On windows system

Download and install

Putty and PsFTP
(http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html)

Xming (http://sourceforge.net/projects/xming/) and xming fonts
(http://en.sourceforge.jp/projects/sfnet_xming/downloads/Xming-fonts/7.5.0.34/Xming-fonts-7-5-0-34-setup.exe/)

VMD
(http://www.ks.uiuc.edu/Development/Download/download.cgi?PackageName=VMD)

Getting started

Open xming, this starts the X-Server

Open 'putty' -> For IP, type in 140.114.129.170. Click on SSH->X11, Check X11 Forwarding. Then 'Connect' , type in username ~~'ime-#'~~ 'ntu1', password is

~~Untar class.tar. ‘tar –xvf class.tar’~~

Go to 'SoftPhys' by typing in 'cd SoftPhys' and hit <enter>

Go into MD-NP. ‘cd MD-NP’. This is the folder with all the code files.

Go into TEST. ‘cd ../TEST’. This is the folder with input.dat and the test run results.

You will store your materials in a numerical folder '01', '02', '03' ... Choose one and go in 'cd 01'

Copy the input and executable files by
'cp ../MD-NP/md_np .'   and
'cp ../TEST/input.dat .'

The MD-NP code is used to study a mixture of one polymer and several nanoparticles

‘input.dat’ contains the input parameters for a simulation. Type ‘more input.dat’ to display the file. Type ‘gedit input.dat &’ to edit the file, ‘&’ means to run the program in the background or concurrently.

'./md_np . &'

The executable file is ‘md_np’. Type ~~‘./md_np &’~~ to execute the file in the background. ‘. ‘ means the file in the current directory. ‘..’ means the upper directory.

After the code is finished running, the results are written out to

‘config.xyz’ : particle configuration file for VMD

‘Rg.dat’ : data for the polymer, including polymer Rg, extension, and displacement

‘run_time.dat’: The actual run time of the program.

‘total_energy.dat’ : The total energy of the system, including potential and kinetic energy

To use vmd to read chain configurations, type ‘vmd’ and load ‘config.xyz’

The source code is in MD-NP

Type ‘more readme.txt’ to display the parameter information for input.dat

header.h : Defines the global parameters and global functions.
main.c : The main file that handles initialization, force calculation, trajectory integration, and data output
Init.c : Read in the parameters from input.dat, and initializes the particle positions
euler.c: Integrates the particle trajectories using euler or verlet algorithms
toterg.c : calls to evaluate all the energies and forces
enerbend.c: evaluates the bonded chain bending forces
ener_ev.c: evaluates the non-bonded interparticle forces(Lennard-Jones)
enervib.c: evaluates the bonded particleforces
ev_nlist.c: Use the neighbor list method to evaluate non-bonded interparticle forces
wall_force.c : evaluates forces from the wall boundaries
fileoutput.c : Analyze particle configurations to determine polymer size, displacement, and write output to files.

The potential between particles is defined in the file ‘ener_ev.c’ There are two routines ‘ener_ev’ and ‘ener_wca’. We can use ‘ener_wca’ to calculate the Lennard-Jones potential. In ‘ener_wca’, we define the WCA potential, which is the same as the Lennard-Jones potential but the attractive part of the potential is “cut-off”. We will modify this routine to return it to the Lennard-Jones potential.

Exercise 1: Create a code to calculate sin(x) and print out the results

Enter ‘gedit sinx.c &’

In the gedit window, type in

```
#include <stdio.h>                          <-  stdio is the standard input/output library of functions
#include <stdlib.h>                         <- stdlib is the standard library of functions
#include <math.h>                           <- math is the math library of functions

void main(void) {                           <- main is the first routine C compiler looks to execute
int i;                                      <- declare the variable i to be an integer
double dx;                                  <- declare the variable dx to be a real number to the 10^-32
                                               precision
dx = 0.1;                                   <- set the variable dx to 0.1

for(i=0; i<10; i++)                         <-  Do a loop from i=1 to i=10, print out the value of
  printf(“%lf %lf\n”, i*dx, sin(i*dx));          i*x and sin(i*dx).    %lf means the values  are real  numbers
}
```

Save the file and go to the terminal window

```
Enter ‘cc –o a.out sinx.c –lm’              <-   This compiles the code for machine execution
Enter ‘./a.out’                             <-    This executes the program you just wrote
```

Exercise 2: Modify the potential in ener_ev.c to make it the Lennard-Jones potential

Go into ‘MD-NP’

Enter ‘gedit ener_ev.c &’

In the gedit window, find the line “/*WCA potential */. Below this line, the routine calculates the purely repulsive part of the Lennard-Jones potential. This is known as the Weeks-Chandler-Andersen potential.

The WCA potential is the same as the Lennard-Jone potential, but it keeps only the purely repulsive part. For an interparticle distance r > rc, the WCA potential is set to zero. rc is known as a “cut-off” radius. You need to change the value to rc such that the L-J potential is calculated.

Find the line “rc=1.12*SIGMA_M” and change it to “rc = 5.0*SIGMA_M”
Find the line “eps = TEMP/sigma” and change it to “eps = EPS_M*TEMP/sigma”

Save the file and go back to the terminal window

Enter ‘make’ to recompile the code.

Now you can use ‘./md_np’ to execute

Recommendation:

Before you run the program, it is recommended that you make a new directory, for example ‘mkdir N100EPS1.0’, to denote the parameters ELL=100 and EPS_M=1.0.

Copy the file md_np and input.dat to the new directory
‘cp md_np N100EPS1.0’ and ‘cp input.dat N100EPS1.0’

Then go into the N100EPS1.0 directory to execute the file
‘cd N100EPS1.0’
‘./md_np’

This will allow you to keep the output files for different parameter sets separate.