How problems were solved:

* Singelton pattern of private constructor was used to eliminate the creation of different lists by method returning singleton instance.
* When repeat mode is activated this will make the list as a circle which might create problems when printing or doing other functionalities, a variable repeat was created to track the list if circle or linear.
* Because of the limitation (according to the brief) of adding the last song to genres, the song be added to genre will be automatically deleted from liked list, which will allow to add the next song and not the same song repeatedly.

Abstract Data Types & Interfaces

* Song ADT:  
  Represents a song with attributes (title) and (artist)  
  Provides methods to retrieve the title and artist of the song.
* Genreplaylistinterface:  
  Interface defining methods to add, delete, print, count, search, and manage playlists based on Hip-Hop and Piano genres.
* LikedSongsInterface:  
  Interface defining methods to add, delete, print, count, search, and manage a list of liked songs.

Classes

* Node:  
  represents a node in a single linked list which contains a reference to the song object and reference to the next node, will be referencing to null if it is the last node and repeat is deactivated.
* GenrePlaylistImpl:  
  implements GenrePlaylistInterface which manages playlist for Hip-Hop and Piano, this class takes Node class to maintain playlist.
* LikedSongsImpl:  
  implements LikedSongsInterface which manages playlist of liked songs, this class takes Node class to maintain a list of liked songs as single linked list.

Managing Songs

* Adding Songs:  
  any new song is a node which will be added to the single linked list at the end (newSong = lastNode)
* Deleting Songs:  
  Song gets deleted from the list is by traversing the linked list then finding the specific node that containing the song and remove it from the list, in single linked list condition is that I leave the node with nothing pointing to it so the cabbage collector deals with it.
* Print Songs:  
  by starting from the beginning of the list (head) then traversing to each node till the end of the list retrieving the title and the artist of each node.
* Counting Songs:  
  traversing through the linked list starting from the list (head) till the end of the list and retrieve the number by a counter variable.
* Search Songs:  
  by iterating through the linked list and comparing the title || artist with the search query by the user.
* Repeat Mode:  
  making the last node pointing to the head is when the repeat mode is activated, to deactivate the repeat mode is by changing the pointer of the last node to null instead of head that’s how I manage to toggle between them is by setRepeat() and setNotRepeat() methods to activate or deactivate the repeat mode.

**Notes:**

* GitHub Link for this project: <https://github.com/EskandarAtrakchi/CA1-DSA>
* You can download the zip file <https://github.com/EskandarAtrakchi/CA1-DSA/raw/main/Project%20zipped/MusicLibrary.zip>
* I have added a ReadMe file on GitHub for more explanation for each method. <https://github.com/EskandarAtrakchi/CA1-DSA/blob/main/ReadMe.md>
* To download the executable file and interact with the project without NetBeans <https://github.com/EskandarAtrakchi/CA1-DSA/raw/main/dist/MusicLibrary.jar>

