ES215 - COA - Assignment-4

Name: Daniel Giftson E

Roll No: 20110051

Q3)

- From Bucket 1, I coded Matrix Multiplication in C++. (Github link to the code)
- I have hardcoded the value of N.
- I have made some minor changes to the code I submitted for Assignment-1 to easily work on this question. I have shared the modified code in the zip file.
- I have coded the three loops directly inside the main function rather than coding them as a separate function and calling it inside the main function.
- I have also coded the initialization of every element of the resultant matrix to 0 separately outside the meat part so that it doesn't interfere when I interchange the iterations.
- I have compiled and run this C++ program in this C++ Online Compiler.

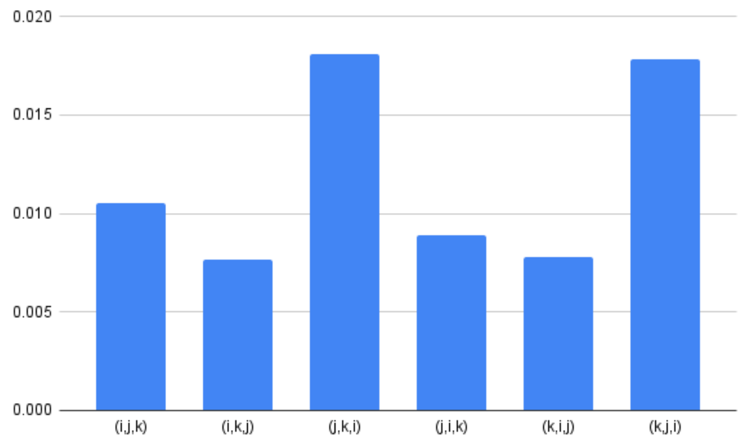
a)

- I have represented the 6 combinations of i,j,k as a tuple (i,j,k) where,
  i = outer loop variable
  j = middle loop variable
  k = inner loop variable

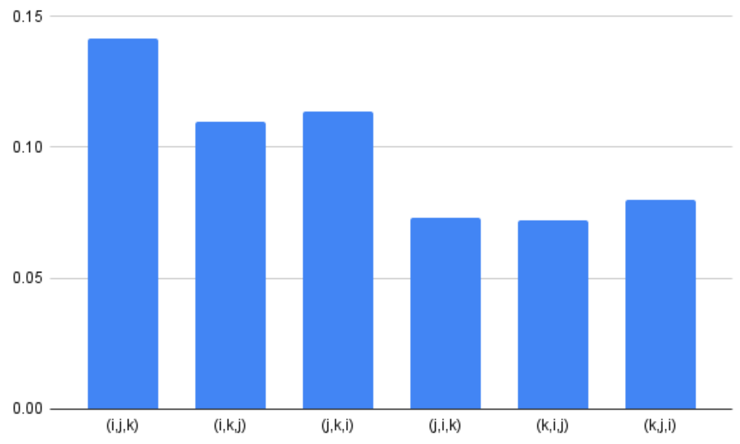
Loop Combination	N = 128	N = 256	N = 512
(i,j,k)	0.0105156 secs	0.141326 secs	2.31662 secs
(i,k,j)	0.00762578 secs	0.109583 secs	1.14169 secs
(j,k,i)	0..018084 secs	0.11343 secs	1.1334 secs
(j,i,k)	0.00887319 secs	0.0730229 secs	0.991935 secs
(k,i,j)	0.00781422 secs	0.0722071 secs	0.566298 secs
(k,j,i)	0.0178483 secs	0.0800174 secs	1.44849 secs

b)

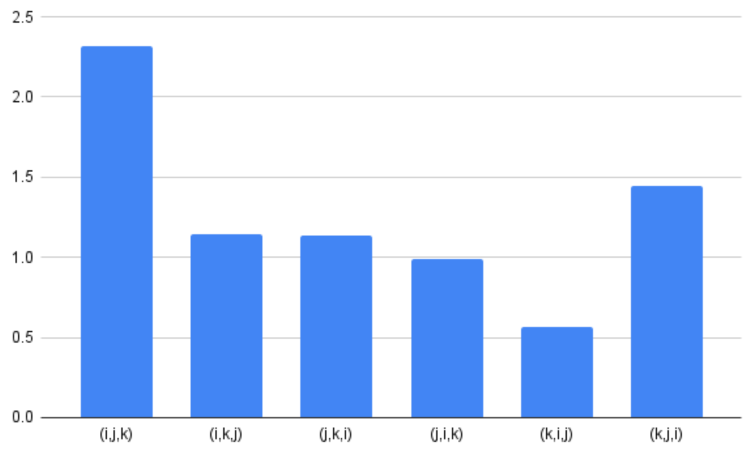
- **Plot of execution time for each iteration: (N = 128)**



- **Plot of execution time for each iteration: (N = 256)**



- **Plot of execution time for each iteration: (N = 512)**



Observations:

- We observe that the execution time increases for each iteration as the value of N increases.
- We also observe that the execution time for the (i,j,k) combination is greater than the other combinations (except for N = 128).
- We also observe that the (k, i, j) combination takes less execution time than the other combinations.

Inference:

- We conclude that as we interchange the position of the three loops, the execution time/performance doesn’t remain the same.
- We also found that the (k, i, j) iteration yields the best performance(less execution time) among all other iterations. This is because we are accessing the memory (1-D) in order w.r.t (i, j) for calculating **mat3** (Resultant matrix) (i.e., (0,0), (0,1), (0,2).......) and, in turn, all memory accesses are not the same because of caches. This means that the CPU will take less time to execute when memory is accessed in order.