Assignment 5: Matrix Transpose using dynamic shared memory

Recall the shared memory implementations of matrix transpose in assignment 4. Implement a version of the code that uses dynamic shared memory instead of static shared memory. For designing a generic implementation, it makes sense to use dynamic shared memory rather than static shared memory where you have to redefine the macros for the shared memory size for every GPU architecture family. Dynamic shared memory can be specified as the third argument while launching the kernel from the host program. For example, the launch parameters can be as follows.

transposeCoalesced<<<grid, block ,sharedMemElements*sizeof(float)>>>

You can query the maximum size of the GPU shared memory and accordingly set up shared memory dynamically for any GPU architecture. Note for this case, the shared memory inside the kernel must be declared as an unsized 1D array as follows.

extern __shared__ float tile[];

Accordingly, implement your transpose operation using 1D Shared Memory Tiles with appropriate padding instead of standard 2D Shared Tiles with padding to avoid bank conflicts. Think carefully about what the dimensions of the shared memory should be and how it should be accessed.

Implement a CUDA program which takes as input:
i) the number of test cases, for each test case
ii) value of N
iii) N lines of floating-point values where each line contains N floating-point numbers separated by spaces.

This represents the NxN input matrix. Your task would be to generate random floating-point matrices of this order and test your implementation while adhering to the input format discussed above. The code should be general enough to handle large matrices. Assume that N is divisible by 32. Your program should print the transposed matrix as N lines of N space-separated floating-point values for each test case.
Again expect N to be of the order 512,1024,2048, etc. Your implementation should be generic enough to identify suitable grid block dimensions and shared memory dimensions as discussed in the previous assignment.