**Universidad San Carlos de Guatemala**
**Facultad de Ingeniería**
**Escuela de Ciencias y Sistemas**
**Introducción a la Programación y Computación 2**

**Inga. Claudia Liceth Rojas Morales**
**Ing. Marlon Antonio Pérez Türk**
**Ing. José Manuel Ruiz Juárez**
**Ing. Edwin Estuardo Zapeta Gómez**
**Ing. Byron Rodolfo Zepeda Arevalo**

**Tutores de curso:**
**Josué Alfredo González**
**Marvin Daniel Rodríguez**
**Daniel Arturo Alfaro**
**Sergio Felipe Zapeta**
**Erwin Fernando Vásquez**

# PROYECTO 1

## OBJETIVO GENERAL

Desarrollar una solución integral que implemente tipos de datos abstractos (TDA) y visualización de datos (Graphviz) bajo el concepto de programación orientada a objetos (POO).

## OBJETIVOS ESPECÍFICOS

- Implementar POO para el desarrollo de la solución a través de lenguaje Python.
- Utilizar estructuras de programación secuenciales, cíclicas y condicionales
- Visualizar TDA's por medio de la herramienta Graphviz.
- Utilizar archivos XML como insumos para la lógica y comportamiento de la solución.

## ENUNCIADO

El Centro de Investigaciones de la Facultad de Ingeniería está experimentando un método para lograr comprimir señales de audio, por lo que se han enfocado en dos parámetros propios de las ondas de sonido: Frecuencia y Amplitud, estos parámetros describen la señal de audio en función del tiempo. La frecuencia es la cantidad de ciclos que se realizan en un segundo y se mide en Hertz (Hz), mientras que la amplitud representa la altura de la onda y hace referencia a la intensidad del sonido, la amplitud se mide en decibelios (Db).

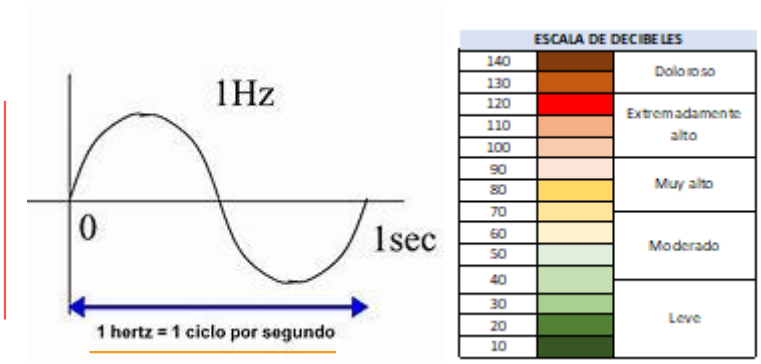


**Figura No. 1** – Frecuencia y Amplitud de una señal de audio

La figura No. 1 muestra gráficamente los parámetros que está considerando el Centro de Investigaciones de la Facultad de Ingeniería para su experimento de compresión de señales de audio.

El Centro de Investigación de la Facultad de Ingeniería ha abordado el problema de diseño para la compresión de señales de audio como un problema combinatorio NP-Hard ya que analizó que algunas de las situaciones comunes observadas cuando se resuelven instancias muy grandes de un problema NP-Hard son: Fuerte requerimiento de tiempo y fuerte demanda de recursos, por lo que optó por un método para resolver este tipo de problemas aplicando una metodología de agrupamiento.

Para implementar esta metodología de agrupamiento, el Centro de Investigación ha definido una matriz de tiempo (t), amplitud (A) y frecuencias (f) para distintas señales de audio S [t][A], transformarla en una matriz de patrones de frecuencia y agrupar las tuplas con el mismo patrón. La matriz de patrones de frecuencias para una matriz de frecuencias dada es la matriz binaria que indica en qué tiempos y amplitudes hay frecuencias en la señal de audio en estudio.

**Ejemplo:**

Para la siguiente señal de audio, se genera la matriz de frecuencia siguiente:

| | | AMPLITUD (db) | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| TIEMPO (seg) | 1 | 2 | 3 | 0 | 4 |
| | 2 | 0 | 0 | 6 | 3 |
| | 3 | 3 | 4 | 0 | 2 |
| | 4 | 1 | 0 | 1 | 5 |
| | 5 | 0 | 0 | 3 | 1 |

A continuación, se construye la matriz de patrones correspondiente a la señal de audio

| | | AMPLITUD (db) | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| TIEMPO (seg) | 1 | 1 | 1 | 0 | 1 |
| | 2 | 0 | 0 | 1 | 1 |
| | 3 | 1 | 1 | 0 | 1 |
| | 4 | 1 | 0 | 1 | 1 |
| | 5 | 0 | 0 | 1 | 1 |

En la matriz de patrones se puede observar que las filas 1 y 3 tienen los mismos patrones de frecuencia, así como también las filas 2 y 5. Entonces, habrá tres grupos considerando el tercer grupo formado simplemente por la fila 4. La cardinalidad de un grupo es definida por el número de tuplas incluidas en él.

La matriz reducida de frecuencias se obtiene sumando las tuplas en los grupos identificados en la matriz de patrones, esta matriz reducida será:

| | | AMPLITUD (db) | | | |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 |
| TIEMPO (seg) | Grupo 1 (tiempo 1 y 3) | 5 | 7 | 0 | 6 |
| | Grupo 2 (tiempo 2 y 5) | 0 | 0 | 9 | 4 |
| | Grupo 3 (tiempo 4) | 1 | 0 | 1 | 5 |

Usted ha sido contratado por la Facultad de Ingeniería para diseñar un programa que acepte "n" señales de audio (expresadas como matrices de frecuencia) y por cada una de estas señales de audio debe obtener las matrices reducidas con la misma matriz de patrones.

## GRÁFICAS

Se deberá utilizar la herramienta Graphviz para crear un grafo que muestre de manera gráfica las señales de audio ingresadas desde el archivo de entrada procesado y la matriz reducida producida a partir de la misma.

La figura 2 y 3 presentan un ejemplo de las gráficas a generar.

En la figura 2, “Prueba 1” será el nombre de la señal de audio, “t” y “A” serán las dimensiones (tiempo y amplitud) de la señal de audio y en la parte derecha se mostrarán las frecuencias que contiene la matriz de frecuencias de dicha señal.

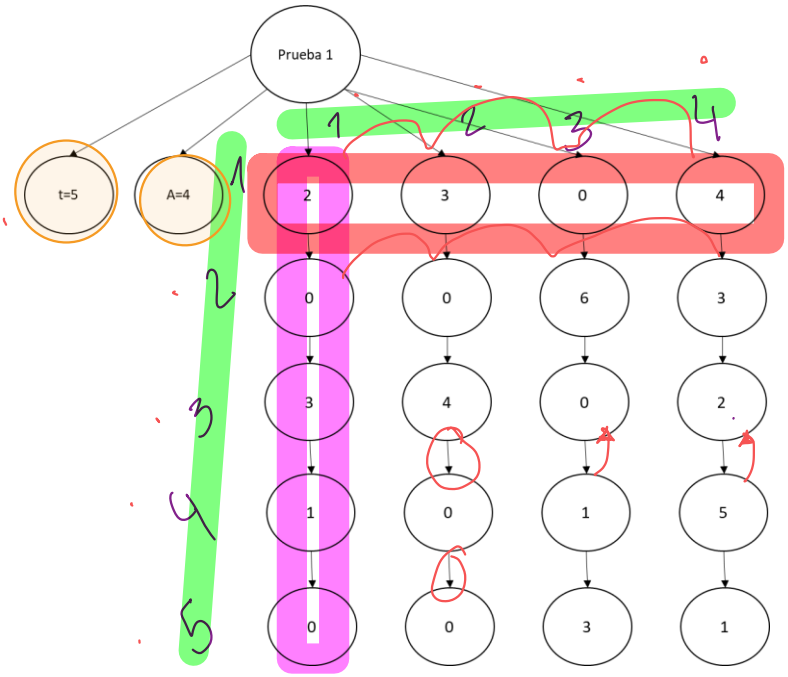


**Figura No. 2** – Onda original de la señal de audio

En la figura 3, se presenta la matriz reducida para la señal “Prueba 1”, “g” y “A” serán las dimensiones (grupos y amplitud) de la señal de audio y en la parte derecha se mostrará la matriz de frecuencias reducida para la señal de audio.

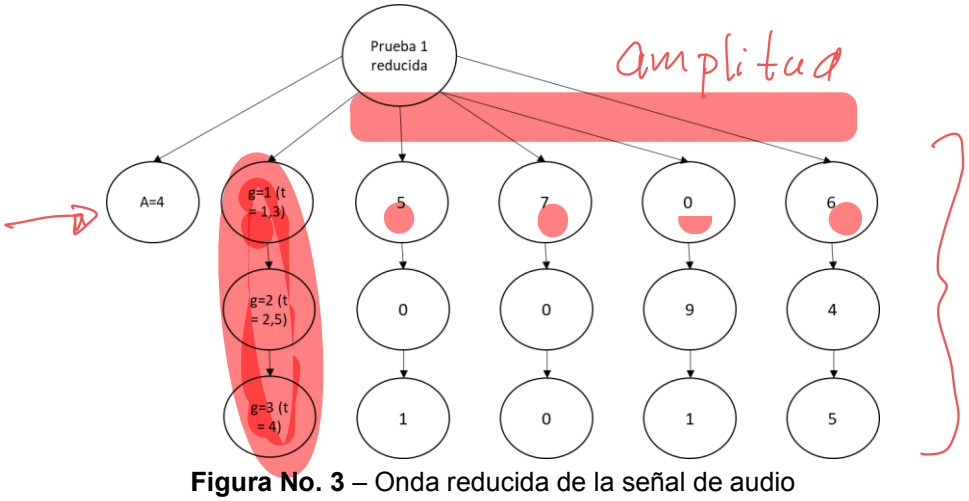


**Figura No. 3** – Onda reducida de la señal de audio

Las figuras 2 y 3 son representativas, deberán crear sus propias gráficas lo más detalladas posible.

## Archivos de Entrada

Los archivos de entrada consistirán en archivos con extensión y estructura xml en el cual se limitará a utilizar únicamente las siguientes etiquetas:

- **senales**: Esta etiqueta engloba toda la información que el archivo contendrá.
- **senal**: esta etiqueta será la que indica que una nueva matriz de frecuencias será creada para su respectivo análisis y únicamente puede estar dentro de la etiqueta **senales** y puede tener los siguientes atributos:
  - **nombre**: este contendrá el identificador de la señal de audio (se deberá validar la existencia de matrices con el mismo nombre, para mantener la consistencia de los datos). Si la señal ya existe, será reemplazada por la nueva señal y el programa debe informar de esta situación.
  - **t**: representa la dimensión tiempo, y definirá la cantidad de filas que tendrá la matriz de frecuencias. Regla: t > 0 y t <= 3600.
  - **A**: representa la dimensión Amplitud, y definirá la cantidad de columnas que tendrá la matriz de frecuencias. Regla: A > 0 y A <= 130.
- **dato**: esta etiqueta únicamente podrá estar dentro de la etiqueta **senal** y contendrá los valores respectivos de la frecuencia en un tiempo y amplitud dados, esta etiqueta puede tener los siguientes atributos.
  - **t**: será el tiempo (fila) de la matriz de frecuencias y no puede ser mayor al atributo **t** de la señal.
  - **A**: será la amplitud (columna) de la matriz de frecuencias y no puede ser mayor al atributo **A** de la señal.

**Ejemplo de Entrada[1]:**

```
<?xml version="1.0"?>
<senales>
  <senal nombre="Prueba 1" t="5" A="4">
    <dato t="1" A="1">2</dato>
    <dato t="1" A="2">3</dato>
    <dato t="1" A="3">0</dato>
    <dato t="1" A="4">4</dato>
    <dato t="2" A="1">0</dato>
    <dato t="2" A="2">0</dato>
    <dato t="2" A="3">6</dato>
    <dato t="2" A="4">3</dato>
    <dato t="3" A="1">3</dato>
    <dato t="3" A="2">4</dato>
    <dato t="3" A="3">0</dato>
    <dato t="3" A="4">2</dato>
```

[1] Si un tiempo y Amplitud no están definidos en una señal ingresada en el archivo de entrada XML, toma como valor default cero (0).

```
    <dato t=”4” A=”1”>1</dato>
    <dato t=”4” A=”2”>0</dato>
    <dato t=”4” A=”3”>1</dato>
    <dato t=”4” A=”4”>5</dato>
    <dato t=”5” A=”1”>0</dato>
    <dato t=”5” A=”2”>0</dato>
    <dato t=”5” A=”3”>3</dato>
    <dato t=”5” A=”4”>1</dato>
  </senal>
  ....
</senales>
```

## Archivos de Salida

Los archivos de salida consistirán en archivos con extensión y estructura xml en el cual se limitará a utilizar únicamente las siguientes etiquetas:

- **senalesReducidas**: Esta etiqueta engloba toda la información que el archivo contendrá.
- **senal**: esta etiqueta será la que indica una matriz reducida generada por el programa y únicamente puede estar dentro de la etiqueta **senalesReducidas** y puede tener los siguientes atributos:
  - **nombre**: este contendrá el identificador de la señal de audio.
  - **A**: representa la dimensión Amplitud, y definirá la cantidad de columnas que tendrá la matriz de frecuencias reducidas. Regla: A > 0 y A <= 130.
- **grupo**: esta etiqueta definirá información de un grupo de la señal reducida y puede tener el siguiente atributo:
  - **g**: Representa el número de grupo
- **tiempos**: Esta etiqueta representa los tiempos que están agrupados en el grupo “g” y únicamente puede estar dentro de una etiqueta **grupo**.
- **datosGrupo**: Esta etiqueta representa el conjunto de frecuencias por amplitud que están agrupados en el grupo “g” y únicamente puede estar dentro de una etiqueta **grupo**.
- **dato**: esta etiqueta únicamente podrá estar dentro de la etiqueta **datosGrupo** y contendrá los valores respectivos de la frecuencia en un grupo y amplitud dados, esta etiqueta puede tener los siguientes atributos.
  - **A**: será la amplitud (columna) de la matriz de frecuencias y no puede ser mayor al atributo **A** de la señal.

**Ejemplo de Salida:**

```
<?xml version="1.0"?>
<senalesReducidas>
  <senal nombre="Prueba 1" A=”4”>
      <grupo g=”1”>
          <tiempos>1,3</tiempos>
          <datosGrupo>
```

```
                <dato A="1">5</dato>
                <dato A="2">7</dato>
                <dato A="3">0</dato>
                <dato A="4">6</dato>
            </datosGrupo>
        </grupo>
        <grupo g="2">
            <tiempos>2,5</tiempos>
            <datosGrupo>
                <dato A="1">0</dato>
                <dato A="2">0</dato>
                <dato A="3">9</dato>
                <dato A="4">4</dato>
            </datosGrupo>
        </grupo>
        <grupo g="3">
            <tiempos>4</tiempos>
            <datosGrupo>
                <dato A="1">1</dato>
                <dato A="2">0</dato>
                <dato A="3">1</dato>
                <dato A="4">5</dato>
            </datosGrupo>
        </grupo>
    </senal>
    ...
</senalesReducidas>
```

La aplicación deberá contar con un menú en consola, con las siguientes opciones

```
Menú principal:
        1. Cargar archivo
        2. Procesar archivo
        3. Escribir archivo salida
        4. Mostrar datos del estudiante
        5. Generar gráfica
        6. Inicializar sistema
        7. Salida
```

1. **Cargar Archivo:** Esta Opción solicitará la ruta del archivo a cargar.

```
Opcion Cargar Archivo:
Ingrese la ruta del archivo:
```

2. **Procesar el Archivo:** Esta opción será la encargada de procesar la información cargada en memoria, durante el proceso se deben ir mostrando mensajes al usuario para tener el conocimiento de lo que está pasando en el sistema.

```
> Calculando la matriz binaria...
> Realizando suma de tuplas...
```

3. **Escribir Archivo de salida:** Esta opción será la encargada de escribir el archivo con la salida específica.

```
Escribir una ruta especifica: C:/escritorio/ipc2/final.xml
Se escribio el archivo satisfactorio
```

permiso

4. **Mostrar datos del estudiante:** Mostrar los datos del estudiante, carné, nombre, curso, carrera y semestre.

```
> Juan Pablo Osuna de Leon
> 201503911
> Introduccion a la Programación y computación 2 seccion "A"
> Ingenieria en Ciencias y Sistemas
> 4to Semestre
```

5. **Generar gráfica**: El programa deberá permitir que el usuario elija una de las señales de audio ingresadas en el archivo de entrada y mostrará la gráfica cómo la que se indica en la sección de gráficas.

## CONSIDERACIONES

Se deberá implementar una lista simplemente enlazada, creada por el estudiante, creando una clase Nodo, y una clase lista, de tal manera que al recorrer la lista se pueda validar la existencia de los nombres de las señales de audio, y en base en esta lista poder realizar un reporte gráfico de la misma.

Debe utilizarse versionamiento para el desarrollo del proyecto. Se utilizará la plataforma **Github** en la cual se debe crear un repositorio en el que se gestionará el proyecto. Se deben realizar 4 releases o versiones del proyecto (se recomienda realizar una por semana del tiempo disponible). Se deberá agregar a su respectivo auxiliar como colaborador del repositorio. El último release será el release final y se deberá de realizar antes de entregar el proyecto en la fecha estipulada.

Szapeta

## DOCUMENTACIÓN

Para que el proyecto sea calificado, el estudiante deberá entregar la documentación utilizando el formato de ensayo definido para el curso. En el caso del proyecto, el ensayo debe tener entre 4 y 7 páginas de contenido, este máximo no incluye los apéndices o anexos donde se pueden mostrar modelos y diseños utilizados para construir la solución. Esta documentación debe incluir los diagramas de clases y opcionalmente los diagramas de actividades que utilice para diseñar la solución de software.

## RESTRICCIONES

- Solo se permitirá la utilización de los IDEs discutidos en el laboratorio.
- Uso obligatorio de programación orientada a objetos (POO).
- El nombre del repositorio debe de ser **IPC2_Proyecto1_#Carnet.**
- El estudiante debe entregar la documentación solicitada para poder optar a la calificación.
- Los archivos de entrada no podrán modificarse.
- Los archivos de salida deben llevar la estructura mostrada en el enunciado obligatoriamente.
- Las estructuras de datos a utilizar en la solución para manejo de matrices y el TDA para el reporte deben ser definidas y manejadas por el estudiante.
- Se calificará de los cambios realizados en el cuarto release. Los cambios realizados después de ese release no se tomarán en cuenta.
- Para dudas concernientes al proyecto se utilizarán los foros en UEDI de manera que todos los estudiantes puedan ver las preguntas y las posteriores respuestas.
- **NO HABRÁ PRÓRROGA.**

## ENTREGA

- La entrega será el **3 de septiembre** a las 11:59 pm como máximo.
- La entrega será por medio de la UEDI.
- La documentación debe estar subida en el repositorio en una carpeta separada.
- Para entregar el proyecto en UEDI se deberá subir un archivo de texto con el link del repositorio.