El programa cuenta con 1427 líneas de código, uso de arreglos de datos y de funciones y subrutinas. El programa hace uso de 2 librerías: la primera “minim” para la reproducción de audio y las segunda “gráfica” para el dibujo de las gráficas de posición, velocidad y tiempo.

Se comienza el programa definiendo e iniciando todas las variables que se van a usar así como los debidos arreglos de datos. luego, En la función setup de processing se cargan todas las imágenes, los audios, se crean las tablas que se usarán para las gráficas y los archivos de texto que se mostrarán en la carpeta del proyecto

luego sigue la función draw de processing que es la encargada de mostrar todo en pantalla. El programa está dividido por etapas donde cada etapa es una pantalla que se mostrará comenzando por la etapa 0 hasta la etapa 5.

Etapa 0: aquí se muestra la primera pantalla con el nombre de nuestro proyecto oscilab, aquí mostramos un botón para reproducir o pausar la música usando la librería minim. si presionamos el botón de continuar se nos llevará a la etapa 1. si presionamos el botón de salida se deja de ejecutar el programa, y si presionamos el botón de instrucción se mostrará las instrucciones para el usuario, esté con los debidos botones pueden ver las diferentes instrucciones y al finalizar se le mostrará un botón que lo lleve a la etapa 1.

Etapa 1: aquí se le pide al usuario el sistema que desea trabajar, sea sistema no amortiguado, amortiguado y oscilaciones forzadas o resonancia., y a través del uso de condicionales para el cambio de sistema para trabajar. al presionar continuar se dirige a la etapa 2.

Etapa 2: aquí se le pide al usuario dependiendo del sistema que previamente eligió, los respectivos valores que debe digitar para poder calcular la posición, velocidad y aceleración. Estos menús fueron implementados con una función para optimizar el código, dicha función hace uso de las figuras geométricas de processing para mostrar cada recuadro, así como el uso de cadena de caracteres para digitar los valores. los botones de continuar y borrar son programados en la función de processing de MouseClicked y KeyPressed que se explicará más adelante, después de digitar el último valor el programa automáticamente lo llevará a la etapa 3.

Etapa 3: esta etapa para los casos de sistema no amortiguado y oscilaciones forzadas no hace practicamente nada. pero para el caso de sistema amortiguado se hace cálculo de la condición del oscilador y después lo muestra en pantalla, si pulsa en el botón de continuar lo lleva a la etapa 4.

Etapa 4: esta etapa es la de la animación, para esta etapa es muy importante saber que la subrutina de processing void draw se ejecuta a 60 FPS es decir 60 imágenes por segundo. las gráficas del oscilador se hacen teniendo en cuenta esto, donde una variable llamada “y” hace la animación mediante la opción que cuentan las imágenes de processing para poder ajustar su tamaño horizontal y vertical, por lo que la “y” que representaría su tamaño vertical, esta cambia cada segundo estirando la imagen o comprimiendola, haciendo el efecto de oscilar. y con el uso de condicionales se controla el cambio de esta haciendo o que se comprima o se estire dependiendo si alcanza cierto valor.

esta etapa va a durar el tiempo que el usuario haya registrado y mediante el uso de 2 contadores se puede crear el tiempo pues al ejecutarse a 60 FPS, un contador se suma a sí mismo y mediante un condicional donde si el contador es igual a 60, el contador numero 2 se suma a sí mismo y se resetea el contador 1, logrando así controlar el tiempo de ejecución donde si el contador 2 es igual al tiempo registrado se nos llevará a la etapa número 5.
en este mismo condicional donde se calcula el tiempo, se hace el cálculo de los valores en cada segundo para la posición, velocidad y aceleración y mediante una función esta guarda en una arreglo los valores que se usará para las gráficas y para descargar el archivo excel así como para el archivo de texto. y se muestra a tiempo real los cambios de los valores en las gráficas usando las sentencias que la librería “gráfica” nos permite usar.

Etapa 5: en esta etapa se muestran las gráficas en un tamaño más grande para poderlas ver mejor y con el uso de las herramientas que nos proporciona la librería “gráfica”, podemos mover y hacer zoom a las gráficas.
también está el botón de descargar donde al pulsarlo se guardará en la carpeta del proyecto una imagen en formato jpg y un archivo excel de la debida grafica asi como un archivo txt. esto se logra con una función de processing llamada save() donde toma un screenshot de la pantalla completa por lo que para que solo salga la gráfica esta debe ponerse en tamaño completo tomar la captura de pantalla y volver al tamaño original, igual al ejecutarse a 60 FPS, el cambio del tamaño de la gráfica es casi imperceptible por lo que no molesta la experiencia del usuario y la tabla que en la etapa 4 se llenó, se guarda igualmente con la sentencia saveTable(), todo esto fue optimizado usando subrutinas y siendo esta la última etapa del proyecto.

Sub proceso KeyPressed(): en este subproceso de processing se lee del teclado la tecla que se presione y con el uso de condicionales se restringe a solo poder usar teclas numéricas, el punto y signo negativo. estas se guardan en una varible de tipo carácter por lo que hay que posteriorme convertirla en tipo real, y tienen un tope mediante la longitud de la cadena a solo poder digitar 10 caracteres pues si hay mas de eso, las operaciones pueden que arrojen resultados más grandes que lo que una variable de tipo float puede almacenar, y no se usa variables de tipo double pues estas no se pueden digitar en la funciones de processing de seno, coseno, raiz y exponente que se usarán para calcular posición, velocidad y aceleración.

Sub proceso MouseClicked(): en este subproceso de processing se programan todos los botones que el programa utiliza mediante el uso de la herramienta de processing Mousex y Mousey para saber la posición del ratón o mouse en la pantalla y saber si se encuentra en la posicion del boton.