



Laboratorio de Estructuras de Datos

Práctica 6. Aplicaciones de arreglos unidimensionales

Unidad Temática: 1. Introducción a las estructuras de datos y estructuras fundamentales

📅 👤 Profesor: Dr. Aldonso Becerra Sánchez

Índice

1	Objetivo de la tarea	1
2	Tiempo aproximado de realización	1
3	Fecha de entrega	1
4	Fecha de entrega con extensión y penalización	1
5	Introducción	1
6	Actividades a realizar	1
6.1	Actividad Inicial	1
6.2	Actividad 1	1
6.3	Actividad 2	1
6.4	Actividad 3	4
6.5	Actividad 4	4
6.6	Actividad 5	4
6.7	Actividad 6	4
6.8	Actividad 7	4
7	Contáctame	4
	References	4

6. Actividades a realizar

6.1. Actividad Inicial

Lea primero toda la práctica ⚠️. No inicie a programar sin leer todo cuidadosamente primero. Recuerde que debe generar el reporte en formato IDC con todos sus componentes.

6.2. Actividad 1

Primero genere la Introducción 📄.

6.3. Actividad 2

Información importante

Esta actividad debe entrar en la parte de *Desarrollo* 📄.

El sonido es una onda de presión que requiere un transductor de presión (un micrófono) para convertir las ondas de presión de aire (ondas sonoras) en señales eléctricas (señales analógicas). La conversión contraria se realiza mediante un altavoz (bocina), el cual convierte las señales eléctricas en ondas de presión de aire. Sele pide que grabe un frase auditiva usando el programa Wavesurfer. En la Figura 1 se puede ver este programa.

Oprimiendo el botón de botón rojo (circular) comienza la grabación de voz, sin embargo, antes de eso se requiere realizar lo siguiente:

- Oprima botón derecho sobre la zona que se encuentra debajo de la palabra “Sound”, aparecerá un menú como el de la Figura 2, ahí debe seleccionar agregar panel de “waveform”.
- Después puede oprimir el botón rojo para comenzar la grabación, se le pide que inmediatamente después de presionar el botón diga su “nombre completo” con todo y apellidos y la frase indicada abajo (después de completar oprima el botón negro cuadrado para detener el proceso). Por ejemplo el audio debe quedar grabado con “Aldonso Becerra Sánchez, quiero que sepan que mi mamá me mima y cuando fui a cantar cantar, me encontré a un ratón, ratón, le pedí papel papel, y el tonto ratón, ratón...”. Y quedará grabado algo como lo de la Figura 3.
- Posteriormente grabe el archivo como archivo .wav.

Si se oprime botón derecho sobre la zona de la forma de onda aparece un menú, ahí oprima el botón del ratón sobre propiedades (Figura 4). Ahí se pueden apreciar las características de la señal grabada; por ejemplo, la tasa de muestreo (el número de puntos que son obtenidos/discretizados (ver Figura 5) de la señal analógica continua [forma de onda], el número de canales [generalmente se llama sonido

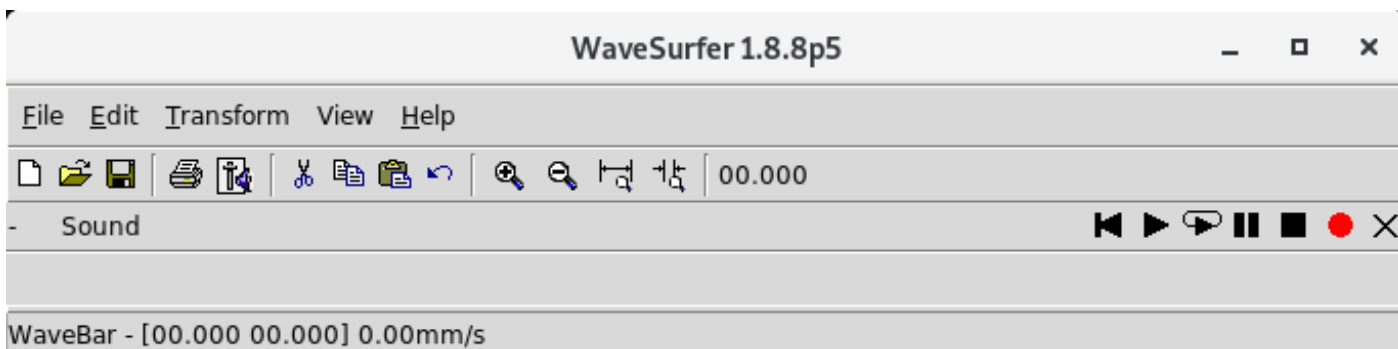


Figura 1. Wavesurfer.

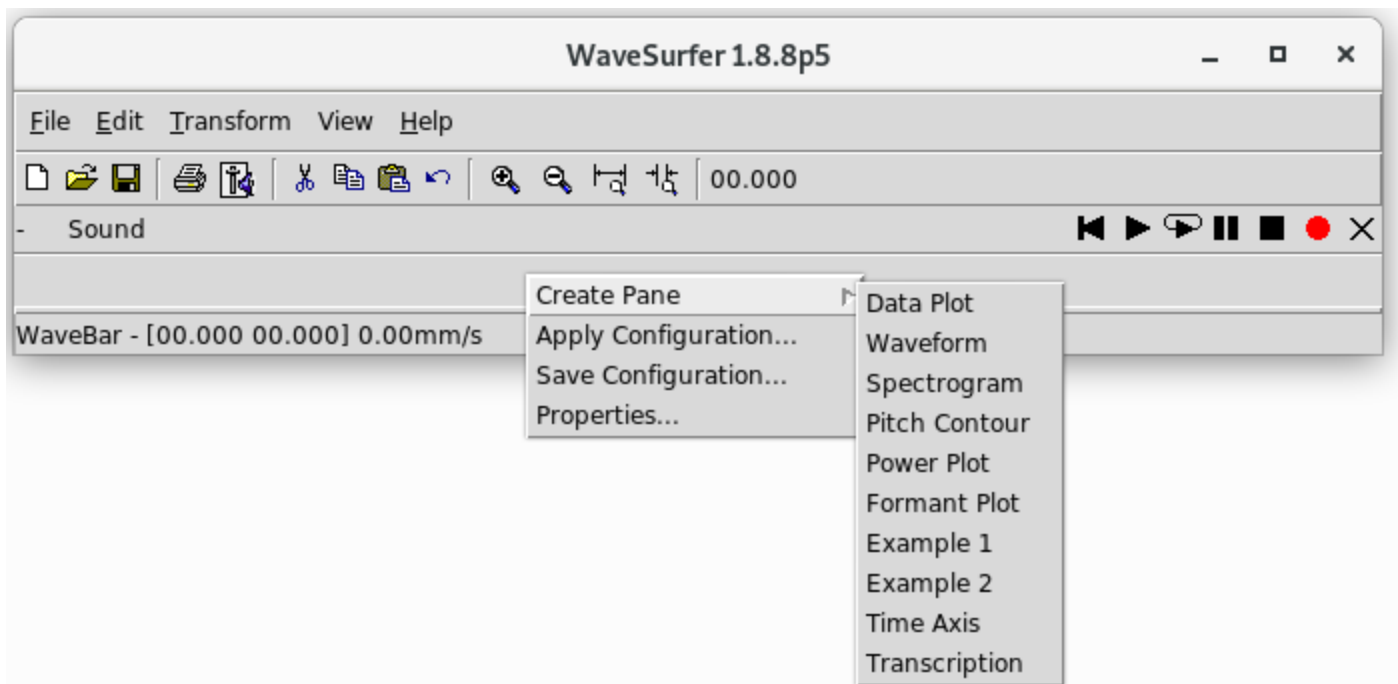


Figura 2. Parámetros audio.

estereofónico o estéreo al sonido grabado y reproducido en dos canales. Los discos de vinilo, los CD audio, la mayoría de las estaciones de radio FM, casetes y la totalidad de canales de TV y televisión vía satélite transmiten señales de audio estéreo. El propósito de grabar en sonido estereofónico es el de recrear una experiencia más natural al escucharlo, y donde, al menos en parte, se reproducen las direcciones izquierda y derecha de las que proviene cada fuente de sonido grabada, y el número de bits usados para codificar/discretizar la señal numérica que es convertida por el micrófono.

Tomando como base el programa que se proporciona en Java, el cual hace una lectura de un archivo de audio (wav), se le pide lo siguiente:

1. Acople el programa de la clase `AudioFileRecord` para que los datos del `buffer2[]` del método `leerAudio` ahora queden guardados en la clase `ArregloListaInfoEstaticaNumeros`. Para esto debe crear un método, en la clase `ArregloListaInfoEstatica`, que reciba el arreglo `buffer[]` (para que se pueda tener esta funcionalidad desde la super clase, no solo con valores numéricos) y lo guarde en el contenido del arreglo interno de la clase `ArregloListaInfoEstatica` (en la variable `datos[]`, sobrescriba la variable interna del arreglo de los datos para que no se dupliquen los almacenamientos de memoria). La idea es que todo procesamiento se haga sobre una variable de tipo `ArregloListaInfoEstaticaNumeros`.
2. Acople el programa para que el método `escribirAudio` pueda

guardar el `buffer2[]` como un arreglo[], para eso debe crear un método en la clase `ArregloListaInfoEstatica` (por ejemplo, `public Object[] leerArreglo()`) que proporcione una copia de los datos (no debe regresar el arreglo original, sino sólo una copia, esto garantiza la encapsulación) para después usar esa copia y almacenarla en `buffer2[]` como `double`, para que así pueda ser usado el método `writeFrames`. La idea es que la información guardada en la variable de tipo `ArregloListaInfoEstaticaNumeros` sea extraída para que `wirteFrames` pueda hacer su trabajo como arreglo con [].

3. A partir de aquí todas las modificaciones que se hagan sobre el archivo de audio deberán ser sobre la variable temporal de tipo `ArregloListaInfoEstaticaNumeros`.
4. Se pide que haga una serie de métodos de manipulación del audio. Por lo cual deberá elegir el lugar correcto para guardarlos.
5. Con la señal de voz discretizada ($x[n]$, valor obtenido ya en el archivo .wav, y que el método `leerAudio()` ya proporciona) aplicar un filtro FIR de primer orden en las altas frecuencias, dado por $y[n] = x[n] + \alpha x[n - 1]$, donde α es un valor configurable, habitualmente con 0.95 o 0.97. Este proceso se llamará `void preEnfasis()`. Verifique en qué cambio el audio, documento este hecho.

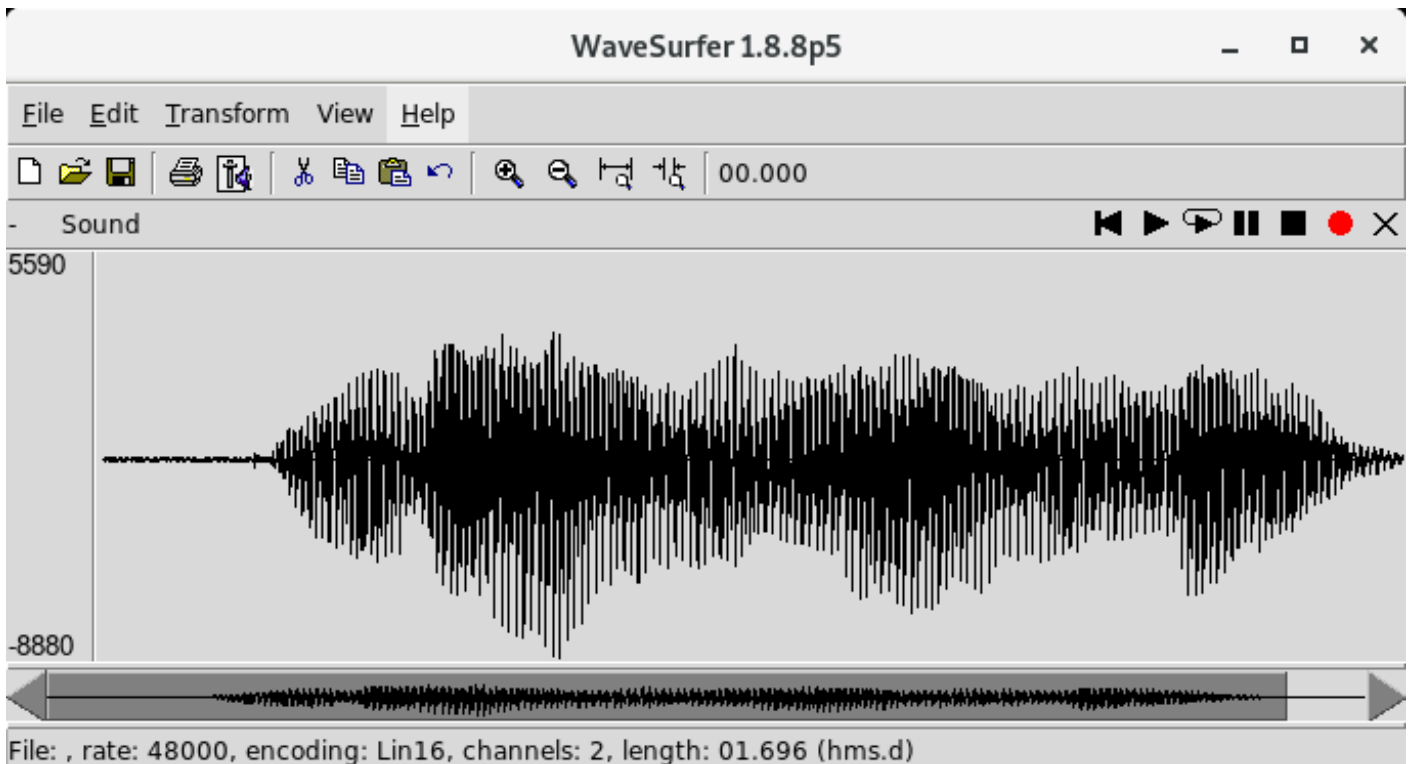


Figura 3. Ejemplo audio.

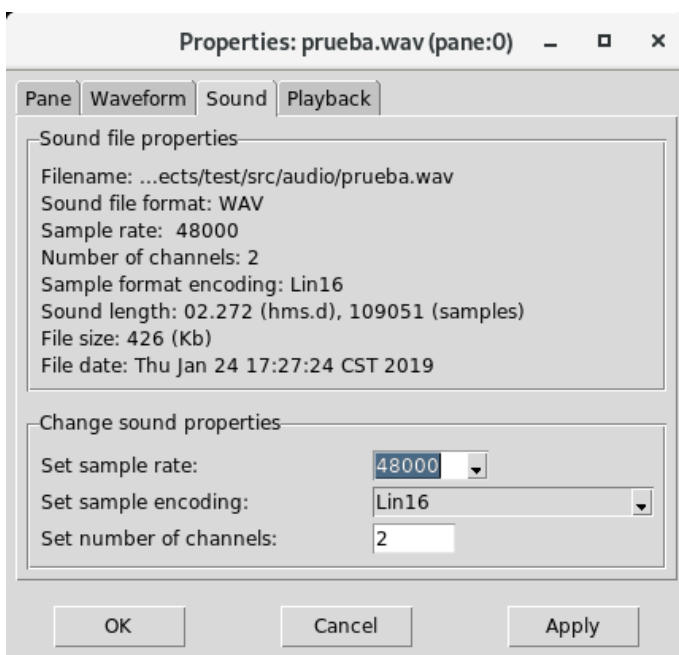


Figura 4. Parámetros Wavesurfer.



Figura 5. Cuantificación.

6. Un método en la clase `AudioFileRecord` que incremente el volumen del archivo de audio almacenado en la variable temporal `buffer2` de tipo `ArregloListaInfoEstaticaNumeros`: public void `volumen(int intensidad)`. Intensidad es la cantidad de volumen a subir, la cual debe ser proporcionada por el usuario, si es positiva, se sube el volumen, si es negativa se disminuye el volumen.
7. Pruebe la funcionalidad de estos métodos guardando el archivo de audio (variable temporal de tipo `ArregloListaInfoEstaticaNumeros` convertida a arreglo `[]` para ser pasada a `writeFrames`) cada vez que hace una modificación del archivo temporal de audio.

8. Haga un método para acelerar la pista de audio (sonido tipo Alvin y las ardillas). Para hacer esto existen varias maneras, la que se pide que use es reducir el número de muestras por unidad de tiempo (por ejemplo a la mitad, las muestras como en la figura 5). Por ejemplo, si originalmente eran 10000 muestras (tramas* número de canales), el nuevo archivo de audio quedará de 5000. El proceso para conseguir esto es agarrar dos muestras consecutivas y promediarlas (esto es acelerar el sonido al doble), creando un nuevo arreglo ahora con la mitad de muestras. Recuerde que todo este proceso es sobre una variable de tipo `ArregloListaInfoEstaticaNumeros`. El incremento en la velocidad debe ser un parámetro, No siempre debe ser el doble, este valor debe ser configurable en el método.
9. Haga un método para retrasar la pista de audio (sonido fantasma). Para hacer esto existen varias maneras, la que se pide que use es aumentar el número de muestras por unidad de tiempo (ver figura 5). Por ejemplo, si originalmente eran 10000 muestras (tramas* número de canales), el nuevo archivo de audio quedará de 20000 (en este caso fue al doble). El proceso para conseguir esto es insertar la muestra "n" en un nuevo arreglo de tipo `ArregloListaInfoEstaticaNumeros`, y después de cada muestra "n" (o antes) se debe insertar una muestra inventada que será el relleno, la cual se obtiene como un promedio de la muestra actual "n" más la muestra "n+1" (o n-1, dependiendo). El decremento en

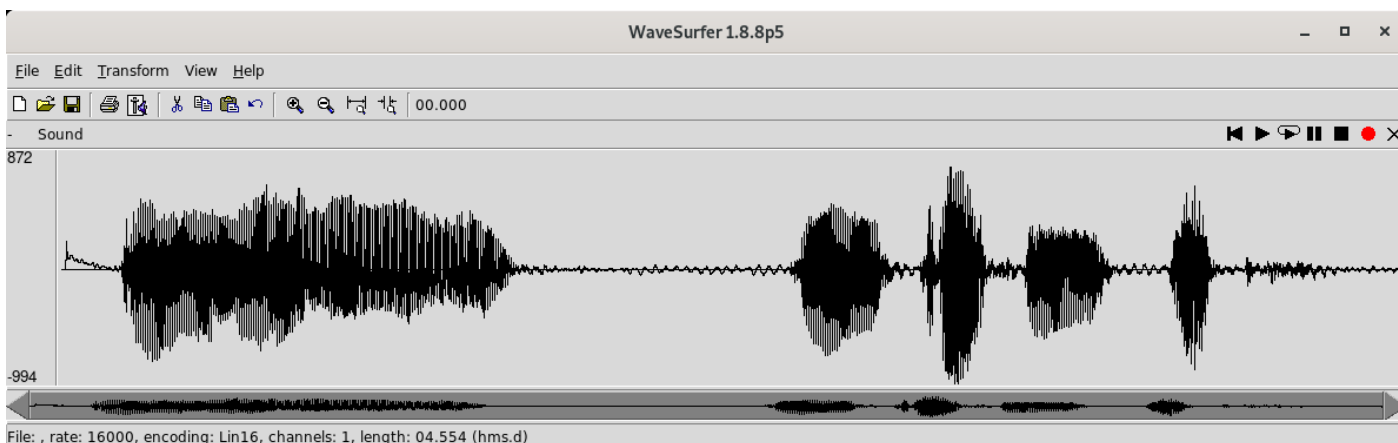


Figura 6. Ejemplo silencio audio.

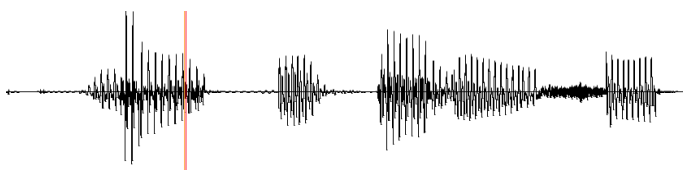


Figura 7. Ejemplo 1 de inversión de audio en X.

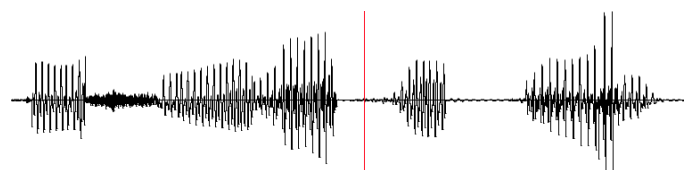


Figura 9. Ejemplo 1 de inversión en Y.

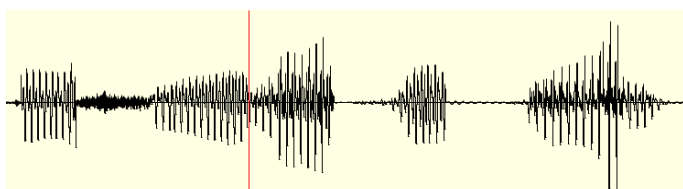


Figura 8. Ejemplo 2 de inversión de audio en X.

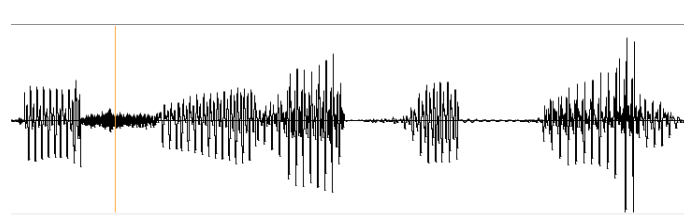


Figura 10. Ejemplo 2 de inversión en Y.

la velocidad debe ser un parámetro, el cual debe ser configurado por el usuario.

10. Haga un método para que usted diseñe la manera de eliminar el silencio existente en un archivo de audio (analiza cómo distinguir el silencio del no silencio). Por ejemplo el mostrado en la Figura 6:

Fíjese en el silencio puede existir en cualquier posición del audio, normalmente es antes o después de palabras pronunciadas (flechas en rojo). Ese silencio debe ser removido.

11. Haga que el audio grabado sufra una inversión en el eje x (se volteé al revés en tiempo, es decir lo que estaba a la izquierda ahora estará a la derecha, y así sucesivamente). Tal como se muestra la Figura 7 y 8.


12. Haga que el audio grabado sufra una inversión en el eje y (se volteó de cabeza en intensidad, es decir lo que estaba arriba ahora estará abajo, y así sucesivamente). Tal como se muestra la Figura 9 y 10.

13. Haga unas grabaciones de vocales (pronunciadas una vocal sola por cada archivo). Grabe varias veces una misma vocal, por ejemplo unas 5 veces. Posteriormente obtenga la energía que producen cada una de ellas (la energía de una señal de audio se obtiene elevando al cuadrado cada una de las muestras individuales, posteriormente se suman todas). Verifique si a través de la energía puede clasificar vocales. Haga pruebas y documente los resultados. 14. Pruebe el funcionamiento del programa grabando el arreglo resultante ArregloListaInfoEstaticaNumeros en un archivo .wav correspondiente en disco.


6.4. Actividad 3

Pruebe el funcionamiento del programa de la actividad 2 con todo y sus capturas de pantalla.

6.5. Actividad 4

Realice la sección de Código agregado  (diagrama de clases UML).





6.6. Actividad 5

Realice la sección de Pre-evaluación  (use los lineamientos establecidos).

6.7. Actividad 6

Finalmente haga las Conclusiones .


6.8. Actividad 7


Subir los entregables (pdf  y zip  con código )  a Moodle.


7. Contáctame

Puedes contactarme a través de los siguientes medios.

 <https://moodle.ingsoftware.uaz.edu.mx/>

 a7donso@gmail.com

 Cubículo

 Salón CC2-IS

Referencias

- [1] O. Cairo y S. Guardati, *Estructura de datos*. McGraw-Hill.

- 182 [2] L. Joyanes Aguilar, *Fundamentos de programación, algoritmos*
183 *u estructura de datos*. McGraw-Hill.
- 184 [3] M. A. Weiss, *Estructura de datos en Java*. Addison Wesley.