Kinematics

Homework 1

The first problem we will look at is basic kinematics—how objects move. This is probably the simplest motion that there is to model and we can use it to get used to programming for science and to see how powerful it can be.

The basic approach is to calculate the position of an object at many different points in time. To do this, we can use the basic definition of velocity: . For computational purposes, this is the same as saying , as long as is sufficiently small.

Our general strategy is to calculate the position of an object at many times, each one apart. We can use the velocity equation to find the next position because .

**Level I**

Write a program that calculates the position of a particle moving at a constant velocity of 5 m/s. You should output the position of the particle every tenth of a second for 100 s. Have your program produce an output as a table:

|  |  |  |
| --- | --- | --- |
| Time (s) | Position (m) | Velocity (m/s) |
| 0 | 0 | 5 |
| 0.1 | 0.5 | 5 |
|  |  |  |

It’s easiest if you use tabs to separate the columns. You can add a tab to a string by typing \t. Paste this table into Excel and make two scatterplots out of it: position vs. time and velocity vs. time. Make sure these numbers make sense before you continue. Make sure to label your axes!

**Level II**

Now we should consider the case where there is acceleration. This just requires us to add in the change in velocity, because we know that . Thus, in each step of our loop, we can change the velocity, and then use that value to change our position at each point in time.

Write a program that calculates the position of a particle beginning with a velocity of 5 m/s and accelerating with a constant acceleration of 9.8 m/s2 (both in the positive direction). Add a fourth column, acceleration, to your table, and use Excel to make three scatterplots: position vs. time, velocity vs. time, and acceleration vs. time. Use the kinematic equation to calculate what you would expect to get at the end of 100 s, and make sure your program agrees. Note that your program does not use the kinematic equation itself! You should compare your program’s computational answer with the analytic answer from basic kinematics.

**Level III**

Now we will include air resistance. While the phenomenon of air resistance is too difficult for us to model completely with our understanding, we can use the commonly accepted formula that relates the force of air resistance to the square of the velocity, or . Air resistance, of course, always acts to slow a particle down without changing its direction.

Use an air resistance force with kg/m on a particle with a mass of 4 kg. Create the same table and plots as the last problem.

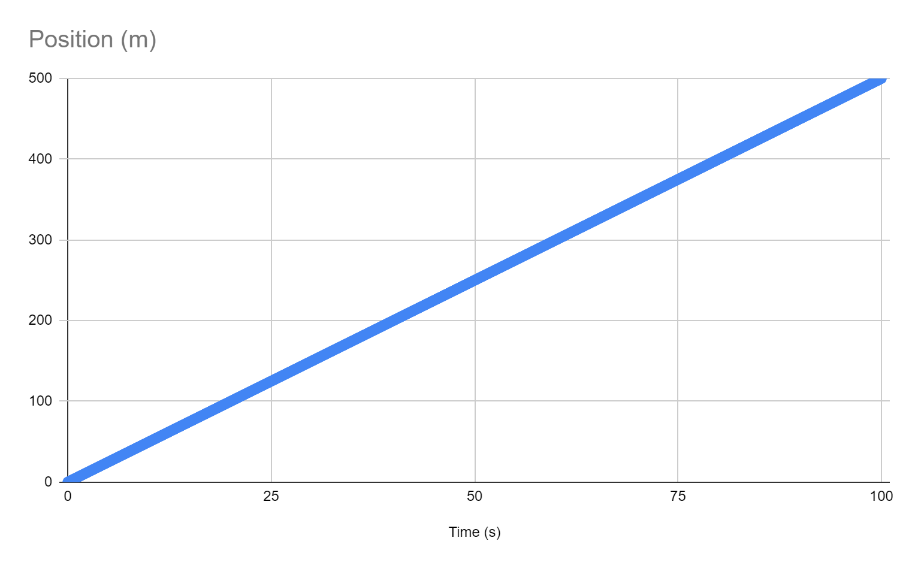
**Challenge**

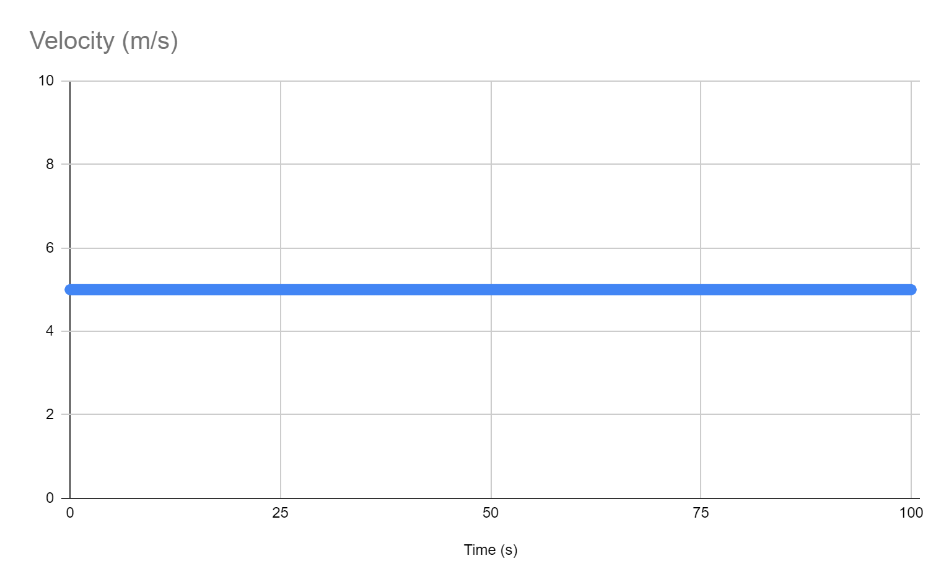
Now calculate the behavior of a mass attached to a spring. This follows Hooke’s law, which says that the force is proportional to the displacement from the position of the unstretched spring: . Let the spring constant *k* be 8 kg/s2 and let the spring be attached to the origin with an unstretched length of 2 m. Have the particle start with a velocity of 5 m/s at position m. Have gravity (still acting in the positive x direction) and air resistance act just as in the previous problem, and make the same plots for this situation.

Homework 1 Answer Sheet

**Level I**

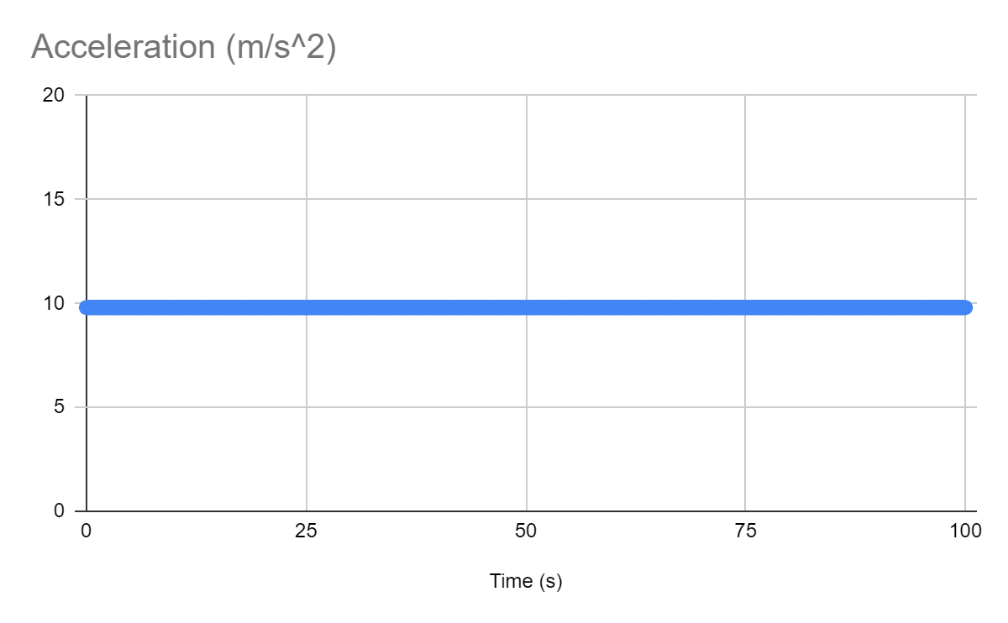
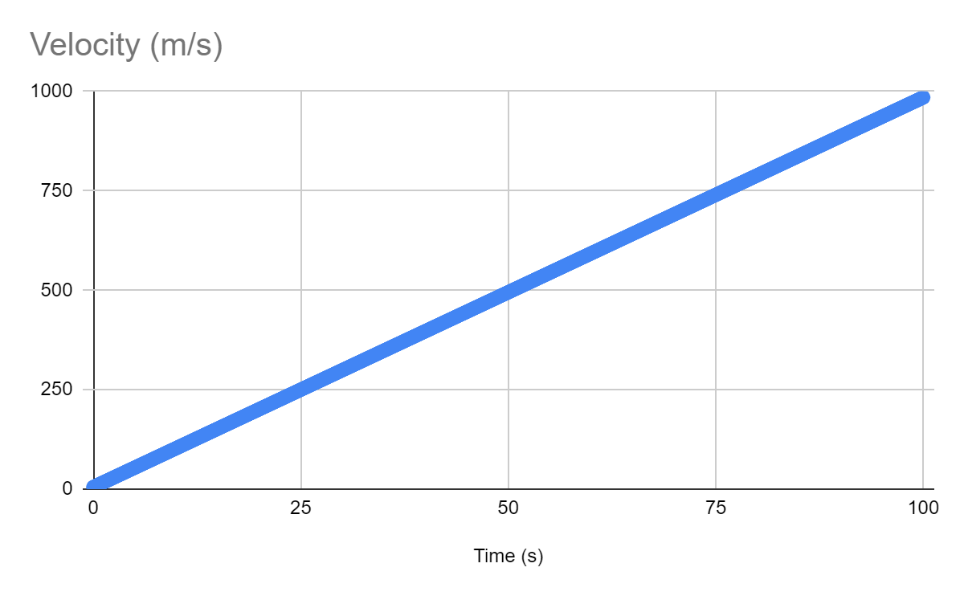
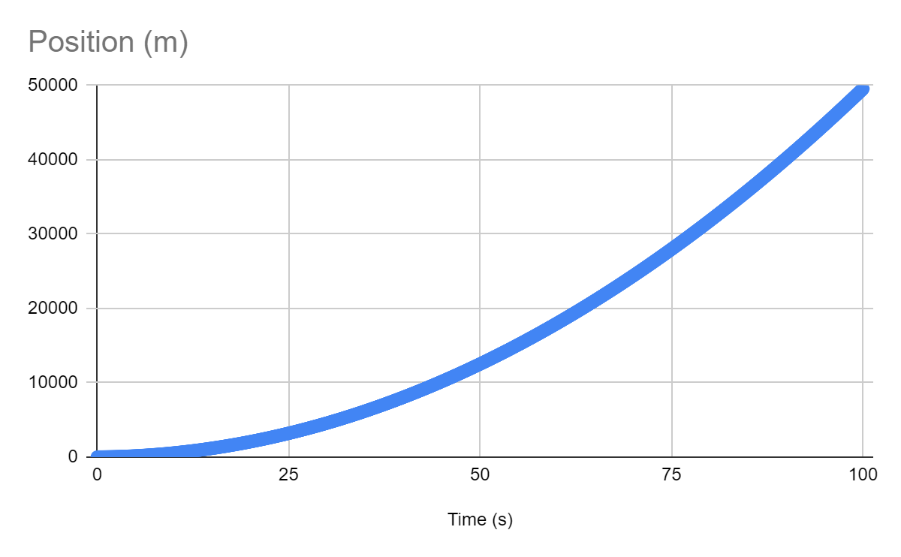
Insert your two graphs (position vs. time and velocity vs. time) below.





**Level II**

Insert your three graphs (position vs. time, velocity vs. time, and acceleration vs. time) below.



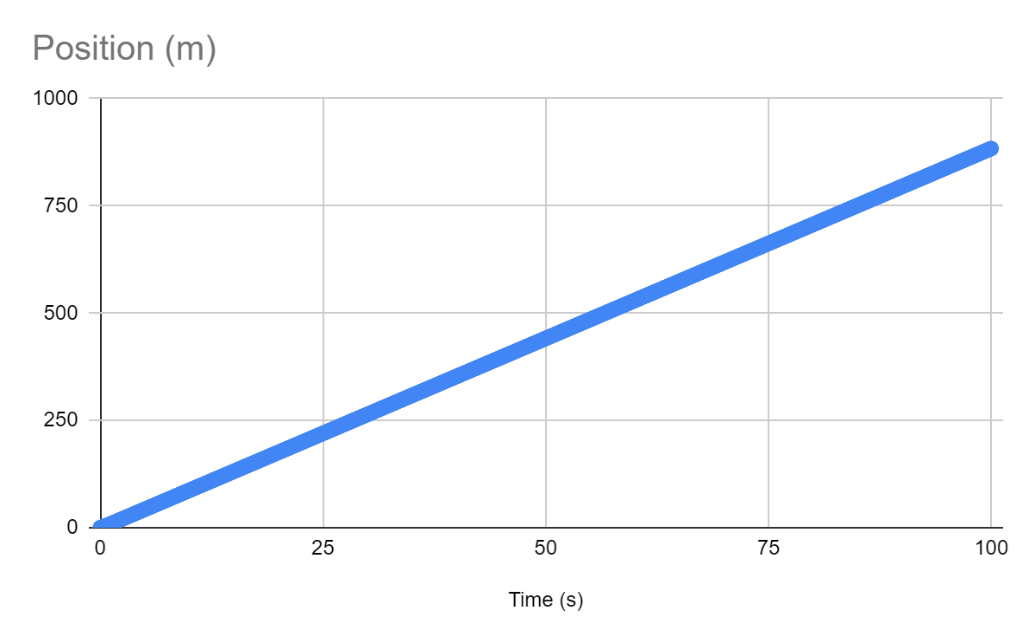
Find the expected position from the kinematic equation given: **49500m**

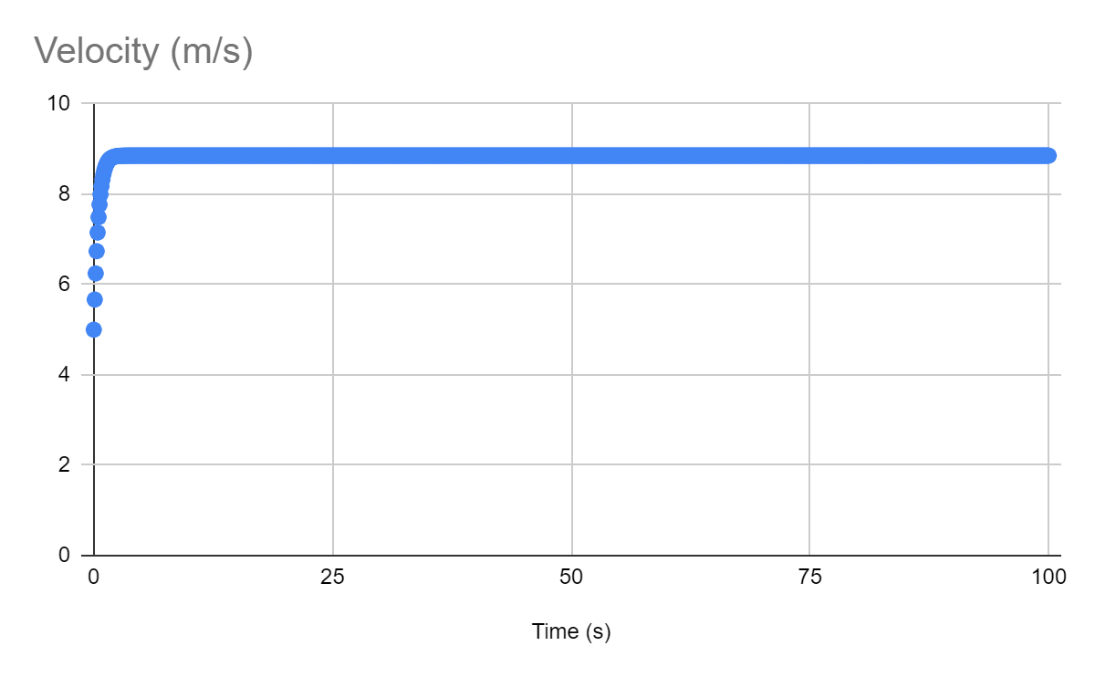
Is your program in perfect numerical agreement? If not, why not?

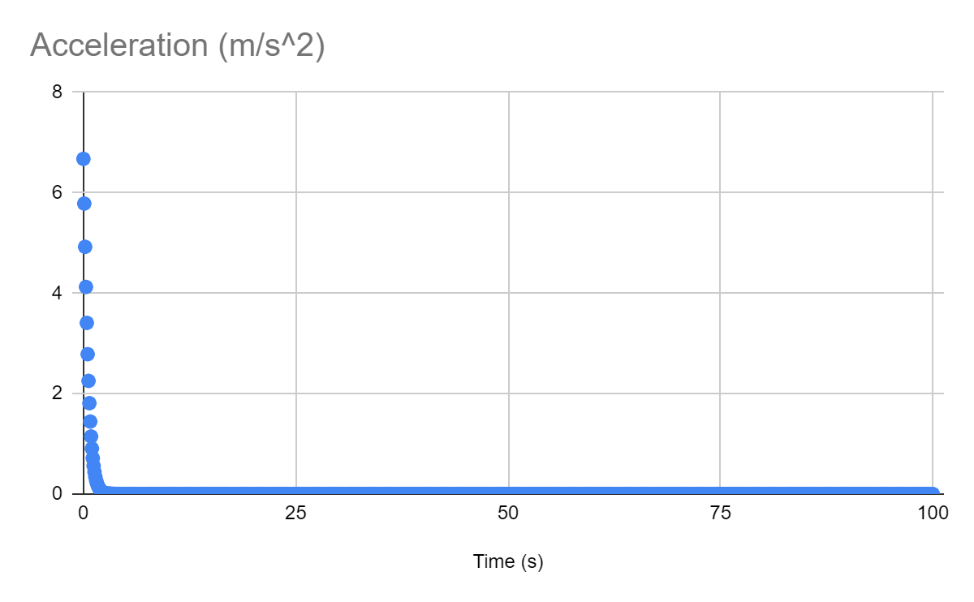
**No, my program gives me an answer of 49549m, as a result of the increment of time being 0.1. Decreasing this value would allow for a much more accurate answer because it would be recalculating the velocity and the position more often, making it much closer to the actual value.**

**Level III**

Insert your three graphs (position vs. time, velocity vs. time, and acceleration vs. time) below.

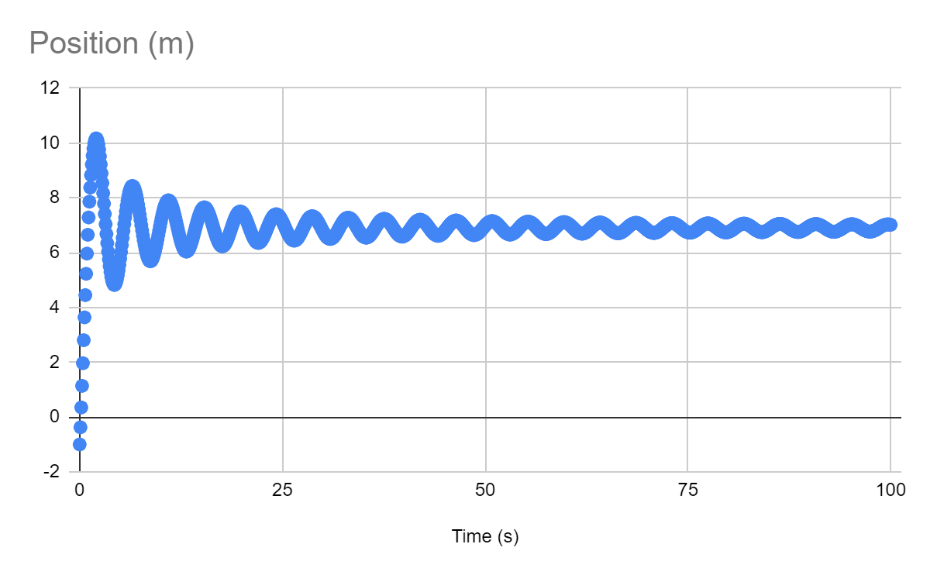


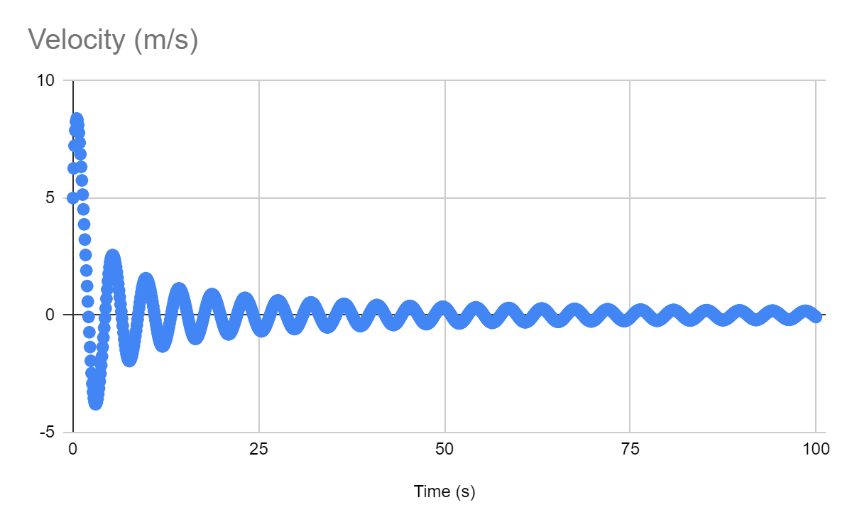


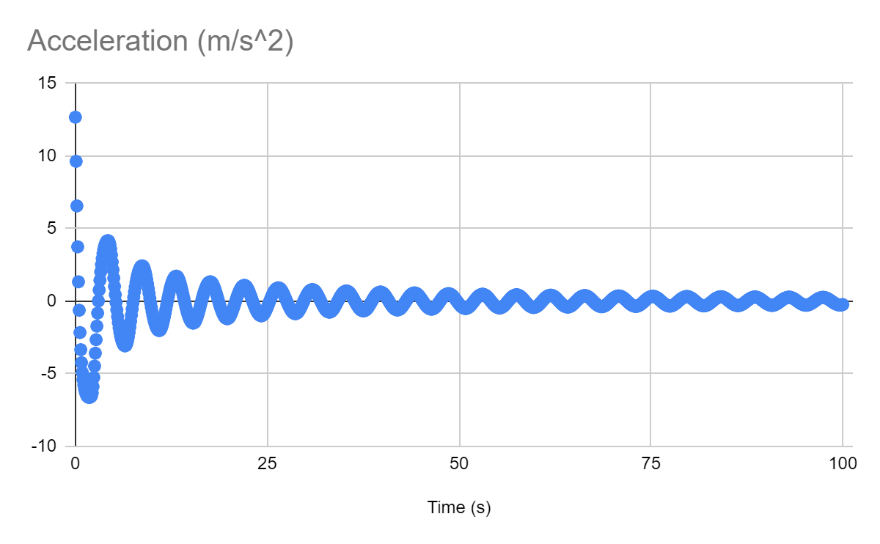


**Challenge**

Insert your three graphs (position vs. time, velocity vs. time, and acceleration vs. time) below.







**Code**

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

using System;

namespace Kinematics

{

class Program

{

static void Main(string[] args)

{

Challenge();

}

private static void LevelOne()

{

//v = dx/dt ---> dx = v\*dt

//dt = 0.1

//initialize velocity variable to 5 m/s

//using decimal instead of double to get rid of annoying floating point errors

decimal velocity = 5;

//initialize position variable

decimal position = 0;

Console.WriteLine("Time (s)" + "\tPosition (m)" + "\tVelocity (m/s)");

for (decimal time = 0; time <= 100; time += (decimal)0.1)

{

Console.WriteLine(time + "\t\t" + position + "\t\t" + velocity);

position = position + velocity \* (decimal)0.1;

}

Console.WriteLine("Done");

}

private static void LevelTwo()

{

//v = dx/dt ---> dx = v\*dt

//a = dv/dt = d^2x/dt^2

//dt = 0.1

//initialize velocity variable to 5 m/s

//using decimal instead of double to get rid of annoying floating point errors

decimal velocity = 5;

//initialize position variable

decimal position = 0;

//initialize acceleration as 9.8

decimal acceleration = (decimal)9.8;

Console.WriteLine("Time (s)" + "\tPosition (m)" + "\tVelocity (m/s)" + "\tAcceleration");

for (decimal time = 0; time <= 100; time += (decimal)0.1)

{

Console.WriteLine(time + "\t" + position + "\t" + velocity + "\t" + acceleration);

velocity = velocity + acceleration \* (decimal)0.1;

position = position + velocity \* (decimal)0.1;

}

Console.WriteLine("Done");

}

private static void LevelThree()

{

//v = dx/dt ---> dx = v\*dt

//a = dv/dt = d^2x/dt^2

//dt = 0.1

//initialize velocity variable to 5 m/s

//using decimal instead of double to get rid of annoying floating point errors

decimal velocity = 5;

//initialize position variable

decimal position = 0;

//initialize mass variable

decimal mass = 4;

//need to calculate F\_net

//F\_g = mass\*9.8

decimal f\_g = mass \* (decimal)9.8;

//F\_air = 0.5\*v^2

decimal f\_air = (decimal)(0.5) \* velocity \* velocity;

//F\_net = f\_g - f\_air

decimal f\_net = f\_g - f\_air;

//initialize acceleration as 9.8

decimal acceleration = f\_net/mass;

Console.WriteLine("Time (s)" + "\tPosition (m)" + "\tVelocity (m/s)" + "\tAcceleration");

for (decimal time = 0; time <= 100; time += (decimal)0.1)

{

Console.WriteLine(time + "," + position + "," + velocity + "," + acceleration);

velocity = velocity + acceleration \* (decimal)0.1;

position = position + velocity \* (decimal)0.1;

f\_air = (decimal)(0.5) \* velocity \* velocity;

f\_net = f\_g - f\_air;

acceleration = f\_net / mass;

}

Console.WriteLine("Done");

}

private static void Challenge()

{

//v = dx/dt ---> dx = v\*dt

//a = dv/dt = d^2x/dt^2

//dt = 0.1

//initialize velocity variable to 5 m/s

//using decimal instead of double to get rid of annoying floating point errors

decimal velocity = 5;

//initialize position variable

decimal position = -1;

//initialize mass variable

decimal mass = 4;

//need to calculate F\_net

//F\_g = mass\*9.8

decimal f\_g = mass \* (decimal)9.8;

//F\_air = 0.5\*v^2

decimal f\_air = (decimal)(0.5) \* velocity \* velocity;

//F\_spring = -8\*(position relative to spring) = -8\*(position - 2)

decimal f\_spring = -8 \* (position - 2);

//F\_net = sum of all forces

decimal f\_net = f\_g - f\_air + f\_spring;

//initialize acceleration as 9.8

decimal acceleration = f\_net / mass;

Console.WriteLine("Time (s)" + "\tPosition (m)" + "\tVelocity (m/s)" + "\tAcceleration");

for (decimal time = 0; time <= 100; time += (decimal)0.1)

{

Console.WriteLine(time + "," + position + "," + velocity + "," + acceleration);

velocity = velocity + acceleration \* (decimal)0.1;

position = position + velocity \* (decimal)0.1;

if (velocity > 0)

{

f\_air = (decimal)(0.5) \* velocity \* velocity;

}

else

{

f\_air = (decimal)(-0.5) \* velocity \* velocity;

}

f\_spring = -8 \* (position - 2);

f\_net = f\_g - f\_air + f\_spring;

acceleration = f\_net / mass;

}

Console.WriteLine("Done");

}

}

}