2. In this problem, you will modify the matmul.c program provided, optimizing the execution of the matrix multiplication with first a dense matrix, and second with a sparse matrix. You are welcome to use pthreads, OpenMP or any of the optimizations that were presented in class to accelerate this code. There will be prizes awarded for the fastest dense and the fastest sparse implementations.

System: Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz
Thread(s) per core: 1
Core(s) per socket: 14
Socket(s) 2

The running result of the original matmul.c:

```
[luo.qiu@c2177 hw3]$ ./a.out
starting dense matrix multiply
a result 4.44488e+07
The total time for matrix multiplication with dense matrices = 7109.618744 ms
starting sparse matrix multiply
A result 0
The total time for matrix multiplication with sparse matrices = 7067.301979 ms
The sparsity of the a and b matrices = 0.750977
```

The running result of the optimized matmul.c:

```
[luo.qiu@c2177 hw3]$ ./a.out
starting dense matrix multiply
a result 4.44488e+07
The total time for matrix multiplication with dense matrices = 6618.839163 ms
starting sparse matrix multiply
A result 0
The total time for matrix multiplication with sparse matrices = 6523.709434 ms
The sparsity of the a and b matrices = 0.750977
```

Based on Pthreads implementation, I use 16 threads to breakdown the whole task into sub-tasks, and each sub-task calculate the multiplication of 512 / 16 = 32 sub-rows or sub-columns of the matrix.
And the speedup of dense matrix multiplication is 107.41%, the speedup of sparse matrix multiplication is 108.37%.