1. Is there any performance problem with the following CUDA kernel? Clearly explain. [5 pts]

```
__global__ void incrementOddEven(int *data, int n) {
int idx = threadIdx.x + blockIdx.x * blockDim.x;
if (idx < n) {
    if (idx % 2 == 0) {
        data[idx] += 2;  // Increment even indexed elements by 2
    } else {
        data[idx] += 1;  // Increment odd indexed elements by 1
    }
}
}
```

Rewrite the kernel to fix the highlighted performance problem.

2. Consider the following CUDA code which applies 5 x 5 kernel on a single channel image of size 1024 x 1024. While answering the following questions, you can specify the assumed grid and block dimensions wherever required.

```
__global__ void applyConvolution(float *input, float *output, \\
                                 float *kernel, int width, int height)
{
    int x = blockIdx.x * blockDim.x + threadIdx.x;
    int y = blockIdx.y * blockDim.y + threadIdx.y;
    if (x < width && y < height) {
        float sum = 0.0f;
        for (int ky = -2; ky <= 2; ky++) {
            for (int kx = -2; kx <= 2; kx++) {
            // don't worry too much about the following min-max code
            // They handle when the kernel is applied on the boudary pixels.
                int ix = min(max(x + kx, 0), width - 1);  - 2 FLOPs
                int iy = min(max(y + ky, 0), height - 1); - 2 FLOPs
                sum += input[iy * width + ix] * kernel[(ky + 2) * 5 + (kx + 2)];
            }                                   2 FLOPs.
        }
        output[y * width + x] = sum;
    }
}
```

(a) What is the FLOPS/word for the above program? [1 pt]

(b) If the DDR bandwidth is 150 GB/s, what is the acheivable GFLOPs/sec? [1 pt]

(c) How can we improve the GFLOPs/sec for the given kernel? Explain the optimization strategy. Provide the estimated final GFLOPs/sec after the optimizations. [2 pts]

(d) Rewrite the kernel code implementing the above optimizations. You can use pseudo-code or put comments wherever you are unsure about the exact syntax. [3 pts]

3. Answer the following questions related to block and thread scheduling on GPUs.
   (a) Can a thread block, once assigned to a Streaming Multiprocessor (SM), be rescheduled to a different SM during its execution? Explain [2 pts]
   (b) Can threads within a single block be scheduled for execution on different SMs? Explain. [2 pts]

4. Answer the following questions related to matrix-vector multiplications.
   (a) Formulate matrix-vector multiplication using `dotp`. [1½ pts]
   (b) Formulate matrix-vector multiplication using *axpy* [1½ pts]
   (c) Which one is more beneficial under what circumstances? Explain. [2 pts]

5. Consider the following two matrices.

$$A = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

   (a) Compute their product using outer product formulation. Clearly show the computational steps. [2 pts]
   (b) Split matrix A into 2 horizontal panels of dimension 2 x 4. Perform matrix multiplication using outer product formulation and the paritioned matrix strategy. [2 pts]
   (c) Perform matrix multiplication using matrix-vector multiplication which adopts axpy strategy. [2 pts]

6. State the work and span laws. Explain briefly. [2 pts]

7. When we multiply 2 matrices of dimension B x M and M x B, for small values of B, outer product formulation is beneficial. Clearly explain the reason. [2 pts]

8. What is the potential issue with the following OpenMP code snippet? [1 pt]

```
#pragma omp parallel for
for (int i = 0; i < N; i++) {
    sum += data[i];
}
```

9. What is Amdahl's law? [1 pt]

10. What are the three walls which lead to the multi-core era. Explain. [2 pts]