```cpp
#include <iostream>
#include "io.h"
#include "analysis.h"
#include "container.h"
#include "potentials.h"
#include "countdown.h"
typedef pot_coulomb T_pairpot;          // Specify pair potential
#include "markovmove.h"

using namespace std;

int main() {
  cell::cell con(100.);                 // Use a spherical container
  canonical nvt;                        // Use the canonical ensemble
  pot_setup cfg;                        // Setup pair potential (default)
  interaction<T_pairpot> pot(cfg);      // Functions for interactions
  countdown<int> clock(10);             // Estimate simulation time
  macromolecule protein;                // Group for the protein
  ioaam aam(con);                       // Protein input file format is AAM
  protein.add( con, aam.load(
        "calbindin.aam" ) );            // Load protein from disk
  protein.move(con, -protein.cm);       // ..translate it to origo (0,0,0)
  protein.accept(con);                  // ..accept translation
  group salt;                           // Group for salt and counter ions
  salt.add( con, particle::NA, 34+19);  //   Insert sodium ions
  salt.add( con, particle::CL, 34 );    //   Insert chloride ions
  saltmove sm(nvt, con, pot);           // Class for salt movements
  aam.load(con, "confout.aam");         // Load old config (if present)
  chargereg tit(nvt,con,pot,salt,7.6);  // Prepare titration. pH 7.6
  systemenergy sys(pot.energy(con.p));  // System energy analysis
  cout << con.info() << tit.info();     // Some information

  for (int macro=1; macro<=10; macro++) {    // Markov chain
    for (int micro=1; micro<=1e3; micro++) {
      switch (rand() % 2) {                  // Randomly chose move
        case 0:
          sys+=sm.move(salt);                // Displace salt particles
          break;
        case 1:
          sys+=tit.titrateall();             // Titrate sites on the protein
          protein.charge(con.p);             // Re-calc. protein charge
          protein.dipole(con.p);             // Re-calc. dipole moment
          break;
      }
    }                                        // END of micro loop
    sys.update(pot.energy(con.p));           // Update system energy averages
    aam.save("confout.aam", con.p);          // Save configuration to disk
    cout << "Macro step " << macro
         << " completed. ETA: " << clock.eta(macro);
  }                                          // END of macro loop
  cout << sys.info() << sm.info()            // Print results
       << tit.info() << salt.info() << protein.info();
}
```