In this exercise, we are mostly following Chapters 1-2 of "The Art of Molecular Dynamics Simulation" by D.C. Rapaport, 1995. The goal of the exercise is to generate a minimum viable product for performing molecular dynamics simulations. The model starts from a lattice configuration of spherical particles in 2D. We truncate the Lennard-Jones (6-12 potential), which represents the vdW interactions between neighboring atoms, and perform simple MD with this potential. We can enhance our understanding by simply plotting the basic potentials and make inferences before we run a simulation. However, with more complicated potentials this is not so straight forward. Additionally, there are other elements involved in accurate simulation which discussed in the book.

```
def u(rij):
    return 4*eps*((sig/rij))**12 - (sig/rij)**6)
```

A template is provided for one to generate their own potential energy functions.

```
def plot_potential(potential, span=(0, 2)):
    ...
```

A file containing input parameters to be passed as an argument to the MD engine is also written to save user input parameters.

We define a class to hold computations relating to molecular dynamics.

```
class MD(object):
    ...
```

This class carries out a series of operations to compute a single step of molecular (particle) dynamics which are as follows:

1. Define the potential energy function for all particles
2. Compute the pair-wise force generated between nearest-neighbor particles
3. Compute the pair-wise accelerations generated from the force computed in the previous step using the "leapfrog" method.

Once the forces are computed and the velocities / positions updated, we have completed a single step of molecular dynamics for our simple system. The general math is as follows for the above steps.

1.

$$U = -\int_{r_1}^{r_2} F(x)dx$$

Where, U is the potential energy and F is the force between particles.

2.

$$\frac{dU}{dx} = F(x)$$

We take the gradient, which is the derivative in this example to acquire $F(x)$.

3.

$$F_i = m_i a_i$$

We set the mass, m_i, equal to 1 so that the acceleration for the i^{th} particle, a_i, is equal to the force. See the $compute\_forces()$ function for details.

If you're interested, each function is provided with a documentation string which explains its purpose. This code is not optimized for production level data aquisition and analysis.