```
//
//  SPRINGDAMPER.cpp
//  SPRINGDAMPER
//
//  Created by Ben Stager on 5/11/21.
//

#include "SPRINGDAMPER.hpp"

springDamper::springDamper():springDamper(0.0, 0.0, 0.0, 0.0, 0.0){
}

springDamper::springDamper(double y0, double v0, double mass, double k, double
c):mass(mass), k(k), c(c){
        Y[0] = y0;
        Y[1] = v0;
}

springDamper::springDamper(const springDamper& p) : mass(p.getMass()),
k(p.getSpring()), c(p.getDamper()){
        Y[0] = p.getPosition();
        Y[1] = p.getVelocity();
}


void springDamper::operator()(const boost::array<double, 2>& y,
boost::array<double, 2>& yprime, const double runningTime) {
        yprime[0] = y[1];
        yprime[1] = (-getSpring() / getMass() * y[0]) + (-getDamper() / getMass() *
y[1]) - gravity;
        Y = y;
}

std::ostream& operator << (std::ostream& strm, const springDamper& ff){
        strm << ff.Y[0] << " " << ff.Y[1];
        return strm;
}


boost::array<double, 2>& springDamper::vector(){
        return Y;
}


double springDamper::getPosition() const{
        return Y[0];
}


double springDamper::getVelocity() const{
        return Y[1];
}


double springDamper::getMass() const{
        return mass;
}
```

```
double springDamper::getSpring() const{
        return k;
}


double springDamper::getDamper() const{
        return c;
}


void springDamper::setPosition(const double newy0){
        Y[0] = newy0;
}


void springDamper::setVelocity(const double newv0){
        Y[1] = newv0;
}


void springDamper::setMass(const double newMass){
        mass = newMass;
}


void springDamper::setSpring(const double newSpring){
        k = newSpring;
}


void springDamper::setDamper(const double newDamper){
        c = newDamper;
}
```