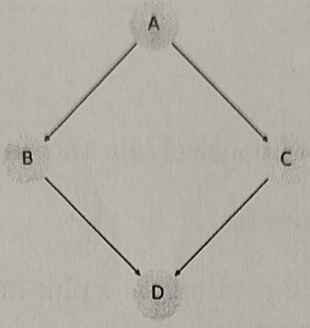
1. Write OpenMP code snippet using *tasks* for the following task graph. Is it beneficial to use *tied* clause in your code. [4 pts]



2. Given a scenario where you have a list of N independent tasks with varying execution times, write an OpenMP code snippet that demonstrates how to dynamically schedule these tasks to ensure efficient execution. Assume each task is encapsulated in a function called `processTask(int taskId)`. [5 pts]

3. What is the raw GFLOPS/sec for a processor with 4 cores, no hyper-threading, AVX512 instruction set, running at 3 GHz? Provide GFLOPS/sec assuming both with and without FMA operation. Any other reasonable assumptions you make, please clearly write them down. [5 pts]

4. Clearly reason about the performance scaling of the following code. [5 pts]

```
#pragma omp parallel num_threads(NUM_THREADS)
{
    int thread_id = omp_get_thread_num();
    int start = thread_id * (ARRAY_SIZE / NUM_THREADS);
    int end = start + (ARRAY_SIZE / NUM_THREADS);
    for (i = start; i < end; ++i) {
        partial_sums[thread_id] += array[i];
    }
}

// Aggregate the partial sums
for (i = 0; i < NUM_THREADS; ++i) {
    total_sum += partial_sums[i];
}
```

5. Consider a program designed to perform a large-scale matrix multiplication (C = A × B) where matrices A, B, and C are all square matrices of size N×N. Assume the matrices are stored in row-major order and the program uses a straightforward triple nested loop approach for the multiplication without any optimizations like tiling or the use of libraries optimized for matrix operations.

   The target hardware for running this program has the following characteristics:

- Peak computational performance: 100 GFLOPS.
- Memory bandwidth: 25 GB/s.
- Size of a floating-point operation (FLOP): 4 bytes

Assume the compiler does not perform any auto-vectorization or other optimization that significantly alters the basic characteristics of the algorithm's memory access pattern or its computation.

(a) What is the arithemtic (operational) intensity of the program? [2 pts]

(b) Is the problem compute or memory bound? [2 pts]

(c) What is ridge point? Is ridge point a function of hardware or the program? Explain using the given program and the hardware specification. [2 pts]