Universidad Rafael Landívar
Facultad de Ingeniería
Ingeniería
Programación avanzada
José Daniel Molina Galindo
Carné: 1007420

PROYECTO 1

PARTY MIX

Guatemala 17 de octubre de 2021

Introducción

Este proyecto número 1 trata sobre las playlist de Spotify en el cual son usadas actualmente por todo el mundo para escuchar sus canciones o álbumes de artistas de todo el mundo, en este caso se realiza una playlist fácil para el usuario y comprensible de la mejor manera para ordenar las canciones sin necesidad de realizar mucha complejidad en su uso, también la reproducción de este y el exportar las canciones que han sido agregadas en el trabajo.

Análisis

Entrada: Pide al usuario las canciones iniciales de su álbum.

Proceso: enlistar las canciones que se tienen en el archivo CSV.

Salida: Devuelve las canciones ordenadas y su respectivo reproductor.

Pseudocodigo:

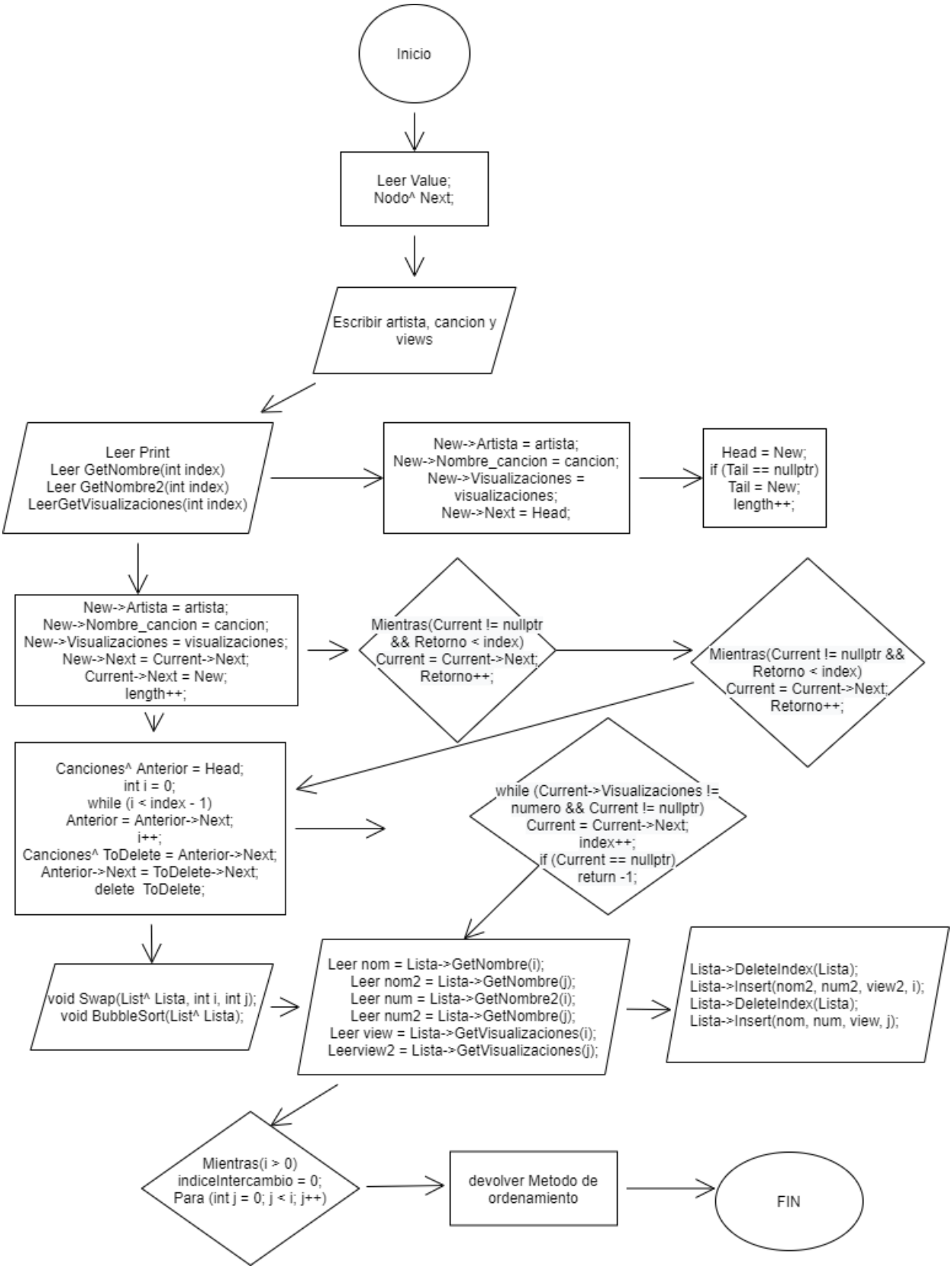
```
// void Orden::Swap(List^ Lista, int i, int j)

	SI (i != j)

		Escribir nom = Lista->GetNombre(i)
		Escribir nom2 = Lista->GetNombre(j)
		Escribir num = Lista->GetNombre2(i)
		Escribir num2 = Lista->GetNombre(j)
		Escribir view = Lista->GetVisualizaciones(i)
		Escribir view2 = Lista->GetVisualizaciones(j)
		Lista->DeleteIndex(Lista)
		Lista->Insert(nom2, num2, view2, i)
		Lista->DeleteIndex(Lista)
		Lista->Insert(nom, num, view, j)



// void Orden::BubbleSort(List^ Lista)

	Escribir i;
	Escribir indiceIntercambio
	i = (Lista->length) - 1
	Mientras (i > 0)

		indiceIntercambio = 0;
		Para (int j = 0; j < i; j++)

			Si (Lista->GetVisualizaciones(j + 1) < Lista->GetVisualizaciones(j))

				Swap(Lista, j, j + 1)
				indiceIntercambio = j


		i = indiceIntercambio

Canciones^ New = gcnew Canciones()
	New->Artista = artista;
	New->Nombre_cancion = cancion
	New->Visualizaciones = visualizaciones
	New->Next = Head
	Head = New
	Si (Tail == nullptr)
		Tail = New
	length++
```

```
	Si (Head == nullptr || index == 0)

		Insert(artista, cancion, visualizaciones)
		return true

	int i = 0
	Canciones^ Current = Head;
	Mientras (i < (index - 1) && Current != nullptr)

		Current = Current->Next
		i++

	Si (Current == nullptr)
		return false
	Canciones^ New = gcnew Canciones()
	New->Artista = artista
	New->Nombre_cancion = cancion
	New->Visualizaciones = visualizaciones
	New->Next = Current->Next
	Current->Next = New
	length++
	Leer res = nullptr
	Canciones^ Current
	Current = Head
	Mientras (Current != nullptr)

		res += Current->Artista + "\t" + Current->Nombre_cancion + "\t" + Current-
>Visualizaciones.ToString() + "\n";
		Current = Current->Next

	Devolver res


	Canciones^ Current = Head
	Leer Retorno = 0
	Mientras (Current != nullptr && Retorno < index)

		Current = Current->Next
		Retorno++

	devolver Current->Artista

	Canciones^ Current = Head
	Leer Retorno = 0
	Mientras (Current != nullptr && Retorno < index)

		Current = Current->Next
		Retorno++

	Devolver Current->Nombre_cancion

	Canciones^ Current = Head
	Leer Retorno = 0
	Mientras (Current != nullptr && Retorno < index)

		Current = Current->Next
		Retorno++
```

```
	Devolver Current->Visualizaciones;
	Escribir index;
	Si (index == 0)

		Canciones^ ToDelete = Head
		Head = Head->Next
		delete ToDelete


	Si_No
		Canciones^ Anterior = Head
		int i = 0
		Mientras (i < index - 1)

			Anterior = Anterior->Next
			i++

		Canciones^ ToDelete = Anterior->Next
		Anterior->Next = ToDelete->Next
		delete  ToDelete


	length--

	Leer index = 0;
	Canciones^ Current = Head
	Mientras (Current->Visualizaciones != numero && Current != nullptr)

		Current = Current->Next
		index++

	SI (Current == nullptr)
		Devolver -1
	Si_no
		Devolver index

	bool Sorted = true
	Canciones^ Current = Head
	Mientras (Current != nullptr && Current->Next != nullptr)

		Si (Current->Visualizaciones >= Current->Next->Visualizaciones)
			Sorted = false
		Current = Current->Next

	Devolver Sorted


	Canciones^ Head
	Canciones^ Tail
public:

	Leer length
	void Insert(String^ artista, String^ cancion, int visualizaciones)
	Leer Insert(String^ artista, String^ cancion, int visualizaciones, int index)
	Leer^ Print()
	Leer ^ GetNombre(int index)
	Leer ^ GetNombre2(int index)
	Leer GetVisualizaciones(int index)
```

```
	Leer Search(int value)
	Leer DeleteIndex(List^ Lista)
	Leer IsSorted()


public:
	Escribir Artista
	Escribir Nombre_cancion
	Escribir Visualizaciones
	Canciones^ Next

public:
	Leer Value
	Leer Next

//TEXT EN BOTONES
richTextBox1->Visible = false
		richTextBox2->Visible = false

		try

			OpenFileDialog dialogoLectura
			Si (dialogoLectura.ShowDialog() ==
System::Windows::Forms::DialogResult::OK)

				StreamReader^ reader = gcnew
StreamReader(dialogoLectura.FileName)
				Mientras (reader->Peek() >= 0)

				Leer line = reader->ReadLine();
				Para (int i = 0; i < line->Split(',')->Length; i = i + 3)

					Leer Artista = line->Split(',')[i]
					Artista = Artista->Trim()
					Leer Nombre_cancion = (line->Split(',')[i + 1])
					Nombre_cancion = Nombre_cancion->Trim()
					Leer Visualizaciones = (line->Split(',')[i + 2])
						Visualizaciones = Visualizaciones->Trim()
						Leer view = int::Parse(Visualizaciones)
						Lista->Insert(Artista, Nombre_cancion, view)
		catch (...)

		MessageBox::Show("El archivo fue cargado exitosamente");
	MessageBox::Show("Ordenado Exitosamente")
		richTextBox1->Text = "ARTISTA" + "\t" + "           CANCIONES" + "\t" + "
VIEWS" + "\n" + Lista->Print()

		ord->BubbleSort(Lista)

		Leer lista = Lista->Print()
		richTextBox2->Text = "ARTISTA" + "\t" + "           CANCIONES" + "\t" + "
VIEWS" + "\n" + lista
		richTextBox1->Visible = true
		richTextBox2->Visible = true
```

Diagrama de flujo



Conclusiones

1. Analizar el comportamiento de una lista en base a un grupo de métodos para ordenar palabras o números.
2. Comprender la diferencia entre pila y cola y su funcionamiento.
3. Determinar el concepto de punteros y nodos en cómo influye drásticamente en el funcionamiento de ordenar algo.

Referencias

T. (2021, 3 agosto). *list Class*. Microsoft Docs. https://docs.microsoft.com/en-us/cpp/standard-library/list-class?view=msvc-160

T. (2021a, agosto 3). *Creating Stack and Queue Collections*. Microsoft Docs. https://docs.microsoft.com/en-us/cpp/mfc/creating-stack-and-queue-collections?view=msvc-160

https://docs.microsoft.com/en-us/cpp/cpp/templates-cpp?view=msvc-160