3. In this problem, you will utilize the OpenBLAS library available on Discovery. To use OpenBLAS, you will need to issue load openblas/0.3.6. Using the malmul.c program, replace the math with a call to appropriate gemm library function. Compare the speed of your solution for problem 2 with the gemm method you use.

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 malmul.c based on OpenBLAS:

```
[luo.qiu@c2177 hw3]$ ./a.out
starting dense matrix multiply
a result 4.44488e+07
The total time for matrix multiplication with dense matrices = 105.017391 ms
starting sparse matrix multiply
A result 0
The total time for matrix multiplication with sparse matrices = 59.584548 ms
The sparsity of the a and b matrices = 0.750977
```

The runtime of the OpenBlas-based program is orders of magnitude less than the runtime of the Pthreads-based program I wrote. The initial analysis is that OpenBlas performs vector operations, while my Pthreads-based implementation still performs scalar operations.