

Fundamentals of Programming

Name: Syed Asad Raza Zaidi

Section: ME-15A

Roll no: 464239

LAB TASK8

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

```
#include<iostream>

using namespace std;

int main()
{
    int a[3][3];

    cout<<"Enter elements in the 3x3 matrix :";

    for(int i=0; i<3; i++){          //loop to take input.
        for(int j=0; j<3; j++){
            cin>>a[i][j];
        }
    }

    cout<<"Matrix :"<<endl;

    for(int i=0; i<3; i++){          //loop to show matrix.
        for(int j=0; j<3; j++){
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }

    cout<<endl;
```

```

int lsum=0, rsum=0;

for(int i=0; i<3; i++){    //loop to sum left diagonal.
    for(int j=i; j<=i; j++){
        lsum = lsum + a[i][j];
    }
}

cout<<"Sum of Left Daigonal = "<<lsum<<endl;

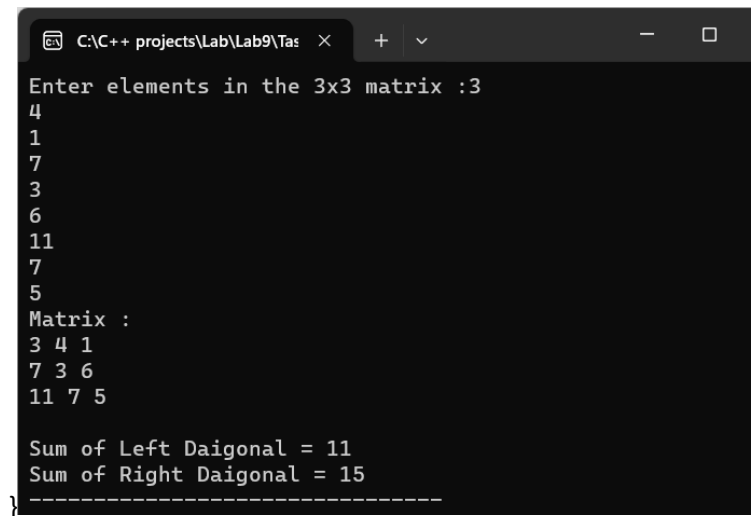
int l=0, m=0;

for(int i=0; i<3; i++){    //loop to sum right diagonal.
    m=2;
    l=m-i;
    rsum = rsum + a[i][l];
}

cout<<"Sum of Right Daigonal = "<<rsum;

return 0;

```



The screenshot shows a Windows-style application window titled "C:\C++ projects\Lab\Lab9\Tas". The window contains a text-based interface for a C++ program. It prompts the user to "Enter elements in the 3x3 matrix :3". The user has entered the following values: 4, 1, 7, 3, 6, 11, 7, 5. The program then displays the matrix as follows:

3	4	1
7	3	6
11	7	5

Below the matrix, the program outputs the sums of the diagonals: "Sum of Left Daigonal = 11" and "Sum of Right Daigonal = 15". The window ends with a series of dashes and a closing brace "}".

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

```
#include<iostream>
```

```
using namespace std;
```

```
void sum(int x[3][3], int y[3][3], int z[3][3]){  
    for(int i=0; i<3; i++){  
        for(int j=0; j<3; j++){  
            z[i][j] = x[i][j]+y[i][j];  
        }  
    }  
}
```

```
int main()  
{  
    int a[3][3], b[3][3];  
    cout<<"Enter elements in Matrix A :";  
    for(int i=0; i<3; i++){        //loop to take input.  
        for(int j=0; j<3; j++){  
            cin>>a[i][j];  
        }  
    }  
    cout<<"Enter elements in Matrix B :";  
    for(int i=0; i<3; i++){        //loop to take input.  
        for(int j=0; j<3; j++){  
            cin>>b[i][j];  
        }  
    }  
    int c[3][3];  
    sum(a, b, c);
```

```

        for(int i=0; i<3; i++){          //loop to output.
            for(int j=0; j<3; j++){
                cout<<c[i][j]<<" ";
            }
            cout<<endl;
        }

        return 0;
    }

```

```

C:\C++ projects\Lab\Lab9\Task9.cpp
Enter elements in Matrix A :2
7
0
3
5
1
4
8
9
Enter elements in Matrix B :1
2
7
0
5
3
5
7
2
Sum of Both Matrices :
3 9 7
3 10 4
9 15 11

```

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

```
#include <iostream>
```

```
using namespace std;
```

```
void transposeMatrix(int matrix[3][3]) {    // Function to find the transpose of a 3x3 matrix
```

```
    int temp;
```

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

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

```
            temp = matrix[i][j];                // Swap elements at [i][j] and [j][i]
```

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

```
            matrix[j][i] = temp;
```

```
        }
```

```
    }
```

```
}
```

```
int main() {
```

```
    int matrix[3][3];
```

```
    cout<<"Enter Elements in a 3x3 matrix :";
```

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

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

```
            cin >> matrix[i][j] ;
```

```
        }
```

```
    }
```

```
    cout<<"Original Matrix :"<<endl;
```

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

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

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

```
        }
```

```
    cout<<endl;
```

```
}
```

```

cout<<endl;

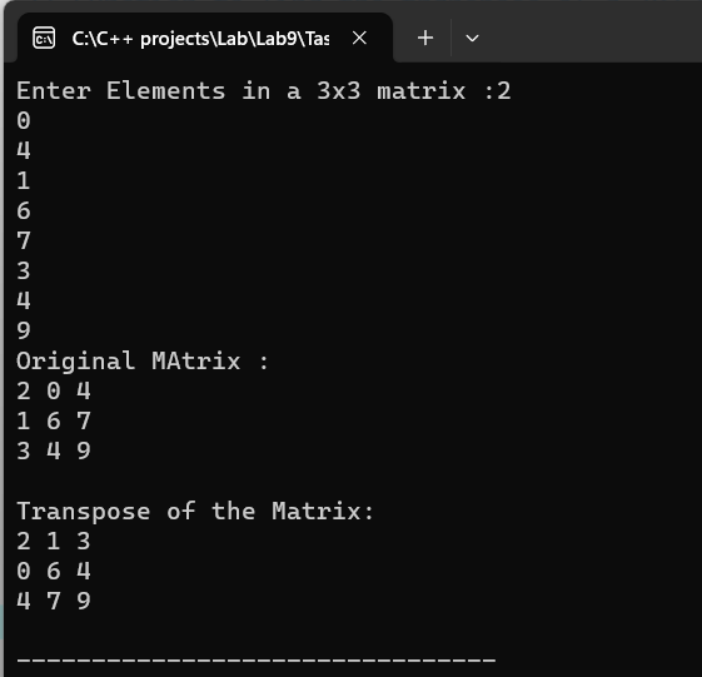
transposeMatrix(matrix);           // Call the transpose function

cout<<"Transpose of the Matrix:"<<endl;    // Display transposed matrix

for (int i = 0; i < 3; ++i) {
    for (int j = 0; j < 3; ++j) {
        cout<< matrix[i][j] << " ";
    }
    cout << endl;
}

return 0;
}

```



```

C:\C++ projects\Lab\Lab9\Task9.cpp
Enter Elements in a 3x3 matrix :2
0
4
1
6
7
3
4
9
Original Matrix :
2 0 4
1 6 7
3 4 9

Transpose of the Matrix:
2 1 3
0 6 4
4 7 9
-----

```

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

```

#include<iostream>

using namespace std;

void Product(int x[3][3], int y[3][3], int z[3][3] ){
    for( int i=0; i<3; i++){
        for( int j=0; j<3; j++){
            z[i][j] = 0;
            for( int k=0; k<3 ; k++){
                z[i][j] += x[i][k] * y[k][j];
            }
        }
    }
}

```

```

int main()
{
    int a[3][3] , b[3][3] , c[3][3] ;

    cout<<"Enter Elements in the Matrix A :";

    for (int i = 0; i < 3; ++i) {
        for (int j = 0; j < 3; ++j) {
            cin >> a[i][j] ;
        }
    }
}

```

```

cout<<"Enter Elements in the Matrix B :";

    for (int i = 0; i < 3; ++i) {
        for (int j = 0; j < 3; ++j) {
            cin >> b[i][j] ;
        }
    }
}

```

```

    }
    Product(a, b, c);
    for(int i=0; i<3; i++){
        for( int j=0; j<3; j++){
            cout<<a[i][j]<<" ";
        }
        cout<<endl;
    }
    cout<<" x  " <<endl;
    for(int i=0; i<3; i++){
        for( int j=0; j<3; j++){
            cout<<b[i][j]<<" ";
        }
        cout<<endl;
    }
    cout<<" =  " <<endl;

//  cout<<"Resultant Matrix :";
    for(int i=0; i<3; i++){
        for( int j=0; j<3; j++){
            cout<<c[i][j]<<" ";
        }
        cout<<endl;
    }
    return 0;
}

```



```
C:\C++ projects\Lab\Lab9\Tas x + v
7
8
9
Enter Elements in the Matrix B :9
8
7
6
5
4
3
2
1
1 2 3
4 5 6
7 8 9
x
9 8 7
6 5 4
3 2 1
=
30 24 18
84 69 54
138 114 90
-----
```

5. Print the multiplication table of 15 using recursion.

```
#include <iostream>
```

```
using namespace std;
```

```
void Table(int N, int X) {
```

```
    if (X> 10) {        // Print up to 10 multiples
```

```
        return;
```

```
    }
```

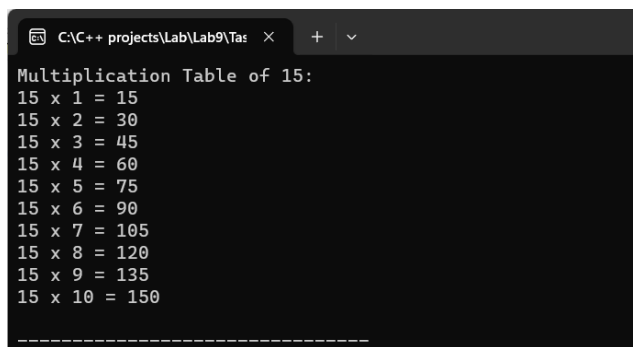
```
    cout<<N <<" x " <<X<<" = " <<N * X<<endl;
```

```
    // Recursively call the function to print the next multiple
```

```
    Table(N, X + 1);
```

```
}
```

```
int main() {  
  
    int tableOf = 15;  
  
    cout << "Multiplication Table of " << tableOf << ":" << std::endl;  
    Table(tableOf, 1); // Start from 1st multiple  
  
    return 0;  
}
```



A screenshot of a C++ program's output in a terminal window. The window title is "C:\C++ projects\Lab\Lab9\Task9.exe". The output displays the title "Multiplication Table of 15:" followed by ten lines of multiplication results: 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, and 15 x 10 = 150. A dashed line is visible at the bottom of the output.

```
C:\C++ projects\Lab\Lab9\Task9.exe  
Multiplication Table of 15:  
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  
-----
```