In [ ]:

```
code = """
#include<iostream>

using namespace std;

__global__
void matrixMultiplication(int *a, int *b, int *c, int m, int n, int k)
{
    int row = blockIdx.y*blockDim.y + threadIdx.y;
    int col = blockIdx.x*blockDim.x + threadIdx.x;
    int sum=0;

    if(col<k && row<m) {
      for(int j=0;j<n;j++)
      {
          sum += a[row*n+j] * b[j*k+col];
      }
      c[k*row+col]=sum;
    }

}

void init_result(int *a, int m, int k) {
    for(int i=0; i<m; i++) {
      for(int j=0; j<k; j++) {
        a[i*k + j] = 0;
      }
    }
}

void init_matrix(int *a, int n, int m) {
    for(int i=0; i<n; i++) {
      for(int j=0; j<m; j++) {
        a[i*m + j] = rand()%10 + 1;
      }
    }
}

void print_matrix(int *a, int n, int m) {
    for(int i=0; i<n; i++) {
      for(int j=0; j<m; j++) {
        cout<<"  "<<a[i*m + j];
      }
      cout<<endl;
    }
    cout<<endl;
}

int main()
{

    int *a,*b,*c;
    int *a_dev,*b_dev,*c_dev;
    int m=5, n=4, k=3;

    a = new int[m*n];
    b = new int[n*k];
    c = new int[m*k];

    init_matrix(a, m, n);
    init_matrix(b, n ,k);
    init_result(c, m, k);

    cout<<"Initial matrix : "<<endl;

    print_matrix(a, m, n);
    print_matrix(b, n, k);
```

```
    print_matrix(c, m, k);

    cudaMalloc(&a_dev, sizeof(int)*m*n);
    cudaMalloc(&b_dev, sizeof(int)*n*k);
    cudaMalloc(&c_dev, sizeof(int)*m*k);

    cudaMemcpy(a_dev, a, sizeof(int)*m*n, cudaMemcpyHostToDevice);
    cudaMemcpy(b_dev, b, sizeof(int)*n*k, cudaMemcpyHostToDevice);

    dim3 dimGrid(1,1);
    dim3 dimBlock(16,16);

    matrixMultiplication<<<dimGrid, dimBlock>>>(a_dev,b_dev,c_dev, m, n, k);

    cudaMemcpy(c, c_dev, sizeof(int)*m*k, cudaMemcpyDeviceToHost);

    cout<<"Result : "<<endl;
    print_matrix(c, m, k);

    cudaFree(a_dev);
    cudaFree(b_dev);
    cudaFree(c_dev);

    delete[] a;
    delete[] b;
    delete[] c;

    return 0;
}
"""
```

In [ ]:

```
text_file = open("matMulti.cu", "w")
text_file.write(code)
text_file.close()
```

In [ ]:

```
!nvcc matMulti.cu
```

In [ ]:

```
!./a.out
```

```
Initial matrix :
  4  7  8  6
  4  6  7  3
  10  2  3  8
  1  10  4  7
  1  7  3  7

  2  9  8
  10  3  1
  3  4  8
  6  10  3

  0  0  0
  0  0  0
  0  0  0
  0  0  0
  0  0  0

Result :
  0  0  0
  0  0  0
  0  0  0
  0  0  0
  0  0  0
```