Name: **Nitesh Kumar**
Roll No. : **10MA20033**

Q.> Write a program to implement the Thomas Algorithm and solve 2nd order ODE using that.

Solution. >

```
//including the library files
# include <stdio.h>
# include <math.h>
# include <malloc.h>

int main()
{
	int i, j, num;
	float **A, *d, **new_A, *new_d, *c, *x, s;

	//prompting the user to enter the values
	printf("\nenter the total number of rows: ");
	scanf("%d", &num);

	//forming required arrays
	A = (float **)malloc(num * sizeof(float *));
	d = (float *)malloc(num * sizeof(float));
	c = (float *)malloc(num * sizeof(float));
	x = (float *)malloc(num * sizeof(float));

	new_A = (float **)malloc(num * sizeof(float *));
	new_d = (float *)malloc(num * sizeof(float));

	//forming the matrix
	for(i=0; i<num;i++)
	{
		A[i] = (float *) malloc( num * sizeof(float) );
		new_A[i] = (float *) malloc( num * sizeof(float) );
		printf( "\nenter values for eqution %d:", (i + 1) );
		for(j=0; j<num; j++)
		{
			scanf("%f", &A[i][j]);
			new_A[i][j] = ((j < i) || (j > i + 1)) ? 0 : 1;
		}
		scanf("%f", &d[i]);
	}

	//calculating the starting values
	new_A[0][1] = A[0][1] / A[0][0];
```

```
        c[0] = new_A[0][1];
        new_d[0] = d[0] / A[0][0];

        for(i=1; i<num; i++)
        {
                j = i + 1;
                float temp = A[i][i] - A[i][i - 1] * new_A[i - 1][i];
                new_A[i][j] = A[i][j] / temp;
                c[i] = new_A[i][j];
                new_d[i] = (d[i] - (A[i][i - 1] * new_d[i - 1])) / temp;
        }

        //displaying the matrix
        for(i=0; i<num; i++)
        {
                for(j=0; j<num; j++)
                {
                        printf("%f ", new_A[i][j]);
                }
                printf("%f %f\n", new_d[i], c[i]);
        }

        x[num-1] = new_d[num-1];

        //back substitution
        for(i=num-2; i>=0; i--)
        {
                x[i] = new_d[i] - c[i] * x[i + 1];
        }

        //displaying the result
        for(i=0; i<num; i++)
        {
                printf("D = %f; Y = %f\n", new_d[i], x[i]);
        }
        return 0;
}
```