# Teoría de la Computación 2025

Primer Proyecto

13.agosto.2025

## Descripción

El proyecto consiste en la implementación de algoritmos básicos para construcción de autómatas finitos a partir de expresiones regulares y visualizaciones a partir de estos algoritmos.

## Algoritmos a implementar

- Implementación del algoritmo *Shunting Yard* para convertir una regexp a postfix.
- Implementación del algoritmo de conversión regexp a AFN (Thompson o Glushkov).
- Implementación del algoritmo de conversión AFN a AFD: Construcción de Subconjuntos.
- Implementación del algoritmo de minimización de Hopcroft: AFD a AFD minimal.
- Implementación de simulaciones de un AFD.

## Especificaciones

Entrada: Solamente se ingresarán textualmente una expresión regular $r$. Esta expresión regular, debe permitir el ingreso de letras minúsculas o mayúsculas (sin acentos), digitos, el caracter $\varepsilon$ (o algún sustituto para representar la cadena vacía), y los operadores `|`, `+`, `*`, `(`, `)`. Por ejemplo:

$$r = \texttt{(b|b)*abb(a|b)*}$$

El símbolo que representa la cadena vacía $\varepsilon$ será designado por los programadores (debe ser algo razonable, no una letra o un número con altas probabilidades de ser usado en otro aspecto del proyecto).

Todos los símbolos del alfabeto son de longitud 1. El alfabeto de símbolos para la expresión regular será conformado por todos los símbolos (no operadores) distintos que se encuentren en la expresión regular.

Salida: Para cada autómata construido, deberá generar un archivo de texto que describa al autómata con la siguiente estructura:

- ESTADOS $= \{0, 1, \ldots, n\}$
- SIMBOLOS $= \{a, b, c, \ldots, z\}$
- INICIO $= \{0\}$
- ACEPTACION $= \{0, 1, \ldots, n\}$
- TRANSICIONES $= \{(0, a, 1), (0, , 2), (3, b, n)\}$

Este archivo puede estar en formato `.json`, `.yml` o similares según su preferencia. Además, su programa debe permitir la visualización del autómata construido mediante una imagen o un ambiente interactivo que muestre el grafo del autómata generado.

Para el desarrollo de los algoritmos, puede implementar una clase autómata que funcione de manera general, y los algoritmos pueden ser métodos asociados a esta clase. Conviene usar estructuras de datos que faciliten en trabajo de autómata como: árboles, grafos. Para la visualización, puede hacer uso de librerías como *graphviz*, *node4j* o similares, que faciliten la construcción visual de un autómata a partir de la estructura de datos.

Puede tomar ideas a partir de las siguientes referencias:

- Toolbox: Regexp to NFA `https://cyberzhg.github.io/toolbox/regex2nfa`.
- Toolbox: Regexp to DFA `https://cyberzhg.github.io/toolbox/nfa2dfa`.
- Toolbox: Regexp to minDFA `https://cyberzhg.github.io/toolbox/min_dfa`.

## Simulaciones

A partir del grafo, su programa debe permitir ingresar una cadena $w$ constituida por símbolos válidos de su autómata. Por ejemplo la cadena $w =$babbaaaaa. Deberá simular las transiciones o la derivación de esta cadena en su autómata AFD minimal, y mostrar en pantalla o en forma visual, cada uno de los pasos de esta derivación-

Al final deberá indicar si la cadena es aceptada o no es aceptada por el autómata. Esto es, devolver un `SÍ` cuando la cadena $w$ pertenece al lenguaje descrito por el autómata $L(r)$ o un `NO` en caso contrario.

Puede tomar ideas a partir de las siguientes referencias:

- Toolbox: Regexp to NFA `https://www.automataverse.com/simulator/dfa`.
- Toolbox: Regexp to DFA `https://www.automataverse.com/simulator`.

## A entregar

:

1. **Documentación**: para tener derecho a nota. La documentación o informe debe de abarcar:
   - Diseño de la aplicación (el método de modelado o estructuras utilizadas para los autómatas).
   - Discusión (obstáculos encontrados, recomendaciones, etc.).
   - Ejemplos y pruebas realizadas para varios autómatas (al menos 2) y para varias cadenas.
   - Referencias utilizadas.
2. **Código**: Este puede subirse como un archivo empaquetado (`.zip`, `.rar`, `.tar` o similares) directo al enlace en Canvas. También pueden dejar su código como un repositorio en Git. En este caso, en el informe deberán indicar el enlace a su repositorio.

   **Importante!** El repositorio debe estar público (si no desean que esté público deberán incluir mi usuario con permisos de lectura). Si al momento de la revisión el repositorio no se puede acceder, se asignará automáticamente una nota de 0.

## Ponderación

El Proyecto tiene un valor de 20 puntos.

| Característica | Ponderación |
|---|---|
| Conversión de *regexp* a *postfix* (Shunting Yard) | 3 puntos |
| Conversión de *regexp* a AFN (archivo y visualización) | 3 puntos |
| Conversión de AFN a AFD (archivo y visualización) | 4 puntos |
| Minimización del AFD (archivo y visualización) | 3 puntos |
| Simulaciones con AFD | 3 puntos |
| Informe técnico | 4 puntos |
| Total | **20 puntos** |

# Fecha de Entrega

**Lunes 08 de septiembre**.
Se asignará un horario específico para cada grupo.