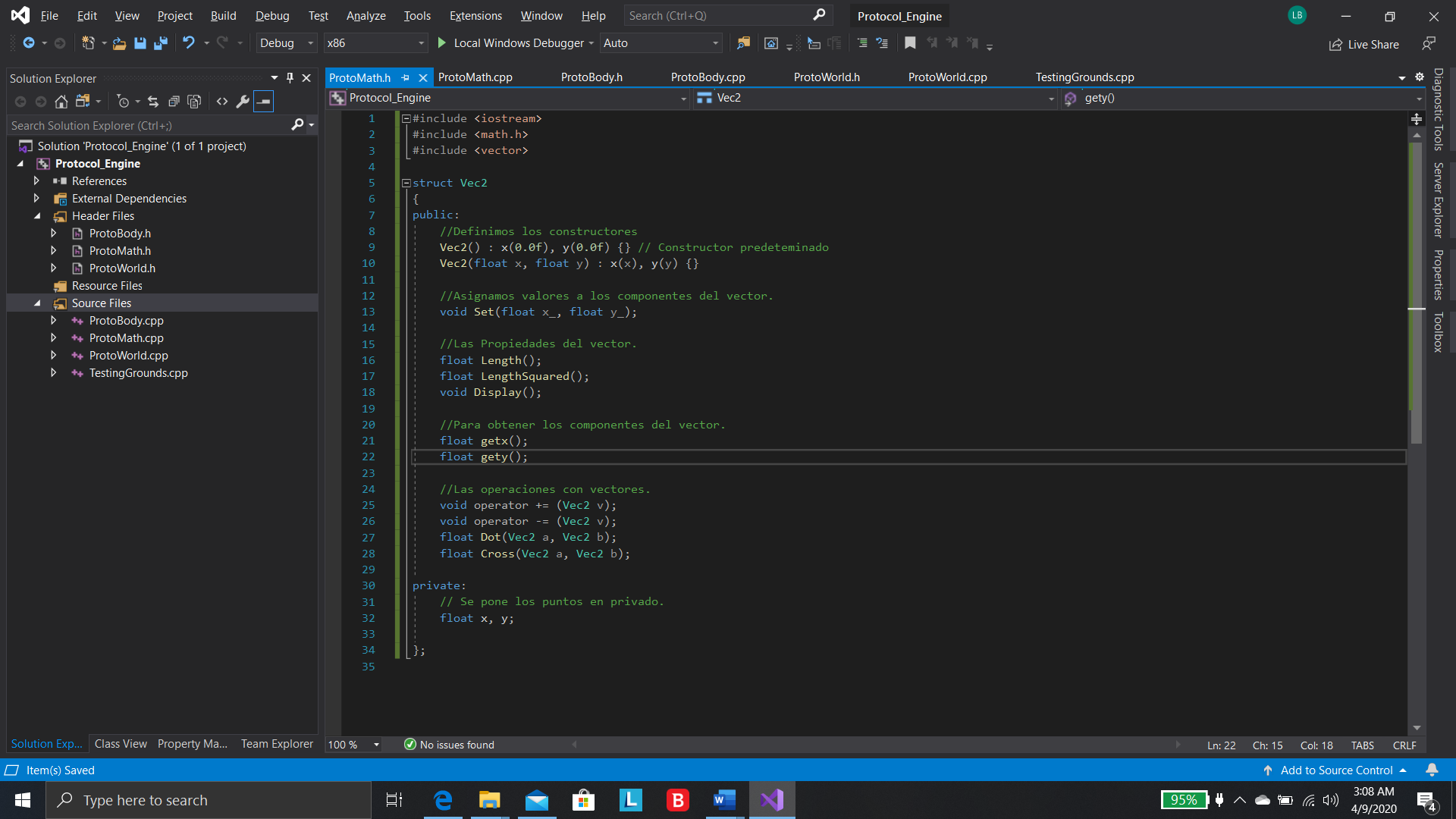
Documentación de Protocol Engine

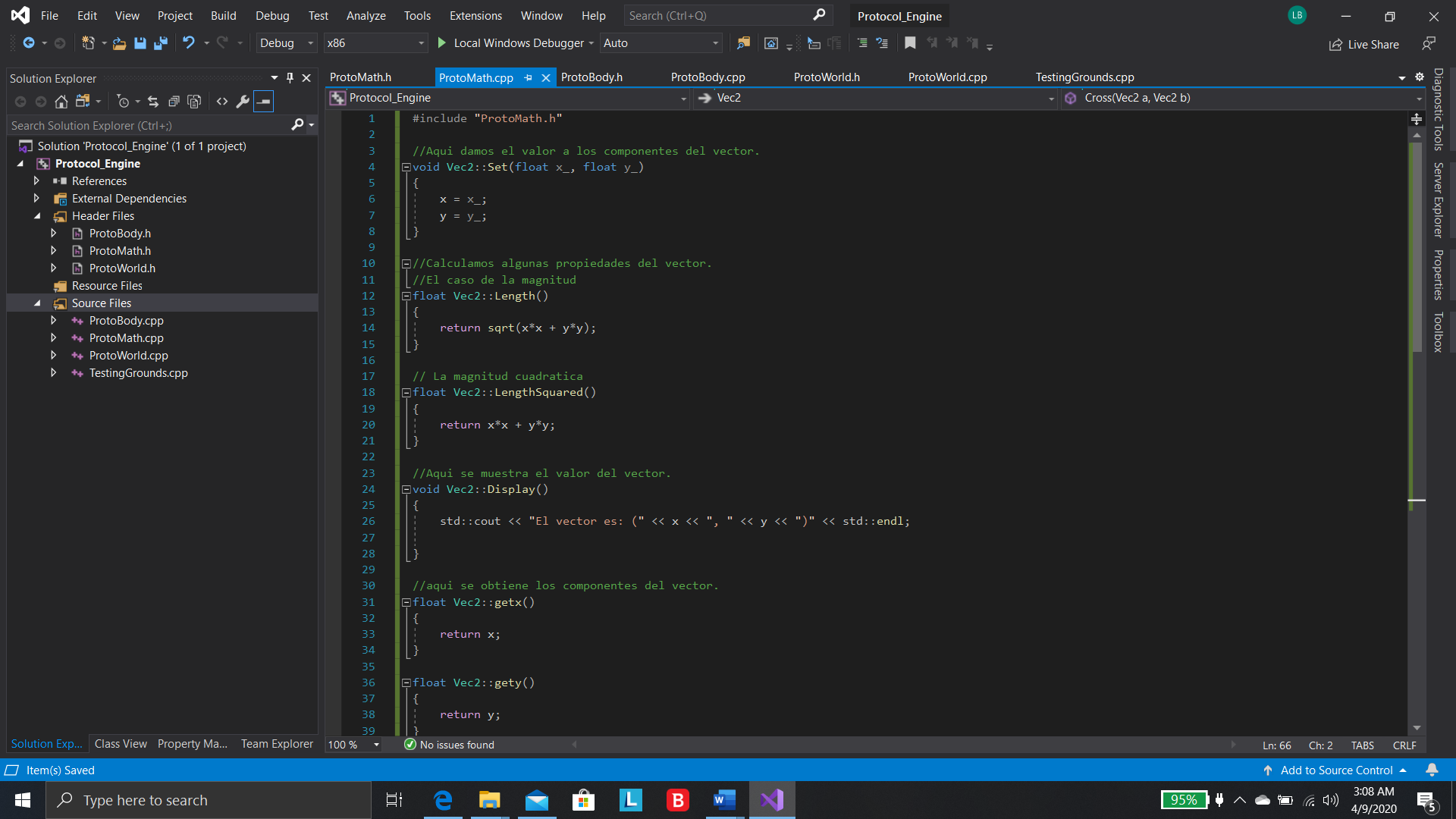
Por: Luis Gutiérrez

El objetivo de esta programación es hacer un game engine básico, dándonos valores importantes de física, poniendo puntos en “x” y “y” dándonos velocidad, posición y gravedad. En primer lugar cada imagen se le explicara con detalle la programación implementada…

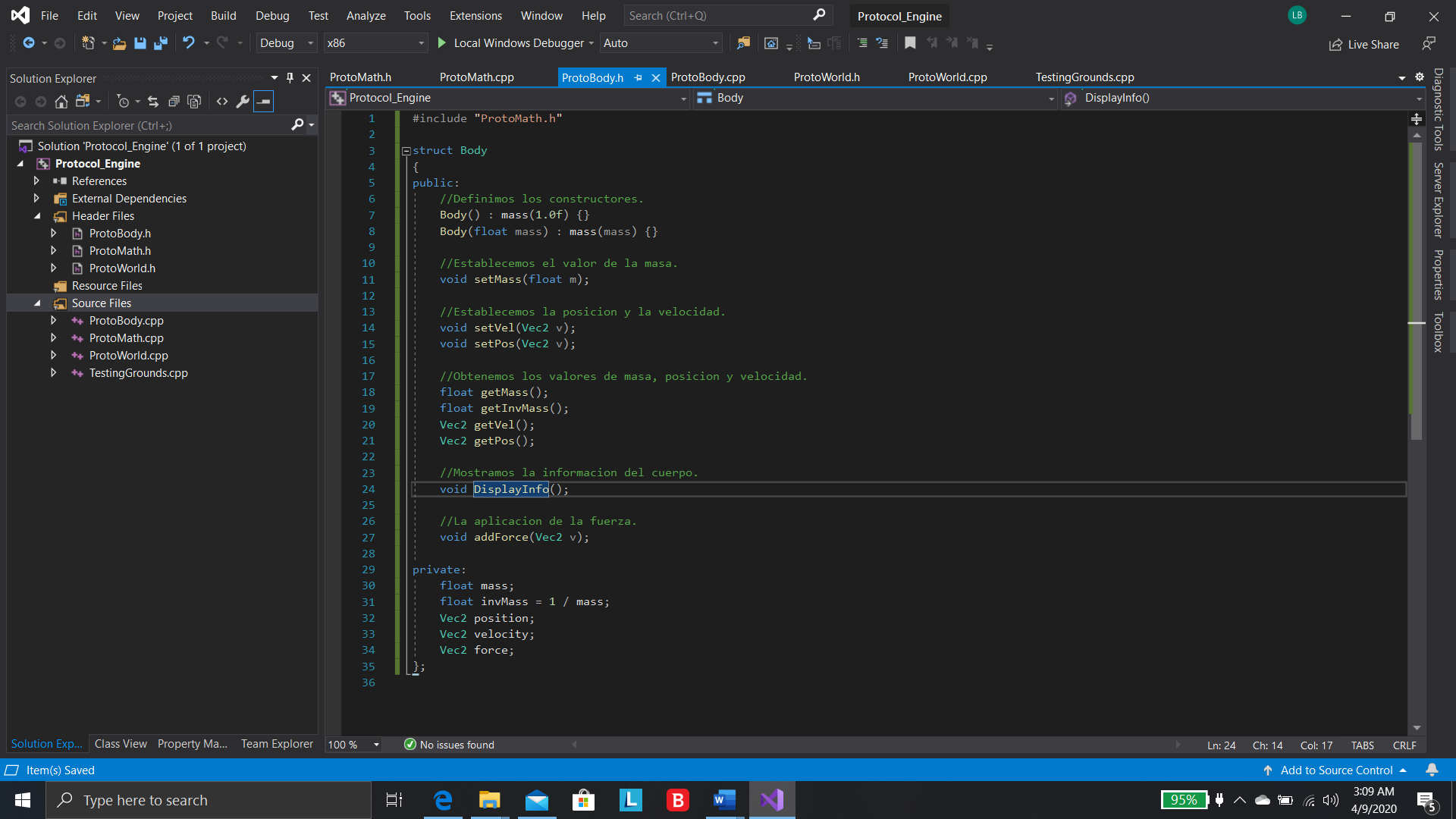


En la librería que creamos ProtoMath.h, aquí se crea esta librería para ingresar los valores matemáticos, por es incluimos la librería math.h y también incluimos el sistema vectorial con vector, creamos una estructura que se llamara Vec2, donde en public se definirá los constructores en cuanto a los vectores con sus parámetros en float. También asignaremos los valores a los componentes del vector con utilizar el set, damos valores en x y y porque lo tenemos en private. También definimos las propiedades del vector en float, que ahí utilizaremos la magnitud del vector, la magnitud cuadrática, y un display donde nos mostrara los valores. Ahora para obtener los componentes del vector utilizando un get para pedir los puntos que tenemos en private. Las operaciones con vectores que serian los siguientes la suma de los vectores, la resta de los vectores, producto punto y producto cruz y en private se pondrá los puntos de x y y con un float para que pongamos y utilicemos valores en números decimales.

En el CPP nombrado ProtoMath.cpp tendrá la librería incluidas de ProtoMath.h y pondremos el contenido y operaciones matemáticos, mostrado en la siguiente imagen.



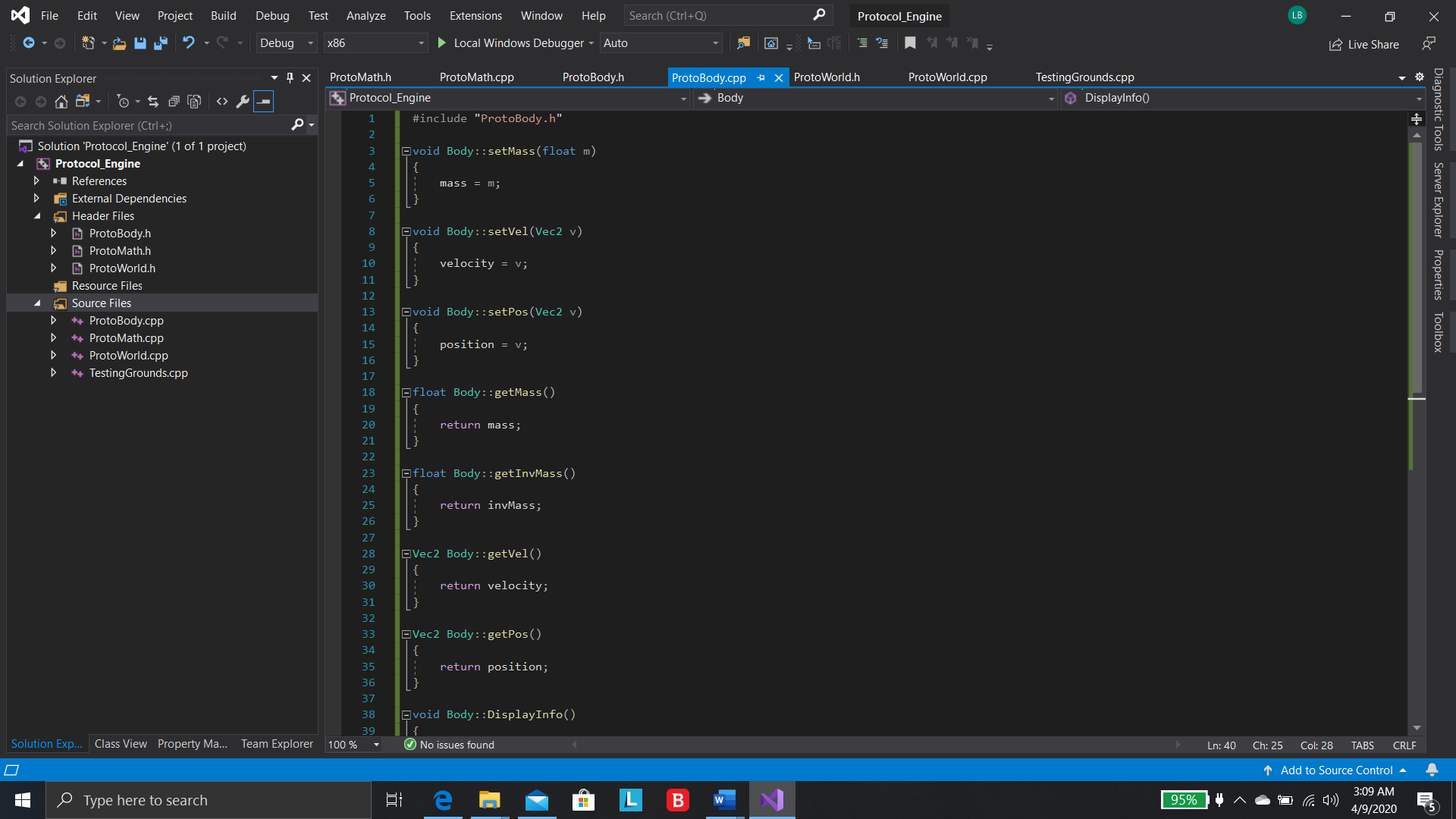
En la imagen como verán se estará poniendo los componentes y formulas como en la magnitud, y la magnitud cuadrática, igual como se imprimirá el resultado del vector.



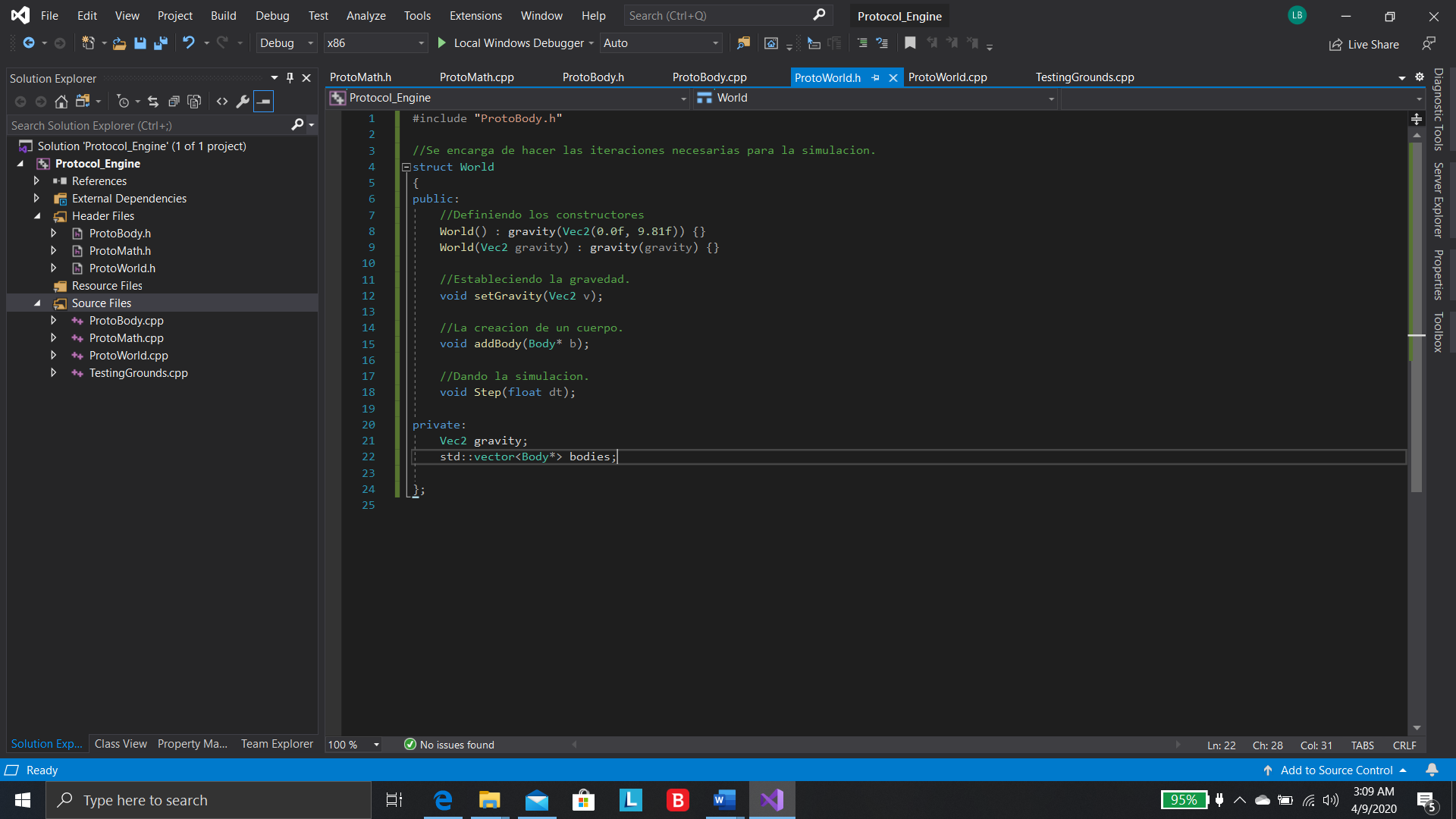
En esta librería que creamos le nombramos ProtoBody.h y esta librería incluye la librería de ProtoMath.h, aquí daremos la estructura de un cuerpo definiendo con el constructor dándonos la masa de ese cuerpo, estableceremos el valor de la masa con un setMass, también estableceremos con un setVel y setPos la velocidad y la posición, y los getmass, getInvmass, y el getVel y el getPos aquí se obtendrá los valores de la masa, la masa invertida y la velocidad y la posición.

Pondremos un void nombrado DisplayInfo para que nos de la información del cuerpo, y pondremos un void que se llamara addforce para aplicar la fuerza en ese cuerpo. En Private pondremos dos variables float nombrado mass para la masa y otro seria invMass dando como resultado 1 entre la masa para darnos la masa invertida, y tres variables vec2 refiriendo un vector de 2 puntos nombrados con position para la posición, velocity para la velocidad y force para la fuerza.

En la siguiente imagen cramos un .cpp nombrado ProtoBody.cpp que este tendrá incluida la librería ProtoBody.h



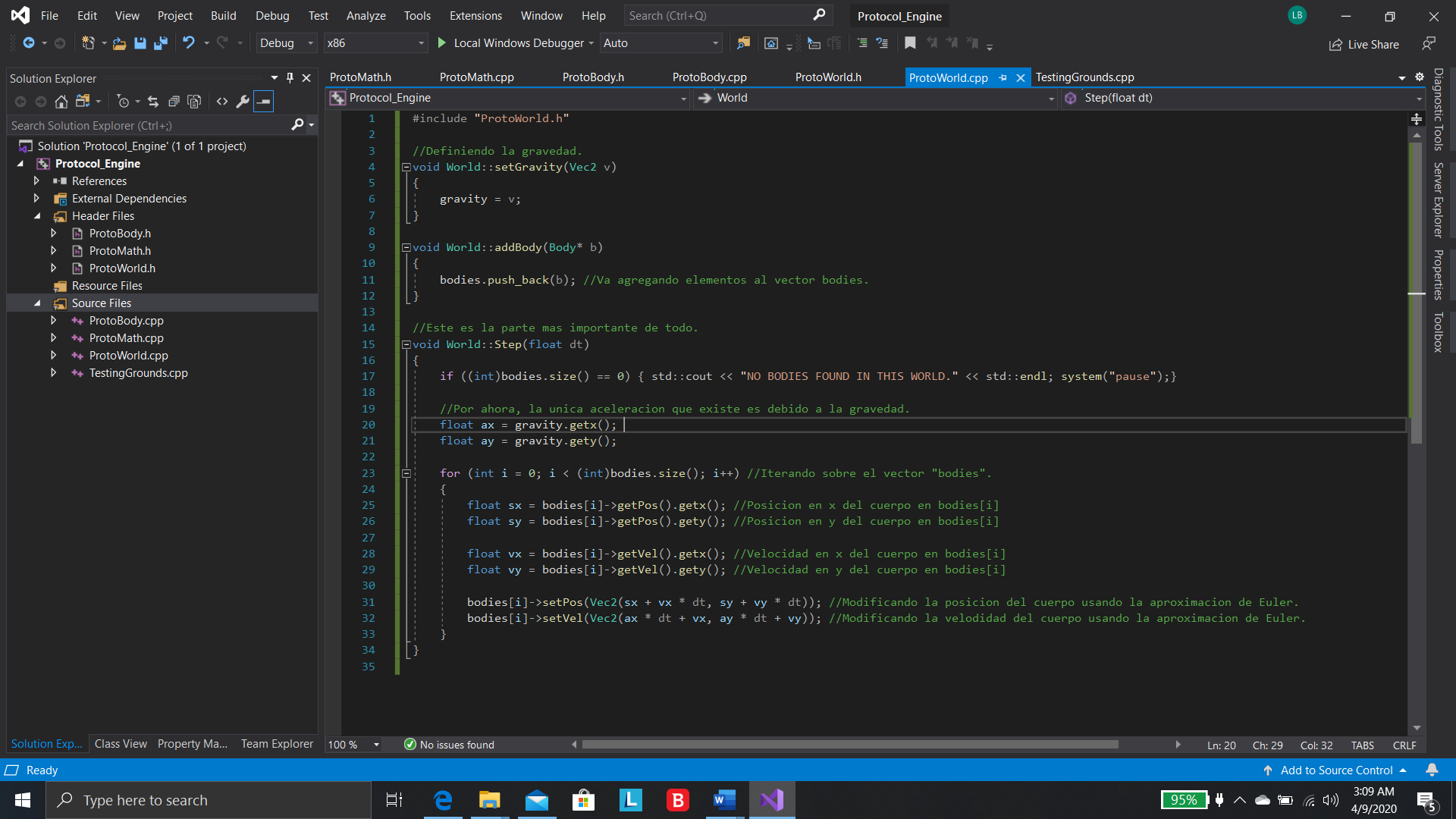
Aquí definiremos con parámetros las funciones en cuando a la mass con una m, el velocity con una v, el position con v, mientras los float y el vec2 regresaran los valores pre dichos, y mas abajo en la información del display se mostrara std::cout << std::endl; std::cout << "m: " << mass << std::endl; std::cout << "s: ", position.Display(); std::cout << "v: ", velocity.Display(); refiriendo que mostrara los valores m, s y v mencionados anteriormente, puesto con un endl para saltar renglón. El void de addforce por el momento lo vamos a tener vacío para hacer procesos avanzados de engine.



En esta librería que creamos le nombramos ProtoWorld.h y esta librería incluye la librería de ProtoBody.h, aquí como dice se va a encargar de hacer las iteraciones necesarias para la simulación refiriéndose a un mundo como tal donde se dará la gravedad y así poniendo una estructura World en public pondremos los constructores de world con su gravedad con un vector de 2 puntos en que el punto y se dará el valor de 9.81 refiriendose la fuerza de la gravedad, en el void estableceremos la gravedad con setGravity y la creación de un cuerpo nombrándolo addBody, dando la simulación nombrándolo step, en private pondremos las variables de Vec2 poniendo el nombre de gravity, y un vector en el cuerpo.

En el CPP que le nombramos ProtoWorld que incluirá la librería de ProtoWorld.h, aquí definiremos la gravedad en este caso gravity será una v, el addBody pondremos un push\_back y esto nos indicara que va agregando elementos al vector bodies.

La parte mas importante del cuerpo en este caso de un todo, indicamos que si el cuerpo tiene un tamaño es igual 0, se imprimirá que no hay cuerpo buscado en este mundo, pusimos que el cuerpo tendrá una aceleración y que también el cuerpo tendrá posición en “x” y en “y”, también lo interesante que vamos a utilizar la posición y la velocidad usando la aproximación de Euler.



Y por ultimo tendremos que el código donde se va a imprimir todos los valores se incluirá la librería ProtoWorld.h que incluye las demás librerías , y aquí definimos las variables con 5 float con nombres de mass para la masa, pos\_x la posición en x, pos\_y la posición en y, vel\_x la velocidad en x, vel\_y la velocidad en y, esto estará afuera del main.

Luego adentro del main imprimeros la bienvenida al usuario pidiéndonos los valores predichos, mas abajo se creara el mundo llamándolo World w1, y el cuerpo Body b1 donde solamente que se de la masa con un setMass, la posición con sePos, y la velocidad con setVel, con su parámetro ingresado por medio de la variables que insertamos al inicio del codigo y daremos el cuerpo con un addBody, y esto se dará valores incrementadose por un paso, hasta 10 veces utilizando con un for el b1.