```
//
//  Body.cpp
//  ps3a
//
//  Created by Jingxian Shi on 2/19/18.
//  Copyright Â© 2018 Jingxian Shi. All rights reserved.
//

#include "Body.hpp"

Body::Body()
{

}

Body::Body(double xpos, double ypos, double xvel, double yvel, double mass, std::string file_name)
{
    _xpos = xpos;
    _ypos = ypos;
    _xvel = xvel;
    _yvel = yvel;
    _mass = mass;
    _image_file = file_name;

    _texture.loadFromFile(_image_file);
    _sprite.setTexture(_texture);

}

Body::~Body()
{

}

std::ostream& operator<<(std::ostream& out, Body& body)
{
    out << body._xpos << " "
    << body._ypos << " "
    << body._xvel << " "
    << body._yvel << " "
    << body._mass << " "
    << body._image_file;

    return out;
}

std::istream& operator>>(std::istream& in, Body& body)
{
    in >> body._xpos >> body._ypos >> body._xvel >> body._yvel >> body._mass >> body._image_file;
    body._texture.loadFromFile(body._image_file);
    body._sprite.setTexture(body._texture);

    return in;
}

void Body::draw(sf::RenderTarget& target, sf::RenderStates states) const
{
    sf::Sprite image = _sprite;

    image.setPosition(_window_size/2.0 + _xpos * _window_size * 0.5 / _univ_size, _window_size/2.0 - _ypos * _window_size * 0.5 / _univ_size);

    target.draw(image, states);
```

```cpp
}

void Body::step(double seconds, double force_x, double force_y)
{
    double accel_x = force_x / _mass;
    double accel_y = force_y / _mass;

    _xvel += accel_x * seconds;
    _yvel += accel_y * seconds;

    _xpos += _xvel * seconds;
    _ypos += _yvel * seconds;
}
```

```
//
//  Body.hpp
//  ps3a
//
//  Created by Jingxian Shi on 2/19/18.
//  Copyright Â© 2018 Jingxian Shi. All rights reserved.
//

#ifndef Body_hpp
#define Body_hpp

#include <stdio.h>
#include <iostream>
#include <string>
#include <SFML/Graphics.hpp>

class Body : public sf::Drawable
{
public:
    Body();
    Body(double xpos, double ypos, double xvel, double yvel, double mass, std::string file_name);
    ~Body();
    void setWindowSize(int window_size) {_window_size = window_size;}
    void setUnivSize(double univ_size) {_univ_size = univ_size;}
    double getUnivSize() {return _univ_size;}
    void step(double seconds, double force_x, double force_y);

    double getPosX() {return _xpos;}
    double getPosY() {return _ypos;}
    double getVelX() {return _xvel;}
    double getVelY() {return _yvel;}
    double getMass() {return _mass;}

    friend std::ostream& operator<<(std::ostream& out, Body& body);
    friend std::istream& operator>>(std::istream& in, Body& body);

private:
    virtual void draw(sf::RenderTarget& target, sf::RenderStates states) const;

    double _univ_size;
    int _window_size;
    double _xpos, _ypos, _xvel, _yvel, _mass;
    std::string _image_file;
    sf::Texture _texture;
    sf::Sprite _sprite;
};


#endif /* Body_hpp */

```