For this exercise, you are given a serial implementation of matrix multiplication (Exercises →Code → CH1 →matmul.c) Use this code as your starting point to answer the following questions

T.1 Parallelizing matrix multiplication using OpenMP

You can and should make the following modifications:

- Include the library needed for OpenMP support.
- Annotate the appropriate parallel section(s) of the code with the appropriate OpenMP directive(s).

Use the appropriate OpenMP clause(s) to control the number of threads (e.g., using the *int nthreads* parameter of the matmul() function)
Discuss the placement of the OpenMP directives. What are the potential places where the OpenMP directives can be placed and why? Try all alternative correct placements. Do you observe differences in performance with different placements. If so, why? If not, why not? Be specific with your analysis and explanation. Report the runtime of your experiments in a table.

T.2: Impact of work-sharing constructs

Start with the best performing implementation from M.1. Discuss the impact of the various *loop work-sharing constructs* (experiment with different types, like "***static"***, "***guided"***, for the "***schedule"*** OpenMP clause). Do you observe differences in performance with the different loop work-sharing constructs. If so, why? If not, why not? Be specific with your analysis and explanation.

T.3: Cache performance profiling for parallel matrix multiplication

Start with the best performing implementation from M.2 and set number of threads at 32. Report the runtime for all possible loop orders. What loop order results in the best performance. Qualitatively and quantitatively analyze the results.

T.4: Leveraging both DLP and TLP

Start with the best performing implementation from M.3 and set the number of threads to 32. Add `#pragma omp simd` at appropriate place to vectorize the execution of matrix multiplication. Compare and analyze the performance of this implementation with M.3
For more details on `#pragma omp simd`, look at https://www.ibm.com/docs/en/xl-c-and-cpp-linux/16.1.0?topic=pdop-pragma-omp-simd

T.5: Impact of matrix transpose operation

Start with the implementation from M.4. Prior to multiplying the matrices, compute a transpose of matrix B. Use the transpose matrix to compute the product, mat_A * mat_B. Make appropriate changes in the matrix multiply function to compute the answer. Report the performance and compare the performance with M.4.

Now, instead of taking a transpose of matrix B, compute matrix multiplication using the the transpose of matrix A. Report the performance and compare the performance with M.4.