NAME:	Vedant Deshmukh
UID:	2021300025
SUBJECT	DAA
EXPERIMENT NO :	03
AIM:	To find shortest path from single source using Dijkstras Algorithm.
PROBLEM STATEMENT 1:	

THEORY

First, we will discuss naïve method and its complexity. Here, we are calculating ***Z = X × Y***. Using Naïve method, two matrices (***X*** and ***Y***) can be multiplied if the order of these matrices are ***p × q*** and ***q × r***. Following is the algorithm.

Algorithm: Matrix-Multiplication (X, Y, Z)

```
for i = 1 to p do
   for j = 1 to r do
      Z[i,j] := 0
      for k = 1 to q do
         Z[i,j] := Z[i,j] + X[i,k] × Y[k,j]
```

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)×(n/2) matrices as represented below –

ALGORITHM

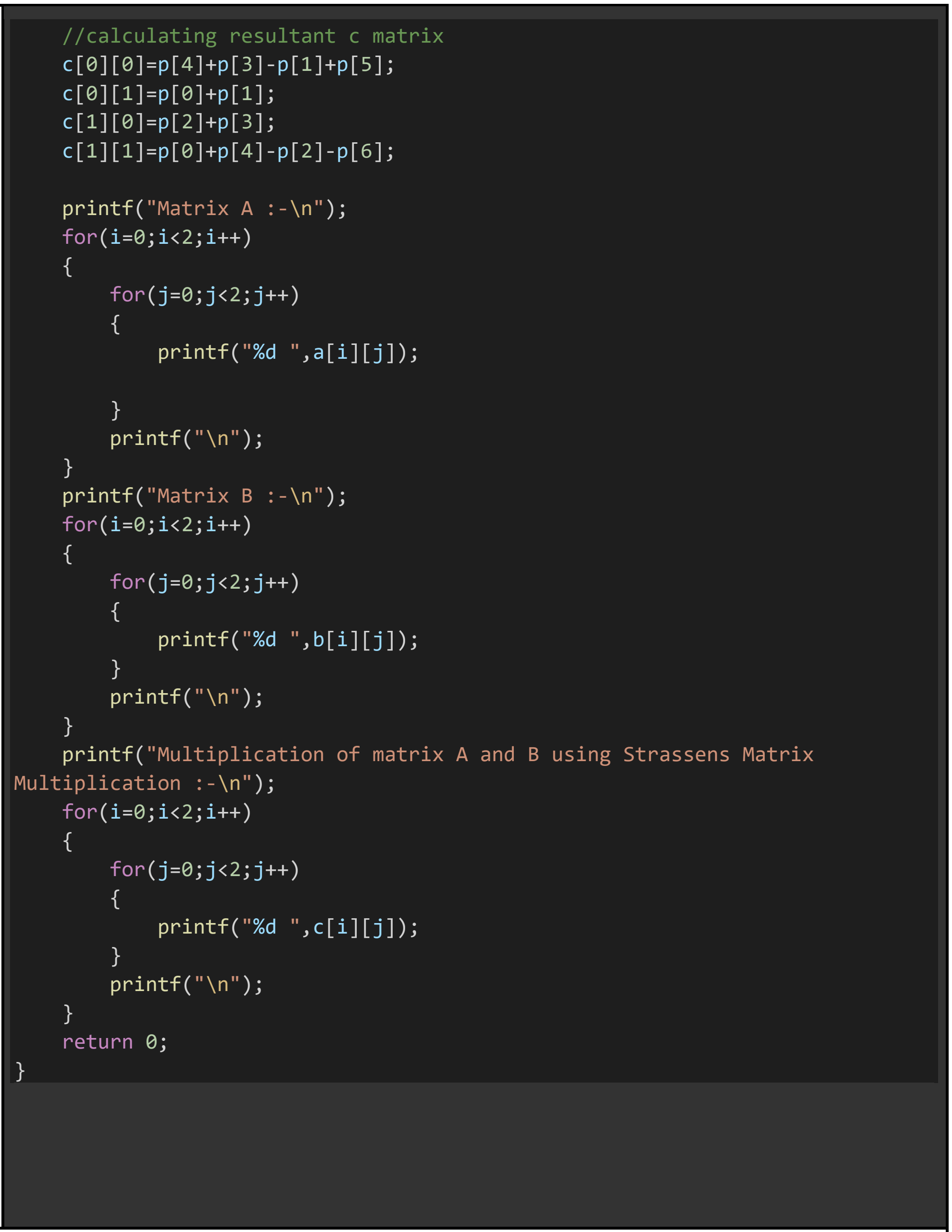
Algorithm: Matrix-Multiplication (X, Y, Z)

```
for i = 1 to p do
  for j = 1 to r do
    Z[i,j] := 0
    for k = 1 to q do
      Z[i,j] := Z[i,j] + X[i,k] × Y[k,j]
```

PROGRAM:

```c
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int a[2][2],b[2][2],i,j,c[2][2];
    int p[7];
    int s[10];
    printf("Enter elements of 1st matrix\n");
    for(i=0;i<2;i++)
    {
        for(j=0;j<2;j++)
        {
            scanf("%d",&a[i][j]);
        }
    }
    printf("Enter elements of 2nd matrix\n");
    for(i=0;i<2;i++)
    {
        for(j=0;j<2;j++)
        {
            scanf("%d",&b[i][j]);
        }
    }
    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];
```

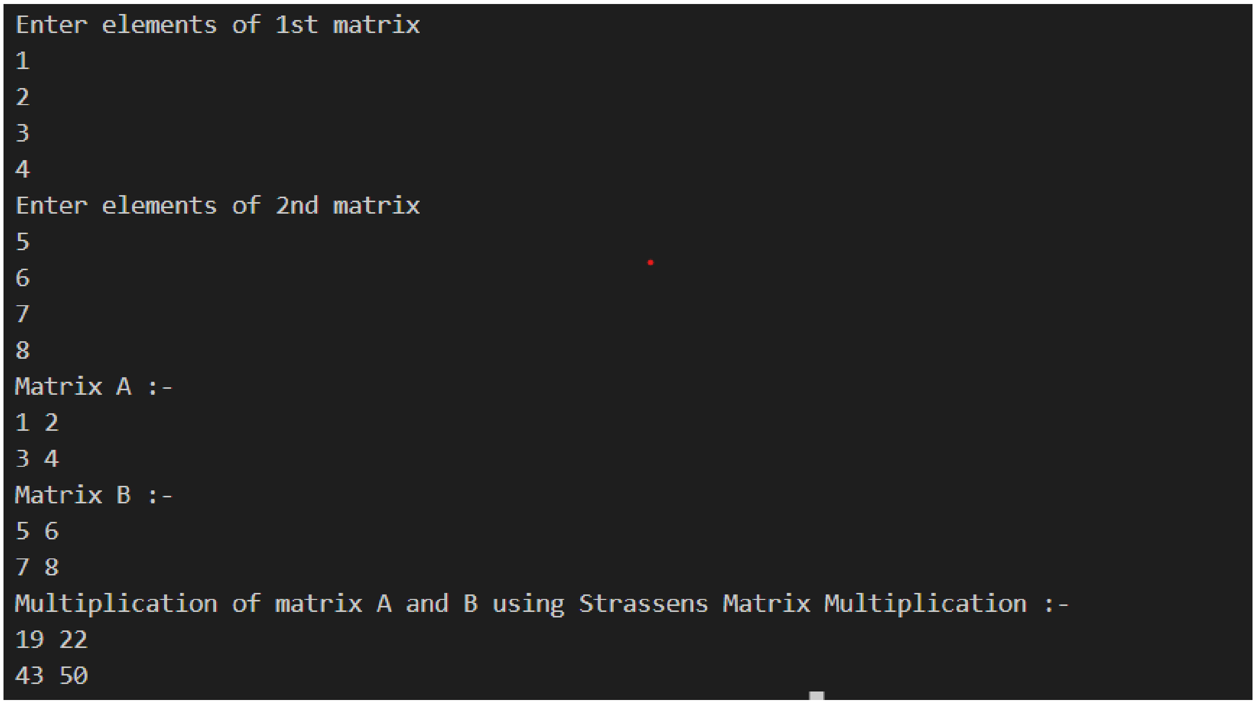


```c
    //calculating resultant c matrix
    c[0][0]=p[4]+p[3]-p[1]+p[5];
    c[0][1]=p[0]+p[1];
    c[1][0]=p[2]+p[3];
    c[1][1]=p[0]+p[4]-p[2]-p[6];

    printf("Matrix A :-\n");
    for(i=0;i<2;i++)
    {
        for(j=0;j<2;j++)
        {
            printf("%d ",a[i][j]);

        }
        printf("\n");
    }
    printf("Matrix B :-\n");
    for(i=0;i<2;i++)
    {
        for(j=0;j<2;j++)
        {
            printf("%d ",b[i][j]);
        }
        printf("\n");
    }
    printf("Multiplication of matrix A and B using Strassens Matrix
Multiplication :-\n");
    for(i=0;i<2;i++)
    {
        for(j=0;j<2;j++)
        {
            printf("%d ",c[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

RESULT (SNAPSHOT)



```
Enter elements of 1st matrix
1
2
3
4
Enter elements of 2nd matrix
5
6
7
8
Matrix A :-
1 2
3 4
Matrix B :-
5 6
7 8
Multiplication of matrix A and B using Strassens Matrix Multiplication :-
19 22
43 50
```

CONCLUSION:	Through this experiment I understood how to implement strassens matrix multiplication