~\Desktop\Outputs\HPC\4-B.cpp

Name:Arshin Mokashi
Roll No.:COBB26

```cpp
#include <cuda_runtime.h>
#include <iostream>

__global__ void matmul(int* A, int* B, int* C, int N) {
    int Row = blockIdx.y * blockDim.y + threadIdx.y;
    int Col = blockIdx.x * blockDim.x + threadIdx.x;
    if (Row < N && Col < N) {
        int Pvalue = 0;
        for (int k = 0; k < N; k++) {
            Pvalue += A[Row * N + k] * B[k * N + Col];
        }
        C[Row * N + Col] = Pvalue;
    }
}

int main() {
    int N = 512;
    int size = N * N * sizeof(int);
    int* A, * B, * C;
    int* dev_A, * dev_B, * dev_C;

    cudaMallocHost(&A, size);
    cudaMallocHost(&B, size);
    cudaMallocHost(&C, size);

    cudaMalloc(&dev_A, size);
    cudaMalloc(&dev_B, size);
    cudaMalloc(&dev_C, size);

    // Initialize matrices A and B
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            A[i * N + j] = i * N + j;
            B[i * N + j] = j * N + i;
        }
    }

    cudaMemcpy(dev_A, A, size, cudaMemcpyHostToDevice);
    cudaMemcpy(dev_B, B, size, cudaMemcpyHostToDevice);

    dim3 dimBlock(16, 16);
    dim3 dimGrid(N / dimBlock.x, N / dimBlock.y);

    matmul<<<dimGrid, dimBlock>>>(dev_A, dev_B, dev_C, N);

    cudaMemcpy(C, dev_C, size, cudaMemcpyDeviceToHost);

```

```
    // Print the result (first 10x10 block)
    for (int i = 0; i < 10; i++) {
        for (int j = 0; j < 10; j++) {
            std::cout << C[i * N + j] << " ";
        }
        std::cout << std::endl;
    }

    // Free memory
    cudaFree(dev_A);
    cudaFree(dev_B);
    cudaFree(dev_C);
    cudaFreeHost(A);
    cudaFreeHost(B);
    cudaFreeHost(C);

    return 0;
}


/*

Output:

1235 1234 1233 1232 1231 1230 1229 1228 1227 1226
1465 1464 1463 1462 1461 1460 1459 1458 1457 1456
1580 1579 1578 1577 1576 1575 1574 1573 1572 1571
1695 1694 1693 1692 1691 1690 1689 1688 1687 1686
...


*/
```