

**Module: R4: Computer Architecture****Section: Caches Task: Loop Ordering****Task 3****Loop Ordering**

---

- **Action Item: Think about what the strides are for the nested loops in other five implementations.**

**1. Loop Order 1: *ijk***

```
A = n;  
B = 1;  
C = 0;
```

**2. Loop Order 2: *ikj***

```
A = 0;  
B = n;  
C = n;
```

**3. Loop Order 3: *jik***

```
A = n;  
B = 1;  
C = 0;
```

**4. Loop Order 4: *jki***

```
A = 1;  
B = 0;  
C = 1;
```

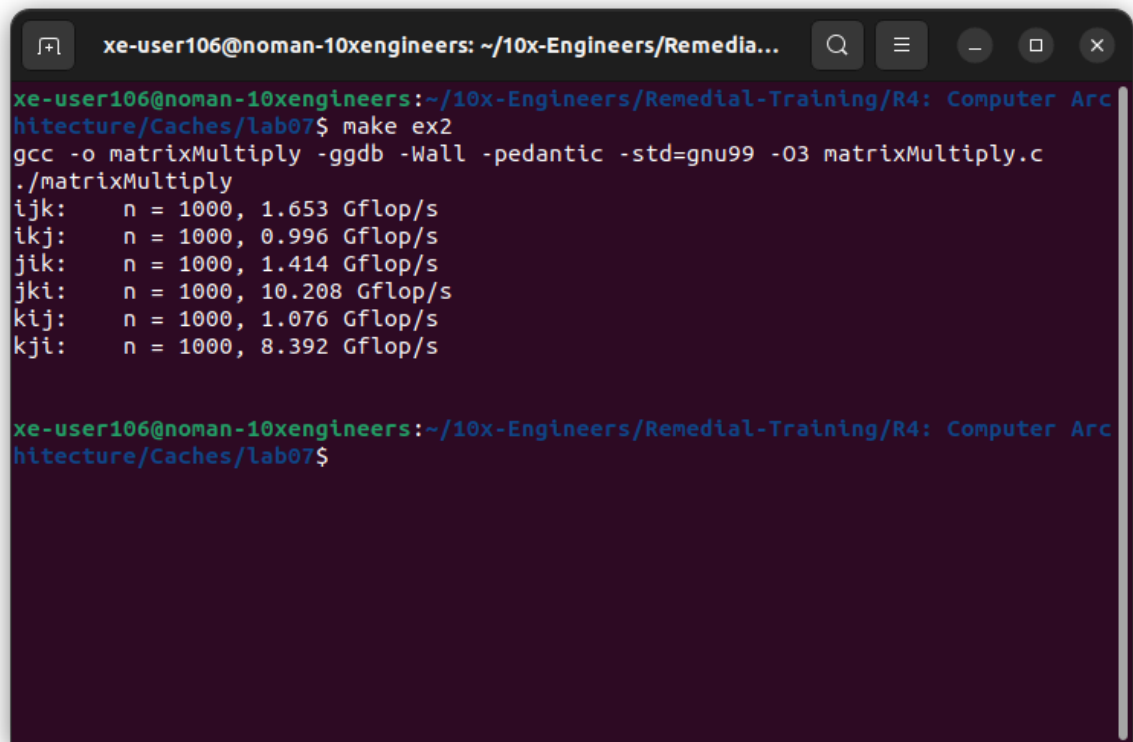
**5. Loop Order 5: *kij***

```
A = 0;  
B = n;  
C = n;
```

**6. Loop Order 1: *ijk***

```
A = 1;  
B = 0;  
C = 1;
```

- **Terminal Output:**



```
xe-user106@noman-10xengineers: ~/10x-Engineers/Remedia...  
xe-user106@noman-10xengineers:~/10x-Engineers/Remedial-Training/R4: Computer Arc  
hitecture/Caches/lab07$ make ex2  
gcc -o matrixMultiply -ggdb -Wall -pedantic -std=gnu99 -O3 matrixMultiply.c  
./matrixMultiply  
ijk:    n = 1000, 1.653 Gflop/s  
ikj:    n = 1000, 0.996 Gflop/s  
jik:    n = 1000, 1.414 Gflop/s  
jki:    n = 1000, 10.208 Gflop/s  
kij:    n = 1000, 1.076 Gflop/s  
kji:    n = 1000, 8.392 Gflop/s  
  
xe-user106@noman-10xengineers:~/10x-Engineers/Remedial-Training/R4: Computer Arc  
hitecture/Caches/lab07$
```

➤ **Questions:**

**1. Which 2 orderings perform best for these 1000-by-1000 matrices?**

The best 2 orderings are:

- i. **“jki”** (Loop Order 4)
- ii. **“kji”** (Loop Order 6)

**2. Which 2 orderings perform the worst?**

The worst 2 orderings are:

- iii. **“ikj”** (Loop Order 2)
- iv. **“kij”** (Loop Order 5)