1. Implement RectMDArrayImpl.H corresponding to the header RectMDArray.H.

- RectMDArray(), RectMDArray(Box a_box): constructors.
- void define(Box a_box): defines an RectMDArray that has been default-constructed.
- RectMDArray(const RectMDArray<T,NC>& a_srcArray): copy constructor
- ~RectMDArray(): destructor.
- void setVal(const T& a_val): set all the elements of the RectMDArray to the constant value a_val.
- const T& operator[](const Point& a_pt) const: indexing operator - return a reference to \( A_p \) for the input Point \( p = a_pt \) (both const and non-const versions).
- const T& operator()(const Point& a_pt, int a_ic = 0) const: indexing operator - return a reference to \( A_p \) for the input Point \( p = a_pt \) for component \( a_ic \) (both const and non-const versions).

Note that const Box RectMDArray<T,NC>::getBox() const is already defined in RectMDArray.H.

You will put this file in the directory resources/homework1/src (the other files required to build your application are already there). To test your implementation, build mdArrayTest in the directory tt homework1/exec1 (the makefile and main are there) for both DIM = 2 and DIM = 3, and run for inputs 64 64 and 64 64 64, respectively. If your program is correct, the output convergence rate for the eigenvalue condition will be 2.0 . . .

2. In the directory homework1/exec2 create a new main program that uses Point Jacobi iteration to compute an approximation to the solution to Poisson’s equation, imitating the style in the example in homework1/exec1 (also, modify a copy of the makefile from that directory as appropriate). Test it for DIM = 2, and, when you think it is correct, build and run it for DIM = 3. Final runs should be for 64x64 in the 2D case (2000 iterations), and 32x32x32 in the 3D case (1000 iterations). The output should be the max norm of the residual written to stdout after the final iteration, as well as a plotfile suitable for viewing using Visit.

To turn in your homework, check into your repo a homework1 directory with src/ exec1/ exec2/ and /utilities populated with the necessary .cpp and .H files as well as the makefiles, but *only* those files. Do not check in .o, .exe, .d, or .vtk files.