**Introduction to the Particle Simulation project**

The project is composed of the classes FilenamesReader, ProcessingInput, InitialiserClass, the abstract base class Particle with its children classes Photon, Neutrino, Electron, the abstract base class Directions with its children classes StochasticDirections and DeterministicDirections, the abstract base class Velocities with its base classes DeterministicVelocities, StochasticVelocities and the class Chamber.

The purpose of the class FilenamesReader is to read from the “Filenames.txt” the names of the files that contain the Directions, the Coordinates, the types of the particles and Terminal coordinates of the chamber where the particles are going to move.

The class ProcessingInput reads the data from those files and stores them as 1D or 2D vectors. It ensures that the number of directions for each particle is the same, the type of directions is the correct one (L,R,U,D only), the number of the chamber terminal coordinates is four( maximum x, minimum x,maximum y, minimum y) the number of particle types at each line is one, the correct type of particle is given( P, N or E), that the initial coordinates of the particle are integers, does not allow negative coordinates, that the number of entries(lines) for the coordinates, types and directions are the same and that the number of coordinates for each line is two.

The purpose of the InitialiserClass is to initialize the vector of pointers of the abstract base class particles that will be passed to the object of TheChamber class as well as the processing input class. The InitialiserClass takes as input references from the Velocities abstract class and the ProcessingInput class. It invokes the ProcessingInput operator() which is responsible for the duties of the ProcessingInput mentioned above. Form the Initialiser class we get back to our main function the initialized vector of the pointers of the Particle class, the Terminal coordinates of the chamber and the 2D char vector of the directions. When initializing the vector<Particle\*> we pass the Velocities object that we passed by reference to the ProcessingInput object. The Velocities object has itself a function which returns its own “this” pointer. We invoke that function in the InitialiserClass and subsequently we pass it to each of the elements of vector<Particle\*> dereferenced.

The Directions class is an abstract base class which has two children classes, the StochasticDirections and the DeterministicDirections. The DeterministicDirections accepts as an input the 2D char Directions vector generated by the ProcessingInput class (which in turn loaded the data from the Directions.txt file) and gives back the number of simulations that we are going to perform( which is the number of directions on each line in the Directions.txt file) as well as the set of directions corresponding to the particular simulation step. The StochasticDirections class accepts the 2D char Directions vector generates by the ProcessingInput class only to determine the number of Particles. We could also do the latter by a setter function in the Chamber class, then a getter function in Chamber class which could be passed back to the StochasticDirections class but we preferred the latter method as the Directions for a particle are not a controlled and self-tuned. The StochasticDirections generate each time they are invoked a vector of Directions at each simulation step.

The Velocities abstract class has two children classes, the Stochastic Velocities and the DeterministicVelocities. Each class has an operator which assigns to each chamber coordinate a velocity. The Velocities’ subclasses get to know the dimensions of the Chamber as follows. The Chamber has a function called “InteractingWithTheParticle”. There it passes to the Particles’ function “EnvironmentAssessment” its terminal coordinates. The latter function finds the number of rows and columns from those terminal coordinates and passes them to the Velocities’ operator. The DeterministicVelocities operator fills in each Chamber’s coordinate with determinist velocities( we put 1 for convenience), whereas the StochasticVelocities fills in each of the Chamber’s coordinates with stochastic velocities( we put 1+mod(rand(),4) for each coordinate).

The “TheParticle” abstract base class has three subclasses, the “Neutrino”, the “Electron” and the “Photon”. Apart from the “operator()” function which updates the Particles’ coordinates in the Chamber according to each Particle’s nature and the “EnvironmentAssessment” function described above, those classes have also the function “IdentifyYourself” which returns a string “P” or “N” or “E” for the Photon, the Neutrino and the Electron respectively. The constructor accepts as inputs the original xcoordinate, the original y coordinate and a reference to the Velocities abstract class.

Finally the Chamber class accepts as parameters the four terminal coordinates of the Chamber (its frames) as well as a vector of pointers of the abstract class “TheParticle” as well as a reference to the abstract class Directions. It prints to a file an image of the chamber at each simulation step as well as an explanation of what happens at each simulation step.