Lab2 General Matrix Multiply

Sample Code:

- Go to the `./Lab2-GEMM/code`
- Run `sh run_matmul.sh` in your Raspberry Pi terminal
- You can turn annotations on or off in the main function to get different runtime performance
  - `matmul()`
  - `matmul_ikj()`
  - `matmul_AT()`
  - `matmul_BT()`

Assignments:

Q1 The shape of matrix **A** is $I \times K$ and the shape of matrix **b** is $K \times J$. Get the different runtime performance under `matmul()`, `matmul_ikj()`, `matmul_AT()`, `matmul_BT()`. The value of I, K, J is be fixed at 64 and 512. Record the performance of these four approaches, and discuss your findings.

Q2 Based on the findings obtained in **Q1**, explore different techniques for cache optimization. For instance, the writing operation of matrix C is not consistent, and how to improve the write caching? Try at least two different optimization techniques to improve the cache hit ratio and reduce the matrix multiply time consumption. The value of I, K, J is be fixed at 512. Here are some examples you may use:

- Loop unrolling
- Writing caching
- Tiling
- Vectorization (SIMD)
- Array packing

Record the performance of your implemented optimization approaches, and discuss your findings.

Useful Materials:

- GEMM Optimization on CPU

Tips: You should learn the code style from the sample code to build your project.