

## Task 3

1. Which 2 orderings perform best for these 1000-by-1000 matrices? Write your answer in the form "[Ordering1], [Ordering2]" (e.g. "ijk, ijk").

**jki:** 10.643 Gflop/s ---> A and C are accessed with stride of 1 and B is accessed with stride of 0. There is no stride of n. So it is fast

**kji:** 8.994 Gflop/s ---> Same strides as of above case

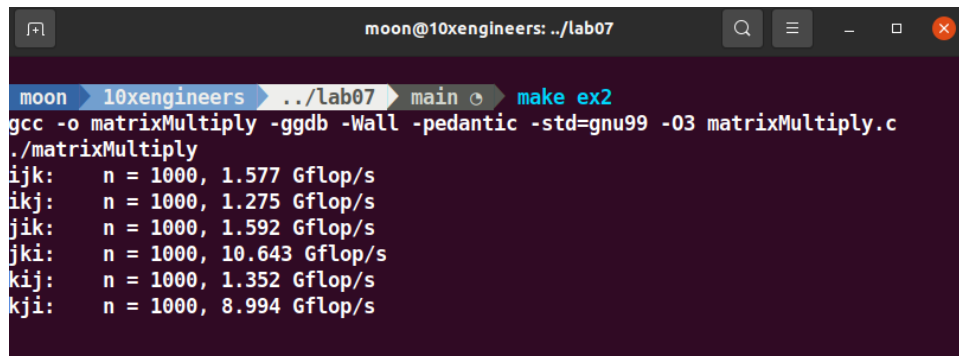
---

2. Which 2 orderings perform the worst?

**ikj:** 1.275 Gflop/s ---> B and C are accessed with strides of n and A is accessed with stride of 0. There are 2 strides of n so it is slow.

**kij:** 1.352 Gflop/s ---> Same strides as of above

---



```
moon@10xengineers: ../lab07 main make ex2
gcc -o matrixMultiply -ggdb -Wall -pedantic -std=gnu99 -O3 matrixMultiply.c
./matrixMultiply
ijk:  n = 1000, 1.577 Gflop/s
ikj:  n = 1000, 1.275 Gflop/s
jik:  n = 1000, 1.592 Gflop/s
jki:  n = 1000, 10.643 Gflop/s
kij:  n = 1000, 1.352 Gflop/s
kji:  n = 1000, 8.994 Gflop/s
```