1. A system has an 8-way set associative 32KB L1 data cache with 64B cachelines.
   a. How many cachelines are in the L1 cache?

   $32KB = 32 \cdot 1024 B = \frac{32,768 B}{64B} = 512$ cache lines

   b. How many C/C++ double words can fit in each cacheline?

   $\frac{64B}{8B/word} = 8$ words

   c. Examine the following code. Suggest a modification that could improve the cache efficiency on this system and explain why.

```
double x[16][512], y[512];

for (int i = 0; i < 512; ++i)
   for (int j = 0; j < 16; ++j)
      y[i] = y[i] + x[j][i];
```

   X[16][513], y{513}
   Pad array to avoid power-of-2.

2. Examine the two matrix multiplication methods below and suggest which would be faster on a CPU with a cache. (This may sound redundant but some CPUs didn't have caches historically or even in modern times.) Assume the input matrices (A,B) are square with N rows and columns and N is a multiple of 4. Explain your answer.

a.
```
double A[N][N], B[N][N], C[N][N];
for (int i = 0; i < N; ++i)
   for (int j = 0; j < N; ++j)
   {
      C[i][j] = 0.0;
      for (int k = 0; k < N; ++k)
         C[i][j] = C[i][j] + A[i][k] * B[k][j];
   }
```

Read + Write N^3 times.

[✓ b.]
```
for (int i = 0; i < N; ++i)
{
   for (int j = 0; j < N; ++j) C[i][j] = 0.0;

   for (int k = 0; k < N; k = k+2)
   {
      double a0 = A[i][k  ];
      double a1 = A[i][k+1];

      for (int j = 0; j < N; ++j)
```

half the loop counts.

N^2

```
         {
            double cij = C[i][j]; — unit stride
            cij = cij + a0 * B[k  ][j];  unit stride
            cij = cij + a1 * B[k+1][j];
            C[i][j] = cij;
         }
      }
```