**NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY (NUST)**

**SCHOOL OF MECHANICAL & MANUFACTURING ENGINEERING (SMME)**

**CLASS:** BE ME-15 Sec-B (1st Semester)

**COURSE:** FOP Lab

**Lab Manual:**  Lab Manual No 09

Lab **Instructor**: Sir Affan

Submitted by Muhammad Rayyan

CMS No: 456847

Submission Date: 8-Dec-2023

**Department Of Mechanical Engineering**

**DME Building CAD/CAM Lab**

#include<iostream>

using namespace std;

**// Q:01**

**// Printing sum of left and right diagonal of 2-D 3\*3 Array.**

/\* int main(){

int n,m;

cout<<" Enter no of rows : ";

cin>>n;

cout<<" Enter no of columns : ";

cin>>m;

int sum\_l=0,sum\_r=0;

int arr[n][m];

cout << " Enter elemens in array :" << endl;

for(int i=0; i<n; i++){

for(int j=0; j<m; j++){

cin>>arr[i][j];

}

}

cout<<" Our 2-D matrix is: "<<endl;

for(int i=0; i<n; i++){

for(int j=0; j<m; j++){

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

}

cout<<endl;

}

for( int i=0; i<n; i++){

for( int j=0; j<n; j++){

if( i==j){

sum\_l=sum\_l+arr[i][j];

}

if(i+j==n-1){

sum\_r=sum\_r+arr[i][j];

}

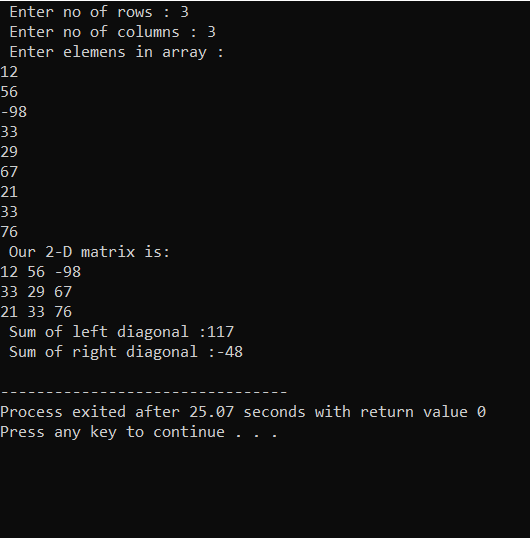
}

}

cout << " Sum of left diagonal :" <<sum\_l<< endl;

cout << " Sum of right diagonal :" <<sum\_r<< endl;

return 0; } \*/



**// Q.02**

**// Addition of two 2-D arrays of 3\*3.**

/\* void add(int arr\_a[3][3],int arr\_b[3][3],int addition[3][3]){

for( int i=0; i<3; i++){

for( int j=0; j<3; j++){

addition[i][j] = arr\_a[i][j] + arr\_b[i][j];

}

}

}

int main(){

int arr1[3][3];

int arr2[3][3];

int addition[3][3];

cout<<" Enter values of 1st array :"<<endl;

for( int i=0; i<3; i++){

for( int j=0; j<3; j++){

cin>>arr1[i][j];

}

}

cout<<" 1st array is : "<<endl;

for( int i=0; i<3; i++){

for( int j=0; j<3; j++){

cout<<arr1[i][j]<<" ";

}

cout<<endl;

}

cout<<" Enter values of 2nd array :"<<endl;

for( int i=0; i<3; i++){

for( int j=0; j<3; j++){

cin>>arr2[i][j];

}

}

cout<<" 2nd array is : "<<endl;

for( int i=0; i<3; i++){

for( int j=0; j<3; j++){

cout<<arr2[i][j]<<" ";

}

cout<<endl;

}

add(arr1,arr2,addition);

cout<<" Addition of 1st & 2nd array is : "<<endl;

for( int i=0; i<3; i++){

for( int j=0; j<3; j++){

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

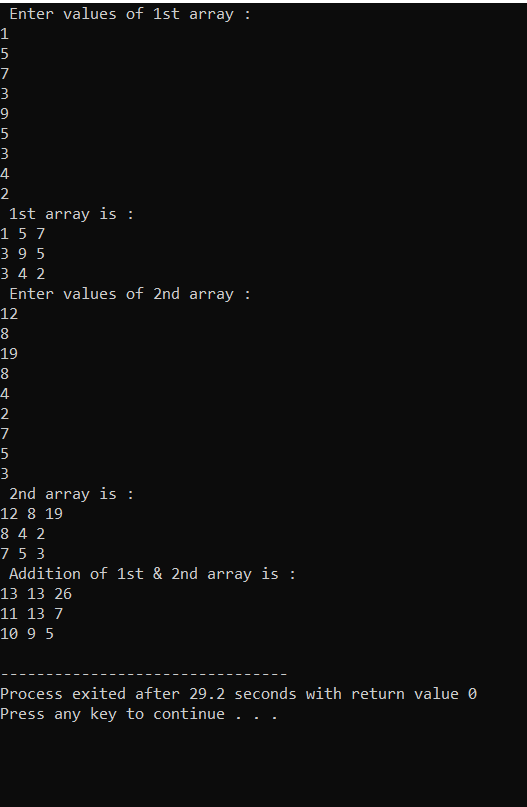
}

cout<<endl;

}

return 0;

} \*/



**// Q.03.**

**// Tranposing the 3\*3 2-D matrix.**

/\* void transpose( int matrix[3][3]){

for(int i = 0; i < 3; i++){

for(int j = i + 1; j < 3; j++){

int temp;

temp = matrix[i][j];

matrix[i][j] = matrix[j][i];

matrix[j][i] = temp;

}

}

}

int main(){

const int n = 3;

const int m = 3;

int arr[n][m];

cout << " Enter values in array :" << endl;

for(int i = 0; i < n; i++){

for(int j = 0; j < m; j++){

cin >> arr[i][j];

}

}

cout << "The original array is:" << endl;

for(int i = 0; i < n; i++){

for(int j = 0; j < m; j++){

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

}

cout << endl;

}

transpose(arr);

cout << "The transposed array is:" << endl;

for(int i = 0; i < n; i++){

for(int j = 0; j < m; j++){

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

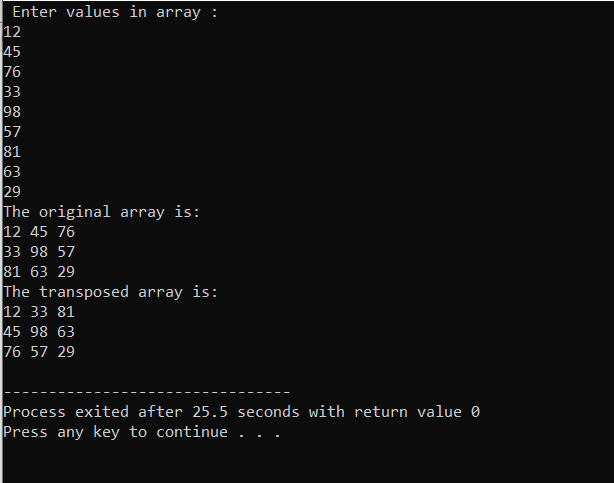
}

cout << endl;

}

return 0;

} \*/



**// Q.04**

**// Matrix Multiplication**

// Making a function of array multiplication.

/\* void multiplyArrays(int smo1[3][3], int smo2[3][3], int result[3][3]) {

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

result[i][j] = 0;

for (int k = 0; k < 3; ++k) {

result[i][j] += smo1[i][k] \* smo2[k][j];

}

}

}

}

int main(){

int row\_a,row\_b,col\_a,col\_b;

cout << "Creating the 1st 2-D array : " << endl;

cout << "Enter no of rows : " ;

cin >> row\_a;

cout << "Enter no of columns : " ;

cin >> col\_a;

int arr1[row\_a][col\_a];

cout << "Enter elements in 1st array :" << endl;

for(int i=0; i<row\_a; i++){

for( int j=0; j<col\_a; j++){

cin >> arr1[i][j];

}

}

cout << "Our 1st 2-D array is :" << endl;

for(int i=0; i<row\_a; i++){

for( int j=0; j<col\_a; j++){

cout<< arr1[i][j]<<" ";

}

cout<<endl;

}

cout << "Now creating the 2nd 2-D array : " << endl;

cout << "Enter no of rows : " ;

cin >> row\_b;

cout << "Enter no of columns : " ;

cin >> col\_b;

int arr2[row\_b][col\_b];

cout << "Enter elements in 2nd array :" << endl;

for(int i=0; i<row\_b; i++){

for( int j=0; j<col\_b; j++){

cin >> arr2[i][j];

}

}

cout << "Our 2nd 2-D array is :" << endl;

for(int i=0; i<row\_b; i++){

for( int j=0; j<col\_b; j++){

cout<< arr2[i][j]<<" ";

}

cout<<endl;

}

int ans[3][3];

multiplyArrays(arr1, arr2, ans);

cout << " The matrix multiplication is:" <<endl;

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

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

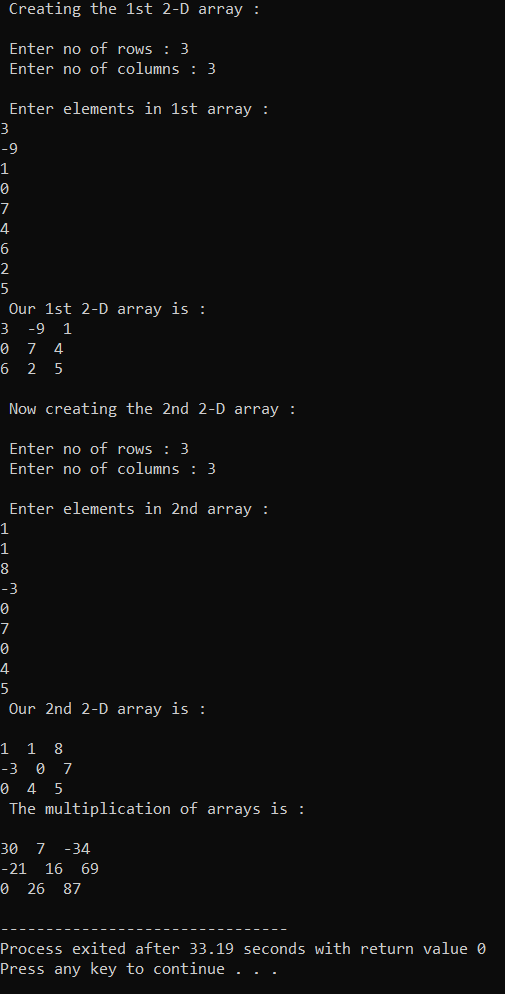
}

cout << endl;

}

return 0;

} \*/



**// Q: 05.**

**// Printing table of 15 by Using recursive function.**

// Declaration of recursive function.

/\* void table(int number, int multiplier = 1) {

if (multiplier > 10) {

return;

}

cout << number << " x " << multiplier << " = " << number \* multiplier << endl;

table(number, multiplier + 1); // Here function again calls itself.

}

int main() {

int num;

cout << " Enter the number : ";

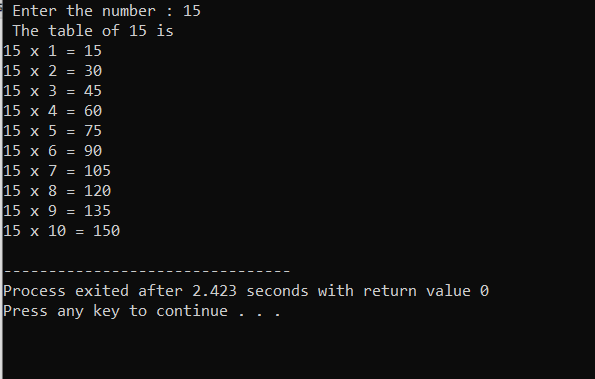
cin >> num;

cout << " The table of "<<num<<" is " << endl;

table(num, 1); // calling the function

return 0;

} \*/



**// Q: 06 ( Home Task )**

**// Inverse of an array.**

// Making adjoint function.

/\* void adjoint(float arr[3][3], float adj[3][3]) {

adj[0][0] = arr[1][1]\*arr[2][2] - arr[2][1]\*arr[1][2];

adj[0][1] = -(arr[1][0]\*arr[2][2] - arr[2][0]\*arr[1][2]);

adj[0][2] = arr[1][0]\*arr[2][1] - arr[2][0]\*arr[1][1];

adj[1][0] = -(arr[0][1]\*arr[2][2] - arr[2][1]\*arr[0][2]);

adj[1][1] = arr[0][0]\*arr[2][2] - arr[2][0]\*arr[0][2];

adj[1][2] = -(arr[0][0]\*arr[2][1] - arr[2][0]\*arr[0][1]);

adj[2][0] = arr[0][1]\*arr[1][2] - arr[1][1]\*arr[0][2];

adj[2][1] = -(arr[0][0]\*arr[1][2] - arr[1][0]\*arr[0][2]);

adj[2][2] = arr[0][0]\*arr[1][1] - arr[1][0]\*arr[0][1];

}

void showArray(float arr[3][3]) {

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

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

}

cout << endl;

}

}

void inverse(float adj[3][3], float det, double inv[3][3]) {

if (det == 0) {

cout << " The matrix is singular so inverse does not exist." << endl;

return;

}

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

inv[i][j] = adj[i][j] / det;

}

}

}

int main() {

float myArray[3][3];

cout << "Enter the elements in array :" << endl;

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

cin >> myArray[i][j];

}

}

cout << " Our 2-D array is :" << endl;

cout << " " << endl;

for( int i = 0; i < 3; i++){

for( int j = 0; j < 3; j++){

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

} cout << endl;

}

float adjArray[3][3];

adjoint(myArray, adjArray);

cout << " " << endl;

cout << " The Adjoint of array is :" << endl;

cout << " " << endl;

showArray(adjArray);

float det=0;

for( int i=0; i<3; i++){

for( int j=0; j<3; j++){

det = myArray[0][0] \* (myArray[1][1] \* myArray[2][2] - myArray[2][1] \* myArray[1][2]) -

myArray[0][1] \* (myArray[1][0] \* myArray[2][2] - myArray[2][0] \* myArray[1][2]) +

myArray[0][2] \* (myArray[1][0] \* myArray[2][1] - myArray[2][0] \* myArray[1][1]);

}

}cout << " " << endl;

cout << " The determinant of array is = " << det << endl;

cout << " " << endl;

double inverseArray[3][3];

inverse(adjArray,det,inverseArray);

cout << " The Inverse of the Array is:" <<endl;

cout << " " << endl;

for (int i = 0; i < 3; ++i) {

for (int j = 0; j < 3; ++j) {

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

}

cout <<endl;

}

return 0;

} \*/

