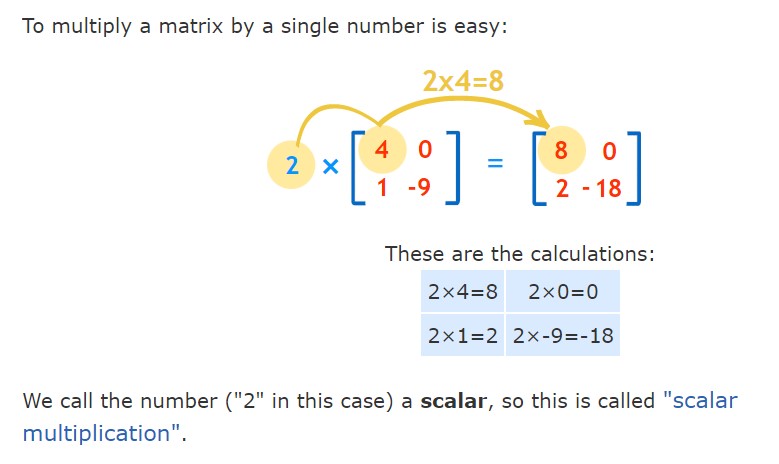
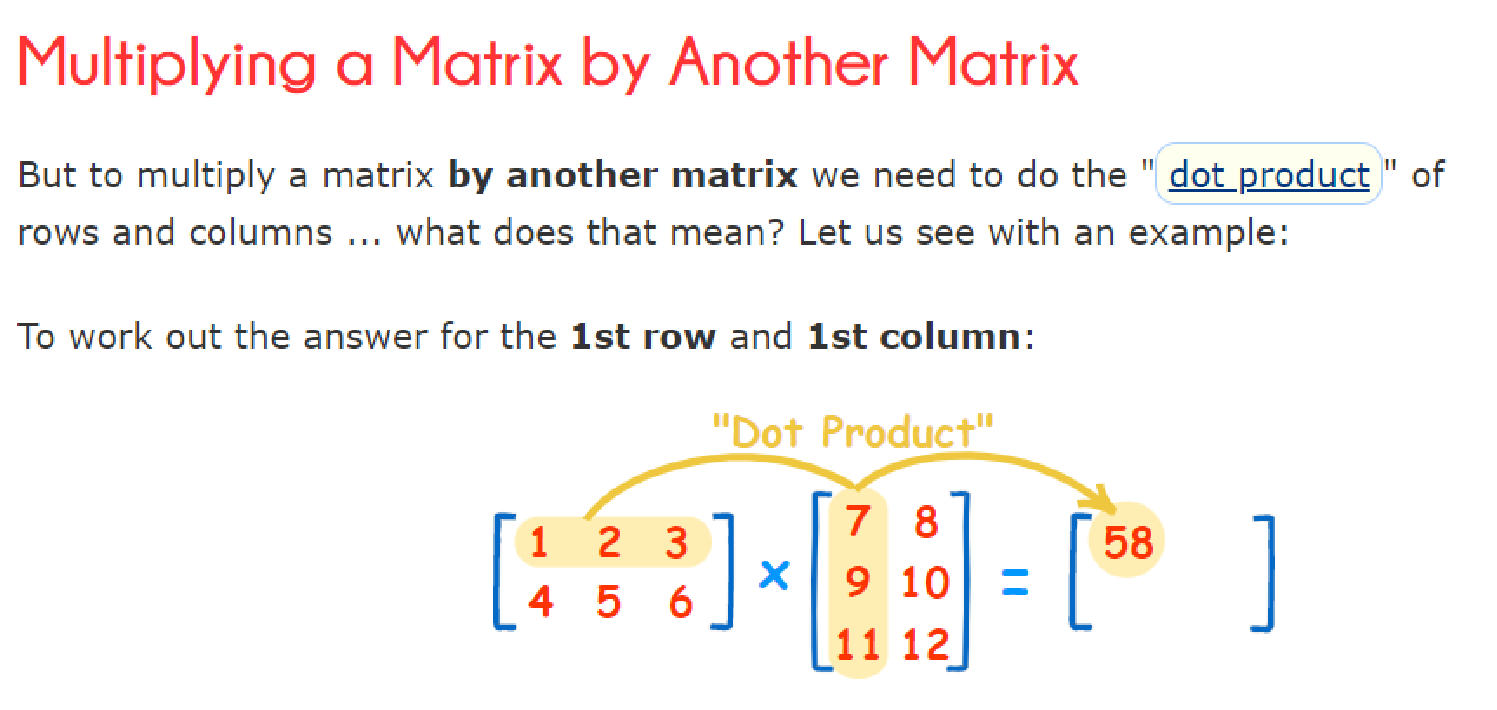
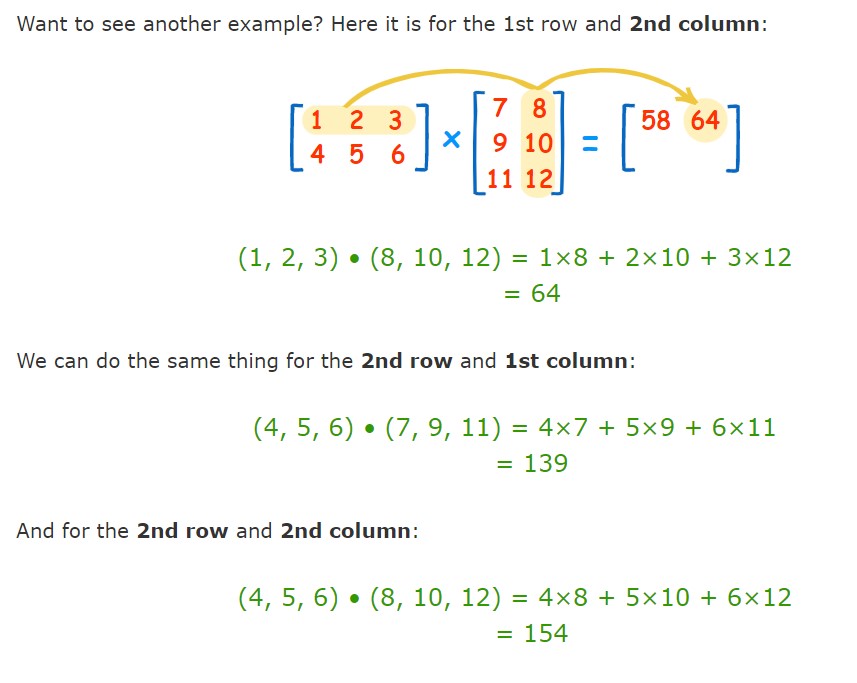
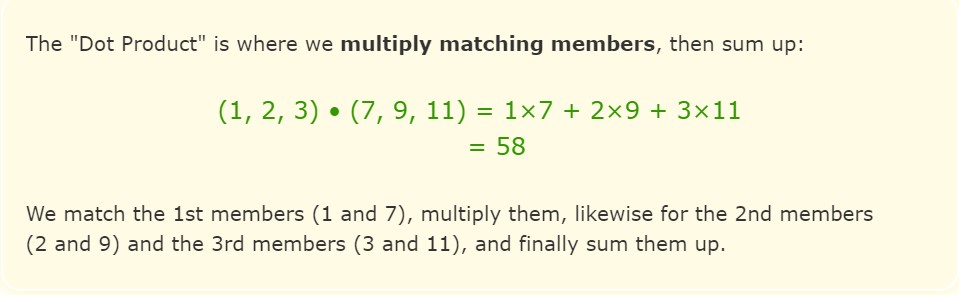
**TITLE: Matrix-Multiplication**

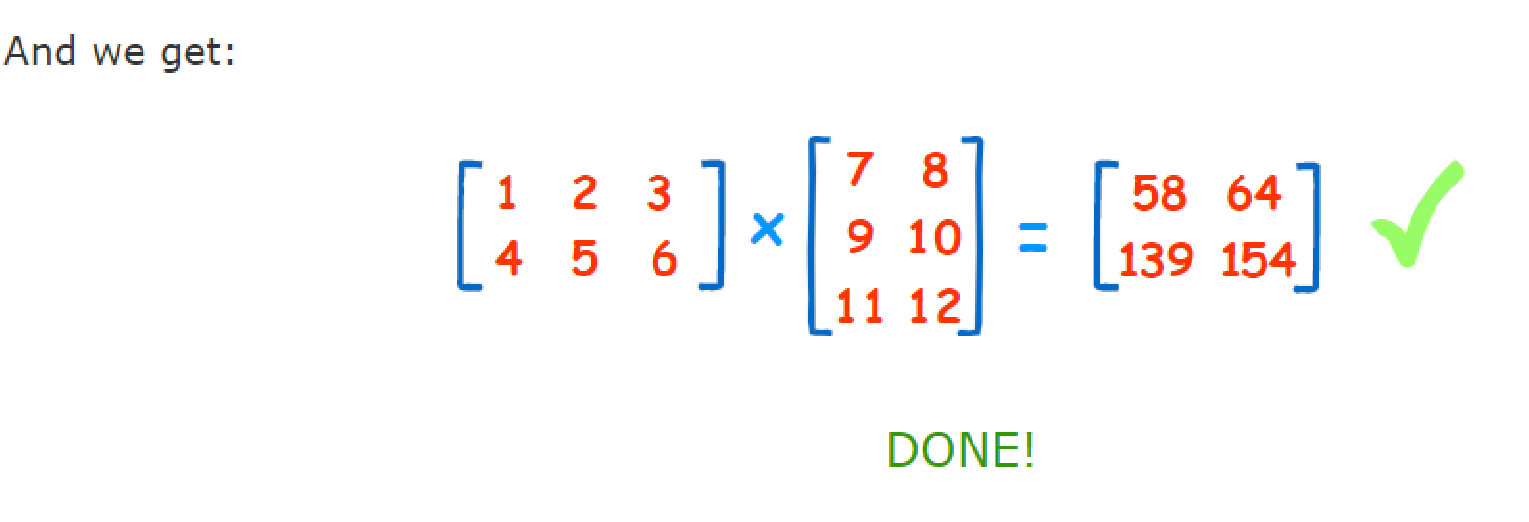
**PROBLEM STATEMENT:** Implement matrix multiplication using multithreading with pthread library.

**THEORY:**









**What are threads?**

Technically, a thread is defined as an independent stream of instructions that can be scheduled to run as such by the operating system.

A thread is a semi-process that has its own stack, and executes a given piece of code.

Unlike a real process, the thread normally shares its memory with other threads (where as for processes we usually have a different memory area for each one of them). A Thread Group is a set of threads all executing inside the same process.

**What are pthreads?**

Historically, hardware vendors have implemented their own proprietary versions of threads. These implementations differed substantially from each other, making it difficult for programmers to develop portable threaded applications. In order to take full advantage of the capabilities provided by threads, a standardized programming interface was required. For UNIX systems, this interface has been specified by the IEEE POSIX 1003.1c standard (1995). Implementations which adhere to this standard are referred to as POSIX threads, or Pthreads. Most hardware vendors now offer Pthreads in addition to their proprietary threads.

**PROGRAM:**

#include<stdio.h>

#include<pthread.h>

#include<stdlib.h>

#include<unistd.h>

#define MAX 3

int a[MAX][MAX]; int b[MAX][MAX]; int c[MAX][MAX];

//Generic function prototypes void\* mult(void\*);

int main()

{

pthread\_t tid1,tid2,tid3; //Threads for row1, row2, and row3 int row1,row2,row3;

int i,j;

printf("\n\nEnter First matrix : ");

for(i=0;i<MAX;i++){ for(j=0;j<MAX;j++){

printf("\nEnter a[%d][%d] : ",i,j); scanf("%d",&a[i][j]);

}

}

printf("\n\nEnter Second matrix");

for(i=0;i<MAX;i++){ for(j=0;j<MAX;j++){

printf("\nEnter b[%d][%d] : ",i,j); scanf("%d",&b[i][j]);

}

}

row1=0;

//Create a thread as tid1; executing mult routine; accepting index of row1 pthread\_create(&tid1, NULL, mult, &row1); // pthread\_create(a1, a2, mult, 0) row2=1;

//Create a thread as tid2; executing mult routine; accepting index of row2 pthread\_create(&tid2, NULL, mult, &row2);

row3=2;

//Create a thread as tid3; executing mult routine; accepting index of row3 pthread\_create(&tid3, NULL, mult, &row3);

pthread\_join(tid1,NULL); pthread\_join(tid2,NULL); pthread\_join(tid3,NULL);

printf("\n\nResult is : \n");

for(i=0;i<MAX;i++){ for(j=0;j<MAX;j++){ printf("%d ",c[i][j]);

} printf("\n");

}

exit(0);

}//End of main

void\* mult(void \* arg) // Address of 0th row

{

int i,j,k;

i = \*(int \* )arg; //value of i = 0, 1, or 2

for(j=0;j<MAX;j++){ c[i][j] = 0;

for(k=0;k<MAX;k++){ c[i][j] += a[i][k] \* b[k][j];

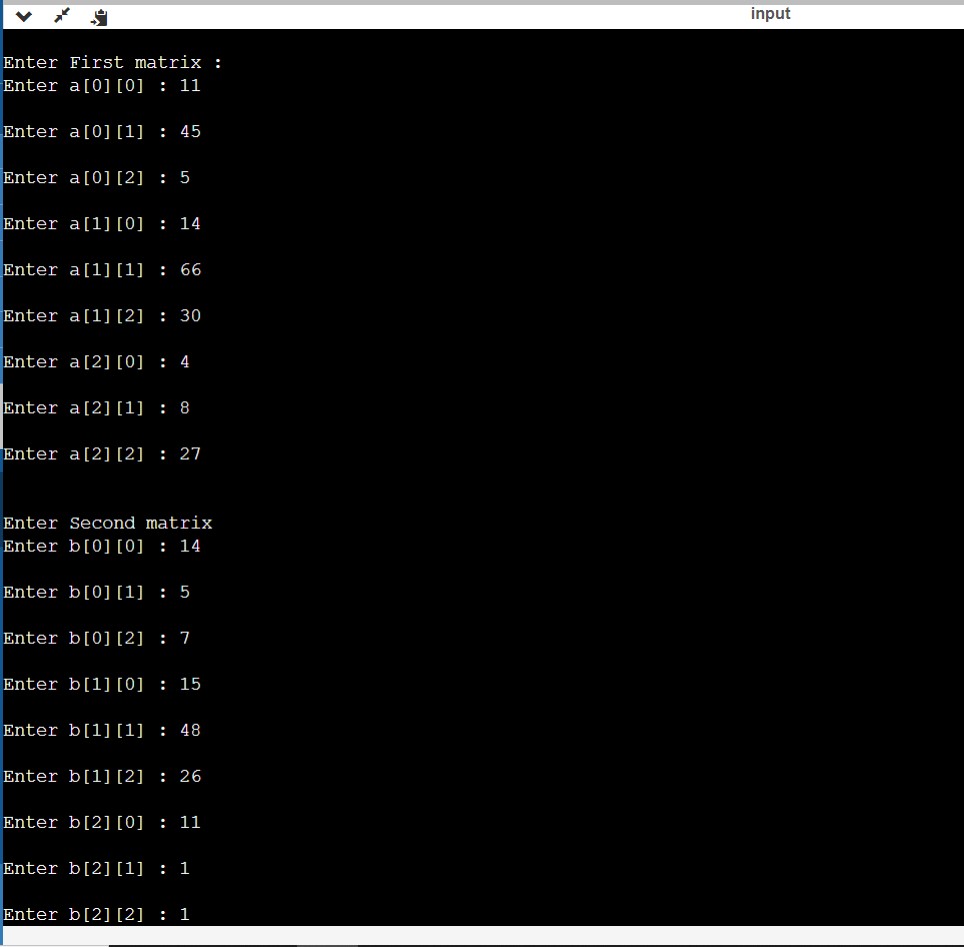
}

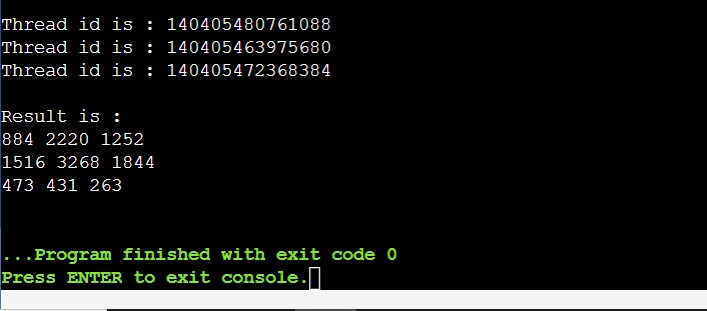
}

printf("\nThread id is : %ld",pthread\_self());

}//END

**OUTPUT:**





**CONCLUSION:**

Hence, we have successfully Implemented matrix multiplication using multithreading with pthread library.