Computer Programming Lab

Manual #9

Submitted by:

Shah Jahan khan

469192 (section C)

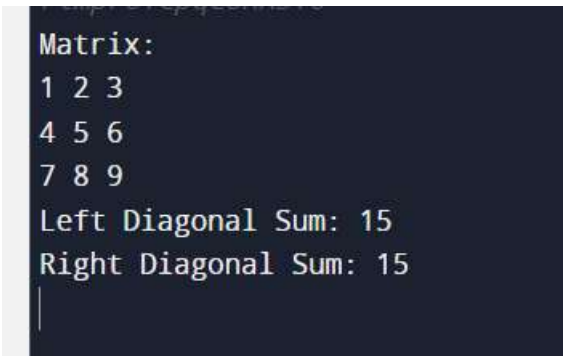
Submitted to:

Sir Affan Tariq

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 matrix[3][3] = {{1, 2, 3},
                       {4, 5, 6},
                       {7, 8, 9}}
   int leftDiagonalSum = 0;
   int rightDiagonalSum = 0;
   for (int i = 0; i < 3; i++) {
      leftDiagonalSum += matrix[i][i];
```

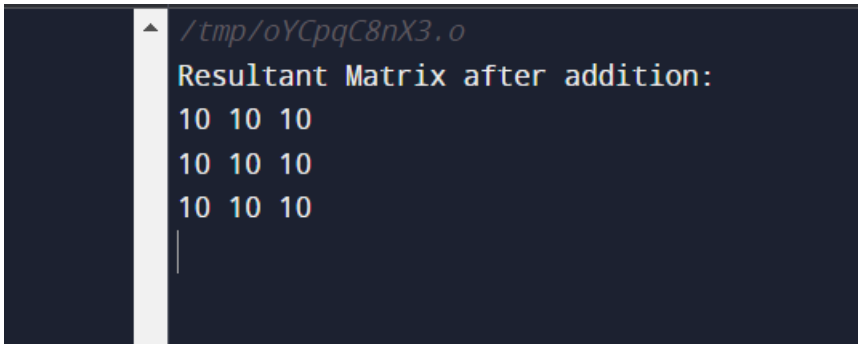
```
    }
    for (int i = 0; i < 3; i++) {
        rightDiagonalSum += matrix[i][2 - i];
    }
    cout << "Matrix:" << endl;
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << matrix[i][j] << " ";
        }
        cout << endl;
    }
    cout << "Left Diagonal Sum: " << leftDiagonalSum << endl;
    cout << "Right Diagonal Sum: " << rightDiagonalSum << endl;

    return 0;
}
```



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

```
#include <iostream>
using namespace std;
void addMatrices(int mat1[3][3], int mat2[3][3], int result[3][3]) {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
```

```
            result[i][j] = mat1[i][j] + mat2[i][j];
        }
    }
}

int main() {
    int matrix1[3][3] = {{1, 2, 3},
                         {4, 5, 6},
                         {7, 8, 9}};

    int matrix2[3][3] = {{9, 8, 7},
                         {6, 5, 4},
                         {3, 2, 1}};

    int resultMatrix[3][3];
    addMatrices(matrix1, matrix2, resultMatrix);

    cout << "Resultant Matrix after addition:" << endl;
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << resultMatrix[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```



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

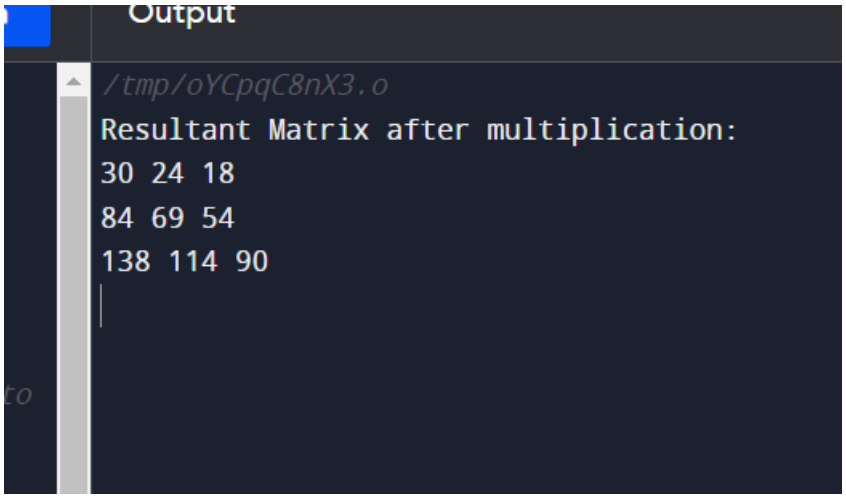
```
#include <iostream>
using namespace std;

// Function to calculate transpose of a 3x3 matrix
void transposeMatrix(int mat[3][3]) {
    int temp;
    for (int i = 0; i < 3; i++) {
        for (int j = i + 1; j < 3; j++) {
            // Swap elements at (i, j) and (j, i)
            temp = mat[i][j];
            mat[i][j] = mat[j][i];
            mat[j][i] = temp;
        }
    }
}

int main() {
    int matrix[3][3] = {{1, 2, 3},
                        {4, 5, 6},
                        {7, 8, 9}};

    // Print the original matrix
    cout << "Original Matrix:" << endl;
```

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

    // Calculate the transpose of the matrix
    transposeMatrix(matrix);

    // Print the transposed matrix
    cout << "\nTranspose of Matrix:" << endl;
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << matrix[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```

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

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

// Function to perform matrix multiplication for two 3x3 matrices
void multiplyMatrices(int mat1[3][3], int mat2[3][3], int result[3][3]) {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            result[i][j] = 0; // Initialize result matrix element at (i, j) to 0
            for (int k = 0; k < 3; k++) {
                result[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }
}

int main() {
    int matrix1[3][3] = {{1, 2, 3},
                         {4, 5, 6},
                         {7, 8, 9}};

    int matrix2[3][3] = {{9, 8, 7},
                         {6, 5, 4},
                         {3, 2, 1}};

    int resultMatrix[3][3];

    // Perform matrix multiplication
    multiplyMatrices(matrix1, matrix2, resultMatrix);

    // Print the result matrix
```

```
    cout << "Resultant Matrix after multiplication:" << endl;
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << resultMatrix[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```



Print the multiplication table of 15 using recursion.

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

void printMultiples(int num, int multiple) {
    if (multiple <= 10) {
        cout << num << " * " << multiple << " = " << num * multiple << endl;
        printMultiples(num, multiple + 1);
    }
}
```

```
int main() {
    int number = 15;

    cout << "Multiplication table of " << number << ":" << endl;
    printMultiples(number, 1);

    return 0;
}
```