Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

IPC MUSIC

201903909– Diego Alexander Acetún Chicol

Resumen

El Reproductor de música permite cargar un archivo xml con las canciones, llevando estas su artista y álbum, como todo reproductor de música permite avanzar y regresar canciones

El reproductor de música permite al usuario crear sus propias listas de reproducción, estas se pueden reproducir de dos diferentes modos, modo normal y modo aleatorio. El reproductor también es capaz de exportar en un archivo xml las listas de reproducción creadas para que después puedan ser cargadas sin necesidad de volver a crearlas

Además también puede exportar un archivo html para ver las canciones más reproducidas, también puede generar un grafo que representa el comportamiento de las listas que contienen los artistas, álbumes y canciones.

Adicional a ello también puede reproducir todas las canciones de los álbumes disponibles con solo seleccionarlo.

Palabras clave
Lista Doble, Lista Circular, Objeto, Pila, Apuntador

Abstract

The music player allows to load an xml file with the songs, taking these its artist and album, like all music player allows to go forward and back songs.

The music player allows the user to create their own playlists, these can be played in two different modes, normal mode and shuffle mode. The player is also capable of exporting the created playlists in an xml file so that they can be loaded later without the need to recreate them.

It can also export an html file to see the most played songs, it can also generate a graph that represents the behavior of the playlists containing the artists, albums and songs.

In addition to that you can also play all the songs of the available albums by simply selecting it.

Key words

Double List, Circular List, Object, Stack, Pointer

Introducción

El siguiente proyecto trata de hacer un reproductor de música, hecho en el lenguaje de programación python utilizando la versión 3.9, utilizando listas enlazadas, tanto dobles como circulares, esto para poder dar los atributos que necesitemos a cada lista, algo que no podríamos hacer con las listas propias de python.

También se utilizará programación orientada a objetos, el reproductor será capaz de mostrar nombre, álbum, artista e imagen de la canción cuando esta se esté reproduciendo.

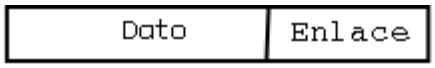
Las listas de reproducción serán almacenadas en listas circulares, esto para que todas las canciones tengan un siguiente y un anterior, incluyendo la primera y la última canción.

Las listas dobles servirán para almacenar los objetos, se utilizará un objeto Canción y las listas, las cuáles tendrán atributos según conveniencia.

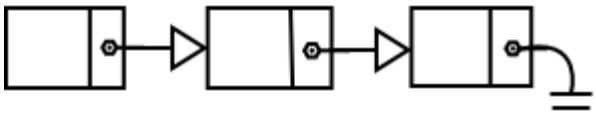
Desarrollo del tema

Listas Enlazadas: Es un TDA que nos permite almacenar datos de forma organizada, esta estructura es dinámica por lo que no necesitamos conocer los elementos que esta contendrá, ya que puede almacenar cuantos elementos necesitemos.

Lista Enlazada Simple: En este tipo de lista cada elemento apunta al siguiente, excepto al último que apunta a nulo.



Estructura de un nodo



Lista enlazada

Figura I Lista Simple Enlazada.

Lista Doblemente Enlazada: En esta lista a diferencia de la lista enlazada simple cada elemento apunta hacia el siguiente y hacia el anterior, excepto el primero que su anterior apunta a nulo y el ultimo que su siguiente apunta a nulo.

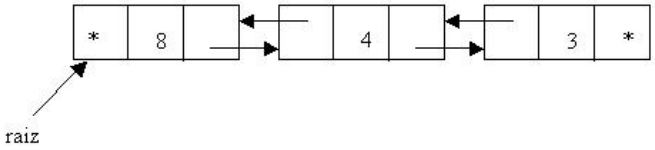


Figura II Lista Doblemente Enlazada.

Lista Doblemente Enlazada Circular: Esta lista es muy similar a la la lista doblemente enlazada, la diferencia es que el anterior del primero es el último y el siguiente del último es el primero.

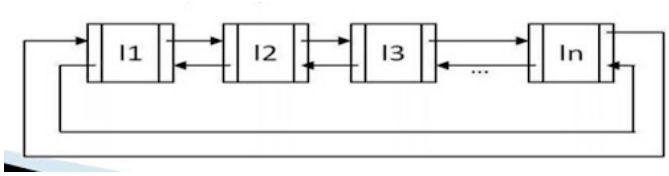


Figura III Lista Doblemente Enlazada Circular.

Interfaz Gráfica

Para el desarrollo de la interfaz gráfica se utilizó tkinter, esta librería se considera un estándar para la interfaz gráfica de usuario (GUI) para Python y es el que viene por defecto con la instalación para Microsoft Windows.

Esta librería contiene una gran variedad de elementos para utilizarlos en la interfaz, algunos de los utilizados en este proyecto fueron:

- Button
- Label
- PhotoImage
- Text
- RadioButton
- ComboBox

Obteniendo la siguiente interfaz.

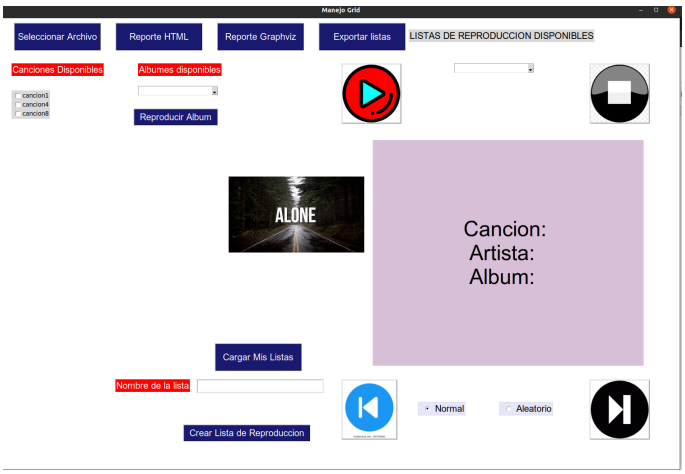


Figura IV Interfaz gráfica.

Manejo de las listas

Las listas se manejaron de la siguiente manera:

Se creaba una lista doblemente enlazada que se llenaba con objetos de tipo Canción.

Se creaba otra lista doblemente enlazada (Lista Álbumes) la cual se llenaba con listas de canciones, creando estas listas con la lista de canciones creada anteriormente.

Por último se creaba otra lista doblemente enlazada (Lista Artistas) la cual se llenaba con listas de álbumes con la lista creada anteriormente.

Se muestra el uso de las listas en la siguiente figura.

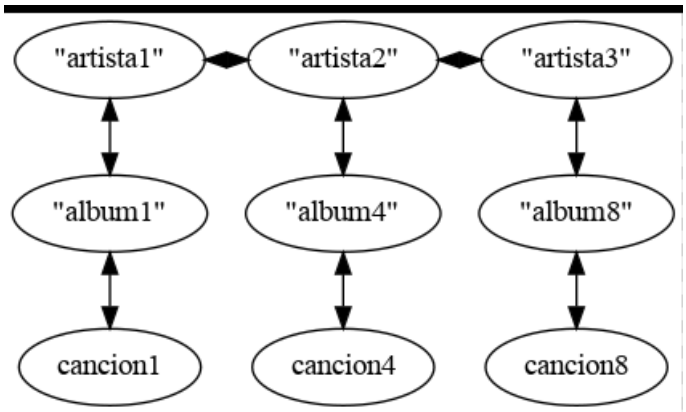


Figura IV Uso de listas.

Manejo de las listas de reproducción

Para la creación de estás listas se hizo uso de listas circulares doblemente enlazadas, ya que las canciones de la lista de reproducción se tenían que seguir reproduciendo siempre que se avanzara o se regresara la canción, sin importar si esta era la primera o la última.

Cómo se podían crear varias listas de reproducción, estas fueron almacenadas en una lista doblemente enlazada.

Modo de reproducción normal

El primer modo de reproducción se logró recorriendo la lista circular doblemente enlazada, cada vez que se presionaba el botón siguiente pasamos al siguiente elemento de la lista circular, de la misma manera cuando se presionaba el botón anterior pasamos al elemento anterior de la lista circular.

Modo de reproducción aleatorio

El segundo modo de reproducción se hizo con la ayuda de una lista propia de python, la cual se utilizó como pila, se trabajó de la siguiente manera:

Se generó un número aleatorio entre 0 y el tamaño de la lista menos uno, esto dado que el índice se comienza a contar a partir de 0

Este número generado se utilizaba para sacar de la lista este índice, y la canción que estaba dentro de este índice era la primera en ser reproducida.

Al presionar el botón siguiente, de igual forma se generó un número aleatorio entre 0 y el tamaño de la lista menos uno, se elimina de la lista la canción que tenía este índice y se agregaba al final de la lista la canción actual que se estaba reproduciendo.

Al presionar el botón anterior se elimina la última canción de la pila se reproduce esta y la que se estaba reproduciendo se agrega al final de la pila.

Manejo de la reproducción por álbum

El tercer modo de reproducción se hizo similar al primero (Modo de reproducción normal) a diferencia que cuando se cargaba el archivo xml se creaban las listas circulares con las canciones correspondientes a cada álbum

Generación de reportes HTML

Para esto se utilizó el manejo de archivos en python.

Lo primero es abrir el fichero, al abrir el fichero nos comunicamos con el sistema operativo, que sabe donde se encuentran almacenados los datos de cada archivo.

Modos para el manejo de archivos

- **r (Solo lectura)** El fichero solo se puede leer. Es el modo por defecto si no se indica.
- **w (Sólo escritura)** En el fichero solo se puede escribir. Si ya existe el fichero, machaca su contenido.
- **a (Adición)** En el fichero solo se puede escribir. Si ya existe el fichero, todo lo que se escriba se añadirá al final del mismo.
- **x** Como ‘w’ pero si existe el fichero lanza una excepción.
- **r+** Lectura y escritura. El fichero se puede leer y escribir.

Para hacer los reportes se utilizó el modo w (sólo escritura) ya que vamos a escribir código html en el archivo.

Debemos recorrer la lista de reproducción seleccionada al momento de generar el reporte, y se crearán las filas y columnas necesarias.

Generación del reporte de Graphviz

Para esta funcionalidad se hizo uso de la librería de graphviz que hay en python.

Esta librería no ayuda a escribir el código de graphviz, la librería ofrece dos clases: Graph y Diagraph.

En este proyecto se utilizó la clase Diagraph para la realización del grafo.

Entre los métodos más importantes están los siguientes:

Para crear el grafo

g = Digraph('nombre', format='png')

Para crear un nodo

g.node(identificador_unico, texto)

Para unir dos nodos

g.edge(identificadorNodo1, identificadorNodo2)

Conclusiones

Al usar listas enlazadas se obtuvieron algunas ventajas y desventajas de las mismas, las cuales se presentan a continuación.

Ventajas

- Trabaja con memoria dinámica, por lo que se puede insertar y extraer elementos en cualquier momento.
- No hay necesidad de grandes cantidades de memoria contigua.
- La memoria se adapta al número de elementos almacenados en la lista.
- Cada lista puede tener los atributos que necesitemos

Desventajas

- Se consume más memoria debido a que hay que almacenar los nodos con sus atributos.

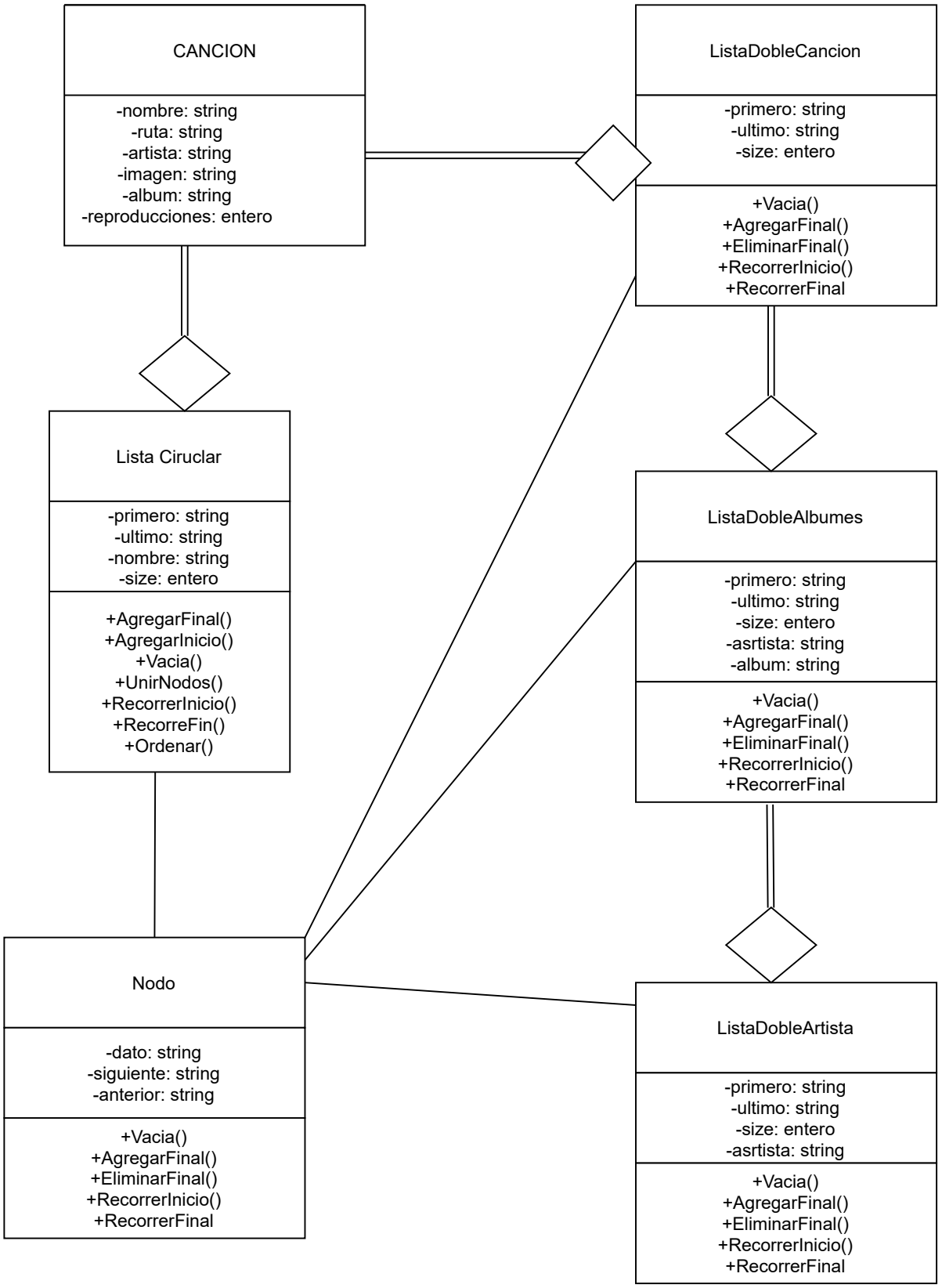
Referencias bibliográficas

https://calcifer.org/documentos/librognome/glib-lists-queues.html

https://programmerclick.com/article/32021840155/

https://ccia.ugr.es/~jfv/ed1/tedi/cdrom/docs/ldoble.html

https://pharos.sh/listas-enlazadas-en-detalle-con-ejemplos-de-python-listas-enlazadas-simples/


CANCION
-nombre: string
-ruta: string
-artista: string
-imagen: string
-album: string
-reproducciones: entero
ListaDobleCancion
-primero: string
-ultimo: string
-size: entero
+Vacia()
+AgregarFinal()
+EliminarFinal()
+RecorrerInicio()
+RecorrerFinal
Lista Ciruclar
-primero: string
-ultimo: string
-nombre: string
-size: entero
+AgregarFinal()
+AgregarInicio()
+Vacia()
+UnirNodos()
+RecorrerInicio()
+RecorreFin()
+Ordenar()
ListaDobleAlbumes
-primero: string
-ultimo: string
-size: entero
-asrtista: string
-album: string
+Vacia()
+AgregarFinal()
+EliminarFinal()
+RecorrerInicio()
+RecorrerFinal
Nodo
-dato: string
-siguiente: string
-anterior: string
+Vacia()
+AgregarFinal()
+EliminarFinal()
+RecorrerInicio()
+RecorrerFinal
ListaDobleArtista
-primero: string
-ultimo: string
-size: entero
-asrtista: string
+Vacia()
+AgregarFinal()
+EliminarFinal()
+RecorrerInicio()
+RecorrerFinal