DESIGN ANALYSIS AND ALGORITHM

PRACTICAL NO 3

027_Abhishek_Ojha

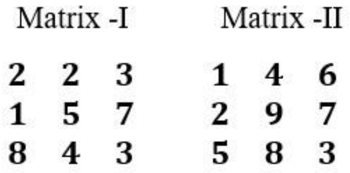
Experiment No – 3 **Date of Experiment : 14th September 2021**

Program : - Write a program on Strassen's algorithm for matrix multiplication and analyze its complexity

Example :-

Matrix Multiplication 3x3

Input :-

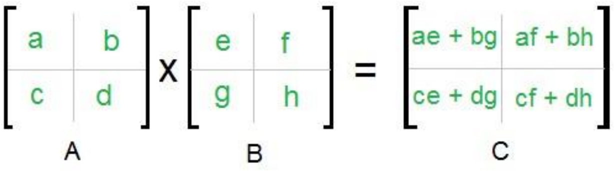


Matrix -I			Matrix -II		
2	2	3	1	4	6
1	5	7	2	9	7
8	4	3	5	8	3

Algorithm :-

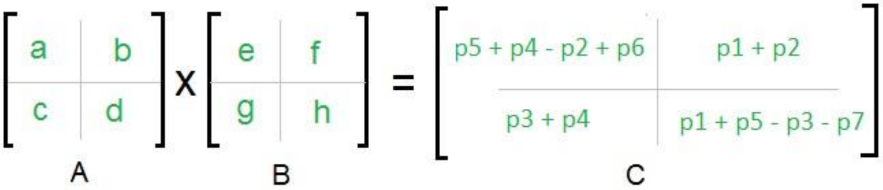
```
for i = 1 to p do
for j = 1 to r do
            Z[i,j] := 0
               for k = 1 to q do
      Z[i,j] := Z[i,j] + X[i,k] × Y[k,j]
```

Fig :-



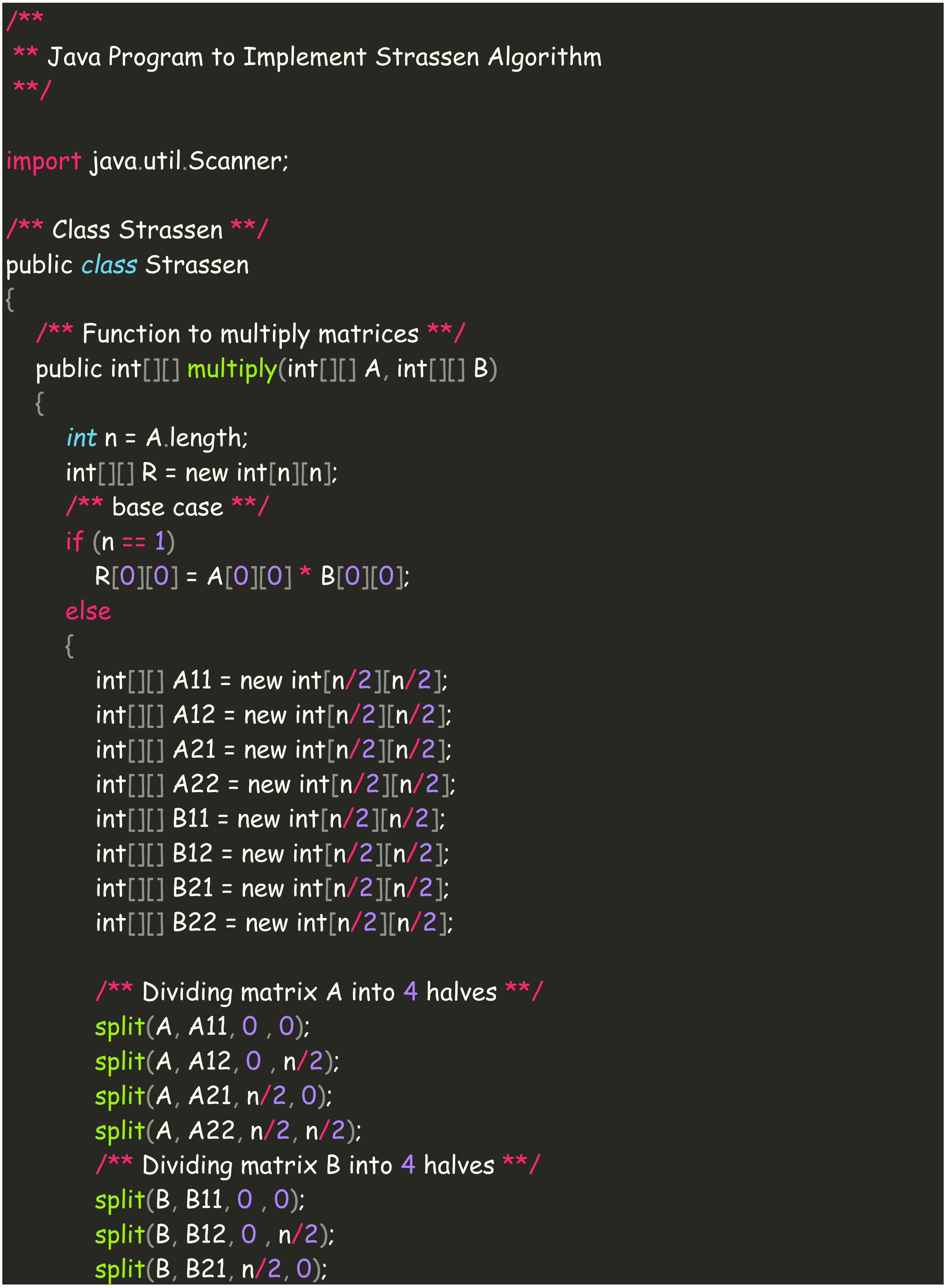
p1 = a(f - h)
p2 = (a + b)h
p3 = (c + d)e
p4 = d(g - e)
p5 = (a + d)(e + h)
p6 = (b - d)(g + h)
p7 = (a - c)(e + f)

The A x B can be calculated using above seven multiplications.
Following are values of four sub-matrices of result C

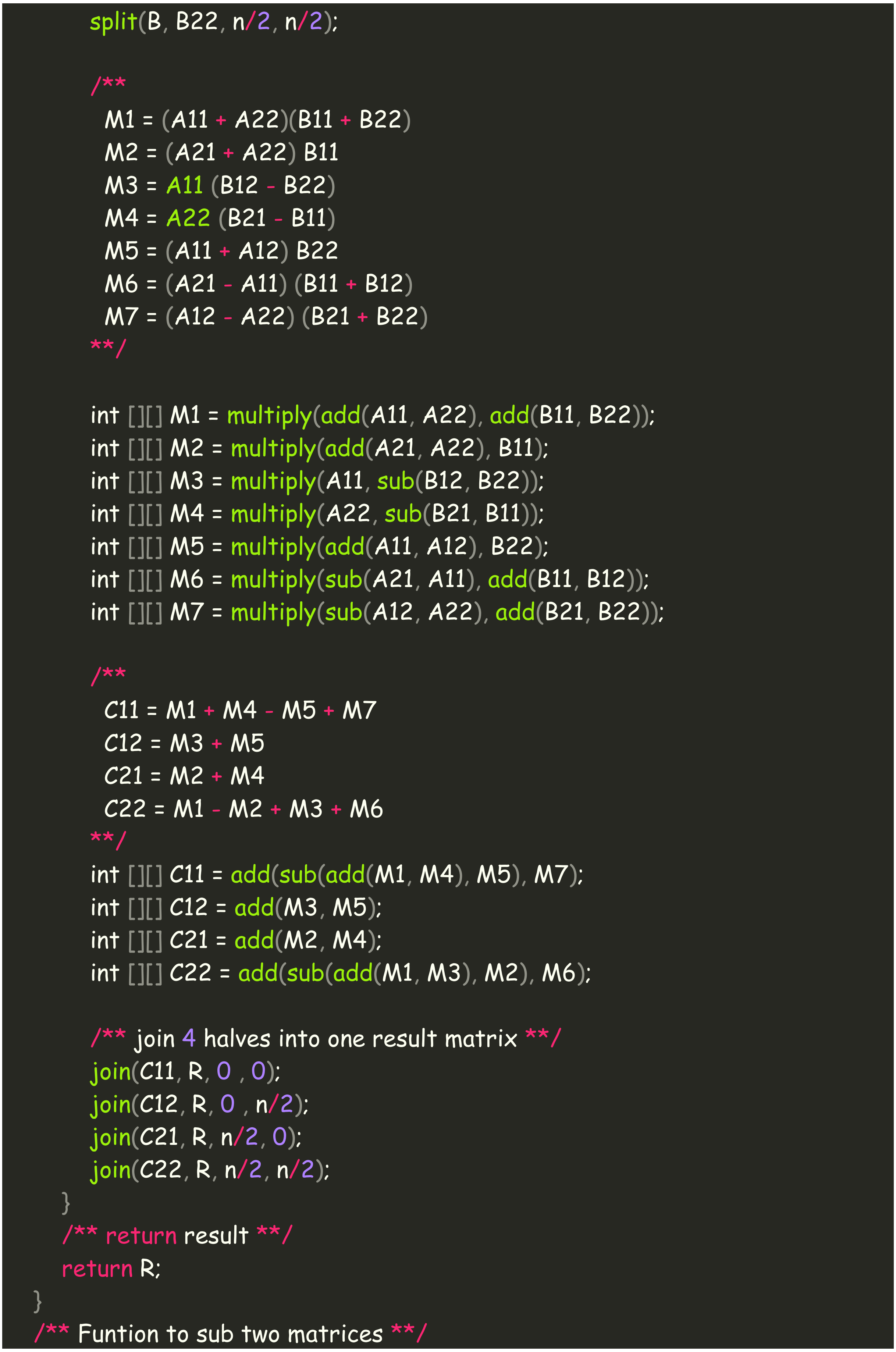


A, B and C are square metrices of size N x N
a, b, c and d are submatrices of A, of size N/2 x N/2
e, f, g and h are submatrices of B, of size N/2 x N/2
p1, p2, p3, p4, p5, p6 and p7 are submatrices of size N/2 x N/2

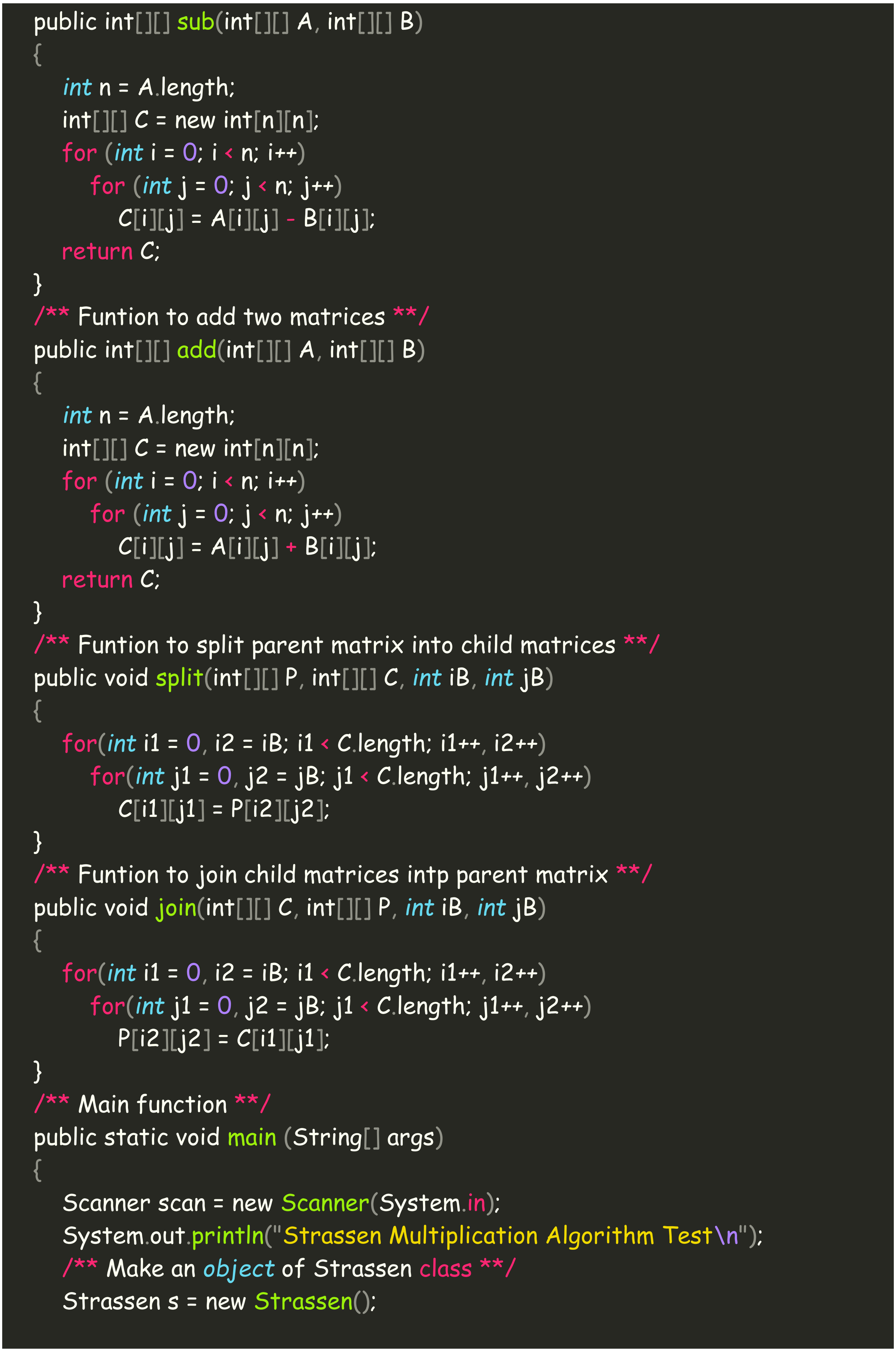
Program : -



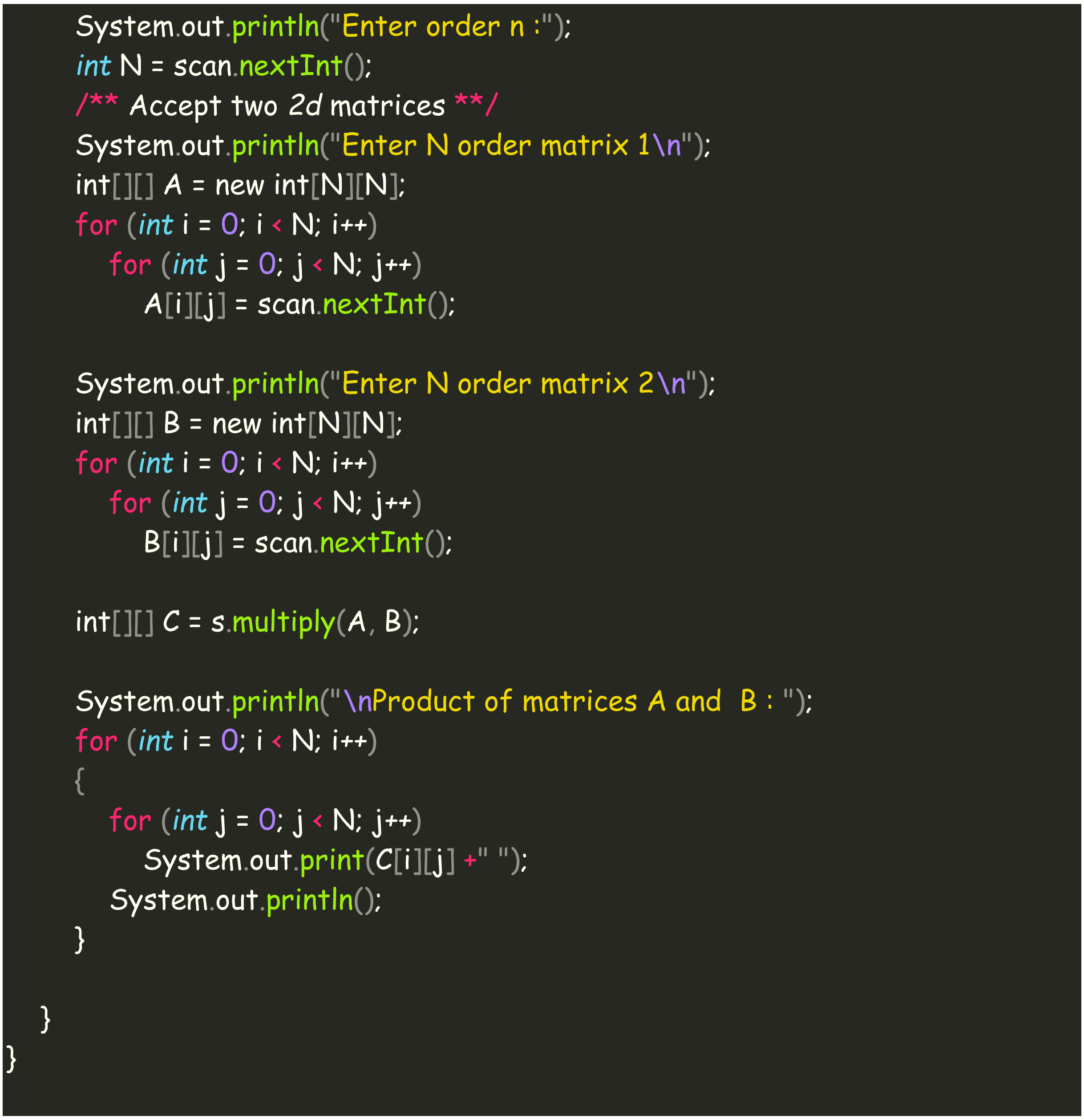
```
/**
 ** Java Program to Implement Strassen Algorithm
 **/

import java.util.Scanner;

/** Class Strassen **/
public class Strassen
{
    /** Function to multiply matrices **/
    public int[][] multiply(int[][] A, int[][] B)
    {
        int n = A.length;
        int[][] R = new int[n][n];
        /** base case **/
        if (n == 1)
            R[0][0] = A[0][0] * B[0][0];
        else
        {
            int[][] A11 = new int[n/2][n/2];
            int[][] A12 = new int[n/2][n/2];
            int[][] A21 = new int[n/2][n/2];
            int[][] A22 = new int[n/2][n/2];
            int[][] B11 = new int[n/2][n/2];
            int[][] B12 = new int[n/2][n/2];
            int[][] B21 = new int[n/2][n/2];
            int[][] B22 = new int[n/2][n/2];

            /** Dividing matrix A into 4 halves **/
            split(A, A11, 0 , 0);
            split(A, A12, 0 , n/2);
            split(A, A21, n/2, 0);
            split(A, A22, n/2, n/2);
            /** Dividing matrix B into 4 halves **/
            split(B, B11, 0 , 0);
            split(B, B12, 0 , n/2);
            split(B, B21, n/2, 0);
```



```
        split(B, B22, n/2, n/2);

        /**
          M1 = (A11 + A22)(B11 + B22)
          M2 = (A21 + A22) B11
          M3 = A11 (B12 - B22)
          M4 = A22 (B21 - B11)
          M5 = (A11 + A12) B22
          M6 = (A21 - A11) (B11 + B12)
          M7 = (A12 - A22) (B21 + B22)
        **/

        int [][] M1 = multiply(add(A11, A22), add(B11, B22));
        int [][] M2 = multiply(add(A21, A22), B11);
        int [][] M3 = multiply(A11, sub(B12, B22));
        int [][] M4 = multiply(A22, sub(B21, B11));
        int [][] M5 = multiply(add(A11, A12), B22);
        int [][] M6 = multiply(sub(A21, A11), add(B11, B12));
        int [][] M7 = multiply(sub(A12, A22), add(B21, B22));

        /**
          C11 = M1 + M4 - M5 + M7
          C12 = M3 + M5
          C21 = M2 + M4
          C22 = M1 - M2 + M3 + M6
        **/
        int [][] C11 = add(sub(add(M1, M4), M5), M7);
        int [][] C12 = add(M3, M5);
        int [][] C21 = add(M2, M4);
        int [][] C22 = add(sub(add(M1, M3), M2), M6);

        /** join 4 halves into one result matrix **/
        join(C11, R, 0 , 0);
        join(C12, R, 0 , n/2);
        join(C21, R, n/2, 0);
        join(C22, R, n/2, n/2);
    }
    /** return result **/
    return R;
}
/** Funtion to sub two matrices **/
```

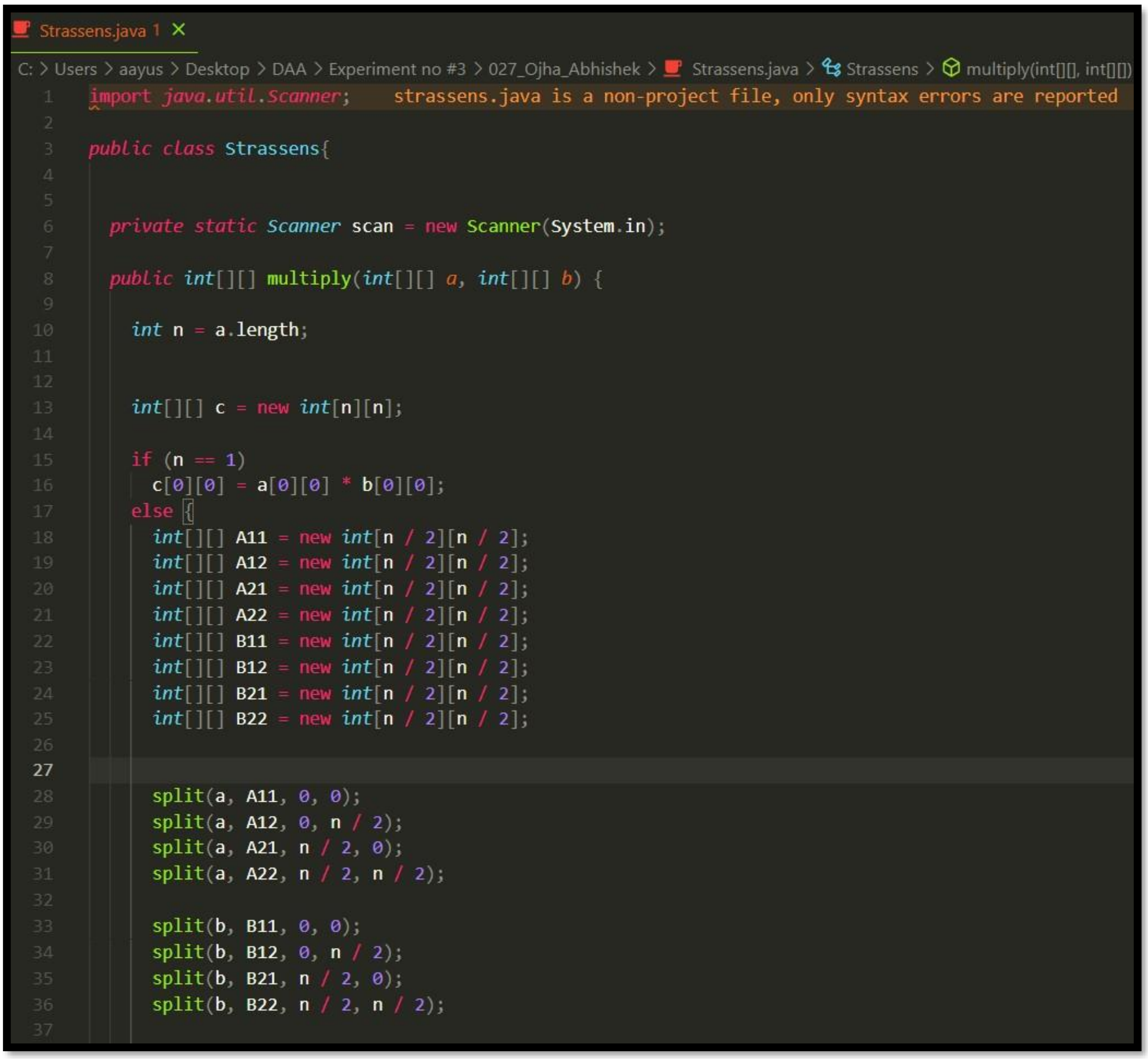


```
public int[][] sub(int[][] A, int[][] B)
{
    int n = A.length;
    int[][] C = new int[n][n];
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            C[i][j] = A[i][j] - B[i][j];
    return C;
}
/** Funtion to add two matrices **/
public int[][] add(int[][] A, int[][] B)
{
    int n = A.length;
    int[][] C = new int[n][n];
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            C[i][j] = A[i][j] + B[i][j];
    return C;
}
/** Funtion to split parent matrix into child matrices **/
public void split(int[][] P, int[][] C, int iB, int jB)
{
    for(int i1 = 0, i2 = iB; i1 < C.length; i1++, i2++)
        for(int j1 = 0, j2 = jB; j1 < C.length; j1++, j2++)
            C[i1][j1] = P[i2][j2];
}
/** Funtion to join child matrices intp parent matrix **/
public void join(int[][] C, int[][] P, int iB, int jB)
{
    for(int i1 = 0, i2 = iB; i1 < C.length; i1++, i2++)
        for(int j1 = 0, j2 = jB; j1 < C.length; j1++, j2++)
            P[i2][j2] = C[i1][j1];
}
/** Main function **/
public static void main (String[] args)
{
    Scanner scan = new Scanner(System.in);
    System.out.println("Strassen Multiplication Algorithm Test\n");
    /** Make an object of Strassen class **/
    Strassen s = new Strassen();
```



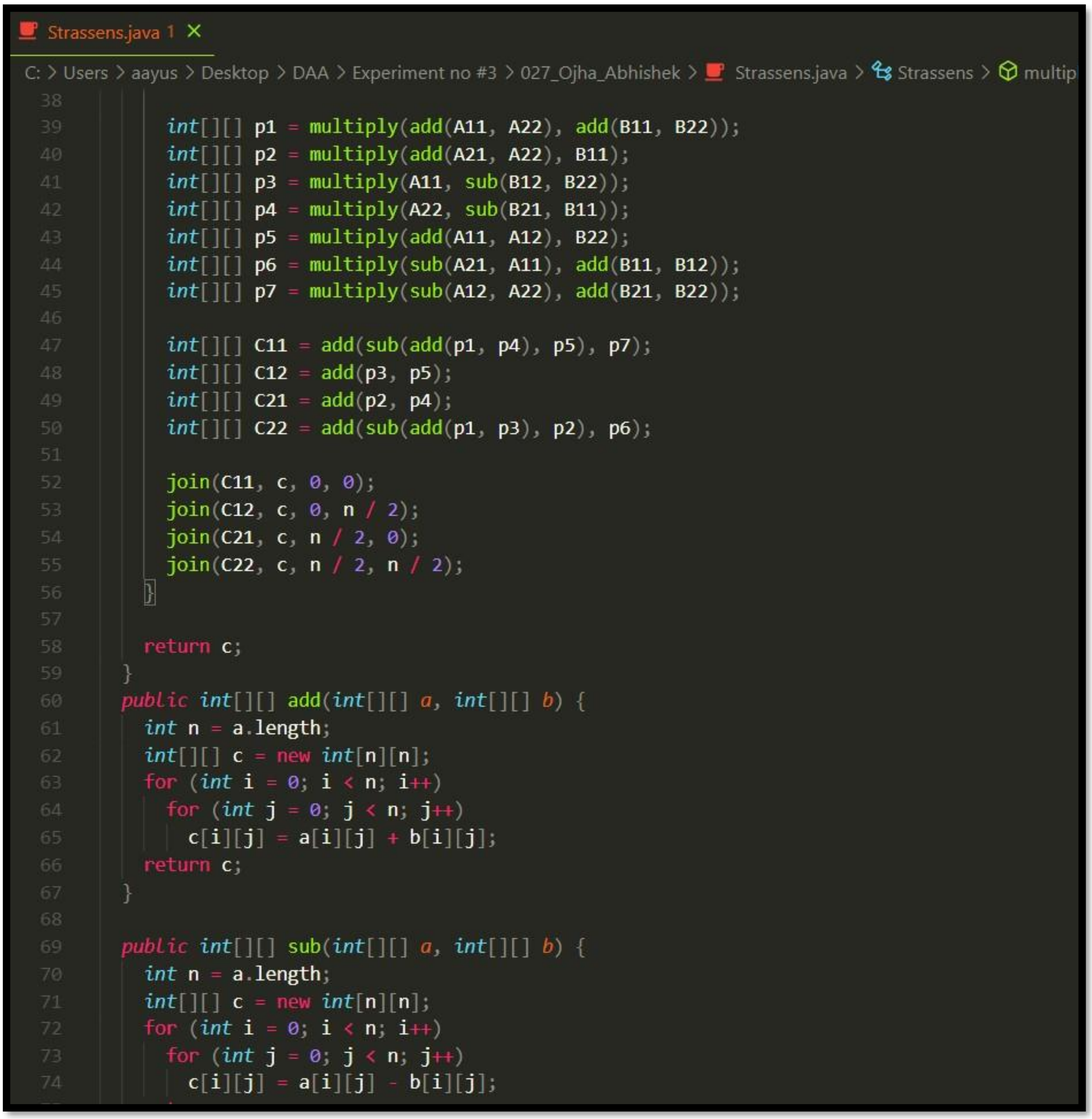
```
        System.out.println("Enter order n :");
        int N = scan.nextInt();
        /** Accept two 2d matrices **/
        System.out.println("Enter N order matrix 1\n");
        int[][] A = new int[N][N];
        for (int i = 0; i < N; i++)
            for (int j = 0; j < N; j++)
                A[i][j] = scan.nextInt();

        System.out.println("Enter N order matrix 2\n");
        int[][] B = new int[N][N];
        for (int i = 0; i < N; i++)
            for (int j = 0; j < N; j++)
                B[i][j] = scan.nextInt();

        int[][] C = s.multiply(A, B);

        System.out.println("\nProduct of matrices A and  B : ");
        for (int i = 0; i < N; i++)
        {
            for (int j = 0; j < N; j++)
                System.out.print(C[i][j] +" ");
            System.out.println();
        }

    }
}
```

Practical Implementation of Strassen's Algorithm :-

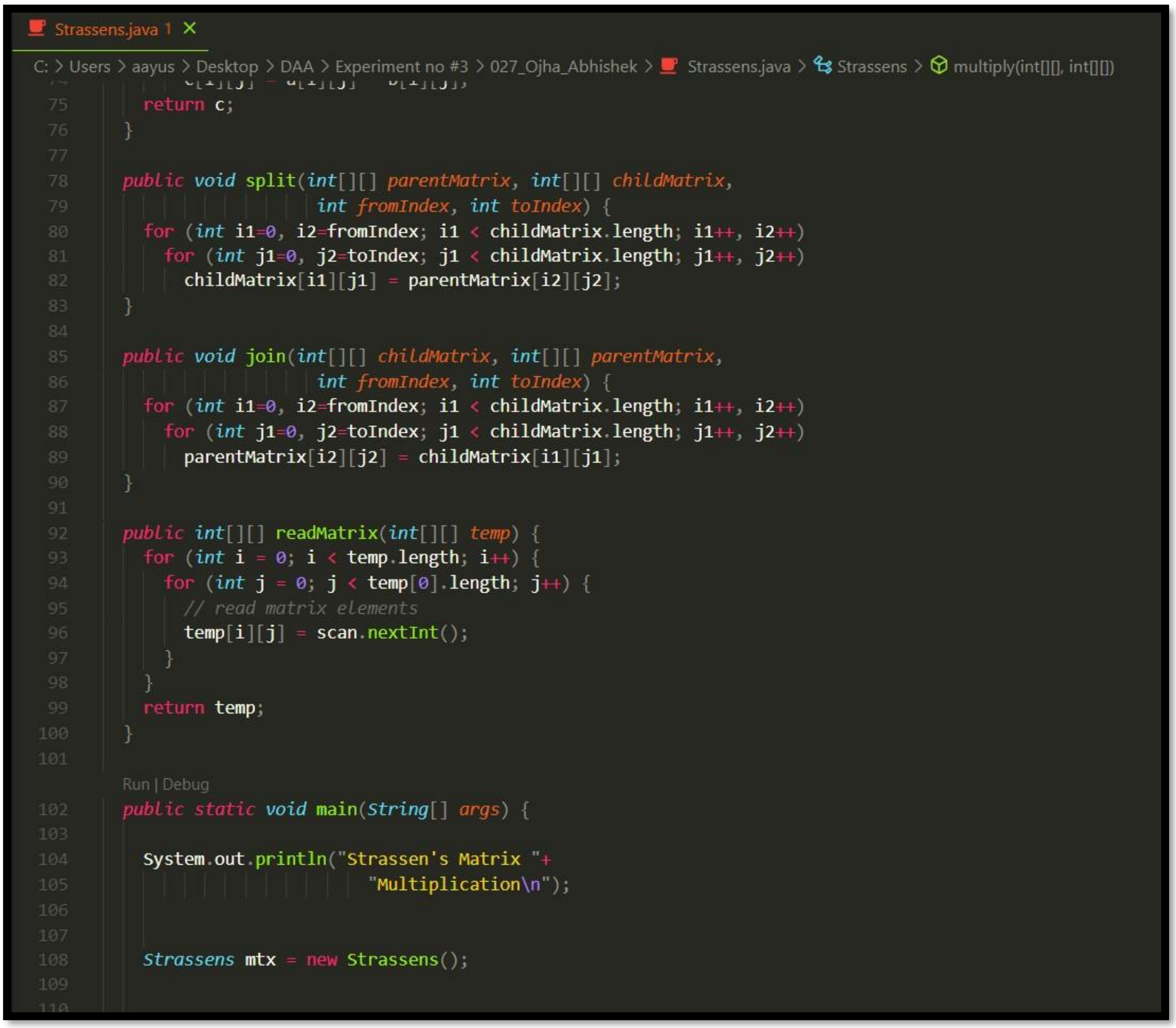

Strassens.java 1 ×
C: > Users > aayus > Desktop > DAA > Experiment no #3 > 027_Ojha_Abhishek > Strassens.java > Strassens > multiply(int[][], int[][])

strassens.java is a non-project file, only syntax errors are reported

```java
import java.util.Scanner;

public class Strassens{


  private static Scanner scan = new Scanner(System.in);

  public int[][] multiply(int[][] a, int[][] b) {

    int n = a.length;


    int[][] c = new int[n][n];

    if (n == 1)
      c[0][0] = a[0][0] * b[0][0];
    else {
      int[][] A11 = new int[n / 2][n / 2];
      int[][] A12 = new int[n / 2][n / 2];
      int[][] A21 = new int[n / 2][n / 2];
      int[][] A22 = new int[n / 2][n / 2];
      int[][] B11 = new int[n / 2][n / 2];
      int[][] B12 = new int[n / 2][n / 2];
      int[][] B21 = new int[n / 2][n / 2];
      int[][] B22 = new int[n / 2][n / 2];


      split(a, A11, 0, 0);
      split(a, A12, 0, n / 2);
      split(a, A21, n / 2, 0);
      split(a, A22, n / 2, n / 2);

      split(b, B11, 0, 0);
      split(b, B12, 0, n / 2);
      split(b, B21, n / 2, 0);
      split(b, B22, n / 2, n / 2);

```

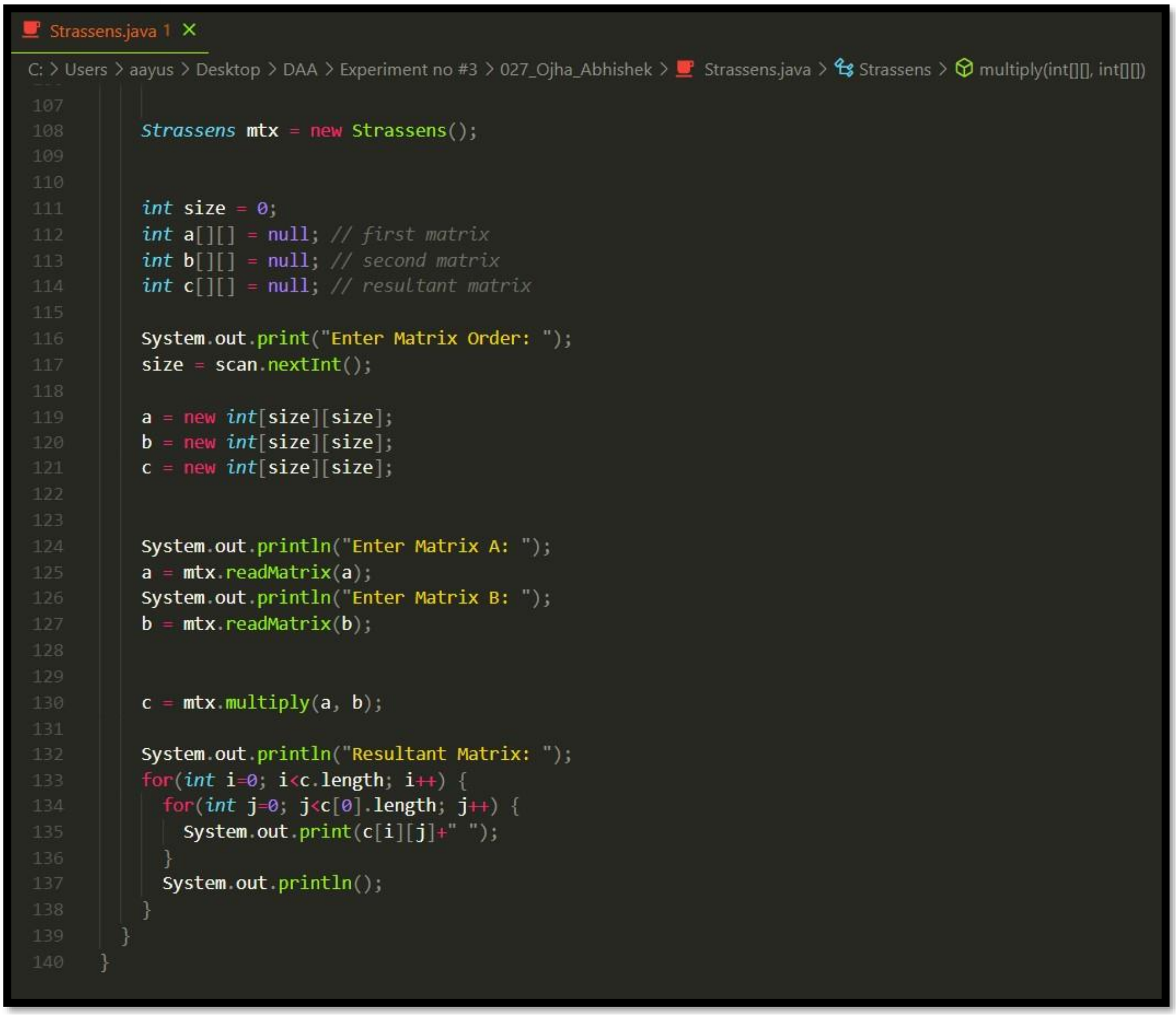

Strassens.java 1 ×
C: > Users > aayus > Desktop > DAA > Experiment no #3 > 027_Ojha_Abhishek > Strassens.java > Strassens > multip

```java

      int[][] p1 = multiply(add(A11, A22), add(B11, B22));
      int[][] p2 = multiply(add(A21, A22), B11);
      int[][] p3 = multiply(A11, sub(B12, B22));
      int[][] p4 = multiply(A22, sub(B21, B11));
      int[][] p5 = multiply(add(A11, A12), B22);
      int[][] p6 = multiply(sub(A21, A11), add(B11, B12));
      int[][] p7 = multiply(sub(A12, A22), add(B21, B22));

      int[][] C11 = add(sub(add(p1, p4), p5), p7);
      int[][] C12 = add(p3, p5);
      int[][] C21 = add(p2, p4);
      int[][] C22 = add(sub(add(p1, p3), p2), p6);

      join(C11, c, 0, 0);
      join(C12, c, 0, n / 2);
      join(C21, c, n / 2, 0);
      join(C22, c, n / 2, n / 2);
    }

    return c;
  }
  public int[][] add(int[][] a, int[][] b) {
    int n = a.length;
    int[][] c = new int[n][n];
    for (int i = 0; i < n; i++)
      for (int j = 0; j < n; j++)
        c[i][j] = a[i][j] + b[i][j];
    return c;
  }

  public int[][] sub(int[][] a, int[][] b) {
    int n = a.length;
    int[][] c = new int[n][n];
    for (int i = 0; i < n; i++)
      for (int j = 0; j < n; j++)
        c[i][j] = a[i][j] - b[i][j];
```



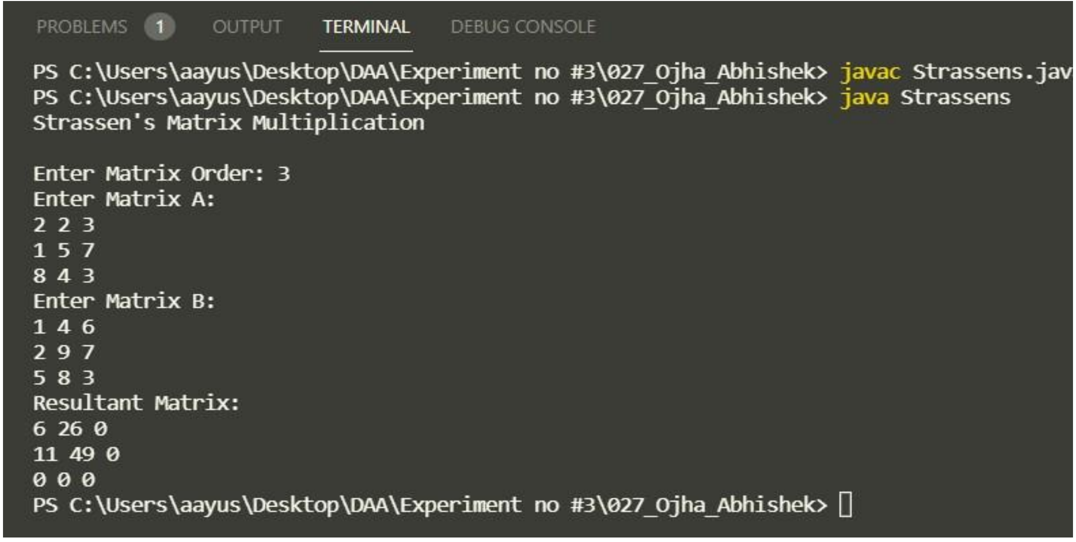
Strassens.java 1 ×
C: > Users > aayus > Desktop > DAA > Experiment no #3 > 027_Ojha_Abhishek > Strassens.java > Strassens > multiply(int[][], int[][])

```java
                c[i][j] = a[i][j] - b[i][j];
        return c;
    }

    public void split(int[][] parentMatrix, int[][] childMatrix,
                      int fromIndex, int toIndex) {
        for (int i1=0, i2=fromIndex; i1 < childMatrix.length; i1++, i2++)
            for (int j1=0, j2=toIndex; j1 < childMatrix.length; j1++, j2++)
                childMatrix[i1][j1] = parentMatrix[i2][j2];
    }

    public void join(int[][] childMatrix, int[][] parentMatrix,
                      int fromIndex, int toIndex) {
        for (int i1=0, i2=fromIndex; i1 < childMatrix.length; i1++, i2++)
            for (int j1=0, j2=toIndex; j1 < childMatrix.length; j1++, j2++)
                parentMatrix[i2][j2] = childMatrix[i1][j1];
    }

    public int[][] readMatrix(int[][] temp) {
        for (int i = 0; i < temp.length; i++) {
            for (int j = 0; j < temp[0].length; j++) {
                // read matrix elements
                temp[i][j] = scan.nextInt();
            }
        }
        return temp;
    }

    Run | Debug
    public static void main(String[] args) {

        System.out.println("Strassen's Matrix "+
                            "Multiplication\n");


        Strassens mtx = new Strassens();

```


Strassens.java 1 ×
C: > Users > aayus > Desktop > DAA > Experiment no #3 > 027_Ojha_Abhishek > Strassens.java > Strassens > multiply(int[][], int[][])

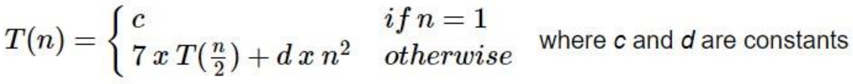
```java
    
    Strassens mtx = new Strassens();
    
    
    int size = 0;
    int a[][] = null; // first matrix
    int b[][] = null; // second matrix
    int c[][] = null; // resultant matrix
    
    System.out.print("Enter Matrix Order: ");
    size = scan.nextInt();
    
    a = new int[size][size];
    b = new int[size][size];
    c = new int[size][size];
    
    
    System.out.println("Enter Matrix A: ");
    a = mtx.readMatrix(a);
    System.out.println("Enter Matrix B: ");
    b = mtx.readMatrix(b);
    
    
    c = mtx.multiply(a, b);
    
    System.out.println("Resultant Matrix: ");
    for(int i=0; i<c.length; i++) {
      for(int j=0; j<c[0].length; j++) {
        System.out.print(c[i][j]+" ");
      }
      System.out.println();
    }
  }
}
```

Output

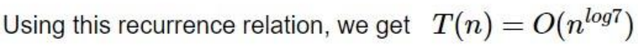


```
PS C:\Users\aayus\Desktop\DAA\Experiment no #3\027_Ojha_Abhishek> javac Strassens.jav
PS C:\Users\aayus\Desktop\DAA\Experiment no #3\027_Ojha_Abhishek> java Strassens
Strassen's Matrix Multiplication

Enter Matrix Order: 3
Enter Matrix A:
2 2 3
1 5 7
8 4 3
Enter Matrix B:
1 4 6
2 9 7
5 8 3
Resultant Matrix:
6 26 0
11 49 0
0 0 0
PS C:\Users\aayus\Desktop\DAA\Experiment no #3\027_Ojha_Abhishek> 
```

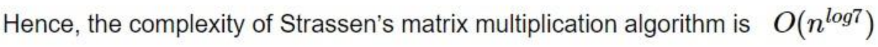
Analysis :



$$T(n) = \begin{cases} c & if\, n = 1 \\ 7\, x\, T(\frac{n}{2}) + d\, x\, n^2 & otherwise \end{cases}$$ where *c* and *d* are constants



Using this recurrence relation, we get $T(n) = O(n^{log7})$



Hence, the complexity of Strassen's matrix multiplication algorithm is $O(n^{log7})$

Conclusion :

integer operations take *O(1)* time. There are three for loops in this algorithm and one is nested in other. Hence, the algorithm takes *O(n^3)* time to execute.