Name: Aditya Somani
Roll No: T1851061
Div: A
PRN NO. 71901204L

ASSIGNMENT NO. 3

TITLE: Thread management using pthread library.

```
#include<stdio.h>
#include<pthread.h>       //standard pthread library
#include<stdlib.h>
#define MAX 50

struct matrix
{
        int p,q,r;
}typedef Matrix;
int a[MAX][MAX],b[MAX][MAX],c[MAX][MAX];
int r1,r2,c1,c2;

pthread_mutex_t lock;

void *multiply(void *temp)
{
        Matrix *m=(Matrix *)temp;
        int mult=0;

        //pthread_mutex_lock(&lock);
        mult=a[m->p][m->r]*b[m->r][m->q];

        //pthread_mutex_unlock(&lock);
        pthread_exit((int *)mult);
}

int main()
{
        pthread_t tid;
        Matrix m;
        int i,j,k;
        void *s;
        int *mult;
        pthread_attr_t attr;
        pthread_attr_init(&attr);

        pthread_mutex_init(&lock,NULL);

        //Accepting row & columns of 1st matrix

        printf("\n\tEnter the data for 1st matrix :\n");
        printf("\trows=");
        scanf("%d",&r1);
        printf("\tColumns=");
        scanf("%d",&c1);
```

```
//Accepting row & columns of 2nd matrix

printf("\n\tEnter the data for 2nd matrix :\n");
printf("\trows=");
scanf("%d",&r2);
printf("\tColumns=");
scanf("%d",&c2);

if(c1!=r2)
{
        printf("Cannot perform multiplication on matrix...!!!!!");
        exit(0);
}

if(pthread_mutex_init(&lock,NULL)!=0)       //initiate the mutex
{
        printf("\n\t mutex init failed\n");
        return 1;
}

printf("Enter the elements of 1st Matrix :\n");

for(i=0;i<r1;i++)
        for(j=0;j<c1;j++)
        {
                printf("\n\tmatrix(%d-%d)=",i+1,j+1);
                scanf("%d",&a[i][j]);
        }
printf("Enter the elements of 2nd Matrix :\n");

for(i=0;i<r2;i++)
        for(j=0;j<c2;j++)
        {
                printf("\n\tmatrix(%d-%d)=",i+1,j+1);
                scanf("%d",&b[i][j]);
        }

for(i=0;i<r1;i++)
{
        m.p=i;
        for(j=0;j<c2;j++)
        {
                m.q=j;
                c[i][j]=0;
                for(k=0;k<r2;k++)
                {
                        m.r=k;
                        pthread_create(&tid,&attr,&multiply,&m);
                        pthread_join(tid,s);
                        c[i][j]=c[i][j]+*(int *)s;
                }
        }
    }

    printf("\t\nThe resultant matrix is :\n");
    for(i=0;i<r1;i++)
    {

        for(j=0;j<c2;j++)
        {
            printf("\t%d",c[i][j]);
        }
        printf("\n");
    }

    pthread_mutex_destroy(&lock);

    return 0;
}
/*
OUTPUT:-
anuj@anuj-Inspiron-5520:~/Desktop/anuj1/2$ gcc -o threads threads.c -lpthread -lrt

anuj@anuj-Inspiron-5520:~/Desktop/anuj1/2$ ./threads

    Enter the data for 1st matrix :
    rows=2
```

```
        Columns=3

        Enter the data for 2nd matrix :
        rows=3
        Columns=2
Enter the elements of 1st Matrix :

        matrix(1-1)=1

        matrix(1-2)=2

        matrix(1-3)=3

        matrix(2-1)=4

        matrix(2-2)=5

        matrix(2-3)=6
Enter the elements of 2nd Matrix :

        matrix(1-1)=1

        matrix(1-2)=2

        matrix(2-1)=3
        matrix(2-2)=4


        matrix(3-1)=5


        matrix(3-2)=6

The resultent matrix is :
        22      28
        49      64
anuj@anuj-Inspiron-5520:~/Desktop/anuj1/2$
*/
```