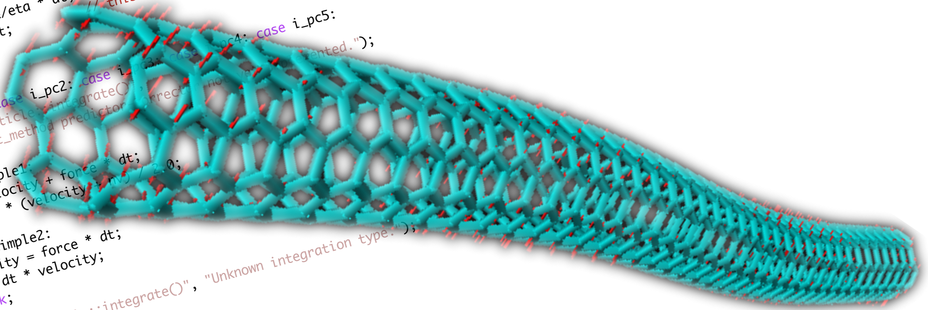

```
{
  xy_vect nv;
  xy_vect dx;
  switch (int_method) {
  case i_euler:  // or leap frog
    if( mass !=0) // standard MD, use leap frog
      {
        nv = velocity + force/mass * dt;
        dx = nv*dt;
      }
    else // langevin dynamics
      {
        dx = force/eta * dt; // this is the integration
        nv = dx/dt;          // this is only to compute velocity
      }

    break;
  case i_pc1: case i_pc2: case i_pc3: case i_pc4: case i_pc5:
    error("Particle::integrate()",
          "int_method predictor-corrector not yet implemented.");

    break;
  case i_simple1:
    nv = velocity + force * dt;
    dx = dt * (velocity + nv) / 2.0;
    break;
  case i_simple2:
    velocity = force * dt;
    dx = dt * velocity;
    break;
  default:
    error("Particle::integrate()", "Unknown integration type.");
  } // switch int_method
  coordinate = coordinate + dx;
  velocity = nv;
  f_hist.push(force);

  reset_force();
  return dx;
}

// apply periodic boundary conditions
int Particle::apply_bc(const Boundary_type& bt,
                       const double& Lx, const double& Ly)
{
  if (Lx <= 0.0 || Ly <= 0.0) {
```