National University of Science and technology

(NUST)

CS-114 - Fundamental of Programing

Lab Manual # 09

Course Instructor: Dr Jawad Khan

Lab Instructor: Muhammad Affan

Student name: Asim Imran

QALAM ID: 476434

Date: 12/12/2023

Lab Tasks

Task No 1:

Make 2D Array in C++ and print left diagonal and right diagonal sum of a 3x3 matrix.

Code:

```
#include <iostream>

using namespace std;

int main(){

	int n,m,sum,sum1;

	cout<<"Enter the Rows of the array:";

	cin>>n;

	cout<<"Enter the Columns of the array:";

	cin>>m;

	int arr[n][m];

	for(int i=1;i<=n;i++){

		for(int j=1;j<=m;j++){

			cout<<"Enter element of "<<i<<","<<j<<":";

			cin>>arr[i][j];

		}

	}

		for(int i=1;i<=n;i++){

		for(int j=1;j<=m;j++){

		if (i==j){

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


		}

			else if(i+j==n){
```

```
                    sum1=sum1+arr[i][j];
            }
            }
        }
        cout<<"Sum of Diagnol elements of matrix ="<<sum<<endl;
        cout<<"Sum of Non-diagnol elements of matrix ="<<sum1<<endl;
        return 0;
}
```

Output:

```
C:\Users\ADMIN\OneDrive\Documents\L-9 T-1.exe
Enter the Rows of the array:3
Enter the Columns of the array:3
Enter element of 1,1:1
Enter element of 1,2:3
Enter element of 1,3:4
Enter element of 2,1:5
Enter element of 2,2:6
Enter element of 2,3:7
Enter element of 3,1:8
Enter element of 3,2:9
Enter element of 3,3:10
Sum of Diagnol elements of matrix =17
Sum of Non-diagnol elements of matrix =9

--------------------------------
Process exited after 12.79 seconds with return value 0
Press any key to continue . . .
```

Task No 2:

Write a function to add two 2D arrays of size 3x3.

Code:

```
#include <iostream>
using namespace std;
```

```
void matrixSum() {
    int n = 3, m = 3;
    int a1[n][m], a2[n][m], sum[n][m];

    for (int i = 1; i <= 3; i++) {
        for (int j = 1; j <= 3; j++) {
            cout << "Enter the Elements of First Array:" << i << "," << j << ":";
            cin >> a1[i][j];
        }
    }

    for (int i = 1; i <= 3; i++) {
        for (int j = 1; j <= 3; j++) {
            cout << "Enter the Elements of Second Array:" << i << "," << j << ":";
            cin >> a2[i][j];
        }
    }

    for (int i = 1; i <= 3; i++) {
        for (int j = 1; j <= 3; j++) {
            sum[i][j] = a1[i][j] + a2[i][j];
        }
    }

    for (int i = 1; i <= 3; i++) {
        for (int j = 1; j <= 3; j++) {
            cout << sum[i][j] << " ";
```

```
        }
        cout << endl;
    }
}

int main() {
    matrixSum();
    return 0;
}
```

Output:

```
C:\Users\ADMIN\OneDrive\Documents\L-9 T-2.exe
Enter the Elements of First Array:1,1:1
Enter the Elements of First Array:1,2:2
Enter the Elements of First Array:1,3:3
Enter the Elements of First Array:2,1:4
Enter the Elements of First Array:2,2:5
Enter the Elements of First Array:2,3:6
Enter the Elements of First Array:3,1:7
Enter the Elements of First Array:3,2:8
Enter the Elements of First Array:3,3:9
Enter the Elements of Second Array:1,1:1
Enter the Elements of Second Array:1,2:2
Enter the Elements of Second Array:1,3:3
Enter the Elements of Second Array:2,1:4
Enter the Elements of Second Array:2,2:5
Enter the Elements of Second Array:2,3:6
Enter the Elements of Second Array:3,1:7
Enter the Elements of Second Array:3,2:8
Enter the Elements of Second Array:3,3:9
2 4 6
8 10 12
14 16 18

--------------------------------
Process exited after 17.35 seconds with return value 0
Press any key to continue . . .
```

Task No 3:

Using 2D arrays in C++, take transpose of a 3x3 matrix. Make a transpose function.

Code:

```
#include<iostream>

using namespace std;

int main(){

int arr1[3][3], transpose[3][3];

int i,j;

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

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

cout<<"Enter Value of Element "<<j<<","<<i<<":";

 cin>>arr1[i][j];

 }

}

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

 cout<<endl;

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

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

  }

}

cout<<endl;

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

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

 transpose[j][i]=arr1[i][j];

 }

}
```

```
cout<<endl<<"Transpose of Given Matrix is: "<<endl;
for(int i=0; i<3; i++){
 cout<<endl;
for(int j=0; j<3; j++){
 cout<<transpose[i][j]<<" ";
 }
}
return 0;
}
```

Output:

```
C:\Users\ADMIN\OneDrive\Documents\L-9 T-3.exe
Enter Value of Element 0,0:14
Enter Value of Element 1,0:25
Enter Value of Element 2,0:36
Enter Value of Element 0,1:96
Enter Value of Element 1,1:85
Enter Value of Element 2,1:74
Enter Value of Element 0,2:58
Enter Value of Element 1,2:52
Enter Value of Element 2,2:44

14 25 36
96 85 74
58 52 44

Transpose of Given Matrix is:

14 96 58
25 85 52
36 74 44
--------------------------------
Process exited after 22.3 seconds with return value 0
Press any key to continue . . .
```

Task No 4:

Using 2D arrays in C++, implement 3x3 matrix multiplication. Make a function.

Code:

```
#include <iostream>
using namespace std;
int main(){
int x1[3][3], x2[3][3], multiple[3][3];
int i, j;
cout<<"Enter the Values for Array 1: "<<endl;
for(i=0; i<3; i++){
 for(j=0; j<3; j++){
 cout<<"Enter Value of Element("<<j<<","<<i<<"):";
 cin>>x1[i][j];
 }
 }
cout<<endl<<"Array 1 Filled! Now Input Array 2: "<<endl;
for(i=0; i<3; i++){
 for(j=0; j<3; j++){
 cout<<"Enter Value of Element ("<<j<<","<<i<<"): ";
 cin>>x2[i][j];
 }
 }
for ( i = 0; i < 3; i++) {
 for ( j = 0; j < 3; j++) {
 multiple[i][j] = 0;
 for (int k = 0; k < 3; ++k) {
 multiple[i][j] += x1[i][k] * x2[k][j];
 }
 }
```

```
 }
cout<<endl<<"Multiple of Given Array is: "<<endl;
for(int i=0; i<3; i++){
 cout<<endl;
 for(int j=0; j<3; j++){
 cout<<multiple[i][j]<<" ";
 }
 }
 return 0;
}
```

Output:

```
C:\Users\ADMIN\OneDrive\Documents\L-9 T-4.exe
Enter the Values for Array 1:
Enter Value of Element(0,0):1
Enter Value of Element(1,0):2
Enter Value of Element(2,0):3
Enter Value of Element(0,1):4
Enter Value of Element(1,1):5
Enter Value of Element(2,1):6
Enter Value of Element(0,2):7
Enter Value of Element(1,2):8
Enter Value of Element(2,2):9

Array 1 Filled! Now Input Array 2:
Enter Value of Element (0,0): 123
Enter Value of Element (1,0): 456
Enter Value of Element (2,0): 789
Enter Value of Element (0,1): 101112
Enter Value of Element (1,1): 131415
Enter Value of Element (2,1): 161718
Enter Value of Element (0,2): 192020
Enter Value of Element (1,2): 21
Enter Value of Element (2,2): 213

Multiple of Given Array is:

778407 263349 324864
1658172 659025 813024
2537937 1054701 1301184
```

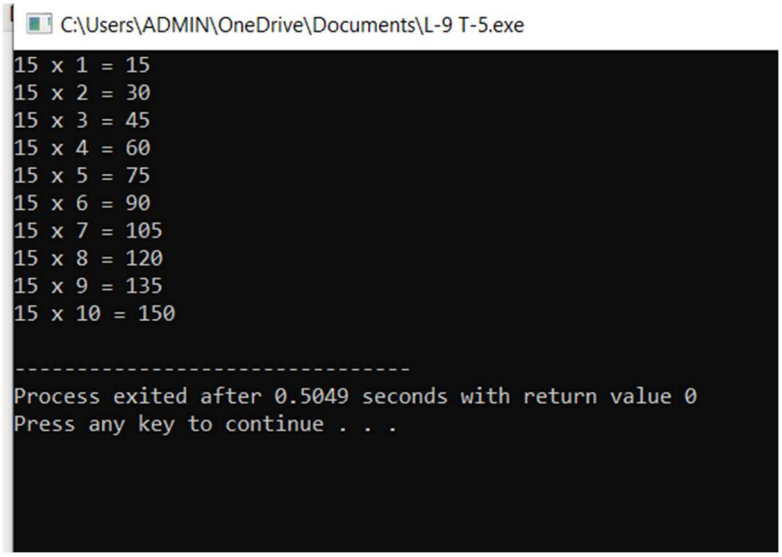
Task No 5:

Print the multiplication table of 15 using recursion

Code:

```
#include <iostream>

using namespace std;

void multiplication(int number, int multiplier = 1) {

 if (multiplier <= 10) {

 int result = number * multiplier;

 cout << number << " x " << multiplier << " = " << result << endl;

 multiplication(number, multiplier + 1);

 }

}

int main() {

 multiplication(15);

 return 0;

}
```

Output:



Home Tasks

Home Task No 1:

Write a C++ program to take inverse of a 3x3 matrix using its determinant and adjoint.

Code:

```
#include <iostream>

using namespace std;

int main() {

 float matrix[3][3];

 cout << "Enter the elements of the 3x3 matrix:" << endl;

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

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

          cout<<"Enter the element of 3x3 matrix("<<i<<","<<j<<"):";

          cin >> matrix[i][j];

 }

 }

   //Taking a 3 by 3 matrix as input from user

 cout << "The entered matrix is equal to :" << endl;

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

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

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

  cout << endl;                //Outputting the entered 3 by 3 matrix

 }

 float det = matrix[0][0] * (matrix[1][1] * matrix[2][2] - matrix[2][1] * matrix[1][2]) -

 matrix[0][1] * (matrix[1][0] * matrix[2][2] - matrix[2][0] * matrix[1][2]) +

 matrix[0][2] * (matrix[1][0] * matrix[2][1] - matrix[2][0] * matrix[1][1]);
```

```
if (det == 0) {                    //Taking the dterminant of  matrix and checking if det is equal
to zero
cout << "The matrix is singular and cannot be inverted!" << endl;
}
else{
                    //If matrix is not singular than taking the adverse of the  same matrix
float adj[3][3];
for (int i = 0; i < 3; ++i)
for (int j = 0; j < 3; ++j)
adj[i][j] = (matrix[(j + 1) % 3][(i + 1) % 3] * matrix[(j + 2) % 3][(i + 2) % 3] -
matrix[(j + 1) % 3][(i + 2) % 3] * matrix[(j + 2) % 3][(i + 1) % 3]);
float inv[3][3];
for (int i = 0; i < 3; ++i)
for (int j = 0; j < 3; ++j)
inv[i][j] = adj[i][j] / det;       //applying the formula for inverse of matrix
cout << "The inverse of the matrix is:" << endl;  //Outputting the   inverse of matrix
for (int i = 0; i < 3; ++i) {
for (int j = 0; j < 3; ++j)
cout << inv[i][j] << " ";
cout << endl;
}
}
return 0;
}
```

Output:

C:\Users\ADMIN\OneDrive\Documents\L-9 H-1.exe

```
Enter the elements of the 3x3 matrix:
Enter the element of 3x3 matrix(0,0):1
Enter the element of 3x3 matrix(0,1):2
Enter the element of 3x3 matrix(0,2):3
Enter the element of 3x3 matrix(1,0):4
Enter the element of 3x3 matrix(1,1):5
Enter the element of 3x3 matrix(1,2):6
Enter the element of 3x3 matrix(2,0):7
Enter the element of 3x3 matrix(2,1):8
Enter the element of 3x3 matrix(2,2):9
The entered matrix is equal to :
1 2 3
4 5 6
7 8 9
The matrix is singular and cannot be inverted!

--------------------------------
Process exited after 8.864 seconds with return value 0
Press any key to continue . . .
```