```
 1: #include "Body.hpp"
 2: #include <cstdlib>
 3:
 4: Body::Body()
 5: {
 6:
 7: }
 8:
 9: Body::Body(double x, double y, double vel_x, double vel_y,
10:            double user_mass, std::string u_filename)
11: {
12:   //initialize variables
13:   xpos = x;
14:   ypos = y;
15:   velocity_x = vel_x;
16:   velocity_y = vel_y;
17:   mass = user_mass;
18:   filename = u_filename;
19:
20:   //Load image
21:   sf::Image image;
22:   if(!image.loadFromFile(filename))
23:     {
24:       exit(1);
25:     }
26:
27:   texture.loadFromImage(image);
28:   sprite.setTexture(texture);
29: }
30:
31: Body::~Body()
32: {
33:
34:
35: }
36:
37: double Body::get_xpos()
38: {
39:   return xpos;
40: }
41:
42: double Body::get_ypos()
43: {
44:   return ypos;
45: }
46:
47: void Body::set_radius(double rad)
48: {
49:   univ_rad = rad;
50: }
51:
52: void Body::set_window(int size)
53: {
54:   window_size = size;
55: }
56:
57: void Body::update_pixel_pos()
58: {
59:   double percent_x, percent_y;
60:
61:   percent_x = (xpos + univ_rad) / (2 * univ_rad);
```

```
   62:   percent_y = (ypos + univ_rad) / (2 * univ_rad);
   63:
   64:   sprite.setPosition((window_size * percent_x),
   65:                       ((window_size * percent_y)));
   66: }
   67:
   68: std::istream& operator >> (std::istream &input, Body &B)
   69: {
   70:   input >> B.xpos >> B.ypos >> B.velocity_x >> B.velocity_y >>
   71:     B.mass >> B.filename;
   72:
   73:   //Load image
   74:   sf::Image image;
   75:   if(!image.loadFromFile(B.filename))
   76:     {
   77:       exit(1);
   78:     }
   79:
   80:   B.texture.loadFromImage(image);
   81:   B.sprite.setTexture(B.texture);
   82:
   83:
   84:
   85:   return input;
   86: }
   87:
   88: void Body::draw(sf::RenderTarget &target, sf::RenderStates states) const
   89: {
   90:   target.draw(sprite, states);
   91: }
```