**Manual técnico Practica AFD**

**LABORATORIO DE TEORÍA DE LENGUAJES**



**Grupo de trabajo:**

**Integrantes**

José Carlos Ortiz Padilla.

Carlos Osvaldo Zapata Arango

**Profesor:**

Roberto Flórez Rueda

**Departamento de Sistemas**

**Facultad de ingeniería**

**Universidad de Antioquia**

**2022-2**

**Índice:**

1. **Introducción**
2. **Como instalar el programa**
3. **Modelos**
4. **Controlador.**
5. **Ruta de ejecución.**
6. **Introducción:**

**En este documento se explicará a detalle cómo funciona el programa basado en autómatas finitos determinístico, su estructura, el recorrido principal, y todas las funcionalidades plasmadas en código.**

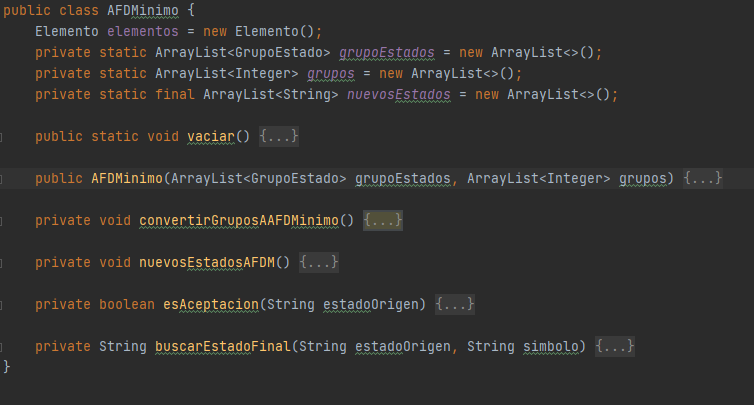
1. **Arquitectura del software:**

**El software se ha creado bajo el patrón de arquitectura Model-View-Controller (MVC) por lo que se abordarán primero estos aspectos del programa:**

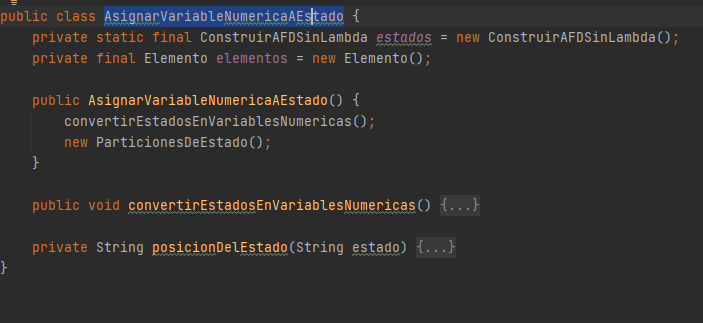
1. **Modelos:**

**El software cuenta con 11 modelos y una interfaz, las cuales son:**

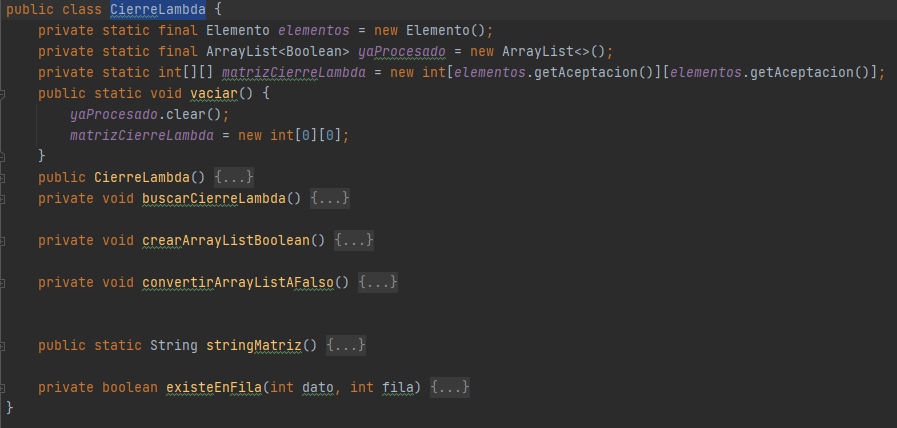
* 1. **Modelo AFDMinimo: Esta clase permite crea el AFD mínimo.**



* 1. **Modelo Asignar Variable Numérica A Estado:** **Esta clase permite asignarles variables numéricas a los nuevos estados obtenidos.**

****

* 1. **Modelo Cierre Lambda:** **Esta clase se encarga de buscar los cierres de lambda de cada estado.**

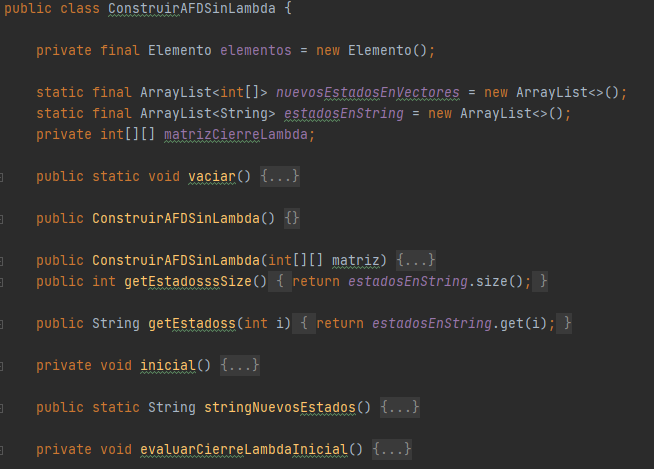
****

* 1. **Modelo Construcción Básica Thompson:** **Esta clase permite evaluar la ER con la respectiva construcción de Thompson.**

**Texto

Descripción generada automáticamente**

* 1. **Modelo Construir AFD Sin Lambda: Esta clase permite crear el AFD sin transiciones lambdas.**

****

* 1. **Modelo Elemento: Esta clase permite crear las listas donde se almacenarán las transiciones de la ER y los diferentes AFD.**

**Texto

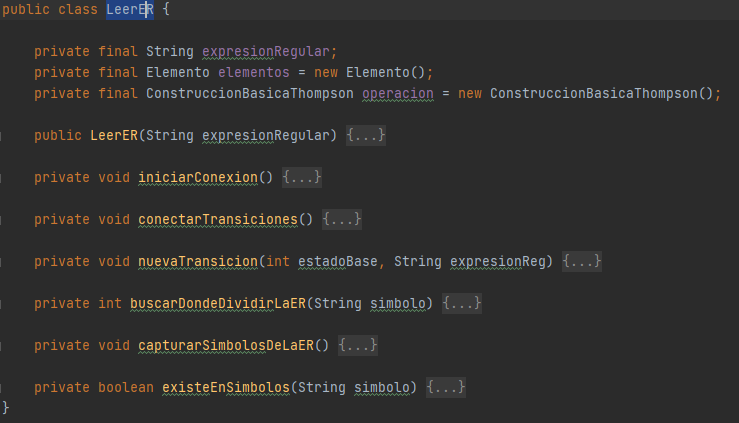
Descripción generada automáticamente**

* 1. **Interfaz Grupo Estado:** **Esta clase permite crear objetos que representan los grupos y los estados.**

**Texto

Descripción generada automáticamente**

* 1. **Modelo Leer ER :** **Esta clase permite leer la expresión regular recibida como parámetro, para procesarla y así comenzar a crear objetos de Transición ER, usando construcción básica de Thompson.**

****

* 1. **Modelo Particiones De Estado\_:** **Esta clase permite agrupar los estados para construir el AFD mínimo.**

**Texto

Descripción generada automáticamente**

* 1. **Modelo Transición AFD: Esta clase permite crear objetos tipo transiciones de**

**los autómatas finitos determinísticos.**

**Texto

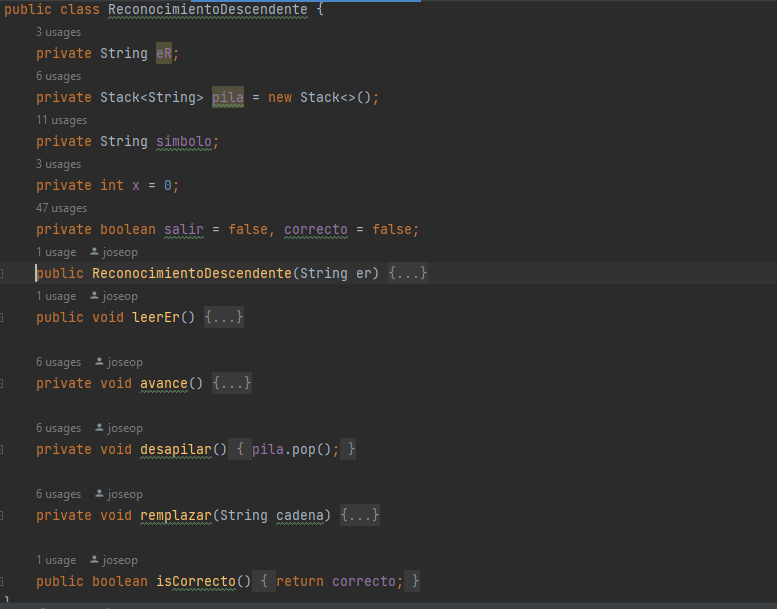
Descripción generada automáticamente**

* 1. **Modelo Transición ER: Esta clase es para crear objetos tipo Transiciones de Expresiones regulares, que representa las transiciones o las construcciones básicas de Thompson.**

**Texto

Descripción generada automáticamente**

**3.12. Modelo Reconocimiento Descendente: Esta clase se encarga de verificar la expresion regular ingresada y mediante gramaticas y Automatas de pila verifica si la expresion regular es correcta.**

****

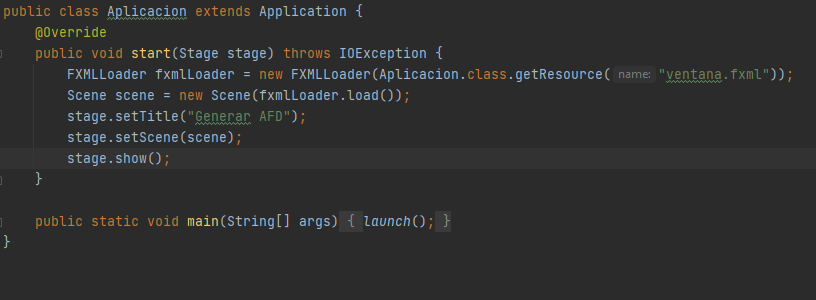
1. **Controladores:**

**Solamente se usan clases controladoras en el programa, las cuales son:**

****

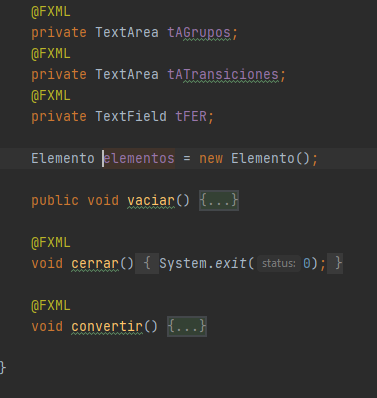
* 1. **Programa:**

**Es la clase principal, que inicia el programa y la ventana JavaFx:**

****

* 1. **VentanaController:**

**Es la clase enlazada con la ventana JavaFx, que permite darle funcionalidad a los botones, labels y textareas:**



**Esta clase implementa la interfaz ClipboardOwner ya que es necesario guardar los links mostrados en la pestaña “Acerca De” en el portapapeles del sistema, los métodos link son quienes efectúan esta tarea.**

1. **Ruta de ejecución:**

**Todo comienza cuando se pulsa el botón “Verificar y Convertir”, el compilador divide el texto en líneas y le pasa las líneas particionadas en caracteres al autómata.**