```
#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
    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:-
615012  616536  618060  619584  621108  622632  624156  625680  627204  628728
```

```
1534136 1539662 1545188 1550714 1556240 1561766 1567292 1572818 1578344 1583870
2453256 2463786 2474316 2484846 2495376 2505906 2516436 2526966 2537496 2548026
3372376 3387910 3403444 3418978 3434512 3450046 3465580 3481114 3496648 3512182
4291496 4312034 4332572 4353110 4373648 4394186 4414724 4435262 4455800 4476338
5210616 5236158 5261700 5287242 5312784 5338326 5363868 5389410 5414952 5440494
6129736 6160282 6190828 6221374 6251920 6282466 6313012 6343558 6374104 6404650
7048856 7084406 7119956 7155506 7191056 7226606 7262156 7297706 7333256 7368806
7967976 8008530 8049084 8089638 8130192 8170746 8211300 8251854 8292408 8332962
8887096 8932654 8978212 9023770 9069328 9114886 9160444 9206002 9251560 9297118
```