```
/*
	Mihir Kulkarni
	33132 L9
	Assignment 3 : Matrix multiplication using pthreads
*/
```

Method 1: Complete Static

```
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
#include<pthread.h>
#include<string.h>
#include<malloc.h>
#define MAX 4
#define MAX_THREADS 4

int mat1[MAX][MAX],mat2[MAX][MAX],mat3[MAX][MAX];
int r1,r2,c1,c2;

void *Multiply(void *args)
{
	for (int i = 0; i < r1; i++)
	{
		for (int j = 0; j < c2; j++)
		{
			mat3[i][j]=0;
			for (int k = 0; k < c1; k++)
			{
	    mat3[i][j] += mat1[i][k] * mat2[k][j];
			}
		}
	}


}

int main()
{
	int i,j;
	printf("\nEnter the rows of matrix 1 :");
	scanf("%d",&r1);

	printf("\nEnter the columns of matrix 1 :");
	scanf("%d",&c1);

	printf("\nEnter the rows of matrix 2 : ");
	scanf("%d",&r2);

	if(c1 != r2){
		printf("\nMatrix multiplication is not possible!!");
		exit(0);
```

```
        }

        printf("\nEnter the columns of matrix 2 : ");
        scanf("%d",&c2);

        //Input mat1
        printf("\nEnter matrix 1 :\n");
        for(int i = 0; i < r1;i++){
                for(int j = 0; j < c1;j++){
                        printf("Enter element matrix [%d][%d] :: ",i+1,j+1);
                        scanf("%d",&mat1[i][j]);
                }
        }
        //Input mat2
        printf("\nEnter matrix 2 :\n");
        for(int i = 0; i < r2;i++){
                for(int j = 0; j < c2;j++){
                        printf("Enter element matrix [%d][%d] :: ",i+1,j+1);
                        scanf("%d",&mat2[i][j]);
                }
        }

        printf("The matrix 1 is \n");
        for(int i = 0; i < r1;i++){
                for(int j = 0; j < c1;j++){
                        printf("%d\t",mat1[i][j]);
                }
                printf("\n");
        }
        printf("The matrix 2 is \n");
        for(int i = 0; i < r2;i++){
                for(int j = 0; j < c2;j++){
                        printf("%d\t",mat2[i][j]);
                }
                printf("\n");
        }
        //Declaring pthreads
        pthread_t threads[MAX_THREADS];

// Creating threads, each evaluating its own part
for (int i = 0; i < MAX_THREADS; i++) {
    int* result;
    pthread_create(&threads[i], NULL, Multiply, (void*)(result));
}
// joining and waiting for all threads to complete
        for (int i = 0; i < MAX_THREADS; i++)
    pthread_join(threads[i], NULL);

        printf("The multiplied matrix is \n");

        for(int i = 0; i < r1;i++){
                for(int j = 0; j < c2;j++){
```

```
                        printf("%d\t",mat3[i][j]);
                }
                printf("\n");
        }
        return 0;

}
```

Output:

```
mihir@pop-os:~/TE/OS-lab/a3$ ./a3

Enter the rows of matrix 1 :2

Enter the columns of matrix 1 :3

Enter the rows of matrix 2 : 3

Enter the columns of matrix 2 : 2

Enter matrix 1 :
Enter element matrix [1][1] :: 1
Enter element matrix [1][2] :: 1
Enter element matrix [1][3] :: 1
Enter element matrix [2][1] :: 1
Enter element matrix [2][2] :: 1
Enter element matrix [2][3] :: 1

Enter matrix 2  :
Enter element matrix [1][1] :: 1
Enter element matrix [1][2] :: 1
Enter element matrix [2][1] :: 1
Enter element matrix [2][2] :: 1
Enter element matrix [3][1] :: 1
Enter element matrix [3][2] :: 1
The matrix 1 is
1       1       1
1       1       1
The matrix 2 is
1       1
1       1
1       1
The multiplied matrix is
3       3
3       3
mihir@pop-os:~/TE/OS-lab/a3$
```

```
mihir@pop-os:~/TE/OS-lab/a3$ ./a3

Enter the rows of matrix 1 :1

Enter the columns of matrix 1 :2

Enter the rows of matrix 2 : 3

Matrix multiplication is not possible!!mihir@pop-os:~/TE/OS-lab/a3$ ./a3

Enter the rows of matrix 1 :2

Enter the columns of matrix 1 :2

Enter the rows of matrix 2 : 2

Enter the columns of matrix 2 : 2

Enter matrix 1 :
Enter element matrix [1][1] :: 1
Enter element matrix [1][2] :: 1
Enter element matrix [2][1] :: 1
Enter element matrix [2][2] :: 1

Enter matrix 2  :
Enter element matrix [1][1] :: 2
Enter element matrix [1][2] :: 2
Enter element matrix [2][1] :: 2
Enter element matrix [2][2] :: 2
The matrix 1 is
1       1
1       1
The matrix 2 is
2       2
2       2
The multiplied matrix is
4       4
4       4
mihir@pop-os:~/TE/OS-lab/a3$
```

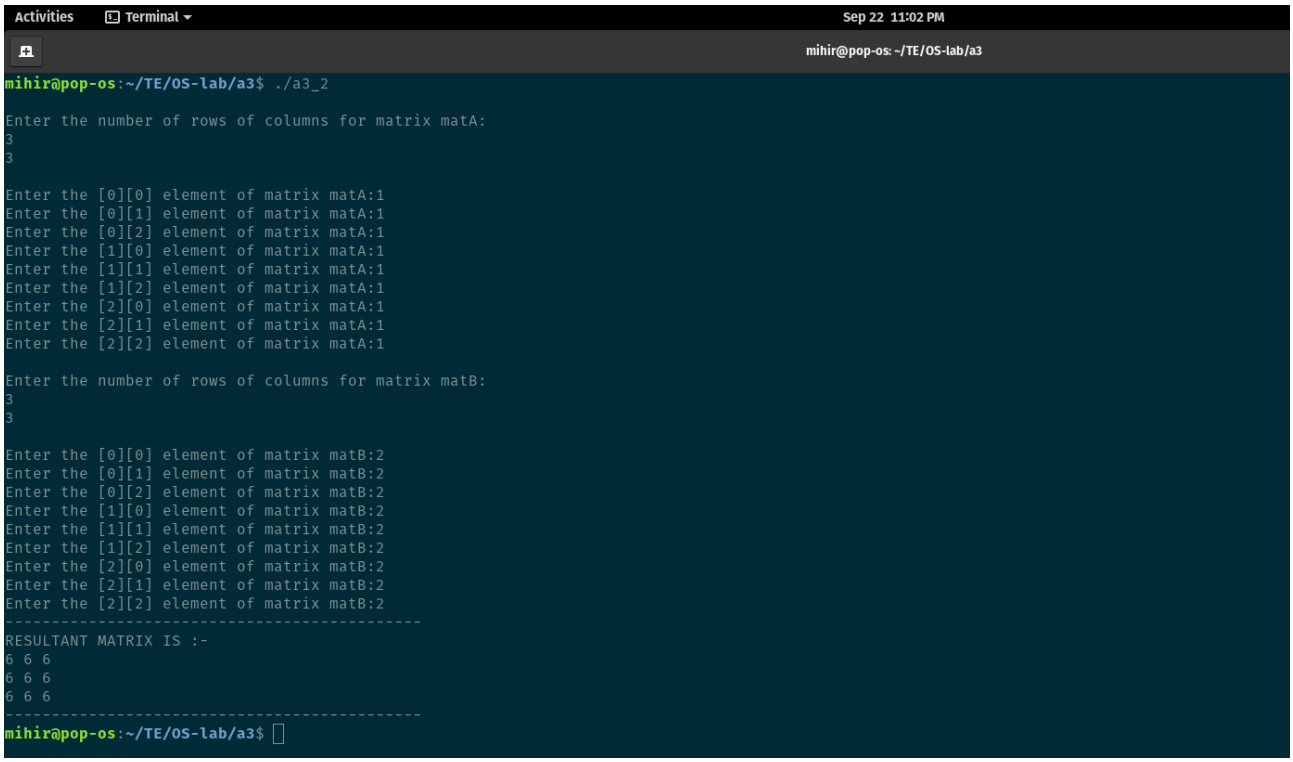
Method 2: r1*c2 threads

```
/*
	Mihir Kulkarni
	33132 L9
	Assignment 3 : Matrix multiplication using pthreads
*/

#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
#include<pthread.h>
#include<string.h>
#include<malloc.h>
#define MAX 4
/* 9 thread solution for a 3*3 matrix */

//Each thread computes single element in the resultant matrix
void *mult(void* arg)
{
	int *data = (int *)arg;
	int k = 0, i = 0;

	int x = data[0];
	//r1=c2=x number of sumations of multiplications in array
	for (i = 1; i <= x; i++)
		k += data[i]*data[i+x];

	int *p = (int*)malloc(sizeof(int));
	*p = k;

	//Used to terminate a thread and the return value is passed as a pointer
	pthread_exit(p);
}

//Driver code
int main()
{
	int r1,c1,r2,c2,i,j,k;

	printf("\nEnter the number of rows of columns for matrix matA:\n");
	scanf("%d %d", &r1, &c1);

	//Initialize matA
	int matA[r1][c1];

	printf("\n");

	//Input of matrix A
	for(int i=0; i<r1; i++)
	{
		for(int j=0; j<c1; j++)
```

```
            {
                        printf("Enter the [%d][%d] element of matrix matA:", i, j);
                        scanf("%d", &matA[i][j]);
            }
      }


      printf("\nEnter the number of rows of columns for matrix matB:\n");
      scanf("%d %d", &r2, &c2);

      //Initialize matA
      int matB[r2][c2];

      printf("\n");

      //Input of matrix B
      for(int i=0; i<r2; i++)
      {
            for(int j=0; j<c2; j++)
            {
                        printf("Enter the [%d][%d] element of matrix matB:", i, j);
                        scanf("%d", &matB[i][j]);
            }
      }


int max = r1*c2;


//declaring array of threads of size r1*c2
pthread_t *threads;
threads = (pthread_t*)malloc(max*sizeof(pthread_t));

int count = 0;
int* data = NULL;
for (i = 0; i < r1; i++)
{
      for (j = 0; j < c2; j++)
      {

            //storing row and column elements in data
            data = (int *)malloc((r1*c2)*sizeof(int));

            //number of sumations of  multiplications in array
            data[0] = c1;

            //Inserting row
            for (k = 0; k < c1; k++)
                  data[k+1] = matA[i][k];
```

```
                //Inserting column after row elements
                for (k = 0; k < r2; k++)
                        data[k+c1+1] = matB[k][j];

                //creating threads
                pthread_create(&threads[count++], NULL, mult, (void*)(data));


        }
    }
    printf("-------------------------------------------\n");
    printf("RESULTANT MATRIX IS :- \n");
    for (i = 0; i < max; i++)
    {
        void *k;

        //Joining all threads and collecting return value
        pthread_join(threads[i], &k);

        int *p = (int *)k;
        printf("%d ",*p);
        if ((i + 1) % c2 == 0)
                printf("\n");
    }
    printf("-------------------------------------------\n");
  return 0;
}
```

Output:



```
mihir@pop-os:~/TE/OS-lab/a3$ ./a3_2

Enter the number of rows of columns for matrix matA:
2
2

Enter the [0][0] element of matrix matA:1
Enter the [0][1] element of matrix matA:1
Enter the [1][0] element of matrix matA:1
Enter the [1][1] element of matrix matA:1

Enter the number of rows of columns for matrix matB:
2
2

Enter the [0][0] element of matrix matB:2
Enter the [0][1] element of matrix matB:2
Enter the [1][0] element of matrix matB:2
Enter the [1][1] element of matrix matB:2
--------------------------------------------
RESULTANT MATRIX IS :-
4 4
4 4
--------------------------------------------
mihir@pop-os:~/TE/OS-lab/a3$
```

Method 2: r1*c2*c1 threads(27 threads)

```
/*
  Mihir Kulkarni
  33132 L9
  Assignment 3 : Matrix multiplication using pthreads
*/

#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include<unistd.h>

// 3*3 matrix solution using 27 threads

void *multiply(void *arg)
{
	//Function to multiply element by element of each row in matrix A
	//with each element in columns of matrix B
	int *data = (int *)arg;
	int k = 0;

	k = data[0] * data[1];

	int *p = (int *)malloc(sizeof(int));
	*p = k;

	pthread_exit(p);
}


int main()
{
	int res, r1, c1, r2, c2, max, count = 0;
	pthread_t *threads;

	printf("\nEnter the number of rows of columns for matrix matA:\n");
	scanf("%d %d", &r1, &c1);

	//Initialize matA
	int matA[r1][c1];

	printf("\n");
	//Input of matrix A
```

```
	for(int i=0; i<r1; i++)
	{
		for(int j=0; j<c1; j++)
		{
			printf("Enter the [%d][%d] element of matrix matA:", i, j);
			scanf("%d", &matA[i][j]);
		}
	}


	printf("\nEnter the number of rows of columns for matrix matB:\n");
	scanf("%d %d", &r2, &c2);

	//Initialize matB
	int matB[r2][c2];

	printf("\n");

	//Input of matrix B
	for(int i=0; i<r2; i++)
	{
		for(int j=0; j<c2; j++)
		{
			printf("Enter the [%d][%d] element of matrix matB:", i, j);
			scanf("%d", &matB[i][j]);
		}
	}

	//each row will take r1*c2 traversals for c1 in columns
	//for e.g. in 3*3 matrices each resultant element requires sumation of 3 multiplications
	//i.e. 3 threads and for 9 resultant elements : 9*3= 27 threads

	max = r1 * c2 * c1;
	threads = (pthread_t *)malloc(max*sizeof(pthread_t));

	int *data = NULL;
	for(int i=0; i<r1; i++)
	{
		for(int j=0; j<c2; j++)
		{
			for(int k=0; k<c1; k++)
			{
				//taking 2 elements at a time
				data = (int *)malloc(2*sizeof(int));
				data[0] = matA[i][k];
				data[1] = matB[k][j];
				//thread creation
				res = pthread_create(&threads[count++], NULL, multiply, (void
*)data);
```

```
                        if (res != 0)
                        {
                                perror("\nThread creation failed!\n");
                                exit(EXIT_FAILURE);
                        }
                }
        }
    }

    printf("\n-------------------------------------------------------\n");
    printf("\nThe resultant matrix is:\n\n");

    for(int i=0; i<max; i++)
    {
        void *k;
        int ans;
        //Joining threads
        res = pthread_join(threads[i], &k);

        if (res != 0)
        {
            perror("\nThread joining failed!\n");
            exit(EXIT_FAILURE);
        }
        //if first sub-part of r1*c2
        if(i % 3 == 0)
        {
            ans = *(int *)k;
        }
        //if second sub-part of r1*c2
        else if(i % 3 == 1)
        {
            ans += *(int *)k;
        }
        //added all three elements and print value of resultant matrix element
        else
        {
            ans += *(int *)k;
            printf("%d ", ans);
            //if all elements in a rows are over of resultant marix => next line
            if((i+1) % (r1*c2) == 0)
                printf("\n");
        }
    }

    return 0;
}
```

Output:

```
mihir@pop-os:~/TE/OS-lab/a3$ gcc -D_REENTRANT 27.c -o 27 -lpthread
mihir@pop-os:~/TE/OS-lab/a3$ ./27

Enter the number of rows of columns for matrix matA:
3
3

Enter the [0][0] element of matrix matA:1
Enter the [0][1] element of matrix matA:1
Enter the [0][2] element of matrix matA:1
Enter the [1][0] element of matrix matA:1
Enter the [1][1] element of matrix matA:1
Enter the [1][2] element of matrix matA:1
Enter the [2][0] element of matrix matA:1
Enter the [2][1] element of matrix matA:1
Enter the [2][2] element of matrix matA:1

Enter the number of rows of columns for matrix matB:
3
3

Enter the [0][0] element of matrix matB:2
Enter the [0][1] element of matrix matB:2
Enter the [0][2] element of matrix matB:2
Enter the [1][0] element of matrix matB:2
Enter the [1][1] element of matrix matB:2
Enter the [1][2] element of matrix matB:2
Enter the [2][0] element of matrix matB:2
Enter the [2][1] element of matrix matB:2
Enter the [2][2] element of matrix matB:2

-------------------------------------------------------

The resultant matrix is:

6 6 6
6 6 6
6 6 6
```