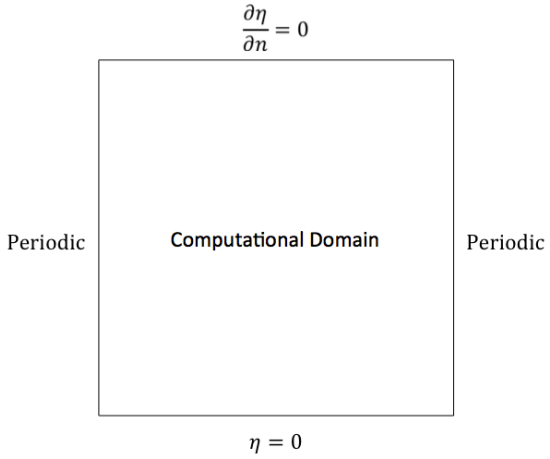
PRISMS-PF Training Exercises:

Here is a set of exercises to familiarize yourself with PRISMS-PF. Most users find that these problems take several hours to complete in a training environment where questions can be answered in real time. The problems are approximately in ascending order of difficulty. We recommend **copying and renaming the example app directories before making modifications** so that you still have the original versions to refer to. **Delete the file "CMakeCache.txt" in the newly created directory**. If the only required changes are to the parameters file (problems 1-2), then you can create a new parameters file with a different name in the original app directory.

1. **Boundary Conditions I:**
   Changing boundary conditions for the Allen-Cahn example problem
   a. Change the BCs in the Allen-Cahn application to zero flux (the natural BC) on the top boundary, eta = 0 on the bottom boundary (a Dirichlet BC), and periodic on the two side boundaries (see diagram below)



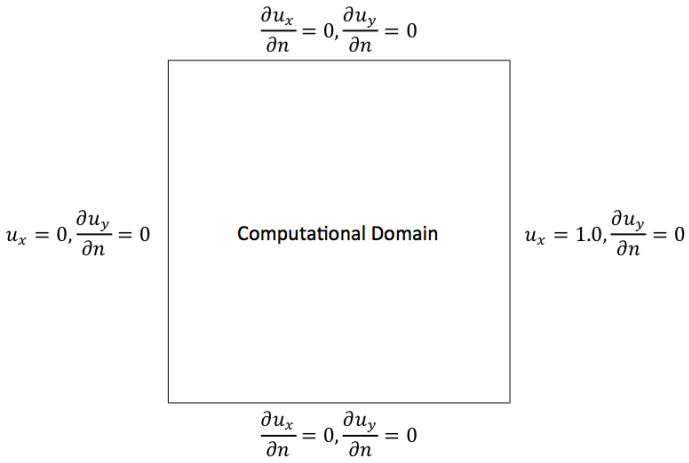
2. **Adaptivity:**
   Add adaptivity to the Allen-Cahn example problem
   a. Copy the adaptivity section of parameters_adaptive.prm from the `cahn_hilliard` application into the `allen_cahn` application, changing the variable that determines the mesh adaptivity from "c" to "n". See how much you can decrease the run time by using adaptivity, adjusting the parameters so that the solution doesn't change significantly. (**Note that the initial refinement should be between the max and min levels of refinement. This is clear if you run `./main-debug`)**
   b. Repeat the simulation, increasing the maximum refine factor by 1. For stability purposes, the time step will need to be decreased by a factor of 4.

3. **Postprocessing:**
   Use PRISMS-PF's postprocessed fields to verify that the integral of a field is conserved. This integral output is only done for postprocessed fields.
   a. Add a second postprocessing variable in the `cahn_hilliard` application for the concentration.
   b. Run a simulation and verify that the integral of the concentration is constant (the same value at every time step) by examining the terminal output.

4. **Initial Conditions and Computational Domain:**
   Adding a third particle to the `coupled_cahn_hilliard_allen_cahn` application
   a. Increase the size of the computational domain by 50% in each direction, using the same mesh size as in the original calculation (i.e. you will need to increase the total number of elements from 192 in each direction to 288).
   b. Add a third particle to the initial condition in a location of your choice that doesn't overlap with either of the other two particles. (Note: if the particles do overlap, the simulation will likely crash. However, you should be able to open the initial output in VisIt.)

5. **Model constants:**
   Add more model constants to an application
   a. In the `allen_cahn` application, the derivative of the free energy is hardcoded in equations.cc through the variable "fnV". The corresponding free energy expression is:
   $$f_{chem} = \eta^4 - 2\eta^3 + \eta^2$$
   b. Make the necessary modifications to `parameters.prm`, `equations.cc`, and `custom_pde.h` to turn this into a generic 4^{th} order polynomial with coefficients set in `parameters.prm` as more "Model constants".
   c. Also make the necessary changes to the postprocessed field so that the free energy expression is correct for total free energy.
   d. Note how the solution changes when the homogenous free energy is changed to:
   $$f_{chem} = 4\eta^4 - 8\eta^3 + 4\eta^2$$
   by changing the constants in `parameters.prm`

6. **Boundary Conditions II:**
   Strained precipitate evolution
   a. Run the un-modified `precipitate_evolution` example and move the output files into a new directory named "no_strain".
   b. Change the boundary conditions for the precipitate evolution application so that the x=40 boundary is displaced by +1 unit, and that the boundary condition on the x component of the displacement along the y=0 and y=40 boundaries are zero derivative (see diagram below). Move the output files to a new directory named "tension" and compare the results to the unstrained results.
   c. Repeat, but with the x=40 boundary displaced by -0.2 units, moving the output files to a new directory named "compression".
   d. Change the boundary condition on the x=40 to a Dirichlet BC that increases from 0 to +1 over the course of the simulation (Hints: use a `TimeDependentNonuniformDirichlet` boundary condition, and the current time can be accessed in non-uniform Dirichlet BC in function via the variable "time". This type of boundary condition must be set in ICs_and_BCs.cc)



7. **Governing Equations I:**
   Add a barrier term to the `precipitate_evolution` *application*
   a. Move two precipitates with different order parameters closer together and observe what happens when they touch (you may have to increase the total number of timesteps to observe this effect).
   b. Add a penalty term for overlapping structural order parameters. The barrier in the free energy is:

$$f_{barrier} = n_1^2 n_2^2 + n_1^2 n_3^2 + n_2^2 n_3^2 + n_1^2 n_2^2 n_3^2$$

      In each Allen-Cahn equation, this adds an extra term to each of the "eq_ni" expressions equal to $-\Delta t M_i \frac{\partial f_{barrier}}{\partial n_i}$, where "i" is the index of the structural order parameter. Run the simulation and observe the effect of adding the barrier term in the precipitates' morphology.
   c. Add the barrier term to the calculation of the total free energy. What is the effect of including this barrier in the total free energy (compare with the total energy from part a at the same timestep)?

8. **Governing Equations II:**
   Create a new application that simulates the growth of particle with a kinetically determined faceted shape. Similar models have been used for simulating selective area epitaxy and electrodeposition.
   a. Create a new application by copying the `cahn_Hilliard` application and renaming it `kinetic_wulff_shape`. Delete the "CMakeCache.txt" file.
   b. Change the initial condition to a circle with radius of 12 units in the center of the domain. Run a (short) simulation to check that the initial condition is correct.
   c. Add a source term to the "eq_c" term:
      `0.05*McV*this->get_timestep()*std::max(c*(1.0-c),constV(0.0))`.
      This source term will add mass at the interface, causing the initial particle to grow. Run a simulation to make sure that the particle is growing.
   d. Multiply the source term by an anisotropy coefficient, γ, where:

$$\gamma = 1 + 0.8 * \left[4 * \left(\left(\frac{\frac{\partial c}{\partial x}}{|\nabla c| + 1.0 \times 10^{-10}} \right)^4 + \left(\frac{\frac{\partial c}{\partial y}}{|\nabla c| + 1.0 \times 10^{-10}} \right)^4 \right) - 3 \right]$$

      Run the simulation to see how the orientation-dependent source term causes the morphology to change.