```
#include <stdlib.h>
#include <GLut/glut.h>
#include <math.h>
#include <iostream>
#include <fstream>

using namespace std;

//variables
double dt=0.01; // this is the time step
float verysmall=0.001;
float totaltime=0.0;
float gravity = 1.0;
GLfloat near=0.1,far=15.0;
GLfloat startdisp=-1.1;
GLfloat ogrepos=7.5;
GLfloat eyepos=7.5;
GLfloat castleradius=0.5;

class ball;
bool eyedef = false;
//ogre class
class ogre{
private:
  double px,py,pz,vx,vz,angle;

public:
       bool show;
  ogre(){
         px=0;py=0;pz=ogrepos; vx = 0.01; vz=-0.01;
         show=true;}
  double getx(){return px;}
  double gety(){return py;}
  double getz(){return pz;}

  void reset() {
         donald.show = true;
              px = 0;
              pz = 0;
              pz=ogrepos;
              vx = 0.01;
              vz=-0.01;

       }

  bool collision(ball a);

  void evolve(){angle+=0.1;
  //movement towards player
if (px < -0.4 || px > 0.4)
       {vx *= -0.5;}
              px += vx;
              pz += vz;
if (pz <= 0.0)// reset ogre if hits player
         {
          cout << "Ogre Donald Hit Player\n";
          donald.reset();
         }
   }
} donald;

//ogre2 class
```

```
class ogre2{
private:
  double px,py,pz,vx,vz,angle;
public:
       bool show2;
  ogre2(){
         px=0/2;py=0/2;pz=ogrepos; vx = 0.01; vz=-0.01;
         show2=true;}
  double getx(){return px;}
  double gety(){return py;}
  double getz(){return pz;}

  void reset() {
         doreen.show2 = true;
              px = 0/2;
              pz = 0/2;
              pz=ogrepos;
              vx = 0.01;
              vz=-0.01;
       }

  bool collision(ball a);

  void evolve(){angle+=0.1;
    //movement towards player
       if (px < -0.2 || px > 0.2)
       {vx *= -0.5;}

              px -= vx;
              pz += vz/2;

  if (pz <= 0.0)// reset ogre if hits player
         {
          cout << "Ogre Doreen Hit Player\n";
          doreen.reset();
         }
   }
} doreen;

 class eye {
private:
       double px,py,pz,angle,vx;
       bool hitEye;

       public:
       double getpx(){return px;}
       double getpy(){return py;}
       double getpz(){return pz;}
       bool gethit(){return hitEye;}
       eye() {px = 0/2; py = 1.0; pz = eyepos;}

       bool collision(ball a);
                void evolve()
                       //movement
                {angle+=0.1;
   pz=ogrepos+-0.05*castleradius*sin(angle);
    px=0.05*castleradius*cos(angle);

   }
} Eyelad;
```

```
class ball{
       private:
       // ax acceleration in x, px position in x, vx velocity in x
       double ax,ay,az,px,py,pz,vx,vy,vz,size,speed,anglexz,angleyz,initialspeed;
       private:  bool ontee;
       public:
              ball()
                     { ontee=true;    px=0;py=0;pz=0;size=0.05;
vx=0;vy=0;vz=0;ay=gravity;speed=0;
                            anglexz=0; angleyz=0; initialspeed=2.0;
                     }

              double getpx(){return px;}double getpy(){return py;};double
getpz(){return pz;};
              double getvx(){return vx;};  double getvy(){return vy;};  double
getvz(){return vz;};
              double getballsize(){return size;};

              void fire(){
              if(ontee==false){return;}
              ontee=false;
              // convert information into vx vy and vz
              double pi=3.14;

              vy=initialspeed*sin((2*pi*angleyz)/360);
           vz=initialspeed*cos((2*pi*angleyz/360))*cos((2*pi*anglexz/360));
              vx=initialspeed*cos((2*pi*angleyz/360))*sin((2*pi*anglexz/360));


              }

              double getspeed(){return  sqrt(py*py+px*px+pz*pz);}
              void alterspeed(double xx)
              {if(ontee==false)return;
              initialspeed+=xx;
              if(initialspeed < 0)initialspeed=0;
              }

              void rotateinxzplane(double xx){if(ontee==false)return;anglexz+=xx;}
              void rotateinyzplane(double xx){if(ontee==false)return;angleyz+=xx;}

              void reset(){
              ontee=true;
              //if ((px=0) || (py=0) || (pz=0.1)|| (pz=-0.1)){

                  px=0;py=0;pz=0;size=0.05;
              vx=0;vy=0;vz=0;ay=gravity;initialspeed=2.0;
              anglexz=0; angleyz=0;
              //}
              }

              void info(){
              if(ontee==false)return;
              cout << "speed=" << initialspeed  << " anglexz=" << anglexz << "
angleyz=" << angleyz << "\n";
              double vya=initialspeed*sin(angleyz);
              double vxa=initialspeed*cos(angleyz)*cos(anglexz);
              double vza=initialspeed*cos(angleyz)*sin(anglexz);
              cout << "corresponds to " << "vx="  << vxa  << " vy=" <<  vya << " vz="
<< vza << "\n";
              cout << "---------------------------------\n";
              }
```

```
  void evolve(){
    if(ontee==true){return;}
    // cout << "vy" << vy << "\n";
      vy+=(-gravity*dt);
      px+=vx*dt;
         py+=vy*dt;
         pz+=vz*dt;
      totaltime+=dt;
              if(py < -0.0005)
                     {
                  py=0;
                     vy=-0.7*vy;
                     }
                            if (pz > far)//Check if ball has reached far plane
                            {

                                   //vz-=dt;//bounce back
                                   golfball.reset();//reset golfball
                            }
              }

       } golfball;

//collision between ogre and ball
bool ogre::collision(ball a){
         double x1=a.getpx();
         double y1=a.getpy();
         double z1=a.getpz();
         double d = sqrt( (px-x1)*(px-x1)+(py-y1)*(py-y1)+(pz-z1)*(pz-z1) );
         //  cout  << " ogre  "<<  px  <<" " << py << " " << pz << "\n";
         // cout  << " cannonball  "<<  x1  <<" " << y1 << " " << z1 << "\n";
         //  cout << "Distance between them" << d << "\n";
         if(d < 0.1)
              {
                            cout << "Hit!\n";
                            eyedef = true; //deformation with eye
                            show = false; // ogre killed
                            golfball.reset();//reset golfball
                            donald.reset();//reset donald
              return true;
              }
       return false;
       }

bool ogre2::collision(ball a){
         double x1=a.getpx();
         double y1=a.getpy();
         double z1=a.getpz();
         double d = sqrt( (px-x1)*(px-x1)+(py-y1)*(py-y1)+(pz-z1)*(pz-z1) );
         //  cout  << " ogre  "<<  px  <<" " << py << " " << pz << "\n";
         // cout  << " cannonball  "<<  x1  <<" " << y1 << " " << z1 << "\n";
         //  cout << "Distance between them" << d << "\n";
         if(d < 0.1)
              {
                            cout << "Hit!\n";
                            eyedef = true; //deformation with eye
                            show2 = false; // ogre killed
                            golfball.reset();//reset golfball
                            doreen.reset();//reset doreen
              return true;
              }
```

```
        return false;
        }

//collision between eye and ball
bool eye::collision(ball a){
         double x1=a.getpx();
         double y1=a.getpy();
         double z1=a.getpz();
         double d = sqrt( (px-x1)*(px-x1)+(py-y1)*(py-y1)+(pz-z1)*(pz-z1) );
         //  cout  << " ogre  "<<  px  <<" " << py << " " << pz << "\n";
         // cout  << " cannonball  "<<  x1  <<" " << y1 << " " << z1 << "\n";
         //  cout << "Distance between them" << d << "\n";
         if(d < 0.1)
              {
                              cout << "hit eye\n";
                              eyedef = true;//deformation of eye
                              //show = false;
              return true;
              }
        return false;

        }



void keyboard(unsigned char key, int x, int y)
{
   switch (key)
        {
        case 'd':// rotate right
                  golfball.rotateinxzplane(5.0);break;
        case 'a': // rotate left
          golfball.rotateinxzplane(-5.0);break;
        case 'w': // rotate up
            golfball.rotateinyzplane(5.0);break;
        case 'x': // rotate down
          golfball.rotateinyzplane(-5.0);break;
        case 'f':   //fire
          golfball.fire(); break;
        case 'r': // reset
          golfball.reset(); break;
         case 'o':
          golfball.alterspeed(0.1); break;
        case 'p':
          golfball.alterspeed(-0.1); break;
        case 'i':
          golfball.info(); break;
           break;
              case 'q':exit(0);
     }
   glutPostRedisplay();
}

void init(void){glClearColor(1.0,1.0,1.0,0.0);}

void display(void)
{

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(40,1,near,far);
       glMatrixMode(GL_MODELVIEW);
```

```
       glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    glClear(GL_COLOR_BUFFER_BIT);

       //draw cannon
       glPushMatrix();

       glColor3f (0.5, 0.6, 1.0);
       glTranslatef(0.0,-0.05,-1.0);
       glRotatef(45, 1.0, 0.0, 0.0);
       glutSolidSphere(0.05,5,20);
       glPopMatrix();

       //draw cannonball
       glPushMatrix();
       glColor3f (0.0, 0.0, 0.0);
    glTranslatef(golfball.getpx(),golfball.getpy(),startdisp-golfball.getpz());
    glutSolidSphere(golfball.getballsize(),20,20);
    glTranslatef(-1.0*golfball.getpx(),-1.0*golfball.getpy(),-1*(startdisp-
golfball.getpz()));
       glPopMatrix();

       //draw eye
       glPushMatrix();
       glColor3f (1.0, 0.0, 0.0);
       if (eyedef == true)// || (ogreeyedef == true))//deformation of eye if ogre is
hit
       {
              glColor3f (0.5, 0.0, 0.0);
       }

       glTranslatef(Eyelad.getpx(),Eyelad.getpy(),startdisp-Eyelad.getpz());

       glutSolidSphere(0.1,20,20);
       glColor3f (1.0, 1.0, 0.0);
       glTranslatef(0.0, 0.0, 0.2);
       glutSolidSphere(0.05,4,20);
       glTranslatef(-Eyelad.getpx(),-Eyelad.getpy(),-(startdisp-Eyelad.getpz()));
       glPopMatrix();

       //draw ogre donald
       glPushMatrix();
       if(donald.show == true) // only draw if ogre has not been hit
       {
       glColor3f (0.0, 1.0, 0.0);

    glTranslatef(donald.getx(),donald.gety(),startdisp-donald.getz());
    glutSolidSphere(0.1,20,20);

       glColor3f (0.0, 0.0, 0.5);
       glTranslatef(0.0, 0.13, 0.0);
       glutSolidSphere(0.05,20,20);
    glTranslatef(-donald.getx(),-donald.gety(),-(startdisp-donald.getz()));
       }
glPopMatrix();

       //draw ogre doreen
glPushMatrix();
       if(doreen.show2 == true) // only draw if ogre has not been hit
       {
       glColor3f (0.0, 1.0, 0.0);
```

```
    glTranslatef(doreen.getx(),doreen.gety(),startdisp-doreen.getz());
    glutSolidSphere(0.1,20,20);

       glColor3f (0.5, 0.0, 0.5);
       glTranslatef(0.0, 0.13, 0.0);
       glutSolidSphere(0.05,20,20);
    glTranslatef(doreen.getx(),doreen.gety(),-(2*startdisp-doreen.getz()));
       }
glPopMatrix();


glFlush();
glutSwapBuffers();
}

void evolve(int a)
{
  // evolve system here
     golfball.evolve();
        donald.evolve();
        doreen.evolve();
        Eyelad.evolve();

     if(donald.collision(golfball)==true)
       {
         cout << "Ogre Donald dead!!!!!!!!!!!!!!!! \n";
      }
        if(doreen.collision(golfball)==true)
       {
         cout << "Ogre Doreen dead!!!!!!!!!!!!!!!! \n";
      }
        if(Eyelad.collision(golfball)==true)
       {
         cout << "Hit Eye!!!!!!!!!!!!!!!! \n";
      }
glutPostRedisplay();
glutTimerFunc(20,evolve,20);
}

int main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB);

   glutInitWindowSize (800, 800);
   glutCreateWindow (argv[0]);   init();
   glutKeyboardFunc (keyboard);
   glEnable(GL_DEPTH_TEST);
   //  glutIdleFunc(evolve);
   glutTimerFunc(100,evolve,100);
   glutDisplayFunc (display);
   cout << "This program models projectile motion, the user picks the direction and
speed of the Cannonball\n";
   cout << "Note the following \n";
   cout << "`a' rotates left , `d' rotates right\n";
   cout << "`w' rotates up, `x' rotates down\n";
   cout << "`o' increases speed of shot, `p' decreases speed of shot\n";
   cout << "`i' info \n";
   cout << "`r' reset \n";
   cout << "`f' fire \n";
   glutMainLoop();
   return 0;
```

```
}
```