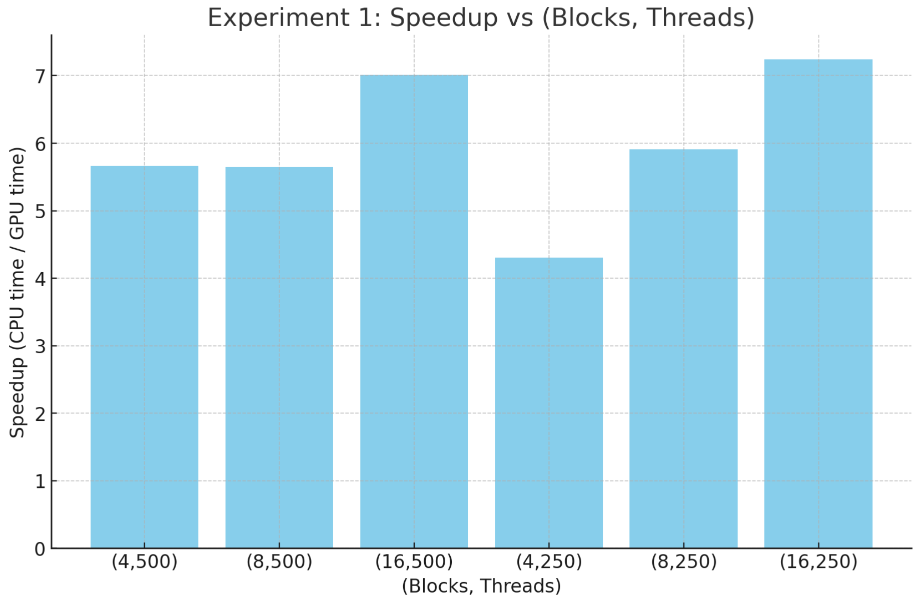
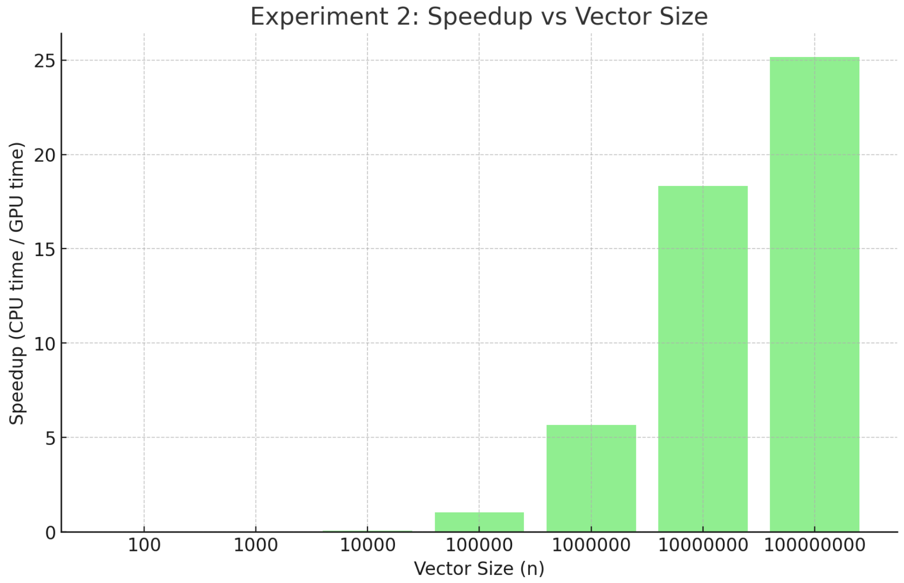
Experiment 1



In general there are improvements with the increase of the block counts to 16 blocks. Which make sense, because the more blocks allocated means the lesser tasks for each thread. And threads are more possible to be allocate to more SPs in parallel as the we have more blocks to be assigned to different SMs.

Experiment 2



The speed up graph is quite straight forward as the larger the vector size to more speed up. But you can see the speed up is not completely linear. My assumption is that when the vector size is small, we are bounded by the clock speed on GPU which is slower than CPU, therefore it is faster on CPU. As the vector size increases we also saturated the resources on GPU, so the speedup growth is slowing down. Although it is obviously faster on GPU.

```
---  Expriment 1 ---
Each vector will have 1000000 elements
Using 500 blocks per grid and 4 threads per block
Total time taken by the sequential part = 0.003313
Total time taken by the GPU part = 0.000585
Each vector will have 1000000 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.003286
Total time taken by the GPU part = 0.000582
Each vector will have 1000000 elements
Using 500 blocks per grid and 16 threads per block
Total time taken by the sequential part = 0.003303
Total time taken by the GPU part = 0.000471
Each vector will have 1000000 elements
Using 250 blocks per grid and 4 threads per block
Total time taken by the sequential part = 0.003286
Total time taken by the GPU part = 0.000764
Each vector will have 1000000 elements
Using 250 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.003327
Total time taken by the GPU part = 0.000563
Each vector will have 1000000 elements
Using 250 blocks per grid and 16 threads per block
Total time taken by the sequential part = 0.003360
Total time taken by the GPU part = 0.000464
--- Expriment 2 ---
Each vector will have 100 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.000002
Total time taken by the GPU part = 0.000377
Each vector will have 1000 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.000004
Total time taken by the GPU part = 0.000317
Each vector will have 10000 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.000034
Total time taken by the GPU part = 0.000402
Each vector will have 100000 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.000332
Total time taken by the GPU part = 0.000419
Each vector will have 1000000 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.003312
Total time taken by the GPU part = 0.000671
Each vector will have 10000000 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.033061
Total time taken by the GPU part = 0.001847
Each vector will have 100000000 elements
Using 500 blocks per grid and 8 threads per block
Total time taken by the sequential part = 0.330647
Total time taken by the GPU part = 0.012033
```