**SIMULADOR DE AUTOMATAS FINITOS DETERMINISTAS (AFD)**

**Autores:** Ivan Durán Vilches, Eduardo Barrera Navarrete, Osvaldo Casas-cordero Navarro

**Profesor:** María Eliana de la Maza W.

**Asignatura:** TEORÍA DE AUTÓMATAS

**Carrera:** Ingeniería civil informática.

**Fecha:**

**1. INTRODUCCIÓN:**

El problema consiste en crear un programa que simule cualquier Autómata Finito Determinista (AFD). Este programa recibirá como entrada; el estado inicial, las transiciones del AFD, los estados finales y una palabra. La salida será un mensaje que indique si la palabra de entrada pertenece o no al lenguaje definido por las transiciones del AFD. El programa debe ser capaz de analizar múltiples palabras de entrada para un mismo AFD sin necesidad de reconfiguración. En resumen, se busca desarrollar una herramienta computacional para el análisis de lenguajes formales mediante la simulación de AFDs.

**2. PLANIFICACIÓN:**

Lenguaje de programación a utilizar: Python.

Estructuras de datos a utilizar: Diccionarios, conjuntos (colecciones).

Se utilizará clases en Python para organizar de manera eficiente las estructuras de datos y la lógica de funcionamiento del AFD. Esto nos permite modularizar el código de forma coherente y reutilizable, facilitando su comprensión y mantenimiento.

* estado inicial: se representará mediante una variable simple “estado inicial”, que almacena un único estado del AFD.

* transiciones: Las transiciones se almacenarán en un diccionario llamado “transiciones”. Cada clave del diccionario es una lista que contiene un estado y un símbolo del alfabeto, y el valor correspondiente es el estado al que se transita desde el estado dado al leer el símbolo dado.

* estados: se almacenarán en un conjunto llamado “estados”.

* estados finales: Los estados finales se almacenarán en un conjunto llamado "estados\_finales", que contiene todos los estados que son finales en el AFD.

* palabra de entrada: La palabra de entrada será solo un argumento que se itera sobre cada símbolo de la palabra para realizar las transiciones en el AFD.

La elección de utilizar diccionarios en nuestra implementación del AFD se debió a su capacidad para asociar datos clave-valor, su eficiencia en la búsqueda y acceso, y su capacidad para organizar de manera estructurada la información.

Algo similar con los conjuntos que los elegimos sobre las listas porque se alineaban mejor con las necesidades específicas de nuestro programa, incluida la garantía de unicidad, la eficiencia en la búsqueda y el acceso a operaciones de conjunto convenientes.

Lenguaje de programación a utilizar: Python.