NAME:	Om Doshi
UID:	2021300030
SUBJECT	Design and Analysis of Algorithm
EXPERIMENT NO :	03
DATE OF PERFORMANCE	27/02/2023
DATE OF SUBMISSION	6/03/2023
AIM:	To multiply two matrices using strassen's matrix multiplication.
PROBLEM STATEMENT 1:	**Strassen's matrix multiplication on a generalized 2x2 matix.**
ALGORITHM and THEORY:	In this context, using Strassen's Matrix multiplication algorithm, the time consumption can be improved a little bit. Strassen's Matrix multiplication can be performed only on square matrices where n is a power of 2. Order of both of the matrices are n × n. Divide X, Y and Z into four $(n/2) \times (n/2)$ matrices as represented below − Z=[IKJL]�=[����] X=[ACBD]�=[����] and Y=[EGFH]�=[����] Using Strassen's Algorithm compute the following − $M_2 := (B+D) \times (G+H)$ �2:=(�+�)×(�+�) $M_3 := (A-D) \times (E+H)$ �3:=(�−�)×(�+�)

	$M_4 := A \times (F-H)$ $M_5 := (C+D) \times (E)$ $M_6 := (A+B) \times (H)$ $M_7 := D \times (G-E)$

Then,

$$I := M_2 + M_3 - M_6 - M_7$$
$$J := M_4 + M_6$$
$$K := M_5 + M_7$$
$$L := M_1 - M_3 - M_4 - M_5$$

Analysis

$T(n) = \begin{cases} c & 7 x T(n2) + d x n2 \end{cases}$ if $n=1$ otherwise where c and d are constants

Using this recurrence relation, we get $T(n) = O(n^{\log 7})$.

Hence, the complexity of Strassen's matrix multiplication algorithm is $O(n^{\log 7})$.

| **PROGRAM:** | ```c
#include<stdio.h>
#include<time.h>
void main()
{
    int a[2][2],b[2][2],c[2][2],i,j;
    int p[7];
    int s[10];
    clock_t start,end;
    printf("Enter the elements of 1st matrix:");
    for(i=0;i<2;i++)
    {
        for(j=0;j<2;j++)
        {
            scanf("%d",&a[i][j]);
        }
    }
    printf("Enter the elements of 2nd matrix:");
``` |

```c
for(i=0;i<2;i++)
{
    for(j=0;j<2;j++)
    {
        scanf("%d",&b[i][j]);
    }
}
start=clock();
s[0]=b[0][1]-b[1][1];
s[1]=a[0][0]+a[0][1];
s[2]=a[1][0]+a[1][1];
s[3]=b[1][0]-b[0][0];
s[4]=a[0][0]+a[1][1];
s[5]=b[0][0]+b[1][1];
s[6]=a[0][1]-a[1][1];
s[7]=b[1][0]+b[1][1];
s[8]=a[0][0]-a[1][0];
s[9]=b[0][0]+b[0][1];

p[0]=s[0]*a[0][0];
p[1]=s[1]*b[1][1];
p[2]=s[2]*b[0][0];
p[3]=s[3]*a[1][1];
p[4]=s[4]*s[5];
p[5]=s[6]*s[7];
p[6]=s[8]*s[9];

c[0][0]=p[4]+p[3]-p[1]+p[5];
c[0][1]=p[0]+p[1];
c[1][0]=p[2]+p[3];
c[1][1]=p[4]+p[0]-p[2]-p[6];

for(i=0;i<10;i++)
{
```

```c
        printf("\nS%d=%d ",i+1,s[i]);
    }
    printf("\n");
    for(j=0;j<7;j++)
    {
        printf("\np%d=%d ",j+1,p[j]);
    }
    printf("\n\n");
    printf("MATRIX A:-\n");
    for(i=0;i<2;i++)
    {
        printf("\n");
        for(j=0;j<2;j++)
        {
            printf("%d\t",a[i][j]);
        }
    }
    printf("\n");
    printf("MATRIX B:-\n");
    for(i=0;i<2;i++)
    {
        printf("\n");
        for(j=0;j<2;j++)
        {
            printf("%d\t",b[i][j]);
        }
    }
    printf("\n");
    printf("MATRIX C:-\n\n");
    printf("%d\t%d\n%d\t%d\n",c[0][0],c[0][1],c[1][0],c[1][1]);
    end=clock();
    printf("The time taken by the program: ");
    printf("%lf",(double)(end-start)/CLOCKS_PER_SEC);
}
```

| **OUTPUT:** | ```
students@students-HP-280-G3-MT:~$ ./a.out
Enter the elements of 1st matrix:1 3 7 5
Enter the elements of 2nd matrix:6 8 4 2

S1=6
S2=4
S3=12
S4=-2
S5=6
S6=8
S7=-2
S8=6
S9=-6
S10=14

p1=6
p2=8
p3=72
p4=-10
p5=48
p6=-12
p7=-84

MATRIX A:-

1        3
7        5
MATRIX B:-

6        8
4        2
MATRIX C:-

18       14
62       66
The time taken by the program: 0.000209students@students-HP-280-G3-MT:~$
``` |
| **CONCLUSION:** | By performing above experiment I have understood how the time complexity of strassen's matix multiplication is better than that of normal mxn matrix multiplication. |