 555-5678
Name: Aditi, Date of Birth: 1992-09-10, Blood Group: O+, Height: 160, Weight: 65, Contact Addr
ess: 789 Elm St, Telephone Number: 555-9876
Name: Ritesh Gupta, Date of Birth: 1995-03-08, Blood Group: AB+, Height: 170.5, Weight: 60, Co
ntact Address: 543 Pine St, Telephone Number: 555-8765

Listing the Library table after insertion:
Accession Number: 101, Author: Avinash, Title: Introduction to C++, Year of Publication: 2010,
 Cost: 29.99
Accession Number: 102, Author: Sumit, Title: Data Structures, Year of Publication: 2015, Cost:
 45.5
Accession Number: 103, Author: Abhay, Title: HDL, Year of Publication: 2018, Cost: 55.25
Accession Number: 104, Author: Abhishek Gupta, Title: Advanced C++ Programming, Year of Public
ation: 2022, Cost: 69.99
```

```
Listing the Personal Information table after editing:
Name: Aman, Date of Birth: 1990-01-15, Blood Group: A+, Height: 165, Weight: 55.5, Contact Address: 123 Main St, Telephone Number: 555-
1234
Name: Aditya, Date of Birth: 1985-05-20, Blood Group: B-, Height: 175.5, Weight: 72, Contact Address: 456 Oak St, Apt 2, Telephone Numb
er: 555-5678
Name: Aditi, Date of Birth: 1992-09-10, Blood Group: O+, Height: 160, Weight: 65, Contact Address: 789 Elm St, Telephone Number: 555-98
76
Name: Ritesh Gupta, Date of Birth: 1995-03-08, Blood Group: AB+, Height: 170.5, Weight: 60, Contact Address: 543 Pine St, Telephone Num
ber: 555-8765

Listing the Library table after editing:
Accession Number: 101, Author: Avinash, Title: Introduction to C++, Year of Publication: 2010, Cost: 29.99
Accession Number: 102, Author: Sumit, Title: Data Structures and Algorithms (Updated Edition), Year of Publication: 2018, Cost: 49.99
Accession Number: 103, Author: Abhay, Title: HDL, Year of Publication: 2018, Cost: 55.25
Accession Number: 104, Author: Abhishek Gupta, Title: Advanced C++ Programming, Year of Publication: 2022, Cost: 69.99

Searching for and printing an entry in Personal Information table:
Entry not found in Personal Information table.

Searching for and printing an entry in Library table:
Accession Number: 103, Author: Abhay, Title: HDL, Year of Publication: 2018, Cost: 55.25

Listing the Personal Information table after sorting:
Name: Aditi, Date of Birth: 1992-09-10, Blood Group: O+, Height: 160, Weight: 65, Contact Address: 789 Elm St, Telephone Number: 555-98
76
Name: Aditya, Date of Birth: 1985-05-20, Blood Group: B-, Height: 175.5, Weight: 72, Contact Address: 456 Oak St, Apt 2, Telephone Numb
er: 555-5678
Name: Aman, Date of Birth: 1990-01-15, Blood Group: A+, Height: 165, Weight: 55.5, Contact Address: 123 Main St, Telephone Number: 555-
1234
Name: Ritesh Gupta, Date of Birth: 1995-03-08, Blood Group: AB+, Height: 170.5, Weight: 60, Contact Address: 543 Pine St, Telephone Num
ber: 555-8765
```

```
Listing the Library table after sorting:
Accession Number: 101, Author: Avinash, Title: Introduction to C++, Year of Publication: 2010, Cost: 29.99
Accession Number: 102, Author: Sumit, Title: Data Structures and Algorithms (Updated Edition), Year of Publication: 2018, Cost: 49.99
Accession Number: 103, Author: Abhay, Title: HDL, Year of Publication: 2018, Cost: 55.25
Accession Number: 104, Author: Abhishek Gupta, Title: Advanced C++ Programming, Year of Publication: 2022, Cost: 69.99
```

Question no 4:

Code:

```
#include <iostream>
using namespace std;
class Polygon {
public:

 virtual double area() const = 0; // Pure virtual function
 virtual double perimeter() const = 0; // Pure virtual function
};
class Rectangle : public Polygon {
private:
 double length;
 double width;
public:
 // Constructor to initialize length and width
 Rectangle(double l, double w) : length(l), width(w) {}

 double area() const override {
 return length * width;
 }

 double perimeter() const override {
 return 2 * (length + width);
 }
```

```
 // Function to calculate charges
 void calculateareaandCharges(double fenceCostPerUnit, double lawnCostPerUnit) const {
 double rectArea =area();
 double rectPerimeter =perimeter();
 double fenceCost = perimeter() * fenceCostPerUnit;
 double lawnCost = area() * lawnCostPerUnit;
 cout<<"area of the land :"<<rectArea<<"square unit"<<endl;
 cout<<"perimeter of the land :"<<rectPerimeter<<" unit"<<endl;
 cout << "Charges for building a fence: RS" << fenceCost << endl;
 cout << "Charges for laying a lawn: RS" << lawnCost << endl;
 }
};
int main() {
 double length, width;

 double fenceCostPerUnit, lawnCostPerUnit;

 cout << "Enter the length of the rectangle: ";
 cin >> length;
 cout << "Enter the width of the rectangle: ";
 cin >> width;
 cout << "Enter the cost per unit length for building a fence: ";
 cin >> fenceCostPerUnit;
 cout << "Enter the cost per unit area for laying a lawn: ";
 cin >> lawnCostPerUnit;

 Rectangle r1(length, width);
 r1.calculateareaandCharges(fenceCostPerUnit, lawnCostPerUnit);
 return 0;
}
```

Output:

```
Enter the length of the rectangle: 20
Enter the width of the rectangle: 12
Enter the cost per unit length for building a fence: 5
Enter the cost per unit area for laying a lawn: 45
area of the land :240square unit
perimeter of the land :64 unit
Charges for building a fence: RS320
Charges for laying a lawn: RS10800
```

SET 2

Question no 1:

Code:

```
#include<iostream>
#include<string>
using namespace std;
class student {
 private :
 string name ;
 int roll_no;
 int total_marks;
 public :
 void setdata(const string& nm,int rn,int tm)
 {
 name =nm;
 roll_no =rn;
 total_marks =tm;
 }
 void showdata()
 {
 cout<<" enter your name :" <<name<<endl;
 cout<<"enter your roll number :"<<roll_no<<endl;
 cout<<" enter your total marks :"<<total_marks<<endl;
 }
};
int main (){
 student s1;
 s1.setdata("vivek",15,70);
 s1.showdata();
 student s2;
 s2.setdata("mohit", 22,80);
 s2.showdata();
 return 0;
}
```

Output:

```
 enter your name :vivek
enter your roll number :15
 enter your total marks :70
 enter your name :mohit
enter your roll number :22
 enter your total marks :80
```

Question no 2:
Code:

```
#include<iostream>
#include <cmath>
using namespace std;
class triangle{
 private:
 double side1,side2,side3;

 public :
 void getdata(double s1,double s2,double s3)
 {
 side1 =s1;
 side2 =s2;
 side3=s3;
 }
 double Area()
 {
 double s=(side1+side2+side3)/2;
 return sqrt(s*(s-side2)*(s-side1)*(s-side3));
 }
 double Perimeter ()
 {
 return (side1+side2+side3);
 }
 void showAreaandPerimeter(){
 double area=Area();
 double perimeter =Perimeter();
 cout<<"area of the triangle :"<<area<<"square unit"<<endl;
 cout<<"perimeter of the triangle :"<<perimeter<<"unit"<<endl;
 }
};
int main() {
 triangle t1;
 t1.getdata(3,4,5);
 t1.showAreaandPerimeter();
 return 0;

}
```

Output:

```
area of the triangle :6square unit
perimeter of the triangle :12unit
```

Question No 3:
Code:

```
#include<iostream>
#include <cmath>
using namespace std;
class rectangle{
 private:
 double length,breadth;

 public :
 void setdim(double l,double b)
 {
 length=l;
 breadth=b;
 }
 double Area()
 {

 return length*breadth;
 }
 double Perimeter ()
 {
 return (2*(length+breadth));
 }
 void showAreaandPerimeter(){
 double area=Area();
 double perimeter =Perimeter();
 cout<<"area of the rectangle :"<<area<<"square unit"<<endl;
 cout<<"perimeter of the rectangle :"<<perimeter<<"unit"<<endl;
 }
};
int main() {
 rectangle r1;
 r1.setdim(4,5);
 r1.showAreaandPerimeter();
 return 0;

}
```

Output:

```
area of the rectangle :20square unit
perimeter of the rectangle :18unit
```

Question No 4:
Code:

```
#include <iostream>
using namespace std;
class Complex {
private:
 double real;
 double imag;
public:
 // Constructor to initialize complex number
 Complex(double r = 0, double i = 0) : real(r), imag(i) {}
 // Function to display the complex number
 void display() const {
 if (imag >= 0)
 cout << real << " + " << imag << "i";
 else
 cout << real << " - " << -imag << "i";
 }
 // Function to add two complex numbers
 Complex add(const Complex& other) const {
 return Complex(real + other.real, imag + other.imag);
 }
 // Function to subtract two complex numbers
 Complex subtract(const Complex& other) const {
 return Complex(real - other.real, imag - other.imag);
 }
 // Function to multiply two complex numbers
 Complex multiply(const Complex& other) const {
 return Complex(real * other.real - imag * other.imag,
 real * other.imag + imag * other.real);
 }
};
int main() {
 double r1, i1, r2, i2;

 cout << "Enter the real part of the first complex number: ";
 cin >> r1;
 cout << "Enter the imaginary part of the first complex number: ";
 cin >> i1;

 cout << "Enter the real part of the second complex number: ";
 cin >> r2;
 cout << "Enter the imaginary part of the second complex number: ";
 cin >> i2;
```

```
 Complex c1(r1, i1);
 Complex c2(r2, i2);
 // Perform operations
 Complex sum = c1.add(c2);
 Complex difference = c1.subtract(c2);
 Complex product = c1.multiply(c2);
 // Display results
 cout << "\nSum: " ;
 sum.display();
 cout << "\nDifference: ";
 difference.display();
  cout << "\nProduct: ";
   product.display();
   cout << endl;
   return 0;
  }
```

Output:

```
Enter the real part of the first complex number: 5
Enter the imaginary part of the first complex number: 3
Enter the real part of the second complex number: 6
Enter the imaginary part of the second complex number: 3

Sum: 11 + 6i
Difference: -1 + 0i
Product: 21 + 33i
```

Question no 5:

Code:

```
#include <iostream>
using namespace std;
class Matrix {
private:
 int rows;
 int cols;
 int elements[10][10];
public:

 Matrix(int r, int c) {
 rows = r;
 cols = c;
 for (int i = 0; i < rows; i++) {
```

```
for (int j = 0; j < cols; j++) {
elements[i][j] = 0;
}
}
}
int getRows() {
return rows;
}
int getCols() {
return cols;
}
// Function to set an element at position (i, j)
void setElement(int i, int j, int value) {
if (i >= 0 && i < rows && j >= 0 && j < cols) {
elements[i][j] = value;
} else {
cout << "Invalid position (" << i << ", " << j << ")" << endl;
}
}
// Function to display the matrix
void display() {
for (int i = 0; i < rows; i++) {
for (int j = 0; j < cols; j++) {
cout << elements[i][j] << " ";
}
cout << endl;
}
}
// Function to add two matrices
Matrix add(Matrix other) {
if (rows != other.getRows() || cols != other.getCols()) {
cout << "Addition not possible. Matrices have different dimensions." << endl;
return Matrix(0, 0); // Return an empty matrix
}
Matrix result(rows, cols);
for (int i = 0; i < rows; i++) {
for (int j = 0; j < cols; j++) {
result.elements[i][j] = elements[i][j] + other.elements[i][j];
}
}
return result;
}
// Function to multiply two matrices
Matrix multiply(Matrix other) {
```

```
 if (cols != other.getRows()) {
 cout << "Multiplication not possible. Columns of first matrix must equal rows of second matrix."
<< endl;
 return Matrix(0, 0); // Return an empty matrix
 }
 Matrix result(rows, other.getCols());
 for (int i = 0; i < rows; i++) {
 for (int j = 0; j < other.getCols(); j++) {
 for (int k = 0; k < cols; k++) {
 result.elements[i][j] += elements[i][k] * other.elements[k][j];
 }
 }
 }
 return result;
 }
};
int main() {
 // Create two 2x2 matrices
 Matrix m1(2, 2);
 Matrix m2(2, 2);
 // Set elements of mat1
 m1.setElement(0, 0, 1);
 m1.setElement(0, 1, 3);
 m1.setElement(1, 0, 4);
 m1.setElement(1, 3, 6);
 // Set elements of mat2
 m2.setElement(0, 0, 5);
 m2.setElement(0, 1, 6);
 m2.setElement(1, 5, 7);
 m2.setElement(1, 0, 8);
 // Display the matrices
 cout << "Matrix 1:" << endl;
   m1.display();
   cout << "Matrix 2:" << endl;
   m2.display();
   // Add the matrices
   cout << "Matrix 1 + Matrix 2:" << endl;
   Matrix sum = m1.add(m2);
   sum.display();
   // Multiply the matrices
   cout << "Matrix 1 * Matrix 2:" << endl;
   Matrix product = m1.multiply(m2);
   product.display();
   return 0;
```

}

Output:

```
Invalid position (1, 3)
Invalid position (1, 5)
Matrix 1:
1 3
4 0
Matrix 2:
5 6
8 0
Matrix 1 + Matrix 2:
6 9
12 0
Matrix 1 * Matrix 2:
29 6
20 24
```

Question no 6:

Code:

```
#include <iostream>
#include <cstring>
using namespace std;
class REPORT {
private:
 int adno;
 char name[20]; // Name (20 characters)
 float marks[5]; // Array of 5
 float average;
 // Function to compute the average obtained in five subjects
 void GETAVG() {
 float sum = 0.0;
 for (int i = 0; i < 5; ++i) {
 sum += marks[i]; // Sum up the marks
 }
 average = sum / 5; // Calculate average
 }
public:
 // Function to accept values for adno, name, and marks
 void READINFO() {
 cout << "Enter Admission Number (4 digits): ";
 cin >> adno;
 cout << "Enter Name: ";
```

```
 cin.ignore(); // Clear the newline character from the input buffer
 cin.getline(name, 20); // Read name
 cout << "Enter Marks for 5 subjects: ";
 for (int i = 0; i < 5; ++i) {
 cin >> marks[i]; // Read marks
 }
 // Invoke the function to compute average
 GETAVG();
 }
 // Function to display all data members of the report
 void DISPLAYINFO() const {
 cout << "\nAdmission Number: " << adno << endl;
 cout << "Name: " << name << endl;
 cout << "Marks: ";
 for (int i = 0; i < 5; ++i) {
 std::cout << marks[i] << " "; // Display marks
 }
 cout << "\nAverage Marks: " << average << endl; // Display average
 }
};
int main() {
 REPORT repo1;
 // Read information
 repo1.READINFO();
 // Display information
 repo1.DISPLAYINFO();
 return 0;
}
```

Output:

```
Enter Admission Number (4 digits): 3248
Enter Name: Badhon Datta
Enter Marks for 5 subjects: 56
34
87
67
98

Admission Number: 3248
Name: Badhon Datta
Marks: 56 34 87 67 98
Average Marks: 68.4
```

Question No 7:
Code:

```
#include <iostream>
#include <string>
using namespace std;
class Movie {
private:
 string title;
 string studio;
 string rating;
public:

 Movie(string t, string s, string r) {
 title = t;
 studio = s;
 rating = r;
 }

 Movie(string t, string s) {
 title = t;
 studio = s;
 rating = "PG";
 }
 // Getter for the rating
 string getRating() {
 return rating;
 }
 // Getter for the title
 string getTitle() {
 return title;
 }
 // Getter for the studio
 string getStudio() {
 return studio;
 }
};
// Function to get all movies with a rating of "PG"
void getPG(Movie movies[], int size) {
 cout << "Movies with PG rating:" << endl;
 for (int i = 0; i < size; i++) {
 if (movies[i].getRating() == "PG") {
 cout << movies[i].getTitle() << " by " << movies[i].getStudio() << endl;
 }
 }
```

```
}
int main() {
 // Create an instance of Movie
 Movie casinoRoyale("Casino Royale", "Eon Productions", "PG13");
 // Create an array of movies
 Movie movieList[] = {
 casinoRoyale,
 Movie("Finding nemo", "Disney", "PG"),
 Movie("The dark knight", "Warner Bros.", "PG13"),
 Movie("Toy Story", "Pixar", "PG")
 };
 // Get movies with a "PG" rating
 getPG(movieList, 4);
 return 0;
}
```

Output:

```
Movies with PG rating:
Finding nemo by Disney
Toy Story by Pixar
```