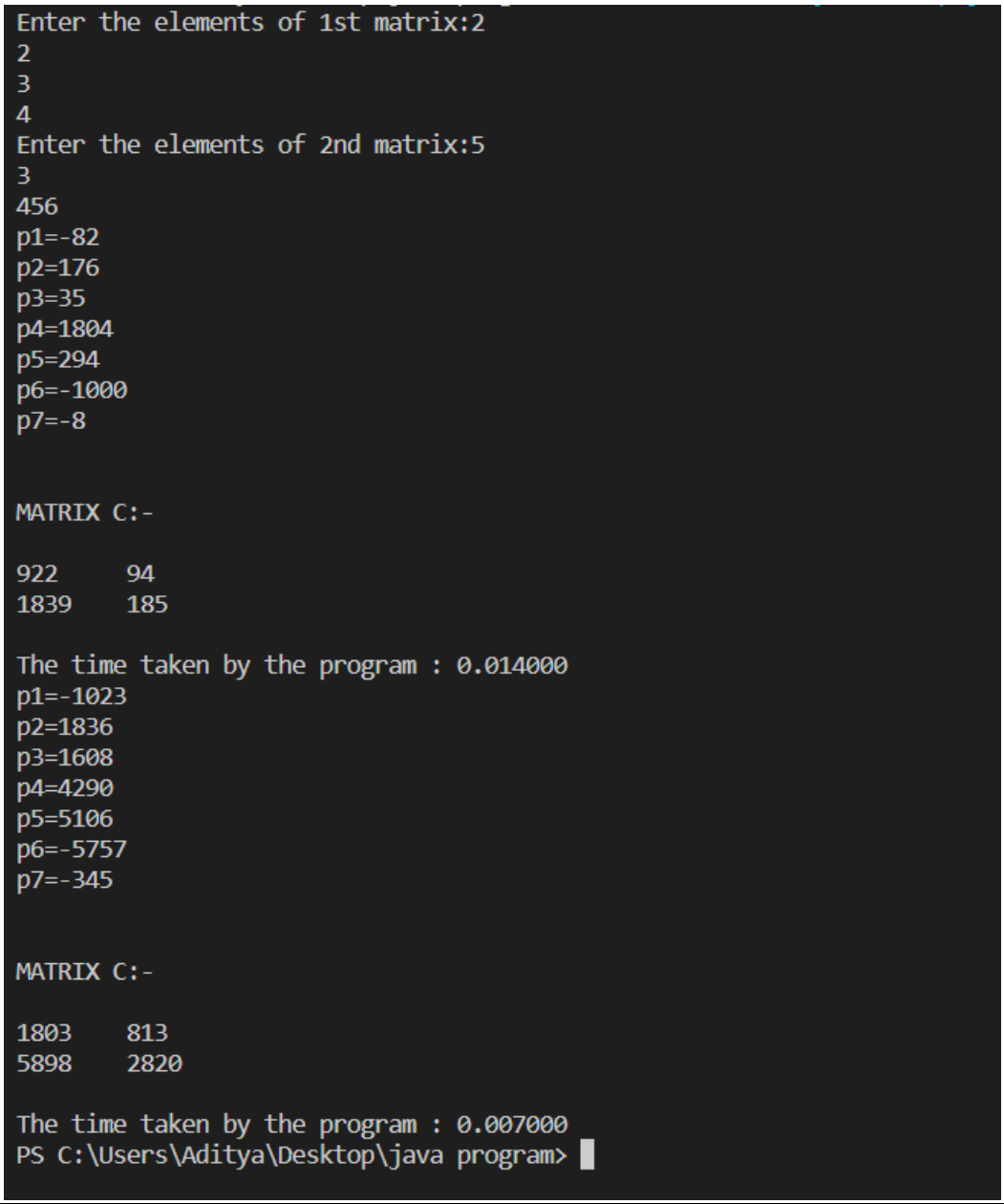
NAME:	Aditya Choudhary
UID:	2021300022
SUBJECT	Design and Analysis of Algorithm
EXPERIMENT NO :	03
DATE OF PERFORMANCE	28/02/2023
DATE OF SUBMISSION	5/03/2023
AIM:	To multiply two matrices using strassen's matrix multiplication.
PROBLEM STATEMENT 1:	**Strassen's matrix multiplication on a generalized 2x2 matix.**
ALGORITHM and THEORY:	**Strassen** has used some formulas for multiplying the two 2*2 dimension matrices where the number of multiplications is seven, additions and subtractions are is eighteen, and in brute force algorithm, there is eight number of multiplications and four addition. When the order **n** of matrix reaches infinity, the utility of Strassen's formula is shown by its asymptotic superiority. For example, let us consider two matrices **A** and **B** of **n*n** dimension, where **n** is a power of two. It can be observed that we can have four submatrices of order **n/2 * n/2** from **A**, **B,** and their product **C** where **C** is the resultant matrix of **A** and **B**. **The procedure of Strassen's matrix multiplication** Here is the procedure : 1. Divide a matrix of the order of 2*2 recursively until we get the matrix of order 2*2. 2. To carry out the multiplication of the 2*2 matrix, use the

	previous set of formulas. 3. Subtraction is also performed within these eight multiplications and four additions. 4. To find the final product or final matrix combine the result of two matrixes. **T(n) = 7T(n/2) + O(n^2)** which leads to **O(n^log(7))** runtime. This comes out to approxiamtely **O(n^2.8074)** which is better than O(n^3)

Program:

```
#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,bend;
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:");


    for(i=0;i<2;i++)
{
for(j=0;j<2;j++)
{
scanf("%d",&b[i][j]);
}
}
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");
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];
 bend=clock();
for(i=0;i<10;i++)
{

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("\n"); printf("MATRIX C:-\n\n"); end=clock();
printf("%d\t%d\n%d\t%d\n",c[0][0],c[0][1],c[1][0],c[1][1]);

printf("\nThe time taken by the program : ");
printf("%lf",(double)(end-start)/CLOCKS_PER_SEC);
}
```

OUTPUT:	
CONCLUSION:	By performing above experiment I have understood how the time complexity of strassen's matix multiplication is better than that of normal nxn matrix multiplication.