ParticleSimulator2

## Project description

This project aims to improve upon the lessons learned from ParticleSimulator1. The objective is to simulate in 2 dimensions, the kinetic and gravitational interactions of n spheroids in an optimized way.

## Objectives

### Primary objectives

* Separate code in header and source files
* Use classes instead of structs
* Use smart pointers. Each particle should only be defined once.
* Implement a quadtree (store the O(n^n) design for comparison)

### Secondary objectives

* Loading of objects/run, user interface to navigate options, using .csv instead of .txt
* Implement a simple helper function that can be wrapped around any function and prints run time to measure efficiency
* Allow for particle radius and particle mass to be individualized
* Separate computation and play-back of simulations

## Recycling from ParticleSimulator1

The elements below are to be recycled from ParticleSimulator1, in accordance with the primary and secondary objectives:

* Kinetic Collission function (including detection, backtracking and resolution)
* Complex object generation (sphere only is OK) (including storage of complex object)
* GNUplot plotting engine
* Kinetic energy, momentum, fps trackers

## Project structure

* Main.cpp file
* Header files:
  + ObjHandler.h : this will contain all functionality in generating, saving and unpacking complex objects
  + PhysEngine.h: this will initialize the simulation, run through the timesteps, update particle locations and save Particle information & meta information (fps etc) at the end of the simulation
  + ParticlePlotter.h: this will set up the plotting engine and plot particles for every timestep
  + Interfacer.h: this will read the input parameters, define scenarios and allow the user to choose which scenario to run.
  + Particles.h: this will define the individual, grouped and time-variant grouped particles and meta information per timestep.
  + Utils.h: this will contain all math heavy functions called upon by PhysEngine
* Source files:
  + ObjHandler.cpp
  + PhysEngine.cpp
  + ParticlePlotter.cpp
  + Interfacer.cpp
  + Particles.cpp
  + Utils.cpp