LAB MANUAL 9

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

```cpp
#include <iostream>
using namespace std;
int main()
{
 int sum1=0,sum2=0;    // Declaring 2D array
 int arr[3][3];       // Initialize 2D array using loop
 cout<<"Enter elements of matrix: "<<endl;
 for (int i=0;i<3;i++)
 {
  for (int j=0;j<3;j++)
  {
    cout<<"Enter element["<<i<<"]["<<j<<"]: ";
    cin>>arr[i][j];
  }
 }
 cout<<"The Matrix Form is: "<<endl; //Printing the element of 2D array
 for(int i=0;i<3;i++)
 {
  for (int j=0;j<3;j++)
  {
     cout<<arr[i][j]<<"  ";
   if(i==j)
   {
    sum1+=arr[i][j];
   }
   if(i+j==2)
   {
    sum2+=arr[i][j];
   }
  }
  cout<<endl;
  }
  cout<<"The sum of left diagonal is = "<<sum1<<endl;
  cout<<"The sum of right diagonal is = "<<sum2;
  return 0;
```

```
D:\FOP\lab manuals\prac.exe
Enter elements of matrix:
Enter element[0][0]: 1
Enter element[0][1]: 2
Enter element[0][2]: 3
Enter element[1][0]: 4
Enter element[1][1]: 5
Enter element[1][2]: 6
Enter element[2][0]: 7
Enter element[2][1]: 8
Enter element[2][2]: 9
The Matrix Form is:
1  2  3
4  5  6
7  8  9
The sum of left diagonal is = 15
The sum of right diagonal is = 15
--------------------------------
Process exited after 6.948 seconds with return value 0
```

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

```cpp
#include <iostream>
using namespace std;
void matrices(int matrix1[3][3],int matrix2[3][3],int addmatrice[3][3])
//function declaration
{
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            addmatrice[i][j]=matrix1[i][j]+matrix2[i][j];
        }
    }
}
int main()
{
    int matrix1[3][3];
    int matrix2[3][3];
    int addmatrice[3][3];
    cout<<"Elements of 1st Matrix are: "<<endl;
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            cout<<"Enter element ["<<i<<"]["<<j<<"] = ";
            cin>>matrix1[i][j];
        }
    }
    cout<<"Elements of 2nd Matrix are: "<<endl;
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            cout<<"Enter element ["<<i<<"]["<<j<<"] = ";
            cin>>matrix2[i][j];
        }
    }
    matrices(matrix1,matrix2,addmatrice); //calling the function
    cout<<"Sum of matrix is = "<<endl;
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            cout<<addmatrice[i][j]<<"  ";
        }
        cout<<endl;
    }
return 0;
}
```

```
D:\FOP\lab manuals\lb9.exe
Elements of 1st Matrix are:
Enter element [0][0] = 1
Enter element [0][1] = 2
Enter element [0][2] = 3
Enter element [1][0] = 4
Enter element [1][1] = 5
Enter element [1][2] = 6
Enter element [2][0] = 7
Enter element [2][1] = 8
Enter element [2][2] = 9
Elements of 2nd Matrix are:
Enter element [0][0] = 1
Enter element [0][1] = 2
Enter element [0][2] = 3
Enter element [1][0] = 4
Enter element [1][1] = 5
Enter element [1][2] = 6
Enter element [2][0] = 7
Enter element [2][1] = 8
Enter element [2][2] = 9
Sum of matrix is =
2  4  6
8  10  12
14  16  18

--------------------------------
Process exited after 25.85 seconds with return value 0
```

LAB MANUAL 9

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

```cpp
#include <iostream>
using namespace std;
void transpose(int arr[3][3]) //declaration of function
{
    cout<<"Transpose is: "<<endl;
        for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            cout<<arr[j][i]<<"  ";
        }
        cout<<endl;
    }
}
int main() {
    int arr[3][3];
    cout<<"Enter elements of matrix "<<endl;
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            cout<<"Enter element["<<i<<"]["<<j<<"]: ";
            cin>>arr[i][j];
        }
    }
    cout<<endl;
    cout<<"Matrix will be "<<endl;
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<3;j++)
        {
            cout<<arr[i][j]<<"  ";
        }
        cout<<endl;
    }
    transpose(arr); //calling the function
    return 0; }
```

```
D:\FOP\lab manuals\prac2.exe
Enter elements of matrix
Enter element[0][0]: 1
Enter element[0][1]: 2
Enter element[0][2]: 3
Enter element[1][0]: 4
Enter element[1][1]: 5
Enter element[1][2]: 6
Enter element[2][0]: 7
Enter element[2][1]: 8
Enter element[2][2]: 9

Matrix will be
1  2  3
4  5  6
7  8  9
Transpose is:
1  4  7
2  5  8
3  6  9

--------------------------------
Process exited after 5.162 seconds with return value 0
Press any key to continue . . .
```

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

```cpp
#include <iostream>
using namespace std;
void multiplymatrices(int matrix1[3][3], int matrix2[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 i=0;i<3;i++) {
for (int j=0;j<3; j++) {
for(int k=0;k<3; k++) {
result[i][j] =result[i][j]+ matrix1[i][k] * matrix2[k][j];
        }
    }
 }
}
int main() {
int  matrix1[3][3];
int matrix2[3][3];
int result[3][3];
cout << "Enter elements of first matrix "<<endl;
for (int i=0;i<3;i++){
for (int j=0;j<3;j++) {
cout<<"Enter elements["<<i<<"]["<<j<<"] : ";
cin>>matrix1[i][j];
```

LAB MANUAL 9

```
27  }
28  }
29  cout<<"Enter elements of second matrix "<<endl;
30  for (int i=0;i<3;i++) {
31  for (int j=0;j<3;j++) {
32  cout<<"Enter elements["<<i<<"]["<<j<<"] : ";
33  cin >> matrix2[i][j];
34  }
35  }
36  multiplymatrices(matrix1, matrix2, result);//Calling the function
37  // Display the result
38  cout << "Multiplication of matrices: "<<endl;
39  for (int i=0;i<3;i++) {
40  for (int j=0;j<3;j++) {
41  cout<<result[i][j]<< " ";
42  }
43  cout<<endl;
44  }
45  return 0;
46  }
```

5. Print the multiplication table of 15 using recursion.

```
1   #include<iostream>
2   using namespace std;
3   int table(int n,int x) //function declaration
4   {
5       if(x<=10){
6       cout<<n<<" X "<<x<<" = "<<n*x<<endl;
7       table(n,x+1);
8   }
9   }
10  int main()
11  {
12      int n=15;
13      table(n,1); //calling the function
14    return 0;
15  }
```

```
D:\FOP\lab manuals\prc5.exe
15 X 1 = 15
15 X 2 = 30
15 X 3 = 45
15 X 4 = 60
15 X 5 = 75
15 X 6 = 90
15 X 7 = 105
15 X 8 = 120
15 X 9 = 135
15 X 10 = 150

--------------------------------
Process exited after 8.601 seconds with return value 0
Press any key to continue . . .
```

HOMETASK:

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

LAB MANUAL 9

```cpp
#include<iostream>
using namespace std;
double Determinant(int mat[3][3])
{
return mat[0][0]*(mat[1][1]*mat[2][2]-mat[1][2]*mat[2][1])-
mat[0][1]*(mat[1][0]*mat[2][2]-mat[1][2]*mat[2][0])+mat[0][2]*(mat[1][0]*mat[2][1]-
mat[1][1]*mat[2][0]);
}
void Adjoint(int mat[3][3], int adj[3][3]) {
for (int i=0;i<3;i++) {
for (int j=0;j<3;j++) {
adj[i][j]=(mat[(j+1)%3][(i+1)%3]*mat[(j+2)%3][(i+2)%3]) -
(mat[(j+1)%3][(i+2)%3]*mat[(j+2)%3][(i+1)%3]);
}
}
}
void Inverse(int mat[3][3], double inv[3][3]) {
double det = Determinant(mat);
if (det==0) {
cout<<"Inverse does not exist"<<endl;
return;
}
int adj[3][3];
Adjoint(mat,adj);
for (int i=0;i<3;i++) {
for (int j=0;j<3;j++) {
inv[i][j]=adj[i][j]/det;
}
}
}
int main() {
int mat[3][3];
cout << "Enter elements of matrix:"<<endl;
for (int i=0;i<3;i++) {
for (int j=0;j<3;j++) {
cout<<"Enter element["<<i<<"]["<<j<<"] : ";
cin>>mat[i][j];
}
}
double inv[3][3];
Inverse(mat, inv);
if (Determinant(mat) != 0) {
cout << "Inverse of the matrix:"<<endl;
for (int i=0;i<3;i++) {
for (int j=0;j<3;j++) {
cout<<inv[i][j] << " ";
}
cout<<endl;
}
}
return 0;
}
```

```
D:\FOP\lab manuals\lb9hmt1.exe
Enter elements of matrix:
Enter element[0][0] : 4
Enter element[0][1] : 2
Enter element[0][2] : 6
Enter element[1][0] : 4
Enter element[1][1] : 6
Enter element[1][2] : 9
Enter element[2][0] : 5
Enter element[2][1] : 3
Enter element[2][2] : 1
Inverse of the matrix:
0.190909 -0.145455 0.163636
-0.372727 0.236364 0.109091
0.163636 0.0181818 -0.145455

--------------------------------
Process exited after 23.49 seconds with return value 0
Press any key to continue . . .
```