CSE 5441
Harsh Gupta
(gupta.749@osu.edu)

Instructor: Dr. Jeffrey S. Jones
Grader: Karan Grover
Grader: Gregory Ochs

Question 1

N	C	Sizeof(element)	B	S	No. of elements in 1 cache line
256	64K	8	128	1	16

Examine value of k for inner loop analysis

	A	**B**	**C**
Miss Rate	**1/16**	**1**	**0**

A is accessed in contiguous fashion with stride = 1 resulting in 1 miss followed by 15 hits
B is accessed column wise (first element of each row) resulting in a miss always
C is accessed with only a single element, independent of k

Question 2

N	C	Sizeof(element)	B	E	No. of elements in 1 cache line
256	64K	8	64	4	8

Examine value of k for inner loop analysis

	A	**B**	**C**
Miss Rate	**1/8**	**1**	**0**

A is accessed in contiguous fashion with stride = 1 resulting in 1 miss followed by 7 hits
B is accessed column wise (first element of each row) resulting in a miss always
C is accessed with only a single element, independent of k

Question 3

N	C	Sizeof(element)	B	E	No. of elements in 1 cache line
256	64K	8	32	2	4

-1

Examine value of k for inner loop analysis

	A	**B**	**C**
Miss Rate	**1/4**	**1**	**0**

A is accessed in contiguous fashion with stride = 1 resulting in 1 miss followed by 3 hits
B is accessed column wise (first element of each row) resulting in a miss always
C is accessed with only a single element, independent of k

Question 4

```
for(I = 0; I < N; I++)
        for(K = 0; K < N; K++)
                for(J = 0; J < N; J++)
                        C[ I ] [ J ] += A[ I ][ K] x B[ K ][ J ]
```

	A	B	C
I	N	N	N
K	N/B	N	N
J	1	N/B	N/B
	N^2/B	N^3/B	N^3/B

Question 5

```
for(K = 0; K < N; K++)
        for(I = 0; I < N; I++)
                for(J = 0; J < N; J++)
                        C[ I ] [ J ] += A[ I ][ K] x B[ K ][ J ]
```

	A	B	C
K	N	N	N
I	N	N	N
J	1	N/B	N/B
	N^2	N^3/B	N^3/B

Question 6

```
for(J = 0; J < N; J++)
        for(K = 0; K < N; K++)
                for(I = 0; I < N; I++)
                        C[ I ] [ J ] += A[ I ][ K] x B[ K ][ J ]
```

	A	B	C
J	N	N	N
K	N	N	N
I	N	1	N
	N^3	N^2	N^3

Question 7

```
for(K = 0; K < N; K++)
        for(J = 0; J < N; J++)
                for(I = 0; I < N; I++)
                        C[ I ] [ J ] += A[ I ][ K] x B[ K ][ J ]
```

	A	B	C
K	N	N	N
J	N	N/B	N
I	N	1	N
	N^3	N^2/B	N^3

Question 8

-2

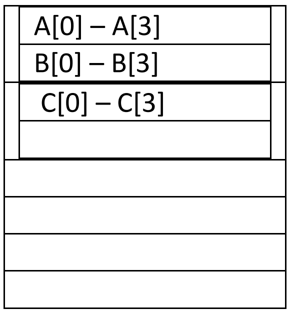
a. Access stride for each loop is 1 (because k is changing by 1 stride which makes C array to be accessed with a stride of 1, j is again changing by 1 stride which makes B array to be accessed with a stride of 1, and finally I is also changing by 1 stride making A array to be accessed with a stride of 1.

-2

b. Number of elements in each cache line is 4, hence for one single miss there will be 3 hits if all the elements are accessed in a way where all the 4 elements which is in a cache line gets accessed once.

Overall hit rate for a is 3/4
Overall hit rate for b is 3/4
Overall hit rate for c is 3/4

c.



After 251 iterations of k loop

A[0] – A[3]
B[0] – B[3]
C[0] – C[3]
C[4] – C[7]
…
C[248] – C[251]

Now at 252 iteration of k loop C will try to access set 0 which already has A and B in its lines, hence there will be thrashing for the next 4 iteration and again the cache contents of the set 0 will be there same and c will move forward

B[0] – B[3]
A[0] – A[3]
C[0] – C[3]
C[256] – C[259]
C[4] – C[7]
C[260] – C[263]
…
C[248] – C[251]
C[504] - C[507]

Again same thing will happen and set 0 will again have A and B. This ll continue for 1024 iteration of K. What we have observed in iteration of K is C moves in all the sets and A and B is accessed from only one set, and if there are other values of A and B by chance that will get evicted by C if they are not accessed in the current iteration.

Now lets see the scenario when I = 1023, J = 1023, k = 0

….
….
C[0] – C[3]
….
…
A[1020] – A[1023]
B[1020] – B[1023]

There will be c[0] to c[3] in set 1 and a and b in set 63. Now again C will be accessed 1024 times where it will pass through all sets and all lines and evict values which are not used recently, so we can say after 1024 iteration all sets except set 63 will have some c values without any contention. Set 63 will suffer thrashing but at the end it will have A and B.

Final contents after C loop going from 0 to 1023 according to me is

C[764] – C[767]
C[1020] – C[1023]
C[512] – C[515]
C[768] – C[771]
….
….
…
A[1020] – A[1023]
B[1020] – B[1023]

Question 9

a.

```
for(I = 0; I < 511; I++)
        for(J = 0; J < 511; J++)
                for(K = 0; K < 511; K++)
                        C[ I ] [ J ] += A[ J ][ K] - B[ K ][ I ]
```

	A	B	C
I	512 (N)	512 (N)	512 (N)
J	512 (N)	512 (N)	512/4 (N/B)
K	512/4 (N/B)	512 (N)	1
	$512^3/4$ (N^3/B)	512^3 (N^3)	$512^2/4$ (N^2/B)

b. No there is no better ordering. There are two reasons for it

   1. I have solved all other 5 combination and didn't find a better combination of I, J and K which will give a better answer. Here are the results of other 5 ordering

	A	B	C
J	N	N	N
I	N	N	N
K	N/B	N	1
	N^3/B	N^3	N^2

	A	B	C
J	N	N	N
K	N/B	N	N
I	1	N/B	N
	N^2/B	N^3/B	N^3

	A	B	C
K	N	N	N
J	N	N	N

I	1	N/B	N
	N^2	N^3/B	N^3

	A	B	C
K	N	N	N
I	N	N/B	N
J	N	1	N/B
	N^3	N^2/B	N^3/B

	A	B	C
I	N	N	N
K	N	N	N
J	N	1	N/B
	N^3	N^2	N^3/B

2. The intuition is as A, B and C are accessed in the pattern of J,K ; K,I ; and I,J respectively, hence if the two outer loop are in the same looping pattern then only there will be a better combination. As in 9 part a we have I, J as the outer loop combination and this is the way we are using it for array C, hence it is a the best ordering possible along with two other ordering which gives the similar result, i.e JKI and KIJ.