Questions à Choix Multiples avec Calculs et Codes en C / Multiple Choice Questions with Calculations and C Codes

Question 1:

Quelle fonction CUDA est utilisée pour allouer de la mémoire sur l'appareil ? / Which CUDA function is used to allocate memory on the device?

a) malloc / malloc
b) cudaMalloc / cudaMalloc
c) free / free
d) cudaMemcpy / cudaMemcpy

Réponse/Answer : b) cudaMalloc / b) cudaMalloc

Question 2:

Quel mot-clé est utilisé pour spécifier une fonction de noyau CUDA ? / Which keyword is used to specify a CUDA kernel function?

a) __global__ / __global__
b) __device__ / __device__
c) __host__ / __host__
d) __shared__ / __shared__

Réponse/Answer : a) __global__ / a) __global__

Question 3:

Quelle est la portée de la mémoire partagée dans CUDA ? / What is the scope of shared memory in CUDA?

a) À l'ensemble du GPU / To the entire GPU
b) Aux threads individuels uniquement / To individual threads only
c) Au bloc de threads / To the thread block
d) Aux grilles de threads / To the thread grids

Réponse/Answer : c) Au bloc de threads / c) To the thread block

Question 4:

Quelle est la formule pour calculer l'index global dans une matrice 2D CUDA ? / What is the formula to calculate the global index in a 2D CUDA matrix?

a) ix = threadIdx.x + blockIdx.x * blockDim.x
b) iy = threadIdx.y + blockIdx.y * blockDim.y

c) idx = iy * nx + ix
d) Toutes les réponses ci-dessus / All of the above

Réponse/Answer : d) Toutes les réponses ci-dessus / d) All of the above

Question 5:

Analysez le code suivant :

```
__global__ void matrixAdd(float *A, float *B, float *C, int nx, int ny) {
   int ix = threadIdx.x + blockIdx.x * blockDim.x;
   int iy = threadIdx.y + blockIdx.y * blockDim.y;
   int idx = iy * nx + ix;
   if (ix < nx && iy < ny) {
      C[idx] = A[idx] + B[idx];
   }
}
```

Que fait ce code CUDA ? / What does this CUDA code do?

a) Multiplie deux matrices / Multiplies two matrices
b) Ajoute deux matrices élément par élément / Adds two matrices element-wise
c) Effectue une somme vectorielle / Performs a vector sum
d) Alloue de la mémoire pour une matrice / Allocates memory for a matrix

Réponse/Answer : b) Ajoute deux matrices élément par élément / b) Adds two matrices element-wise

Question 6:

Quelle est la syntaxe correcte pour invoquer un noyau CUDA avec 32 threads dans un bloc ? / What is the correct syntax to invoke a CUDA kernel with 32 threads in a block?

a) kernel_name<<<1, 32>>>()
b) kernel_name<<<32, 1>>>()
c) kernel_name<<<4, 8>>>()
d) kernel_name<<<32, 32>>>()

Réponse/Answer : a) kernel_name<<<1, 32>>>() / a) kernel_name<<<1, 32>>>()

Question 7:

Quelle fonction CUDA est utilisée pour synchroniser les threads dans un bloc ? / Which CUDA function is used to synchronize threads in a block?

a) cudaMemcpy / cudaMemcpy
b) __syncthreads() / __syncthreads()
c) atomicAdd / atomicAdd
d) cudaMalloc / cudaMalloc

Réponse/Answer : b) __syncthreads() / b) __syncthreads()

Question 8:

Quelle est la différence entre gridDim et blockDim ? / What is the difference between gridDim and blockDim?

a) gridDim représente les dimensions des blocs dans une grille / gridDim represents the dimensions of blocks in a grid
b) blockDim représente les dimensions des threads dans un bloc / blockDim represents the dimensions of threads in a block
c) Les deux / Both
d) Aucune des réponses ci-dessus / None of the above

Réponse/Answer : c) Les deux / c) Both

Question 9:

Quel est le rôle de `atomicAdd` en CUDA ? / What is the role of `atomicAdd` in CUDA?

a) Ajouter deux variables de manière atomique / Add two variables atomically
b) Synchroniser les threads / Synchronize threads
c) Gérer les collisions dans la mémoire globale / Handle collisions in global memory
d) Les réponses a et c / Both a and c

Réponse/Answer : d) Les réponses a et c / d) Both a and c