King Saud University
College of Computer and Information Sciences
Department of Computer Science
CSC453 – Parallel Processing – Tutorial No 6bis – Quarter 3 2023

Question

1. Let's consider the following serial nested loops:

```
for (int i = 0 ; i < N; i ++)
      for (int j = 0 ; j < M; i ++)
            doSomething(i, j);
```

   a. Give a parallel solution using CUDA C.

```
__global__ void kernal(int N, int M){
int i = threadIdx.y;
int j = threadIdx,x;
if (i < N && j < M) doSomthing(i, j);
}

__device__ void doSomthing(int i, int j){...}
```

2. We would like to write a C program that calculates in parallel the matrix-vector product. Let's consider a ***N by N*** square matrix. Let's consider a vector (1-D array) of size ***N***. The matrix-vector product is calculated as follows:

$$\begin{bmatrix} a_1^1 & \cdots & a_n^1 \\ \vdots & \ddots & \vdots \\ a_1^n & \cdots & a_n^n \end{bmatrix} \times [x_1 \quad \dots \quad x_n] = \begin{bmatrix} \sum_{i=1}^{n} a_i^1 \times x_i \\ \dots \\ \sum_{i=1}^{n} a_i^n \times x_i \end{bmatrix}$$

So, we would like to write a kernel that receives 4 parameters:

- a matrix of integers denoted **A (input)**,
- a vector of integers denoted **B (input)**,
- a vector of integers, denoted **C (output)**, where the matrix-vector product will be stored. Elements of vector C will be calculated as follows:

$$C[i] = \sum_{j=1}^{N} \mathrm{A[i][j]} * \mathrm{B[j]}.$$

- an integer denoted **N (input)** which is the size of the vector B; and the number of rows and columns of the matrix A.

a. We would like to run this kernel within a grid composed of multiple blocks each of which is vector of threads. We would like that every thread calculates a single cell of the result (**the vector C**). Give the code of the kernel.

```
__global__ void kernal(int *A, int *B, int *C, int N){
int index = blockIdx.x * blockDim.x + threadIdx.x;
for (int k = 0; k<N; k++)
   C[index] += A[index][k] * B[k];
}
```