Applied Physics - Practical 4

To be completed by November 30th

**Purpose:** Using SFML with C++ allow a circle to move up and down the y-axis under the influence of gravity only, after an initial impulsive force is applied to the sphere.

**To Do:**

1. Using the code given belowcreate and SFML projectand get the project running.
2. Draw a line as a plane.
3. When the space-bar is hit, move the up the y-direction as if it was hit by an impulsive force that imparts an initial velocity of (0, **u**, 0) m s-1 to the sphere in the y-direction only. Allow it to go up and then fall down again under gravity until it hits the plane. It should then stop and rest on the plane.
4. Choose where to put the ground level on screen.
5. Try various different values for the initial speed **u** until the base of the sphere just reaches a height of 100 metres ( 1 meter equals one pixel) above the ground, at its highest point.
6. Output the time taken as the sphere ( the sphere is of size one pixel) goes up and falls back down again, stop the time when it hits the ground. Also output the maximum height that the **base of the sphere** reaches above the ground. Note that the time taken for the box to reach the ground should be  seconds and its maximum height should be , check that your output agrees with these.
7. You should output predicted time and max height as well as actual time and max height. For writing out text, fonts can be found on <http://www.dafont.com/>
8. Amend the project and create a PhysicsObject class which stores the velocity and position of the object and implement an Update(sf::Time t) which updates the position and velocity as below.

Note : To update the position and velocity of the object simply apply the two equations from physics:

=  + t

. . . . . . . . . . . . **(next velocity)**

and

=  + t + t2

. . . . . . . . **(next position)**

In these two equations the acceleration  is constant, defined by:

sf::Vector2f gravity(0.0f, 9.8f);

Why is gravity positive not the usual -9.81 ?

To determine in code how an object moves, use:

**Position = Position + Velocity\*timeChange + 0.5\*acceleration\*(timeChange)2**

**Velocity = Velocity + acceleration\* timeChange**

Code:

#include <SFML/Graphics.hpp>

int main()

{

sf::RenderWindow window(sf::VideoMode(800, 800), "Go Physics!!");

sf::CircleShape shape(0.5f);

shape.setFillColor(sf::Color::Green);

sf::Vector2f velocity(0, 0);

sf::Vector2f position(400,400);

sf::Vector2f gravity(0.0f, 9.8f);

sf::Clock clock;

const float FPS = 60.0f;

const sf::Time timePerFrame = sf::seconds(1.0f / 60.0f);

sf::Time timeSinceLastUpdate = sf::Time::Zero;

clock.restart();

while (window.isOpen())

{

//read keyboard inout

sf::Event event;

while (window.pollEvent(event))

{

if (event.type == sf::Event::Closed)

window.close();

}

//get the time since last update and restart the clock

timeSinceLastUpdate += clock.restart();

//update every 60th of a second

if (timeSinceLastUpdate > timePerFrame)

{

window.clear();

// update position and velocity here using equations in lab sheet using timeChange as timeSinceLastUpdate.asSeconds().

//update shape on screen

shape.setPosition(position);

window.draw(shape);

window.display();

timeSinceLastUpdate = sf::Time::Zero;

}

}

return 0;

}