**BÀI TẬP THỰC HÀNH 2**

***Mô tả bài toán***

Phép nhân ma trận là một phép tính cơ bản trong đại số tuyến tính. Giải thuật Strassen là một phương pháp hiệu quả để nhân hai ma trận vuông cùng bậc. Độ phức tạp thời gian của phương pháp Strassen là O(nlog27), nhanh hơn so với phương pháp thông thường O(n3).

***Dữ liệu đầu vào và dữ liệu đầu ra***

Dữ liệu đầu vào:

- A, B: hai ma trận vuông cùng bậc n

- n: kích thước của ma trận vuông

Dữ liệu đầu ra: C: ma trận tích của A và B

***Code Python***

def strassen(A, B):

n = len(A)

if n == 1:

return [[A[0][0] \* B[0][0]]]

# chia ma trận A và B thành các ma trận con bằng phép chia đôi

A11, A12, A21, A22 = split\_matrix(A)

B11, B12, B21, B22 = split\_matrix(B)

# tính các ma trận phụ trợ

P1 = strassen(A11 + A22, B11 + B22)

P2 = strassen(A21 + A22, B11)

P3 = strassen(A11, B12 - B22)

P4 = strassen(A22, B21 - B11)

P5 = strassen(A11 + A12, B22)

P6 = strassen(A21 - A11, B11 + B12)

P7 = strassen(A12 - A22, B21 + B22)

# tính ma trận tích C

C11 = P1 + P4 - P5 + P7

C12 = P3 + P5

C21 = P2 + P4

C22 = P1 - P2 + P3 + P6

# ghép các ma trận con để tạo ra ma trận C kết quả

C = [[0 for j in range(n)] for i in range(n)]

for i in range(n // 2):

for j in range(n // 2):

C[i][j] = C11[i][j]

C[i][j + n // 2] = C12[i][j]

C[i + n // 2][j] = C21[i][j]

C[i + n // 2][j + n // 2] = C22[i][j]

return C

def split\_matrix(M):

n = len(M)

if n % 2 == 0:

k = n // 2

A = [row[:k] for row in M[:k]]

B = [row[k:] for row in M[:k]]

C = [row[:k] for row in M[k:]]

D = [row[k:] for row in M[k:]]

else:

k = n // 2

A = [row[:k+1] for row in M[:k+1]]

B = [row[k+1:] for row in M[:k