EXPERIMENT-9

Program:

WAP to implement Strassen’s Matrix Multiplication algorithm using c/c++ and write the complexity.

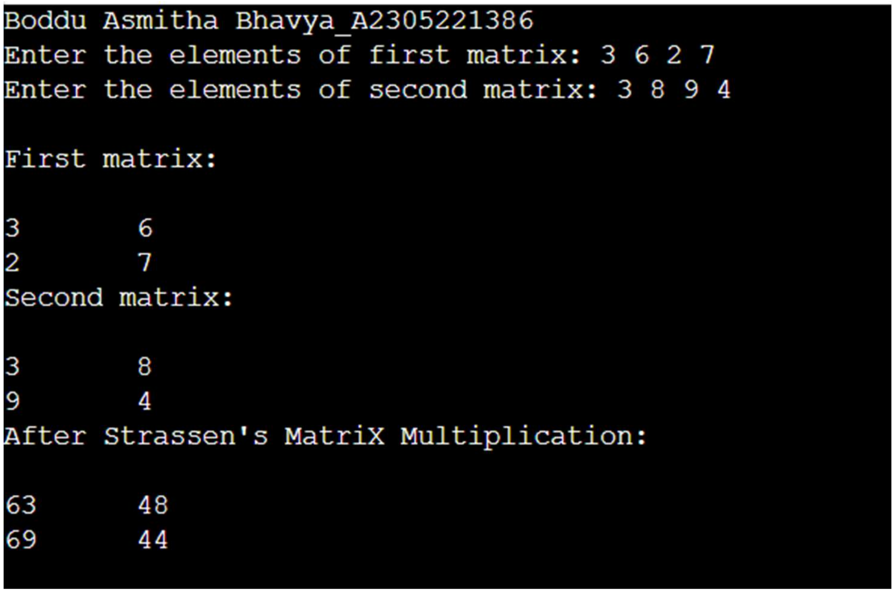
Pseudo code:

```
function strassenMatrixMultiply(A, B):
    if size(A) == 1:
        # Base case: Multiply two 1x1 matrices
        return A * B
    # Split matrices A and B into four submatrices
    A11, A12, A21, A22 = splitMatrix(A)
    B11, B12, B21, B22 = splitMatrix(B)
    # Compute seven products recursively
    P1 = strassenMatrixMultiply(A11, B12 - B22)
    P2 = strassenMatrixMultiply(A11 + A12, B22)
    P3 = strassenMatrixMultiply(A21 + A22, B11)
    P4 = strassenMatrixMultiply(A22, B21 - B11)
    P5 = strassenMatrixMultiply(A11 + A22, B11 + B22)
    P6 = strassenMatrixMultiply(A12 - A22, B21 + B22)
    P7 = strassenMatrixMultiply(A11 - A21, B11 + B12)
    # Calculate resulting submatrices
    C11 = P5 + P4 - P2 + P6
    C12 = P1 + P2
    C21 = P3 + P4
    C22 = P5 + P1 - P3 - P7
    # Assemble the final result matrix
    C = combineMatrices(C11, C12, C21, C22)
        return C
# Function to split a matrix into four equal-sized submatrices
function splitMatrix(M):
    n = size(M)
    m = n / 2
    A11 = M[1:m, 1:m]
    A12 = M[1:m, m+1:n]
    A21 = M[m+1:n, 1:m]
    A22 = M[m+1:n, m+1:n]
    return A11, A12, A21, A22
# Function to combine four submatrices into a single matrix
function combineMatrices(C11, C12, C21, C22):
    n = size(C11)
    m = 2 * n
    C = createEmptyMatrix(m, m)
    C[1:n, 1:n] = C11
    C[1:n, n+1:m] = C12
    C[n+1:m, 1:n] = C21
    C[n+1:m, n+1:m] = C22
    return C
```

Input:

```
#include<stdio.h>
int main(){
  int a[2][2], b[2][2], c[2][2], i, j;
  int m1, m2, m3, m4 , m5, m6, m7;
  printf("Boddu Asmitha Bhavya_A2305221386");
  printf("\nEnter the elements of first matrix: ");
  for(i = 0;i < 2; i++)
      for(j = 0;j < 2; j++)
           scanf("%d", &a[i][j]);
  printf("Enter the elements of second matrix: ");
  for(i = 0; i < 2; i++)
      for(j = 0;j < 2; j++)
           scanf("%d", &b[i][j]);
  printf("\nFirst matrix:\n");
  for(i = 0; i < 2; i++){
      printf("\n");
      for(j = 0; j < 2; j++)
          printf("%d\t", a[i][j]);
  }
  printf("\nSecond matrix:\n");
  for(i = 0;i < 2; i++){
      printf("\n");
      for(j = 0;j < 2; j++)
          printf("%d\t", b[i][j]);
  }
  m1= (a[0][0] + a[1][1]) * (b[0][0] + b[1][1]);
  m2= (a[1][0] + a[1][1]) * b[0][0];
  m3= a[0][0] * (b[0][1] - b[1][1]);
  m4= a[1][1] * (b[1][0] - b[0][0]);
  m5= (a[0][0] + a[0][1]) * b[1][1];
  m6= (a[1][0] - a[0][0]) * (b[0][0]+b[0][1]);
  m7= (a[0][1] - a[1][1]) * (b[1][0]+b[1][1]);
  c[0][0] = m1 + m4- m5 + m7;
  c[0][1] = m3 + m5;
  c[1][0] = m2 + m4;
  c[1][1] = m1 - m2 + m3 + m6;
   printf("\nAfter Strassen's MatriX Multiplication:\n");
   for(i = 0; i < 2 ; i++){
      printf("\n");
      for(j = 0;j < 2; j++)
          printf("%d\t", c[i][j]);
          }
   return 0;
}
```

Output:



```
Boddu Asmitha Bhavya_A2305221386
Enter the elements of first matrix: 3 6 2 7
Enter the elements of second matrix: 3 8 9 4

First matrix:

3       6
2       7
Second matrix:

3       8
9       4
After Strassen's MatriX Multiplication:

63      48
69      44
```

Time complexity: O(n^2.8)