```
// Authors: Joseph Calles and Tharith Sovann

#include "Body.hpp" // include header file

int main(int argc, char* argv[])
{
  int N;
  double R;
  string holder, words;
  int years, months, days;

  getline(cin, holder); // get N
  N = stoi(holder);

  getline(cin, holder); // get R
  R = stod(holder);

  double total_time(stod(argv[1]));//T, the amount of seconds the simulation will run
  double delta_time(stod(argv[2]));//delta T, the step time. simulation stops when (delta T == T)
  double curr_time = 0;

  // initalise background image
  sf::Texture background_texture;
  if(!background_texture.loadFromFile("uni2.png"))
  {
    cout << "Error: could not load background image. . ." << endl;
  }
    // initalise window
  sf::RenderWindow window(sf::VideoMode((background_texture.getSize()).x, (background_texture.getSize()).y), "N-Body Simulation");
  window.setFramerateLimit(30);

  sf::Sprite background(background_texture);
  background.setScale(1.0, 1.0);

  /////initalise all objects in a vector of unique pointers
  std::vector<unique_ptr<Body>> bodies;
  for(int i = 0; i < N; i++)
  {
    unique_ptr<Body> body(new Body);   // declare
    (body)->set_radius(R);    // set radius
    (body)->set_big_G(6.67e-10);
    (body)->set_window_scale(window.getSize());
    // give window size for scaling
    cin >> (*body);                     // load information into object

    bodies.push_back(move(body));
  }
  /////SFML TIMER///////////////
  //  // FONT
  sf::Font font;
  if (!font.loadFromFile("Assistant-Regular.otf"))
  { cout << "Error: could not load font. . ." << endl; }

  // TEXT
  sf::Text text;
  text.setFont(font);
  text.setCharacterSize(20);
  text.setFillColor(sf::Color::White);
```

```cpp
 /////SFML AUDIO////////////////
 sf::Music music;
 if (!music.openFromFile("sound_track.ogg"))
   return -1; // error
 music.play();
//////SFML WINDOW//////////////////////////////
 while( window.isOpen() )
 {
   sf::Event event; // initalise event object

   while( window.pollEvent(event)  )
   {
     if (  (event.type == sf::Event::Closed) ||
          ((event.type == sf::Event::KeyPressed) &&
            (event.key.code == sf::Keyboard::Escape) ) )
       { window.close(); } // close if Esc is pressed or window closed
   }
       // refresh sequence
       window.clear();
       window.draw(background);   // re-draw background
       double x_f = 0;
       double y_f = 0;
   for(int i = 0; i < N; i++) //start calculating the force for all the bodies
   {
       for(int j = 0; j < N; j++){//all bodies besides itself affects others
           if(i != j){            //so each body's forces must be calculated in relation to others through this nested for loop
               x_f += (bodies.at(i))->calc_x_force(*bodies.at(j));
               y_f += (bodies.at(i))->calc_y_force(*bodies.at(j));

           }
       }
           (bodies.at(i))->set_x_force(x_f);
           (bodies.at(i))->set_y_force(y_f);
           x_f = 0;
           y_f = 0;
   //cout << *bodies.at(i);//print out the information of body at x step
   }
   //cout << endl;
   for(int i = 0; i < N; i++)
     {
      (bodies.at(i))->step(delta_time);
       window.draw(*(bodies.at(i)));
     }

   days = ((curr_time / 360)) / 24;
   months = days / 30.45;
   years = months / 12;

   words = "Elapsed time: "  +
           to_string(years)  + " years | "  +
           to_string(months) + " months | " +
           to_string(days)   + " days";

   text.setString(words);
   window.draw(text);

   window.display();
   curr_time += delta_time;
```

```cpp

    if(curr_time >= total_time) break;
  }
  cout << endl;
  for(int i = 0; i < N; i++){//print out the state of the universe at
        cout << *(bodies.at(i));//the end of the simulation
  }
  cout << endl;
  return 0;
}
```