Task#1

You are tasked with implementing a dynamic matrix class in C++ that supports the following operations: Dynamic Matrix Creation: Create a dynamic 2D array (matrix) with rows and columns specified by the user. Matrix Resizing: Implement a method to resize the matrix. The new size should be provided as input (new rows and columns). If the new size is larger, initialize the new elements with a given value. If the new size is smaller, truncate the matrix. Matrix Transposition: Implement a method to transpose the matrix (rows become columns and vice versa). Matrix Printing: Implement a method to print the matrix. After add 2 to each odd index then print the array. Memory Deallocation: Ensure proper memory management, including deallocation of the dynamic matrix when no longer needed.

#include<iostream>

using namespace std;

class Matrix{

    int rows,colums;

    int \*\*ptr = NULL;

    public:

    Matrix(){}

    Matrix(int r,int col):rows(r),colums(col){

        ptr = new int\*[rows];

        for (int i = 0; i < rows; i++)

        {

            ptr[i] = new int[colums];

        }

        for (int i = 0; i < rows; i++)

        {

            cout<<"Enter elements for row "<<i+1<<" : \n";

            for (int j = 0; j < colums  ; j++)

            {

                cout<<"Enter element "<<j+1<<" : ";

                cin>>ptr[i][j];

            }

        }

    }

        void resizeArray(int r,int c){

            for (int i = 0; i < rows; i++)

            {

                delete[] ptr[i];

            }

            delete ptr;

            rows = r;

            colums  =c;

            ptr = new int\*[r];

            cout<<"\nEnter new array\n";

            for (int i = 0; i < r; i++)

            {

                ptr[i] = new int[c];

            }

            for (int i = 0; i < r; i++)

            {

                cout<<"Enter elements for row "<<i+1<<" : \n";

                for (int j = 0; j < c  ; j++)

                {

                    cout<<"Enter element "<<j+1<<" : ";

                    cin>>ptr[i][j];

                }

            }

    }

    void printArray(){

        for (int i = 0; i < rows; i++)

        {

            for (int j = 0; j < colums; j++)

            {

                if (j%2==0)

                {

                    ptr[i][j]+=2;

                }

                cout<<ptr[i][j]<<"  ";

            }

            cout<<endl;

        }

    }

    void transposeArray(){

        for (int i = 0; i < rows; i++)

        {

            for (int j = i+1; j < colums; j++)

            {

                int temp = ptr[j][i];

                ptr[j][i] = ptr[i][j];

                ptr[i][j] = temp;

            }

        }

    }

    ~Matrix(){

        for (int i = 0; i < rows; i++)

        {

            delete[] ptr[i];

        }

        delete[] ptr;

    }

};

int main(){

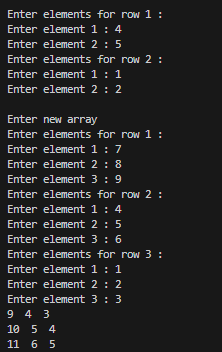
    Matrix m1(2,2);

    m1.resizeArray(3,3);

    m1.transposeArray();

    m1.printArray();

}



Task#2

A school has conducted exams for 5 students in 4 subjects.

You are required to:

 Use a appropriate dynamic array to store marks.

 Calculate and display:

 Total marks of each student.

 Average marks of each subject.

 The topper student (highest total marks).

#include<iostream>

using namespace std;

int main()

{

    int r = 5;

    int c = 4;

    int \*\*array = new int\*[r];

    double totalOfEach[5];

    double average[4];

    double max = -1;

    int ind = -1;

    for (int i = 0; i < r; i++)

    {

        totalOfEach[i] = 0;

        array[i] = new int[c];

    }

    for (int i = 0; i < r; i++)

    {

        cout<<"\nEnter marks for Student "<<i+1<<endl;

        for (int j = 0; j < c; j++)

        {

            cout<<"Enter marks of subject "<<j+1<<" : ";

            cin>>array[i][j];

            totalOfEach[i]  += array[i][j];

        }

    }

    for (int i = 0; i < c; i++)

        { average[i] = 0;

        for (int j = 0; j < r; j++)

        {

            average[i] += array[j][i];

        }

        average[i] /=r;

    }

    for (int i = 0; i < r; i++)

    {

        cout<<"\nStudent "<<i+1<<" marks are: \n[ ";

        for (int j = 0; j < c; j++)

        {

            cout<<array[i][j]<<" , ";

        }

        if (totalOfEach[i]>max)

        {

            max = totalOfEach[i];

            ind = i+1;

        }

        cout<<"]  Total Marks are: "<<totalOfEach[i]<<"\n\n";

    }

    for (int i = 0; i < c; i++)

    {

        cout<<"Average marks in subject "<<i+1<< " is: "<<average[i]<<endl;

    }

    cout<<"Student with max marks is student "<<ind<<" with marks : "<<totalOfEach[ind-1]<<endl;

    for (int i = 0; i < r; i++)

    {

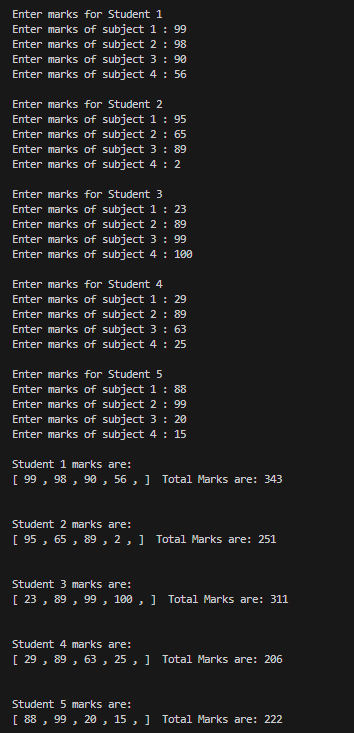
        delete[] array[i];

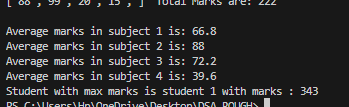
    }

    delete[] array;

return 0;

}





Task#3

A university is managing the marks of students in multiple subjects. Each department has a different number of students. Each student has marks for a fixed number of 5 subjects. You need to store this data in a appropriate dynamic array where rows = number of departments and columns = number of students in each department. Then, calculate the highest, lowest and average marks of each department and display them.

#include<iostream>

using namespace std;

int main()

{

    int departments = 3;

    int students[3];

    int \*\*\*array = NULL;

    array = new int\*\*[departments];

    for (int i = 0; i < departments; i++)

    {

        cout<<"Enter students for department "<<i+1<<" : ";

        cin>>students[i];

    }

    for (int i = 0; i < departments; i++)

    {

        array[i] = new int\*[students[i]];

        for (int j = 0; j < students[i]; j++)

        {

            array[i][j] = new int[5];

        }

    }

    for (int i = 0; i <departments; i++)

    {

        cout<<"\n\tStudents of Department "<<i+1<<" \n";

        for (int j = 0; j < students[i]; j++)

        {

            cout<<"\nEnter marks for student "<<j+1<<" \n";

            for (int k = 0; k < 5; k++)

            {

                cout<<"Enter marks of subject"<<k+1<<" : ";

                cin>>array[i][j][k];

            }

        }

    }

    for (int i = 0; i <departments; i++)

    {

        double total;

        double max = INTMAX\_MIN;

        double min = INTMAX\_MAX;

        int count = 0;

        for (int j = 0; j < students[i]; j++)

        {

            for (int k = 0; k < 5; k++)

            {

                total += array[i][j][k];

                if (array[i][j][k]>max)

                {

                    max = array[i][j][k];

                }

                if (array[i][j][k]<min)

                {

                    min = array[i][j][k];

                }

                count++;

            }

        }

        cout<<"\n\tDepartment "<<i+1<<" \n\n";

        cout<<"Total Marks: "<<total<<endl;

        cout<<"Highest Marks: "<<max<<endl;

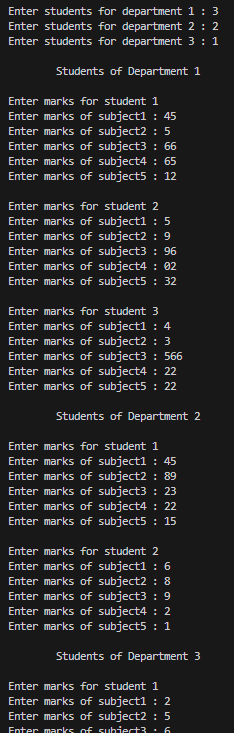
        cout<<"Lowest Marks: "<<min<<endl;

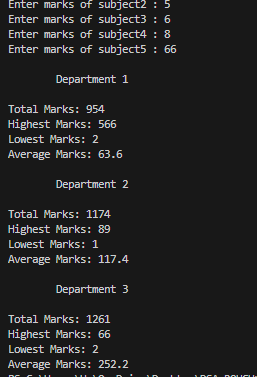
        cout<<"Average Marks: "<<total/count<<endl;

    }

return 0;

}





Task#4

Write a program that creates a 2D array of 5x5 values of type Boolean. Suppose

indices represent people and the value at row i, column j of a 2D array is true just in

case i and j are friends and false otherwise. You can use initializer list to instantiate

and initialize your array to represent the following configuration: (\* means “friends”)

#include<iostream>

using namespace std;

int n = 5;

bool arr1[5][5] = { false, true,  false, true,  true,

                    true,  false, true,  false, true,

                    false, true,  false, false, false,

                    true,  false, false, false, false,

                    true,  true,  false, false, false };

bool \*\*arr2 =  new bool\*[n];

bool findCommonFriend(int a,int b){

    bool flag = 0;

    for (int i = 0; i <n; i++)

    {

        arr2[i] = new bool[n];

    }

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < n; j++)

        {

            arr2[i][j] = arr1[i][j];

        }

    }

    for (int i = 0; i < n; i++)

    {

        if (arr2[a][i] && arr2[b][i])

        {

            flag = 1;

            return flag;

        }

    }

    return flag;

}

int main()

{

    int a,b;

    cout<<"Enter friend 1: ";cin>>a;

    cout<<"Enter friend 2: ";cin>>b;

    cout<<"\nDoes "<<a << " and "<<b<<"  have common friend ? "<<(findCommonFriend(a,b)? "YES" : "NO");

    for (int i = 0; i < n; i++)

    {

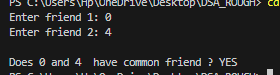
            delete[] arr2[i];

    }

    delete[] arr2;

return 0;

}



Task#5

You are tasked with developing a program to manage and display the Grade Point Average (GPA) for the core courses offered in the first semester of four departments: Software Engineering (SE), Artificial Intelligence (AI), Computer Science (CS), and Data Science (DS). Each department offers a distinct number of core courses for this semester: SE has 3 core courses, AI has 4 core courses, CS has 2 core courses, and DS has 1 core course. To efficiently store and present this data, which type of array structure would you employ? implement a solution using the chosen array structure to display the GPAs of the core courses for each department.

#include<iostream>

using namespace std;

int main()

{

    int dep = 4;

    double \*\*array = new double\*[4];

    int noOfCore[4] = {3,4,2,1};

    double gpa[4];

    for (int i = 0; i < dep; i++)

    {

        array[i] = new double[noOfCore[i]];

        gpa[i] = 0;

    }

    for (int i = 0; i < dep; i++)

    {

        cout<<"\tDepartment "<<i+1<<" \n";

        for (int j = 0; j < noOfCore[i]; j++)

        {

            cout<<"Enter grades of course "<<j+1<< " : ";

            cin>>array[i][j];

            gpa[i]+=array[i][j];

        }

        gpa[i] /=noOfCore[i];

    }

    cout<<"GPA of Software Engineering is : "<<gpa[0]<<endl;

    cout<<"GPA of Artificial Intelligence is : "<<gpa[1]<<endl;

    cout<<"GPA of Computer Science is : "<<gpa[2]<<endl;

    cout<<"GPA of Data Science is : "<<gpa[3]<<endl;

    for (int i = 0; i < dep; i++)

    {

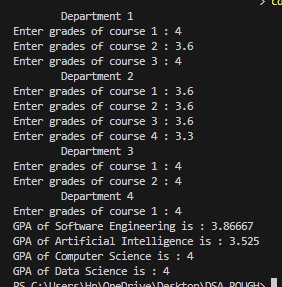
        delete[] array[i];

    }

    delete[] array;

return 0;

}



Task#6

You're developing a program to manage a seating chart for a conference held in a hall with multiple rows of seats. Each row has a different seat capacity. To efficiently handle the seating arrangements, you decide to use a dynamic array. Implement a C++ code that allocates memory for the seating chart and allows attendees' names to be inputted for each seat. Choose and implement the appropriate type of dynamic array for this scenario.

#include<iostream>

using namespace std;

int main()

{

    int rows;

    int seats[rows];

    cout<<"Enter total rows: ";

    cin>>rows;

    string \*\*array = new string\*[rows];

    for (int i = 0; i < rows; i++)

    {

        cout<<"Enter number of seats for row "<<i+1<<" : ";

        cin>>seats[i];

        array[i] = new string[seats[i]];

    }

    for (int i = 0; i < rows; i++)

    {

        cout<<"\nEnter row "<<i+1<<" attendees names: \n";

        for (int j = 0; j < seats[i]; j++)

        {

            cout<<"Enter attendee "<<j+1<<" name: ";

            cin>>array[i][j];

        }

    }

    for (int i = 0; i < rows; i++)

    {

        cout<<"\n\tROW "<<i+1 <<" ATTENDEES ARE: \n\n";

        for (int j = 0; j < seats[i]; j++)

        {

            cout<<"Attendee Number "<<j+1<<" : "<<array[i][j]<<endl;

        }

        delete[] array[i];

    }

    delete[] array;

return 0;

}

