**Batch : T7**

**Practical No. : 4**

**Title of Assignment : Divide and Conquer Strategy Strassen’s Matrix Multiplication**

**Student Name: Om Kulkarni**

**Student PRN: 22510034**

**Problem Statement:**

1. **Implement Naive method multiply two matrices and Justify complexity is o(n3)**
2. Algorithm/Pseudocode

**Input:**

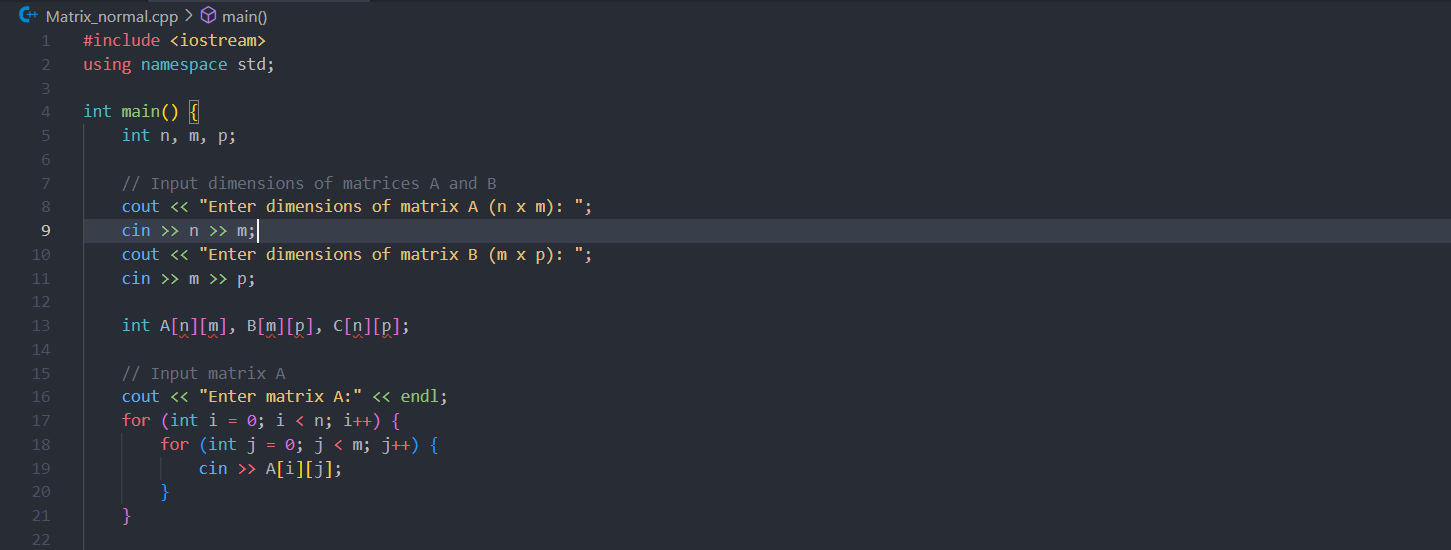
* Matrix A of size n×m
* Matrix B of size m×p

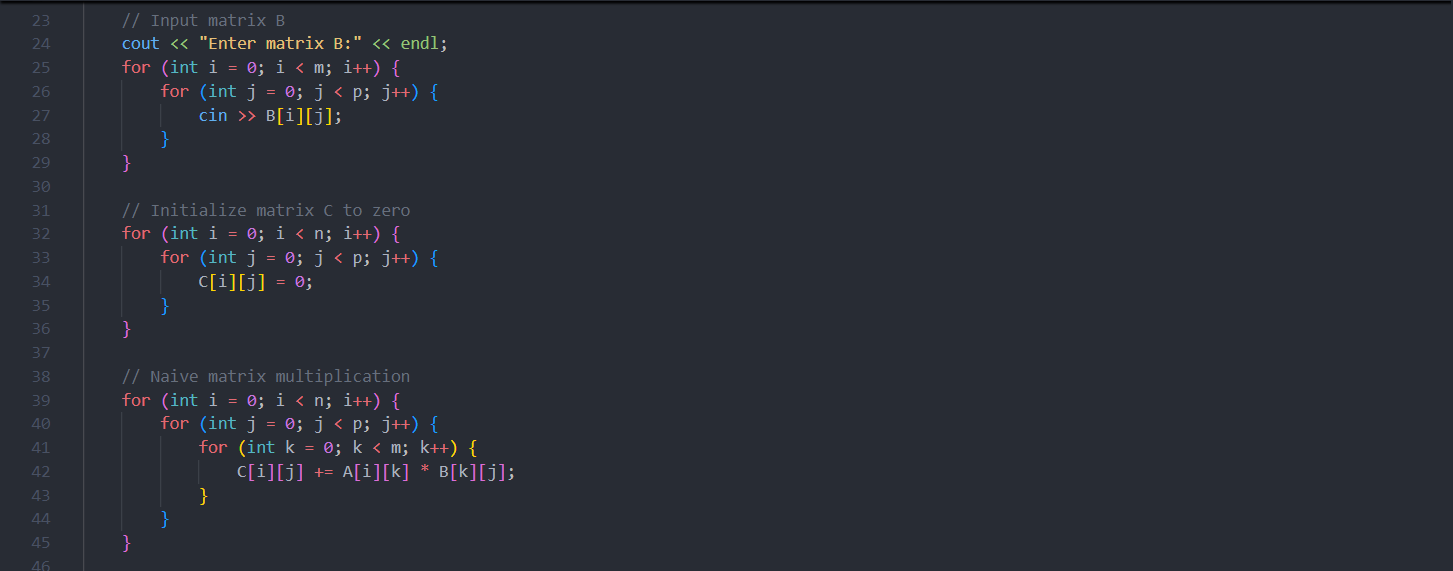
**Output:**

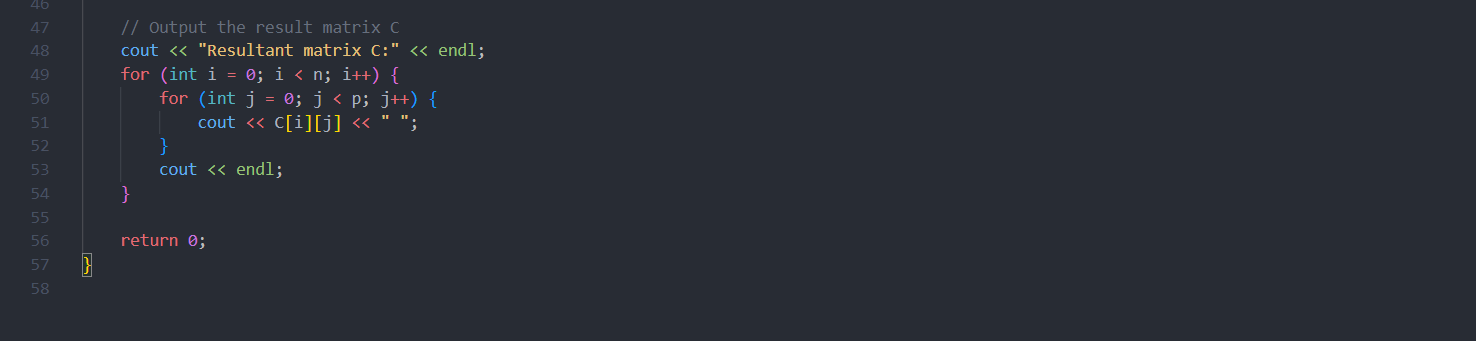
* Matrix C of size n×p where C[i][j] is the result of multiplying row iii of matrix AAA with column j of matrix B.

**Steps:**

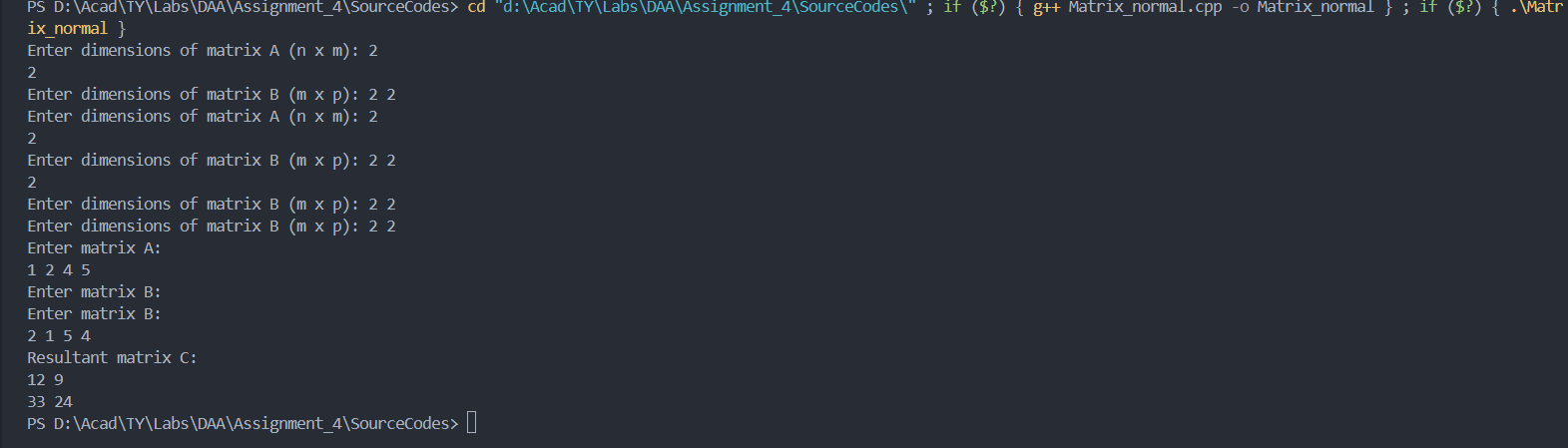
1. **Initialize the Result Matrix:**
   * Create a matrix C of size n×p.
   * Initialize all elements of C to 0.
2. **Iterate Over Rows of Matrix A:**
   * For each row index i from 0 to n−1:
     + Proceed to step 3.
3. **Iterate Over Columns of Matrix B:**
   * For each column index j from 0 to p−1:
     + Proceed to step 4.
4. **Compute the Dot Product:**
   * For each element index k from 0 to m−1:
     + Multiply the element A[i][k] with B[k][j] and add the result to C[i][j].
   * End of the dot product computation.
5. **Repeat for All Elements:**
   * Repeat steps 3 and 4 for all elements i and j of the matrix C.
6. **Return the Result Matrix C:**
   * The matrix C is now the product of matrices A and B.
7. Program Code







1. Output with verity of test cases



1. Analysis in terms of complexity wherever applicable.

**1. Time Complexity:**

* **Naive Matrix Multiplication:**
  + The algorithm involves three nested loops: one for the rows of matrix A, one for the columns of matrix B, and one for the dot product calculation.
  + **Time Complexity:** O(n×p×m), which simplifies to O(n^3) for square matrices (where n=m=p).

**2. Space Complexity:**

* **Space Required:**
  + Input matrices A and B require O(n×m) and O(m×p) space, respectively.
  + Output matrix C requires O(n×p) space.
  + **Space Complexity:** O(n×p) for the result matrix, or O(n^2) for square matrices.

**3. Auxiliary Space Complexity:**

* The auxiliary space complexity is O(1)as no extra space beyond the input and output matrices is used.

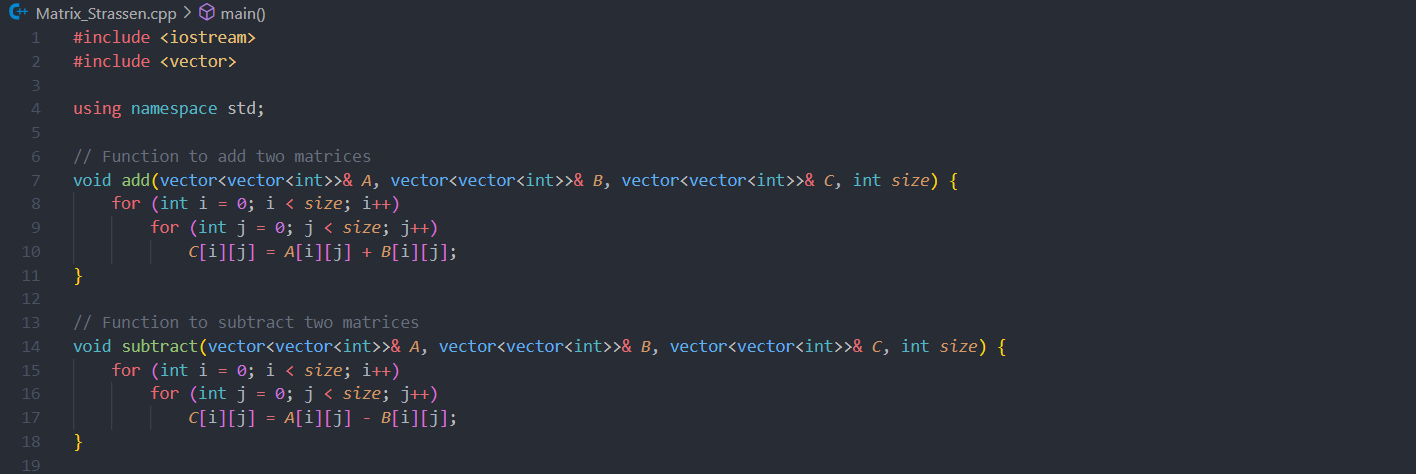
**Problem Statement:**

1. **Implement Strassen’s matrix multiplication for 3\*3 matrix.Do analysis of algorithm with respect to time complexity**

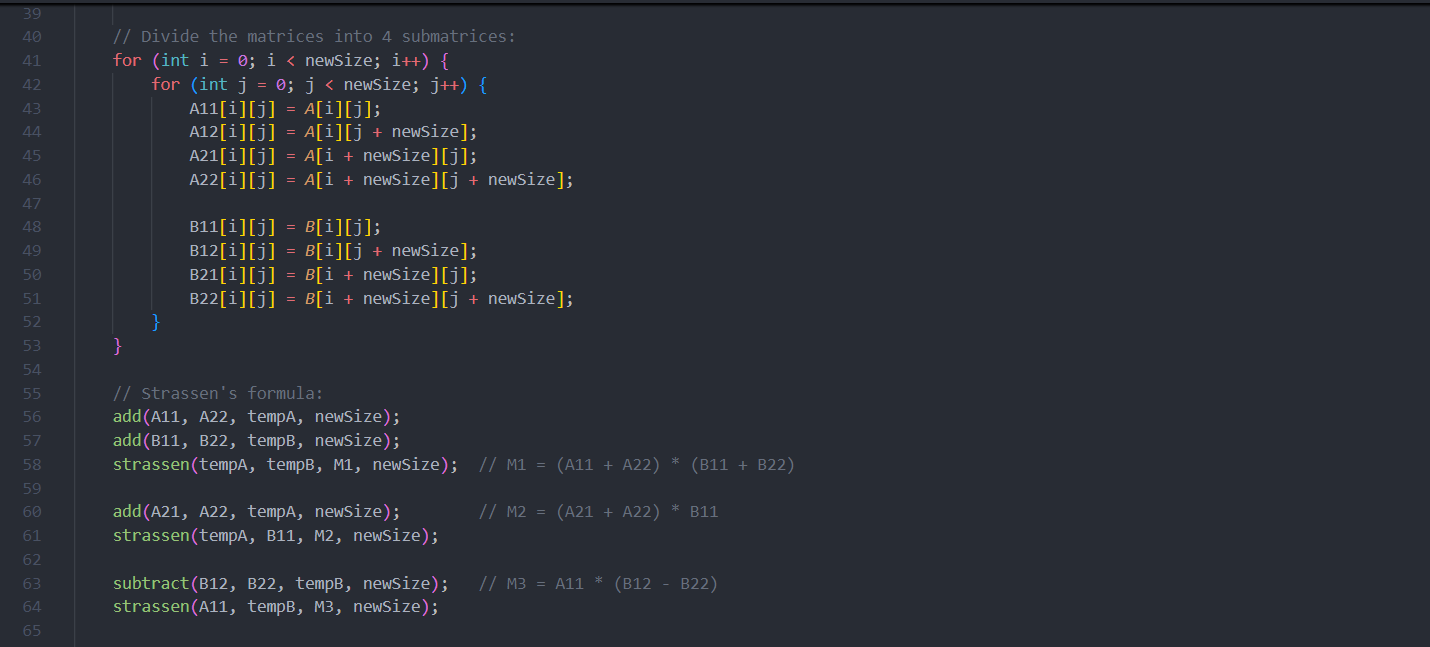
1 Algorithm/Pseudocode

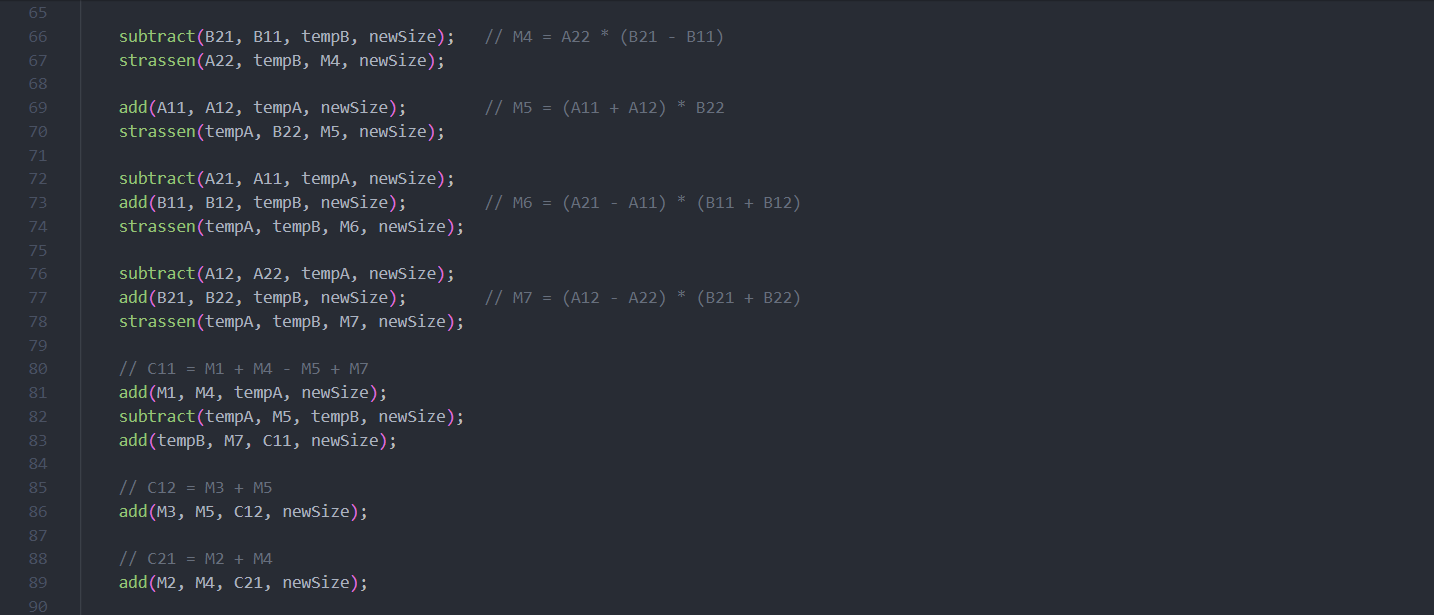
1. **Base Case:**
   * If n=1, compute the single element in the product matrix directly:
     + C[0][0]=A[0][0]×B[0][0]
   * Return C.
2. **Divide:**
   * Split matrices A and B into four submatrices each: A=[A11, A12, A21 ,A22],B=[B11, B12, B21, B22]
3. **Calculate the 7 Strassen Products:**
   * Compute the following intermediate matrices:
     + 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)
4. **Combine:**
   * Compute the submatrices of the result matrix C using the Strassen products: C11=M1+M4−M5+M7​=M1​+M4​−M5​+M7​ C12=M3+M5 ​ C21=M2+M4 C22=M1−M2+M3+M6 ​
   * Combine the submatrices C11, C12​, C21 and C22 ​ to form the full matrix C.
5. **Return:**
   * Return the matrix C as the result of the multiplication.

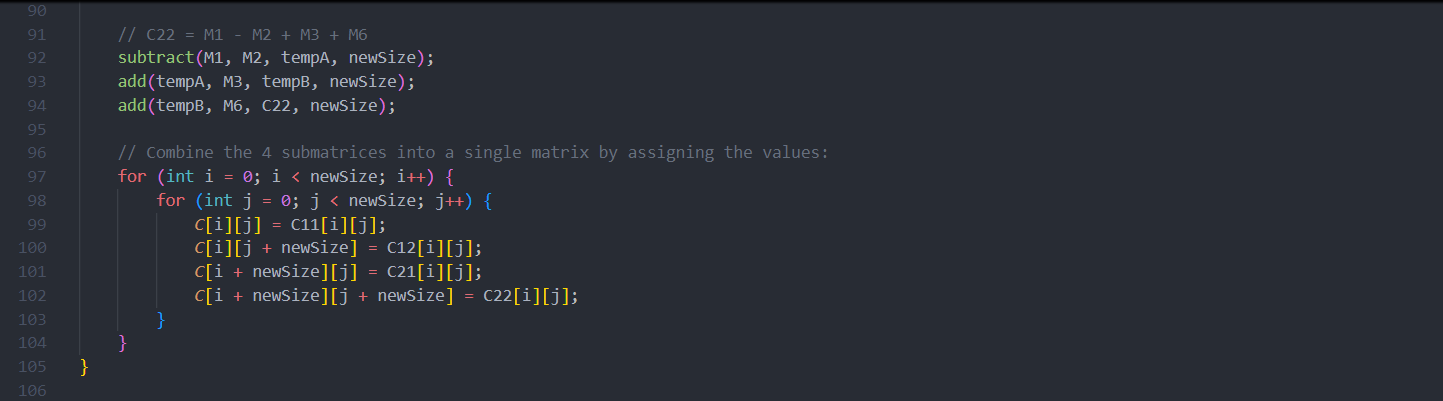
2 Program Code

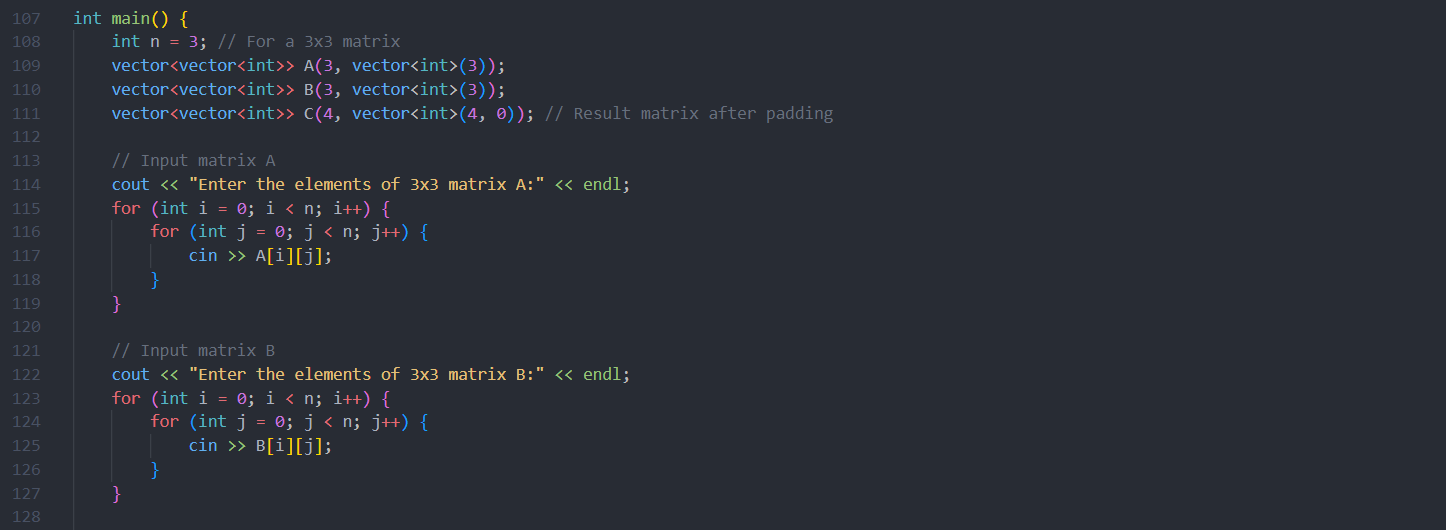


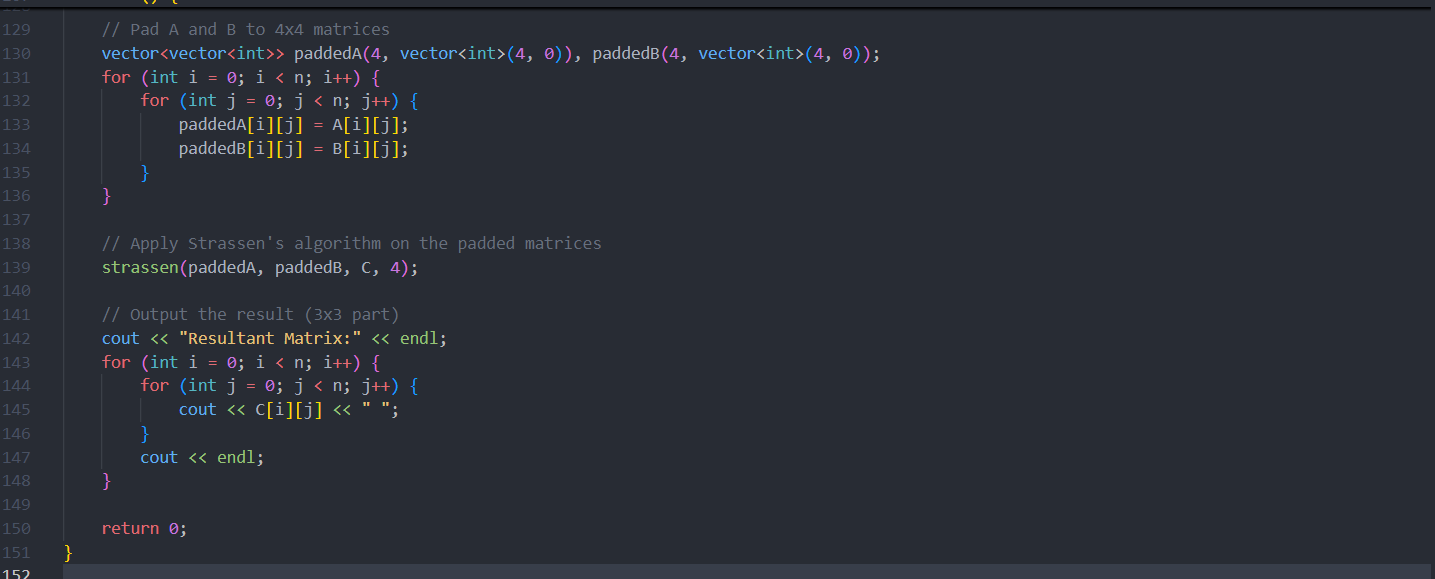




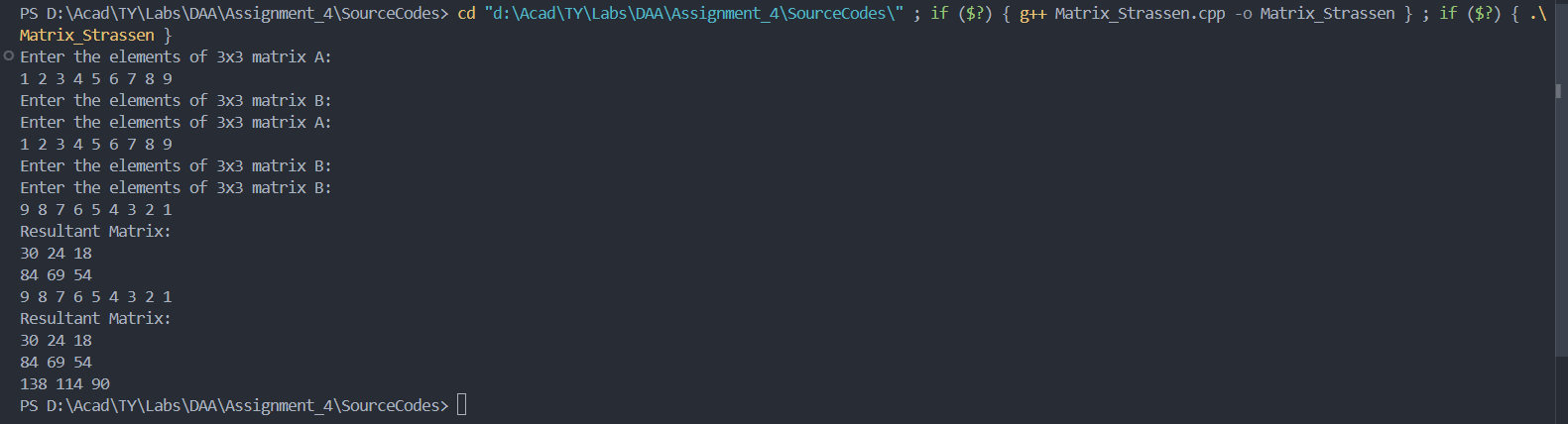








3 Output with verity of test cases



4 Analysis in terms of complexity wherever applicable.

* Recursive Calls**:** Each recursive step splits the matrix into four smaller matrices of size n/2×n/2and makes 7 recursive calls, leading to a complexity relation: T(n)=7T(n2)+O(n^2)
* Solving the Relation**:** Using the Master Theorem, the time complexity is approximately: T(n)=O(n^log27)≈O(n^2.81)
* **Comparison to Naive Multiplication:** The Strassen algorithm is more efficient than the naive O(n3) multiplication for large matrices, but it has higher overhead due to the additional matrix additions and subtractions. For small matrices, the naive approach may actually be faster due to this overhead.

Note:

1. Scan the document and **create a pdf** **file** with **“ExamSeatNum\_P#PS#” as its name**.
2. Create zip folder of your project/assignment which contains source code
3. Add the scanned document into the zip folder and name it as **ExamSeatNum\_P#PS#” as its name**
4. Upload the folder on the **WCE** **Moodle/ ERP** before the given deadline.

(P#: practical number

PS#: Problem statement Number)