Homework 2

Projectile motion

**Level I**

Suppose you throw a 4-kg ball at an angle, so it’s not simple free fall. Let’s say you throw it in the x-z direction with a speed of 5 m/s at an angle of 45 degrees with the horizontal. (Use the z direction for up.) (Note: all trigonometric functions in all major programming languages use radians.) Write a program that prints out the position, velocity, and acceleration of the ball at each point in time. The ball should stop when it reaches the ground again. Make sure your time increment is small enough so that you get a smooth curve. Ignore air resistance. You should print out all three components of each of these plus the magnitude, so that the first row of the table looks something like this:



**Level II**

Now we have to put air resistance back in. Remember that the air resistance is proportional to the *speed* of the ball squared, and that it points in the opposite direction of its velocity. Make sure you add it to the force of gravity before you calculate the overall acceleration. Use a drag coefficient of 0.5 kg/m and a mass of 4 kg as before.

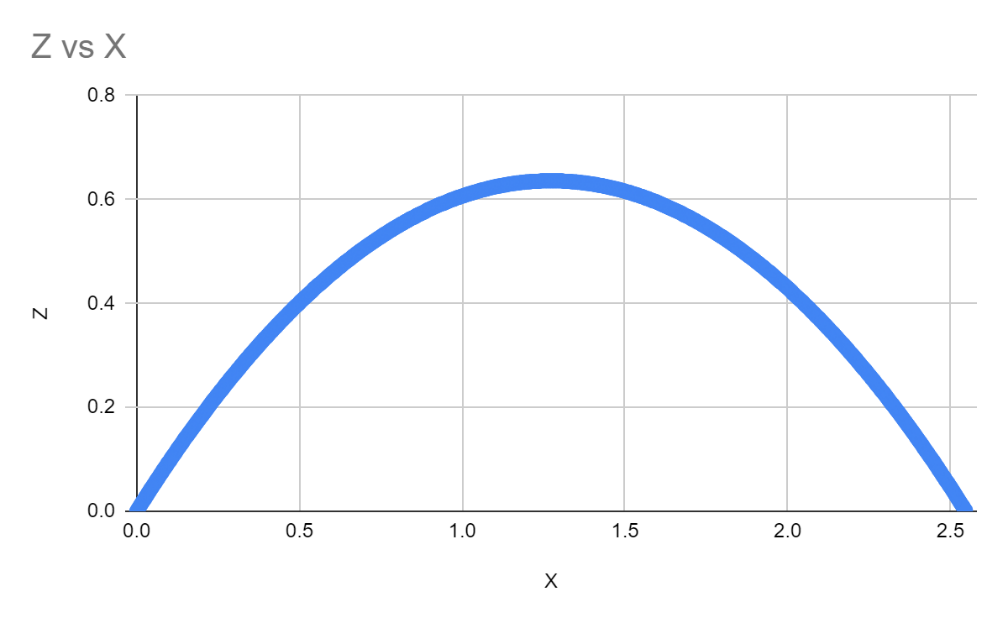
**Level III**

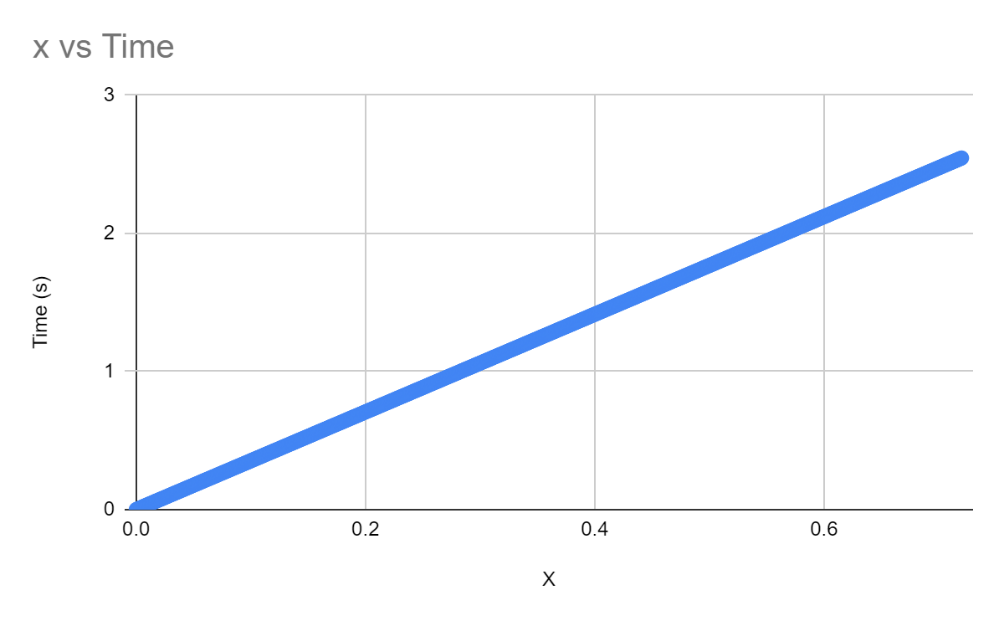
Now imagine a ball suspended from a spring in midair. Put one end of the spring at the origin, and make its unstretched length 2 m and its spring constant 8 kg/s2. Start the ball at position (-1, 1, -1) m and give it an initial velocity of (5, -1, 3) m/s. Make sure to keep gravity and air resistance in. How much time does it take the ball to drop and stay below a speed of 1 m/s? That is, what is the time of the last time the ball reaches 1 m/s?

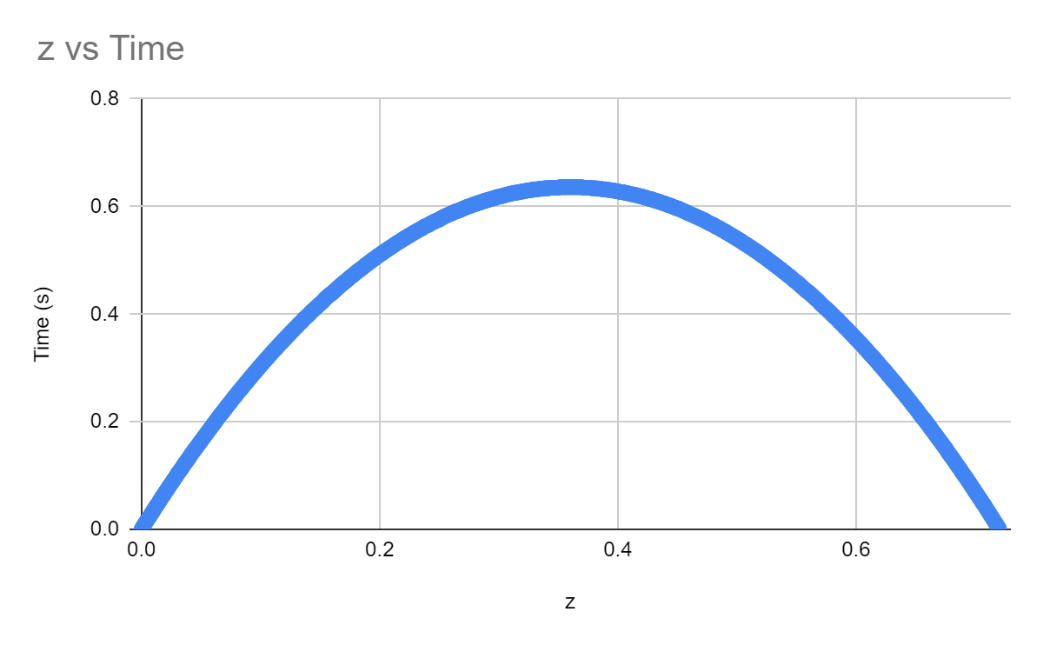
Homework 2 Answer Sheet

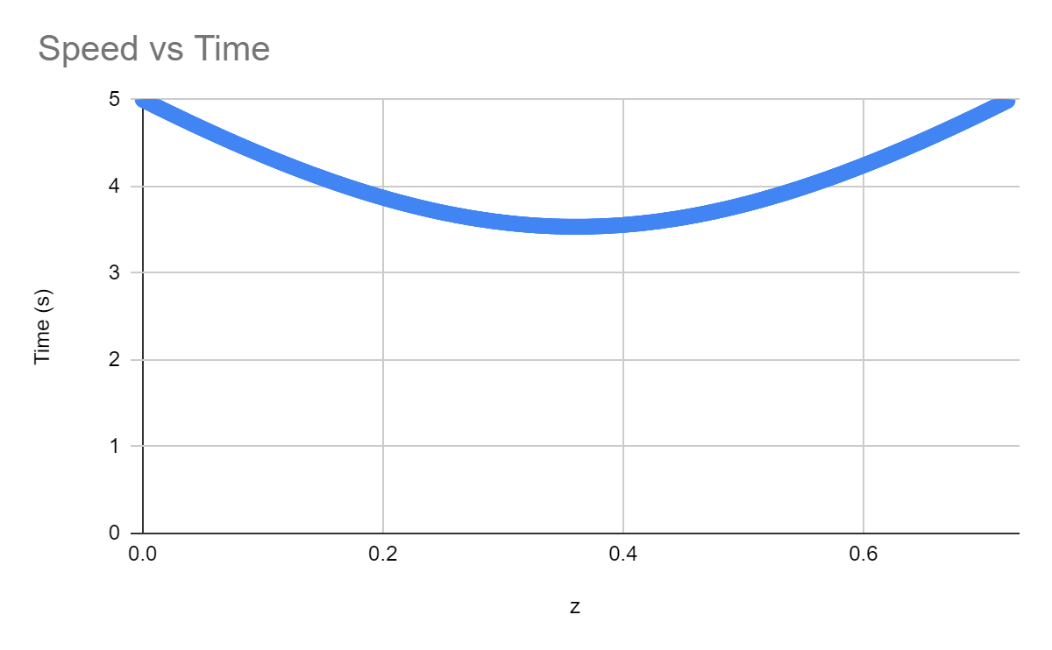
**Level I**

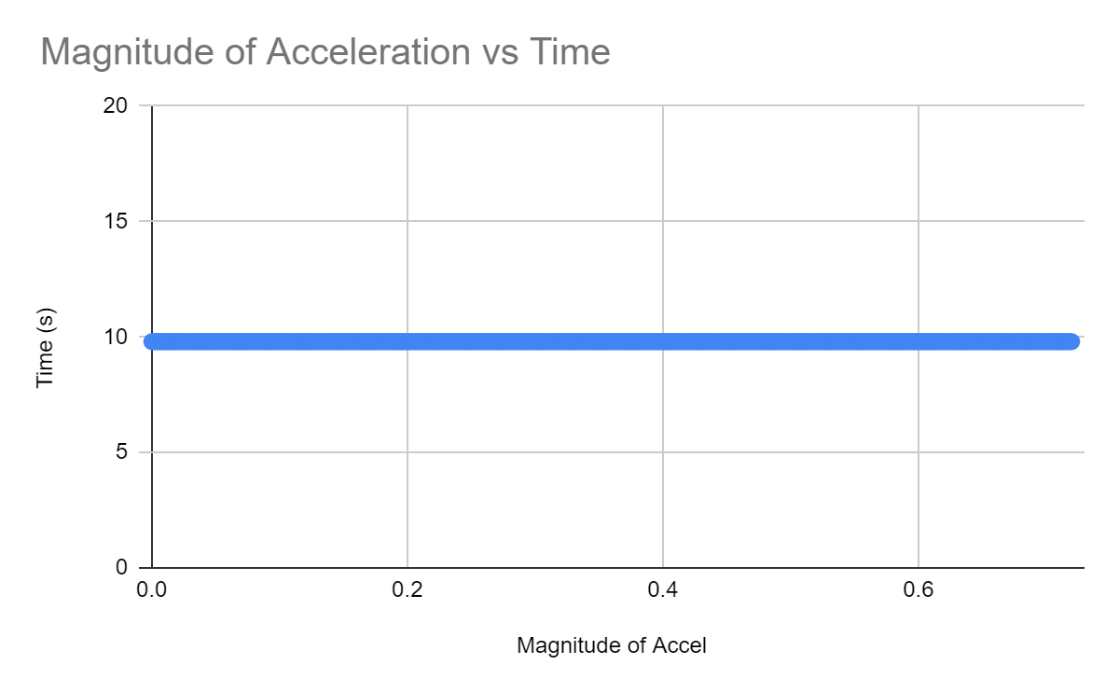
Insert five graphs: z vs. x, x vs. time, z vs. time, speed vs. time, and the magnitude of the acceleration vs. time.





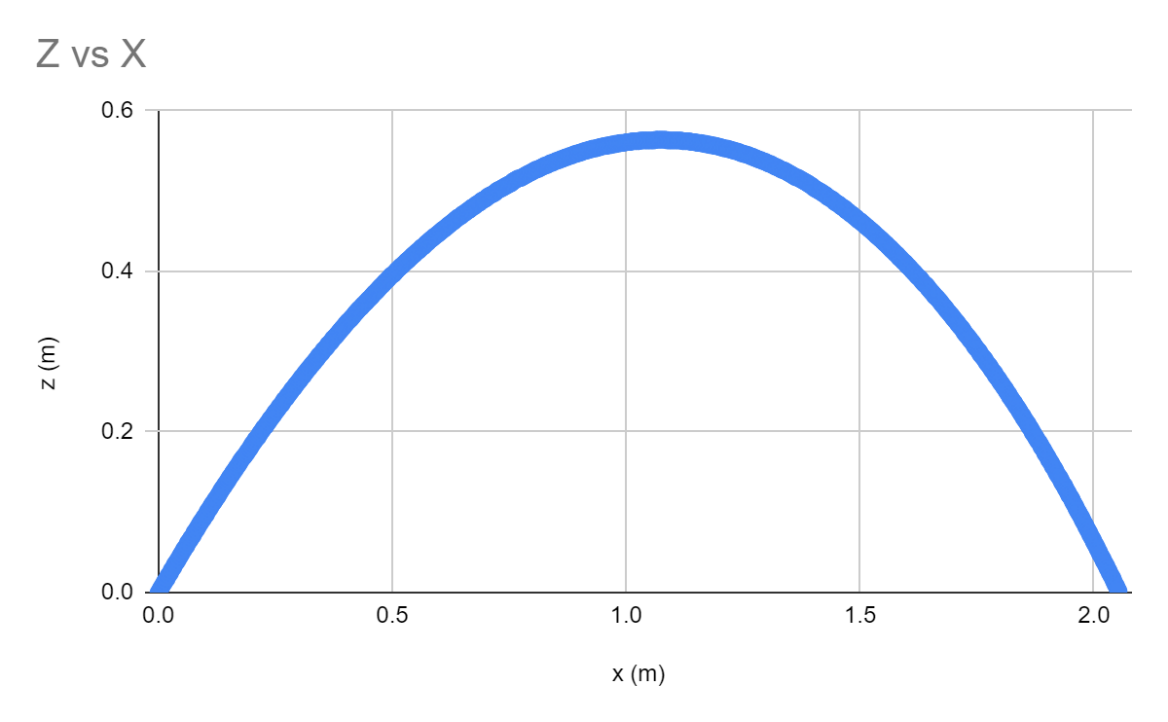
****

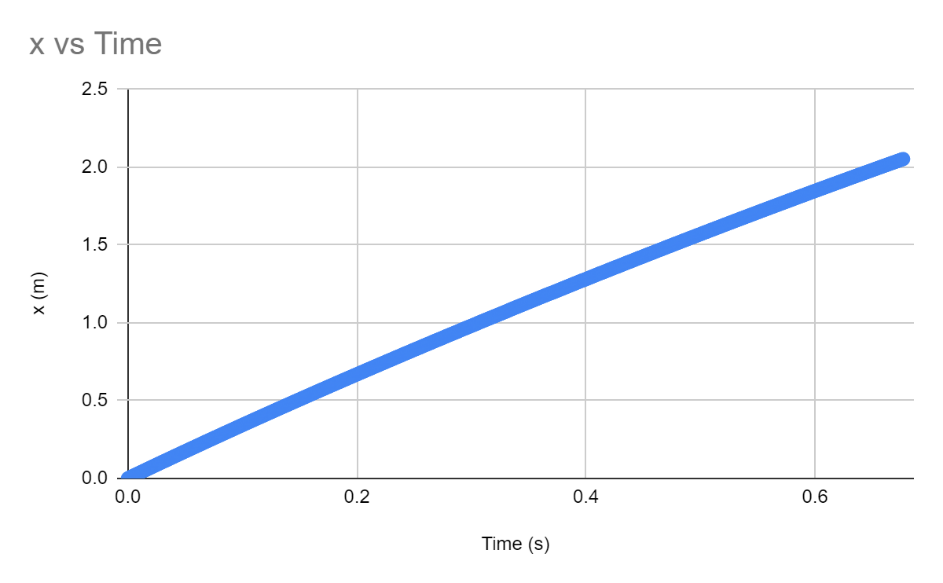
****

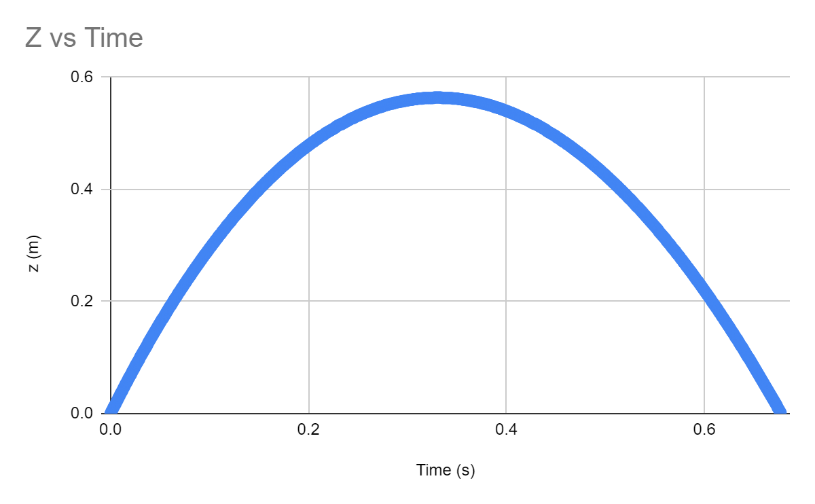
****

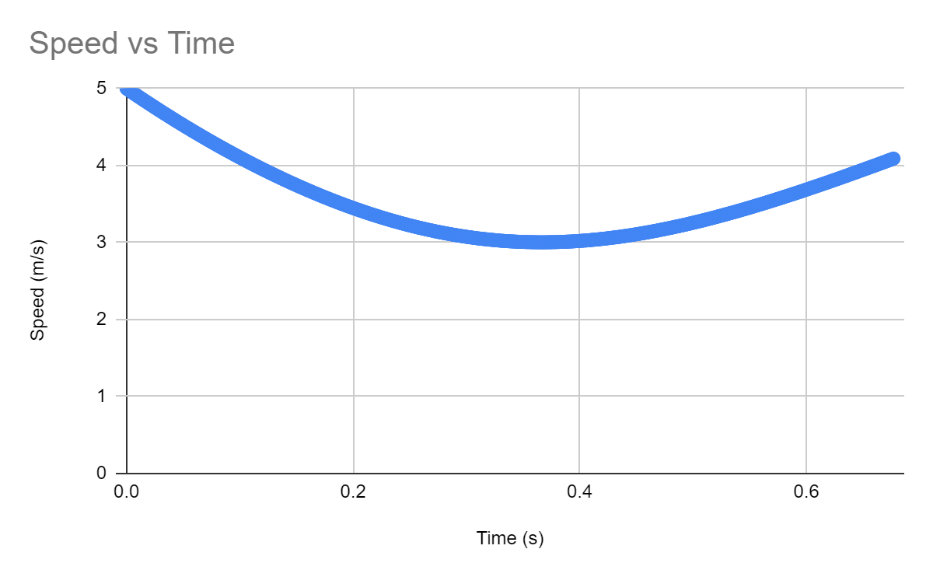
**Level II**

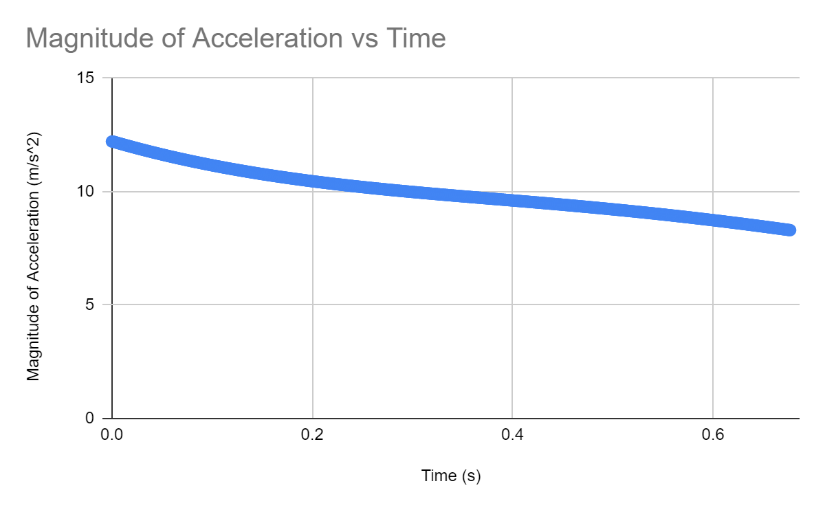
Insert five graphs: z vs. x, x vs. time, z vs. time, speed vs. time, and the magnitude of the acceleration vs. time.



****

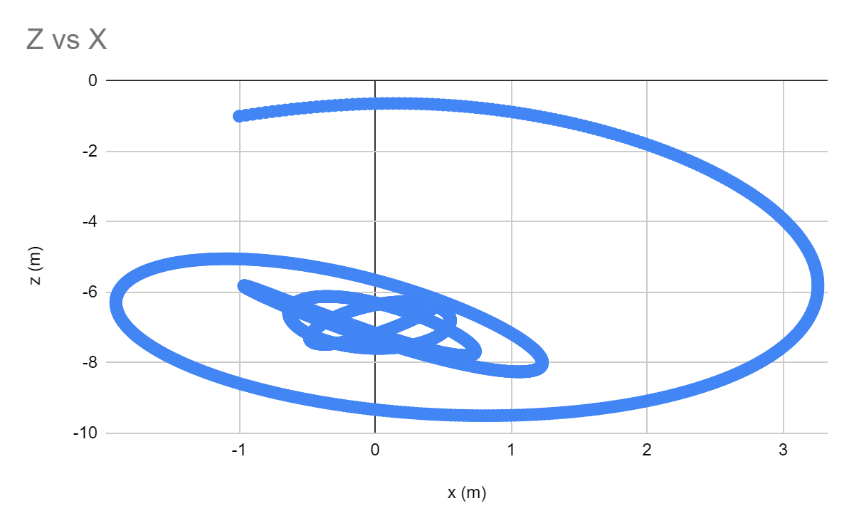
****

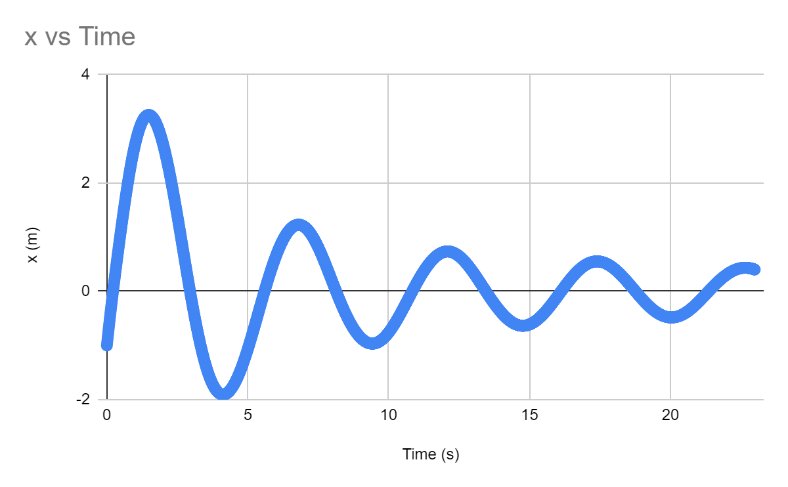
****

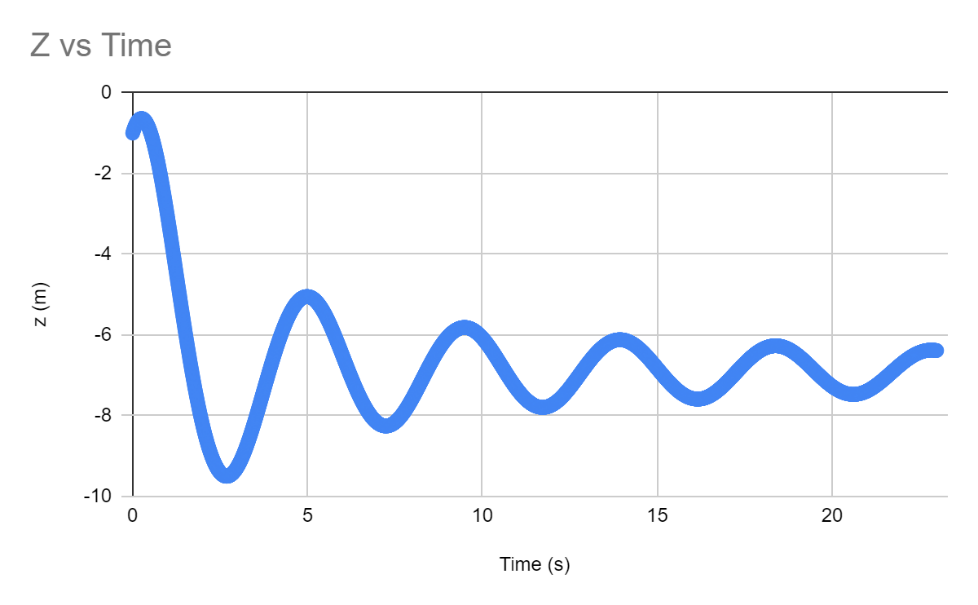
****

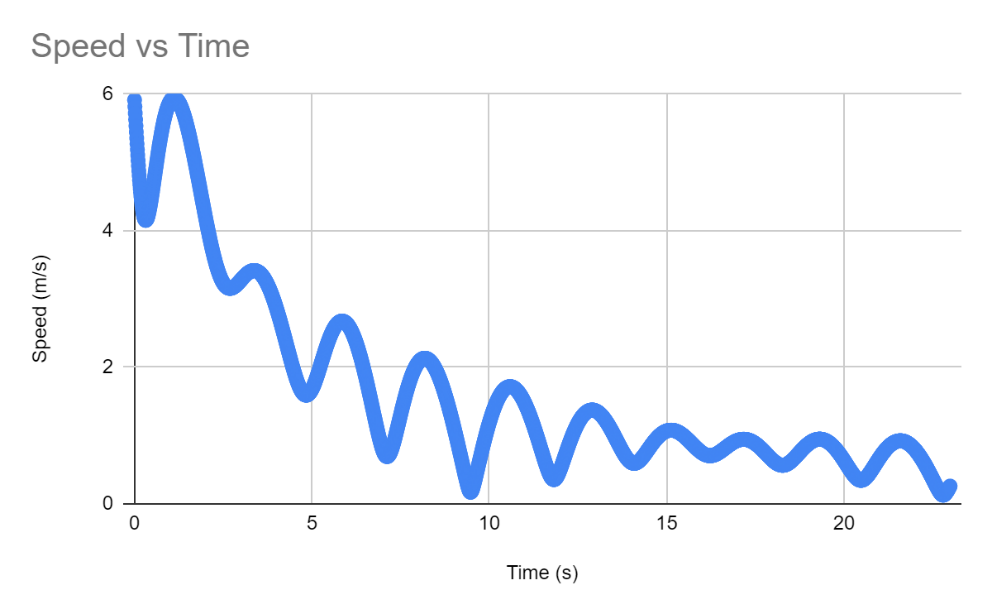
**Level III**

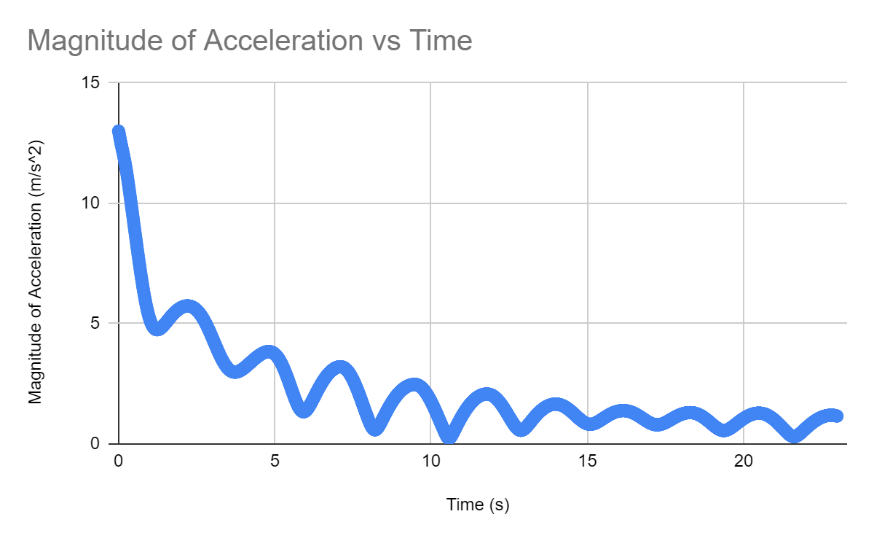
Insert five graphs: z vs. x, x vs. time, z vs. time, speed vs. time, and the magnitude of the acceleration vs. time. If you have one of those fancy 3-D plotters, you can also include an x-y-z graph.











How much time does it take the ball to drop and stay below a speed of 1 m/s? That is, what is the time of the last time the ball reaches 1 m/s?

**15.48 s**

**Code**

Copy and paste all the code that you used below. Please put a page break whenever there is a new file.

using System;

using System.Numerics;

namespace ProjectileMotion

{

class Program

{

static void Main(string[] args)

{

//LevelOne();

//LevelTwo();

LevelThree();

}

static void LevelOne()

{

//theta = 45 deg

float theta = (float)Math.PI/ 4;

float time = 0;

//initialize velocity vector

float speed = 5;

Vector3 velocity = new Vector3((float)Math.Cos(theta)\*speed, 0, (float)Math.Sin(theta)\*speed);

//initialize acceleration vector (only in z axis, everything else is just 0 so it doesn't matter)

Vector3 acceleration = new Vector3(0, 0, -9.8F);

float m\_accel = acceleration.Length();

//initialize position variable

Vector3 position = new Vector3(0, 0, 0);

float distance = 0;

//initialize deltatime variable

float dt = 0.001F;

Console.WriteLine("Time (s)" + "\t" + "x" +"\t" + "y"+ "\t" + "z" + "\t" + "Distance" + "\t" + "vx" + "\t" + "vy" + "\t" + "vz" + "\t"

+ "Speed" + "\t" + "ax" + "\t" + "ay" + "\t" + "az" + "\t" + "m\_accel");

while (position.Z>=0)

{

Console.WriteLine(time.ToString("0.000") + "\t" + position.X.ToString("0.000") + "\t" + position.Y.ToString("0.000") +

"\t" + position.Z.ToString("0.000") + "\t" + distance.ToString("0.000") + "\t" + velocity.X.ToString("0.000") + "\t" + velocity.Y.ToString("0.000") +

"\t" + velocity.Z.ToString("0.000") + "\t" + speed.ToString("0.000") + "\t" + acceleration.X.ToString("0.000") + "\t" + acceleration.Y.ToString("0.000")

+ "\t" + acceleration.Z.ToString("0.000") + "\t" + m\_accel.ToString("0.000"));

time += dt;

velocity += acceleration \* dt;

position += velocity\*dt;

distance = position.Length();

speed = velocity.Length();

}

Console.WriteLine("Done.");

}

static void LevelTwo()

{

//theta = 45 deg

float theta = (float)Math.PI / 4;

float time = 0;

//initialize velocity vector

float speed = 5;

Vector3 velocity = new Vector3((float)Math.Cos(theta) \* speed, 0, (float)Math.Sin(theta) \* speed);

//accel = net force/mass

//force of air resistance = C\*v^2

float c = 0.5F;

float mass = 4;

Vector3 air\_accel = Vector3.Multiply(-c\*velocity.Length()\*velocity.Length()/mass, Vector3.Normalize(velocity));

Vector3 g\_accel = new Vector3(0, 0, -9.8F);

Vector3 acceleration = Vector3.Add(air\_accel, g\_accel);

float m\_accel = acceleration.Length();

//initialize position variable

Vector3 position = new Vector3(0, 0, 0);

float distance = 0;

//initialize deltatime variable

float dt = 0.001F;

Console.WriteLine("Time (s)" + "\t" + "x" + "\t" + "y" + "\t" + "z" + "\t" + "Distance" + "\t" + "vx" + "\t" + "vy" + "\t" + "vz" + "\t"

+ "Speed" + "\t" + "ax" + "\t" + "ay" + "\t" + "az" + "\t" + "m\_accel");

while (position.Z >= 0)

{

Console.WriteLine(time.ToString("0.000") + "\t" + position.X.ToString("0.000") + "\t" + position.Y.ToString("0.000") +

"\t" + position.Z.ToString("0.000") + "\t" + distance.ToString("0.000") + "\t" + velocity.X.ToString("0.000") + "\t" + velocity.Y.ToString("0.000") +

"\t" + velocity.Z.ToString("0.000") + "\t" + speed.ToString("0.000") + "\t" + acceleration.X.ToString("0.000") + "\t" + acceleration.Y.ToString("0.000")

+ "\t" + acceleration.Z.ToString("0.000") + "\t" + m\_accel.ToString("0.000"));

time += dt;

velocity += acceleration \* dt;

position += velocity \* dt;

air\_accel = Vector3.Multiply(-c \* speed \* speed / mass, Vector3.Normalize(velocity));

acceleration = air\_accel+g\_accel;

distance = position.Length();

speed = velocity.Length();

m\_accel = acceleration.Length();

}

Console.WriteLine("Done.");

}

static void LevelThree()

{

float time = 0;

//initialize velocity vector

Vector3 velocity = new Vector3(5, -1, 3);

float speed = velocity.Length();

//accel = net force/mass

//force of air resistance = C\*v^2

float c = 0.5F;

float mass = 4;

float spring\_const = 8;

float spring\_len = 2;

//initialize position variable

Vector3 position = new Vector3(-1, 1, -1);

float distance = position.Length();

//initialize deltatime variable

float dt = 0.01F;

Vector3 spring\_accel = Vector3.Multiply(-spring\_const / mass, (distance - spring\_len) \* Vector3.Normalize(position));

Vector3 air\_accel = Vector3.Multiply(-c \* speed \* speed / mass, Vector3.Normalize(velocity));

Vector3 g\_accel = new Vector3(0, 0, -9.8F);

Vector3 acceleration = air\_accel+g\_accel+spring\_accel;

float m\_accel = acceleration.Length();

Console.WriteLine("Time (s)" + "\t" + "x" + "\t" + "y" + "\t" + "z" + "\t" + "Distance" + "\t" + "vx" + "\t" + "vy" + "\t" + "vz" + "\t"

+ "Speed" + "\t" + "ax" + "\t" + "ay" + "\t" + "az" + "\t" + "m\_accel");

while (time<=23)

{

Console.WriteLine(time.ToString("0.00") + "\t" + position.X.ToString("0.000") + "\t" + position.Y.ToString("0.000") +

"\t" + position.Z.ToString("0.000") + "\t" + distance.ToString("0.000") + "\t" + velocity.X.ToString("0.000") + "\t" + velocity.Y.ToString("0.000") +

"\t" + velocity.Z.ToString("0.000") + "\t" + speed.ToString("0.000") + "\t" + acceleration.X.ToString("0.000") + "\t" + acceleration.Y.ToString("0.000")

+ "\t" + acceleration.Z.ToString("0.000") + "\t" + m\_accel.ToString("0.000"));

time += dt;

velocity += acceleration \* dt;

position += velocity \* dt;

air\_accel = Vector3.Multiply(-c \* speed \* speed / mass, Vector3.Normalize(velocity));

spring\_accel = Vector3.Multiply(-spring\_const / mass, (distance-spring\_len) \* Vector3.Normalize(position));

acceleration = air\_accel + g\_accel+spring\_accel;

distance = position.Length();

speed = velocity.Length();

m\_accel = acceleration.Length();

}

Console.WriteLine("Done.");

}

}

}