**알고리즘 HW1**

최지웅 교수님

Strassen’s Matrix Multiplication Algorithm

/ 2장 Divide-and-conquer

**20150439 소혜빈**

**목차**

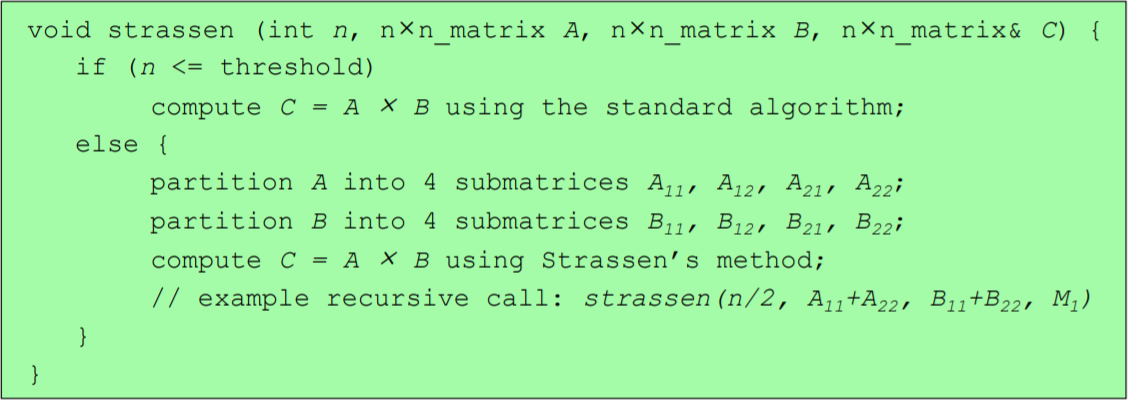
* Problem
* Class
* 재귀 호출 부
* Partition
* Merge
* 결과 확인
* 손 풀이

**Problem :** n 이 2의 거듭제곱일 때 2개의 n x n 행렬의 곱을 구하라.

**Input :** 2의 거듭제곱인 정수 n, 2개의 n x n 행렬 A와 B

**Output :** A와 B의 곱 C

**Algorithm :**



*출처 : 숭실대 스마트캠퍼스 알고리즘 강의자료*

Threshold 값은 1로 한다.

**구현 언어 :** JAVA

**Sample value 1 :**

n = 4,

A = [1,2,3,4] , B = [8,9,1,2]

[5,6,7,8] [3,4,5,6]

[9,1,2,3] [7,8,9,1]

[4,5,6,7] [2,3,4,5]

*출처 : Ex 2.5 / Foundation of Algorithms using C++ Pseudocode 3th*

**Sample value 2 :**

n = 4,

A = [0, 3,2,8] , B = [1,1,0,11]

[1, 0,0,4] [2, 0,0,2]

[5,30,5,5] [1, 2,1,2]

[1, 2,3,4] [0,28,0,7]

Class

public class Matrix

**Fields :**

private int[] matrix;  
private int n; //n by n matrix

**Constructors :**

public Matrix(int n)

public Matrix(int n, int[] matrix)

**Methods :**

public int size()  
public int dimension()

public void print()

public Matrix minus()

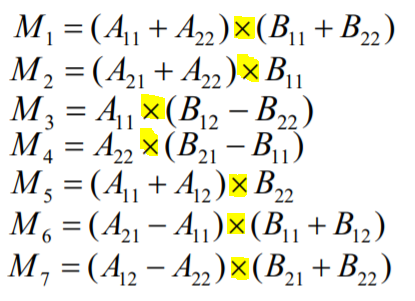
public Matrix sum(Matrix A)

public Matrix Strassen(Matrix B)

private Matrix partition(int i, int j)

private Matrix merge(int n, Matrix C11, Matrix C12, Matrix C21, Matrix C22)

재귀호출부



public Matrix Strassen(Matrix B)

**{**

임계 값 검사

partition

Matrix m1 = a11.sum(a22).Strassen(b11.sum(b22));  
Matrix m2 = a21.sum(a22).Strassen(b11);  
Matrix m3 = a11.Strassen(b12.sum(b22.minus()));  
Matrix m4 = a22.Strassen(b21.sum(b11.minus()));  
Matrix m5 = a11.sum(a12).Strassen(b22);  
Matrix m6 = a21.sum(a11.minus()).Strassen(b11.sum(b12));  
Matrix m7 = a12.sum(a22.minus()).Strassen(b21.sum(b22));

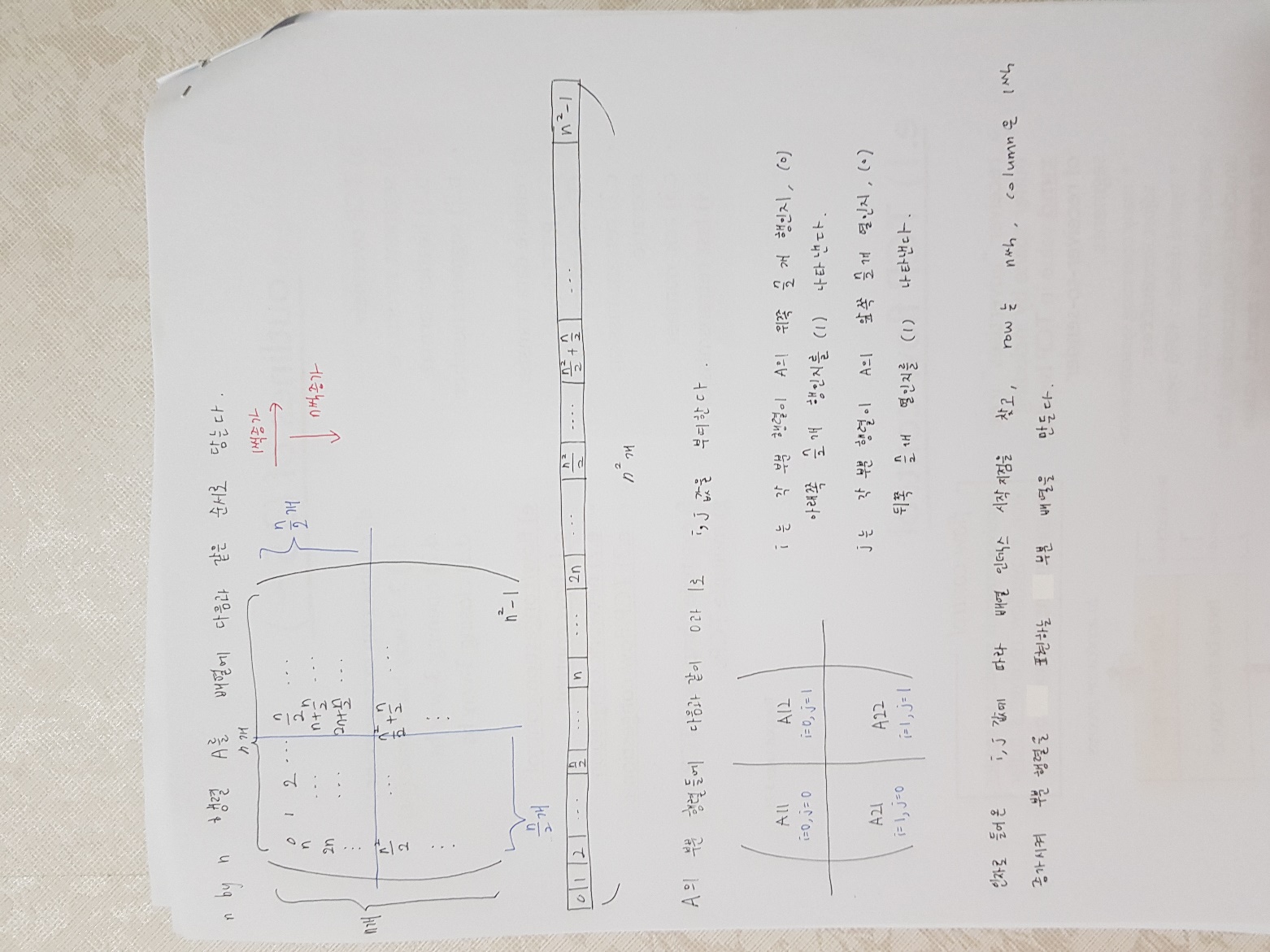
merge

결과 return

**}**

Partition

꽤 어려웠던 부분은 배열로 표현된 행렬을 4개의 부분행렬로 인덱스를 나누는 것이었다.



private Matrix partition(int i, int j)  
{  
 int n = this.dimension()/2;  
 Matrix result = new Matrix(n);  
  
 int index = 0;  
 for(int row = i\*this.size()/2 ; row < i\*this.size()/2+this.size()/2; row+=this.dimension())  
 {  
 for(int col = row+j\*n ; col < row+j\*n+n ; ++col)  
 result.matrix[index++] = this.matrix[col];  
 }  
 return result;  
}

//dimension() { return n\*n; }

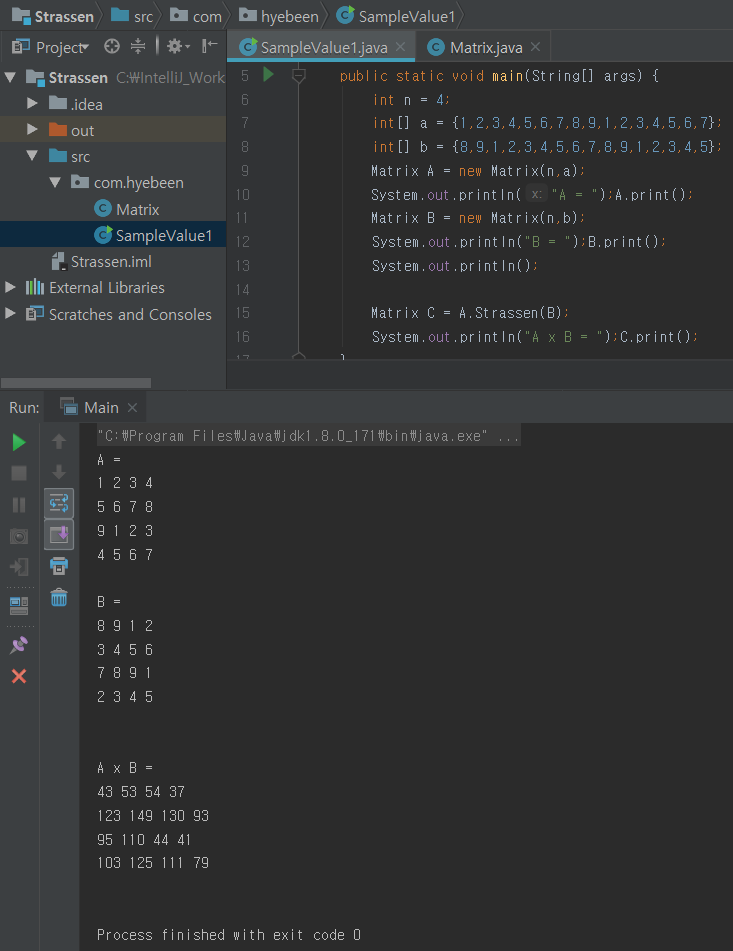
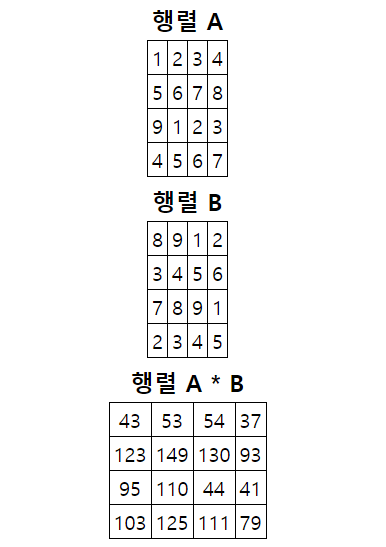
Merge

부분행렬로 나눌 때와 마찬가지로 나뉘어진 행렬을 합칠 때도 다음 조건에 맞게 인덱스를 잘 조절해서 합쳐야 한다.

if(i<C.size()/2 && i%n < n/2) C.matrix[i] = C11.matrix[i11++];  
 else if(i<C.size()/2 && i%n >= n/2) C.matrix[i] = C12.matrix[i12++];  
 else if(i>=C.size()/2 && i%n < n/2) C.matrix[i] = C21.matrix[i21++];  
 else C.matrix[i] = C22.matrix[i22++];

결과 확인

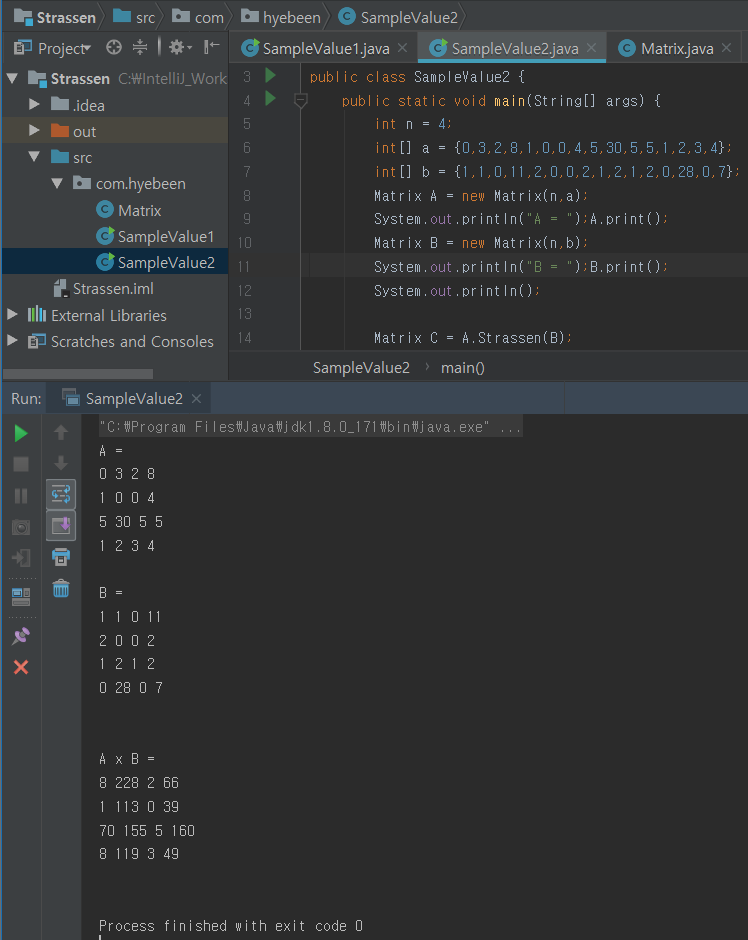
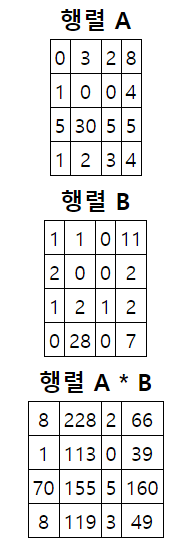
**Sample Value 1**



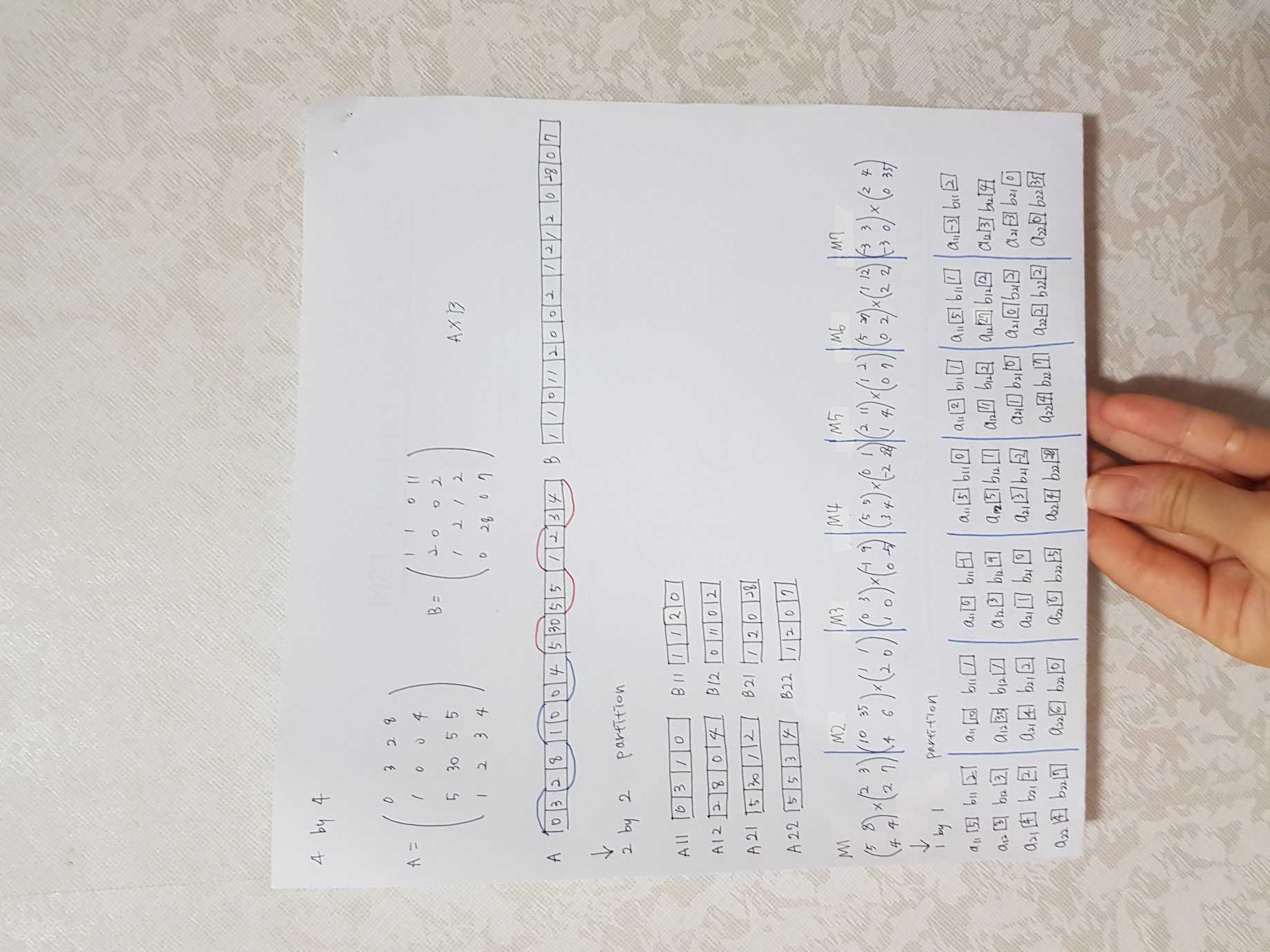
*출처 : https://ko.numberempire.com/matrixbinarycalculator.php*

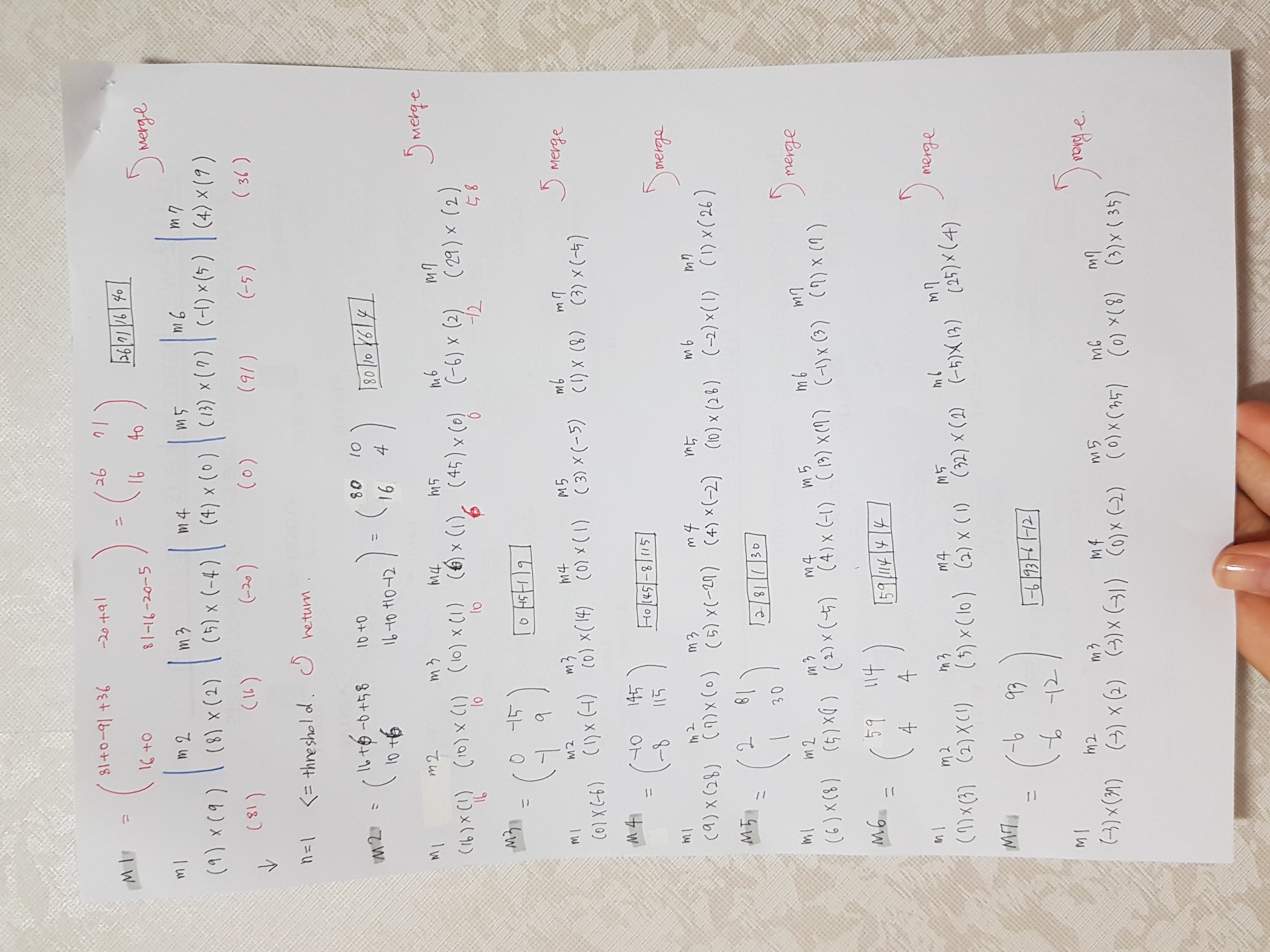
결과 확인

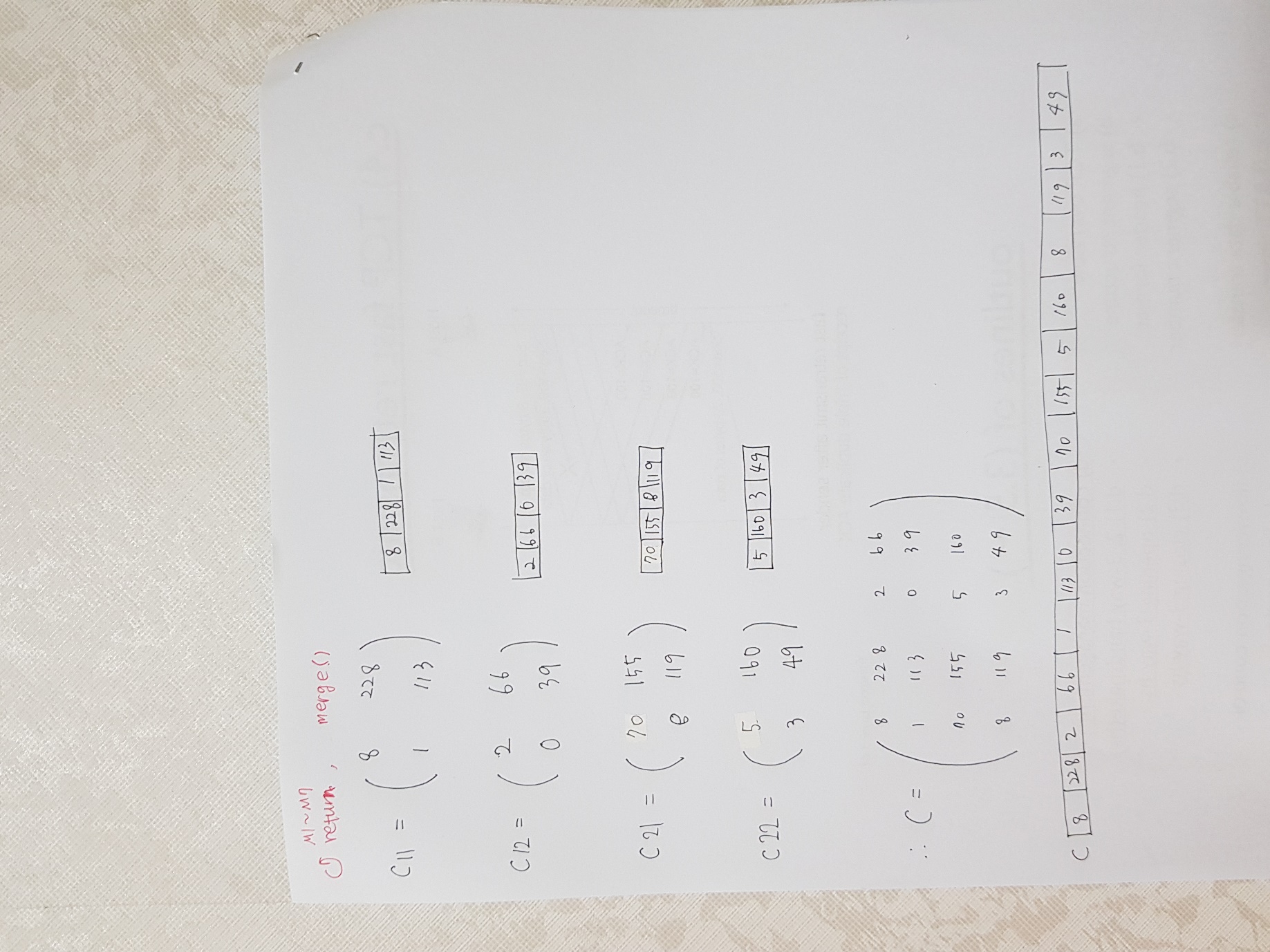
**Sample Value 2**

**Sample Value 2 손 풀이**







결과는 같다.