Matrix 1									Matrix 2							
4	5	7	6	2	3	8	1		6	1	5	4	7	2	3	8
1	2	6	7	5	4	8	3		4	2	7	8	1	5	3	6
4	2	6	1	5	3	8	7		3	1	7	5	2	6	8	4
5	4	8	2	3	1	7	6		3	8	7	1	5	6	4	2
1	2	3	5	7	6	8	4		4	1	6	8	7	5	3	2
2	8	5	1	4	6	3	7		3	1	8	4	2	6	5	7
8	2	6	3	4	5	1	7		8	1	4	5	3	6	7	2
2	6	7	8	1	3	5	4		4	1	8	5	6	3	2	7

- Naïve Method:

```
void multiply(int A[][N], int B[][N], int C[][N])
{
    for (int i = 0; i < N; i++)            -->N=8
    {
        for (int j = 0; j < N; j++)        -->N=8
        {
            C[i][j] = 0;
            for (int k = 0; k < N; k++)    -->N=8
            {
                C[i][j] += A[i][k]*B[k][j];
            }
        }
    }
}
```

T(n) = O(n^3) = 8*8*8= 512;

- Divide and Conquer:

Consider Matrix 1 is X and Matrix 2 is Y

Divide X and Y into 8 sub-matrices A, B, C, and D.

X			Y	
A	B		E	F
C	D		G	H

Do 8 matrix multiplications recursively.

Compute Z by combining results (doing 4 matrix additions).

Lets assume $n = 2^c$ for some constant c and let A, B, C and D be $n/2 \times n/2$ matrices

$T(N) = 8T(N/2) + O(N^2) = 8*(8*(8*(8*(..1..) +(N/8)^2) +(N/4)^2) +(N/2)^2) +O(N^2)$

$a = 8; b = 2; d = 2;$ $\Rightarrow a > b^d$

$\Rightarrow T(N) = O(N^{\log_2 8}) = O(N^3)$

- Strassen's Method

Z	
(S1 + S2 − S4 + S6)	(S4 + S5)
(S6 + S7)	(S2 + S3 + S5 − S7)

$S1 = (B - D) *(G + H)$

$S2 = (A + D) *(E + H)$

$S3 = (A - C) *(E + F)$

$S4 = (A + B) *H$

$S5 = A *(F - H)$

$S6 = D *(G - E)$

$S7 = (C + D) *E$

$T(n)=7T(n/2) + O(n^2)$

$\Rightarrow T(n) = 7T(n/2) + n^2$

$= n^2 + 7(7T(n/2^2)+(n/2)^2)$

$= n^2 + (7/2^2)n^2 + 7^2T(n/2^2)$

$=n^2 + (7/2^2)n^2 + (7/2^2)^2n^2 + 7^3T(n/2^3)$

$=n^2 + (7/2^2)n^2 + (7/2^2)^2n^2 + (7/2^2)^3n^2 + ... + (7/2^2)^{\log(n-1)}n^2 + 7^{\log n}$

$=n^2*O((7/2^2)^{\log(n-1)}) + 7^{\log n}$

$=n^2*O(7^{\log n}/(2^2)^{\log(n-1)}) + 7^{\log n}$

$=n^2*O(7^{\log n}/n^2) + 7^{\log n}$

$=O(7^{\log n})$

$\Rightarrow T(n) = = O(n^{\log 7}) = O(n^{2.8074})$