1. Suppose we have a 16MB (2^{24} bytes) memory that is byte addressable, and a 128KB (2^{17} bytes) cache with 64 (2^6) bytes per block.

a) How many total lines are in the cache?

b) If the cache is direct-mapped, how many cache lines could a specific memory block be mapped to?

c) If the cache is direct-mapped, what would be the format (tag bits, cache line bits, block offset bits) of the address? (Clearly indicate the number of bits in each)

d) If the cache is fully-associative, how many cache lines could a specific memory block be mapped to?

e) If the cache is fully-associative, what would be the format of the address?

f) If the cache is 4-way set associative, how many cache lines could a specific memory block be mapped to?

g) If the cache is 4-way set associative, how many sets would there be?

h) If the cache is 4-way set associative, what would be the format of the address?

2. Consider the following two sections of C code that both sum the elements of a 10,000 x 10,000 two-dimensional array M which contains floating points.

Code A	**Code B**

sum = 0.0;
for (r = 0; r < 10000; r++)
  for (c = 0; c < 10000; c++)
  sum = sum + M[r][c];
	sum = 0.0;
	for (c = 0; c < 10000; c++)
	  for (r = 0; r < 10000; r++)
	  sum = sum + M[r][c];

Explain why Code A takes 1.27 seconds while Code B takes 2.89 seconds. Hint: C uses row-major ordering to store two-dimensional arrays i.e.,