Universidad Rafel Landívar
Facultad de ingeniería
Licenciatura en ingeniería en informática y sistemas
Lenguajes formales y autómatas
Docente: Ing. Julio David Requena Duarte

MANUAL DE USO

"Proyecto 3"

Estudiante: Ponce Valdizón, Ian Andrés

Carné: 1163524

Guatemala, 10 de noviembre de 2025

ÍNDICE

I. DOCUMENTACIÓN

MÓDULOS Y CONSTANTES PRINCIPALES

- **tkinter, ttk, messagebox, scrolledtext, filedialog:** Librerías que se utilizaron para la interfaz gráfica (botones, ventanas, cuadros de texto, diálogos de archivo).
- **re:** Se utilizó para evaluar la cadena final contra la expresión regular seleccionada.
- **time:** Está presente para posibles pausas y animaciones; en el código en sí no se usó fuertemente.
- **REGEXES:** Lista de tuplas (nombre, patrón) con ejemplo de 10 expresiones regulares predefinidas. Se utilizarón para la evaluación final con re.fullmatch.

CLASES

Error

- **Constructor:** __init__(self, tipo, mensaje)
- **Propósito:** Es una estructura simple para representar errores de tipo y mensaje. También sirve para homogeneizar el manejo de errores si se extiende.

Transición

- **Constructor:** __init__(self, estado_origen, simbolo_lectura, estado_destino, simbolo_escritura, movimiento)
- **Propósito:** Es una estructura que representa una transición de la MT es decir estado origen, símbolo que lee, estado destino, símbolo escrito, movimiento L/R/S. En el código la clase existe para modelado y posible extensión, pero no se utilizaron tablas complejas basadas en ella.

MaquinaTuring

- Atributos principales:
    - **BLANK:** símbolo usado para celdas vacías ('_').
    - **cinta:** lista que representa la cinta de la máquina.
    - **head:** índice entero que indica la posición del cabezal en cinta.
    - **estado:** estado actual (cadena, comienza en 'q0').
    - **pasos:** contador de pasos ejecutados.
    - **halted:** booleano que indica si la máquina terminó.
    - **accept:** booleano/flag resultado (se asigna en la evaluación final).
    - **transiciones:** diccionario reservado para definiciones de transiciones si se extiende la MT.

MÉTODOS PRINCIPALES DEL SIMULADOR DE MT

construir_cinta(self, entrada, padding=20)

Prepara la cinta a partir de la entrada del usuario, coloca padding celdas en blanco a la izquierda y derecha, transforma la entrada en una lista de símbolos y posiciona el cabezal en el medio, también reinicia estado, pasos, halted y accept.

paso_simulacion(self)

Realiza un paso de la simulación visual:

- Si halted está True devuelve None.
- Si está en q0 pasa a q_scan (estado de recorrer).
- En q_scan lee el símbolo bajo el cabezal. Si es BLANK marca q_final, halted=True y devuelve un diccionario con la cinta, cabeza, estado, info y halted=True, lo que indica final de ejecución visual. Si no es BLANK, avanza el head a la derecha y continúa.
- Incrementa pasos.
- Devuelve un diccionario con el estado visual (cinta, head, estado, info, halted) para actualizar la GUI.

reiniciar(self)

Vuelve estado a 'q0', reinicia pasos, halted y accept. Cabe aclarar que no reconstruye la cinta; eso lo hace elk método construir_cinta.

Nota: la implementación de paso_simulacion está pensada y creada como demostrativa es decir el mecanismo que decide aceptación final no es una tabla completa de transiciones sino la evaluación final con la expresión regular que se hace con re.fullmatch cuando la simulación termina.

INTERFAZ GRÁFICA

Constructor: __init__(self, root)

- Inicializa ventana principal (root), configura tamaño, color de fondo y componentes internos.
- Crea la instancia self.mt (Máquina de Turing), variables de control (auto_running, visual_cells, etc.) y el historial.
- Llama a configurar_estilos(), crear_layout() y dibuja la cinta vacía inicial con dibujar_cinta_inicial("").

configurar_estilos(self)

- Define estilos ttk (tema, fuentes, colores) para etiquetas y botones de la UI. Ajusta la tipografía a Segoe UI en el código original; puedes cambiarla globalmente si lo deseas.

crear_layout(self)

- Construye la estructura visual:
    - frame_top: título y controles.
    - Etiqueta de título y subtítulo (información del proyecto / autor).
    - Controles para ingresar la cadena (input_var) y seleccionar la expresión regular (regex_combo).
    - Botones principales agrupados: Preparar simulación, Paso, Iniciar automático / Detener, Reiniciar
    - Panel central para mostrar la cinta visual.
    - Panel lateral con lbl_estado, lbl_info, lbl_result y una Listbox que muestra las expresiones disponibles.
    - Enlaza el evento <Configure> del canvas con redibujar_cinta() para redibujar cuando cambia el tamaño de la ventana.
- Inicializa self.preparado = False para bloquear ejecución hasta preparar la simulación.

FLUJO DE SIMULACIÓN

preparar_simulacion(self)

- Llamada al pulsar Preparar simulación.
- Lee la cadena de self.input_var, llama a self.mt.construir_cinta(entrada, padding=20) y luego a self.mt.reiniciar() para preparar la MT.
- Marca self.preparado = True, limpia resultados previos y actualiza etiquetas de estado/información.
- Guarda el índice de la regex seleccionada (current_regex_index) usando _regex_index_from_combo().
- Dibuja la cinta inicial llamando dibujar_cinta_inicial(entrada).

regex_index_from_combo(self)

- Busca en REGEXES cuál índice corresponde al nombre seleccionado en la lista desplegable y lo devuelve. Se usa para obtener luego el patrón regex.

paso(self)

- Ejecuta un paso de la simulación visual:
    - Verifica que preparado sea True.
    - Llama a self.mt.paso_simulacion(); si devuelve None significa que la máquina ya estaba detenida y llama a finalizar_evaluacion().

- Si devuelve un diccionario con la visual, actualiza la cinta en pantalla (actualizar_visual) y las etiquetas lbl_estado y lbl_info.
- Si el paso indica halted, se llama a finalizar_evaluacion().

toggle_auto(self) y _auto_step(self)

- toggle_auto alterna la ejecución automática. Cambia la etiqueta del botón entre "Iniciar automático" y "Detener automático".
- _auto_step realiza iterativamente los pasos con self.root.after(self.auto_delay_ms, self._auto_step) para animación: llama a paso_simulacion() repetidamente hasta que la máquina termine o hasta que el usuario detenga la ejecución.

finalizar_evaluacion(self)

- Cuando la simulación visual termina, evalúa la cadena con la expresión regular seleccionada:
  - Obtiene pattern = REGEXES[idx][1] y usa re.fullmatch(pattern, cadena) para decidir si es aceptada.
  - Asigna self.mt.accept = aceptada, actualiza lbl_result con "ACEPTADA" o "RECHAZADA", y muestra información final (número de pasos y resultado).
  - Actualiza lbl_estado para mostrar el estado final (q_final si la MT terminó).
- Este método es la combinación entre la simulación visual y la decisión teórica. En otras palabras, la visualización es didáctica y la aceptación real se comprueba con re.

VISUALIZACIÓN DE LA CINTA

dibujar_cinta_inicial(self, entrada)

- Borra la pantalla y crea una cuadrícula fija de visual_cells celdas centradas en la ventana.
- Para cada celda crea:
  - Un rectángulo (fondo y borde).
  - Un texto (vacío inicialmente) donde se mostrará el símbolo.
- Crea también un marcador (head_marker) que se mueve para indicar la posición del cabezal.
- Finalmente llama a actualizar_visual(self.mt.cinta, self.mt.head) para mostrar el contenido real de la cinta.

redibujar_cinta(self)

- Handler usado cuando el canvas cambia de tamaño; vuelve a dibujar la cinta llamando dibujar_cinta_inicial() con la cadena actual en la entrada.

actualizar_visual(self, cinta, head_index)

- Calcula la ventana visible de la cinta centrada en head_index.
- Rellena cada celda visible con el símbolo correspondiente (o BLANK si fuera necesario).
- Resalta la celda bajo el cabezal (cambia fondo y contorno).
- Actualiza la posición del triángulo marcador del cabezal para mantenerlo apuntando a la celda central visible.

REINICIO Y LIMPIEZA

reset_simulador(self)

- Reestablece la interfaz y los estados:
    - self.auto_running = False, self.preparado = False.
    - Crea una nueva instancia de MaquinaTuring() para reiniciar estado interno.
    - Limpia la entrada de texto, etiquetas y dibuja cinta vacía.
    - Muestra un messagebox de confirmación.

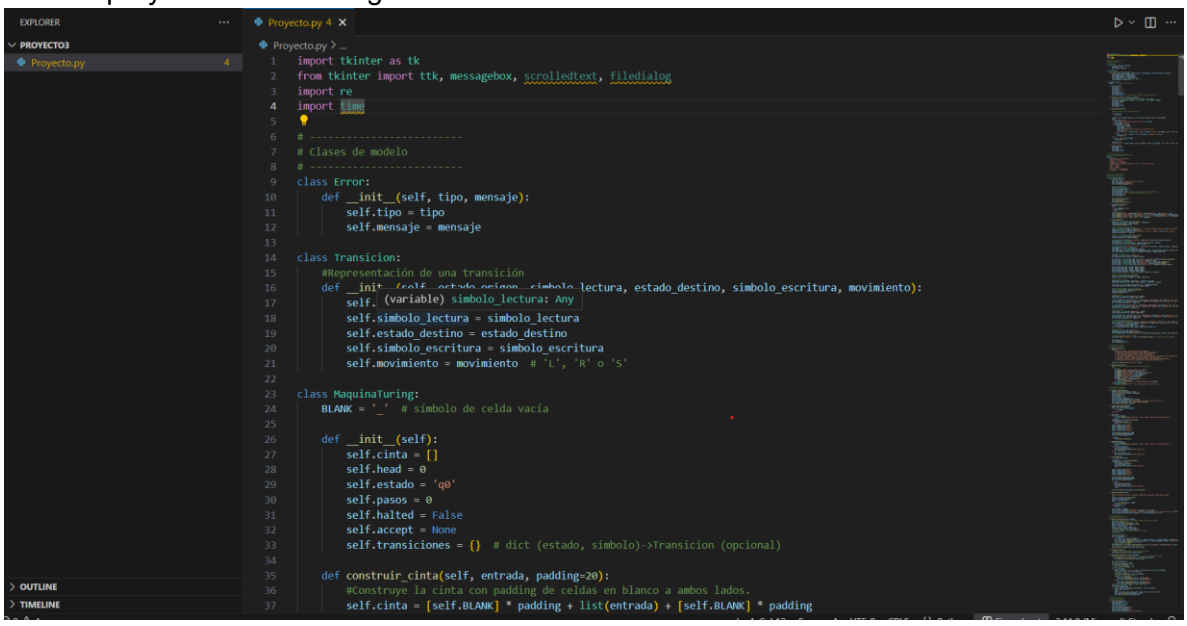
II. REQUISITOS Y EJECUCIÓN DEL PROGRAMA

LIBRERIAS NECESARIAS PARA LA EJECUCIÓN

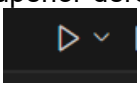
- tkinter
- re
- time

EJEMPLO DE EJECUCIÓN

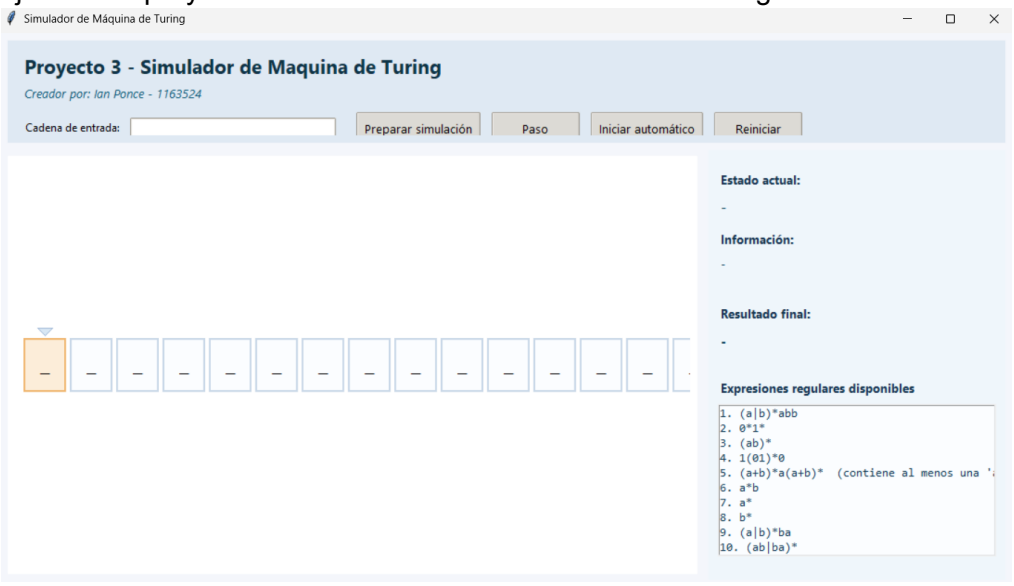
1. Abrir el proyecto con el código



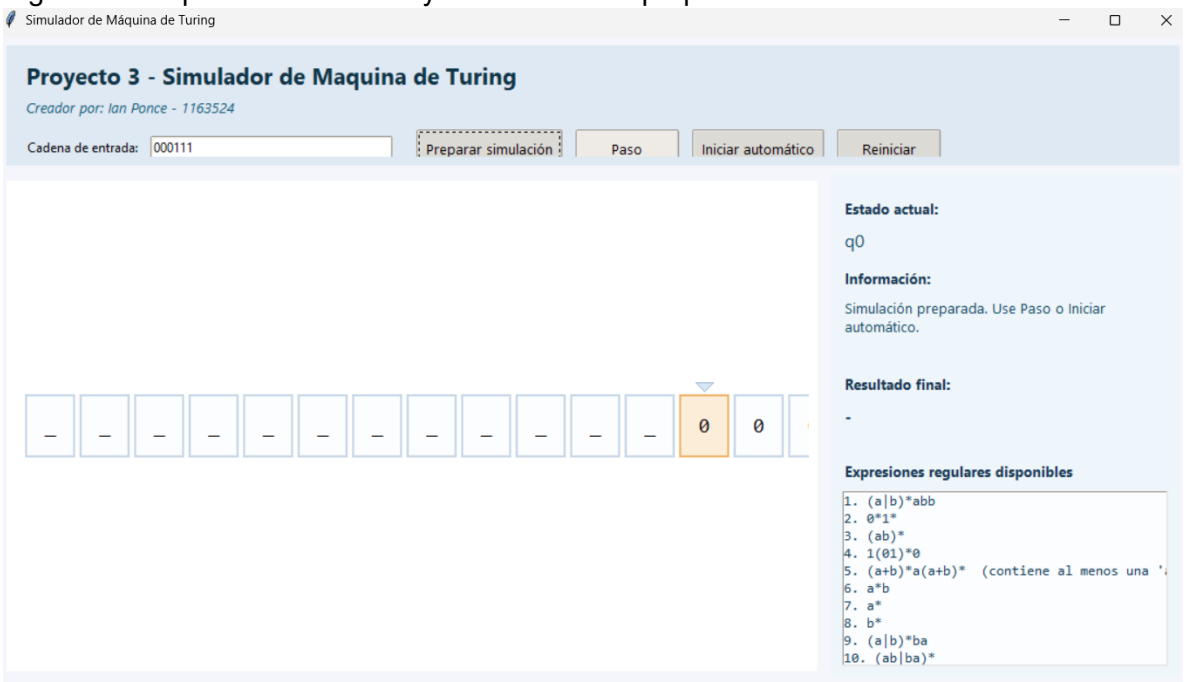
2. Ejecutar el proyecto en la esquina superior derecha o con F5



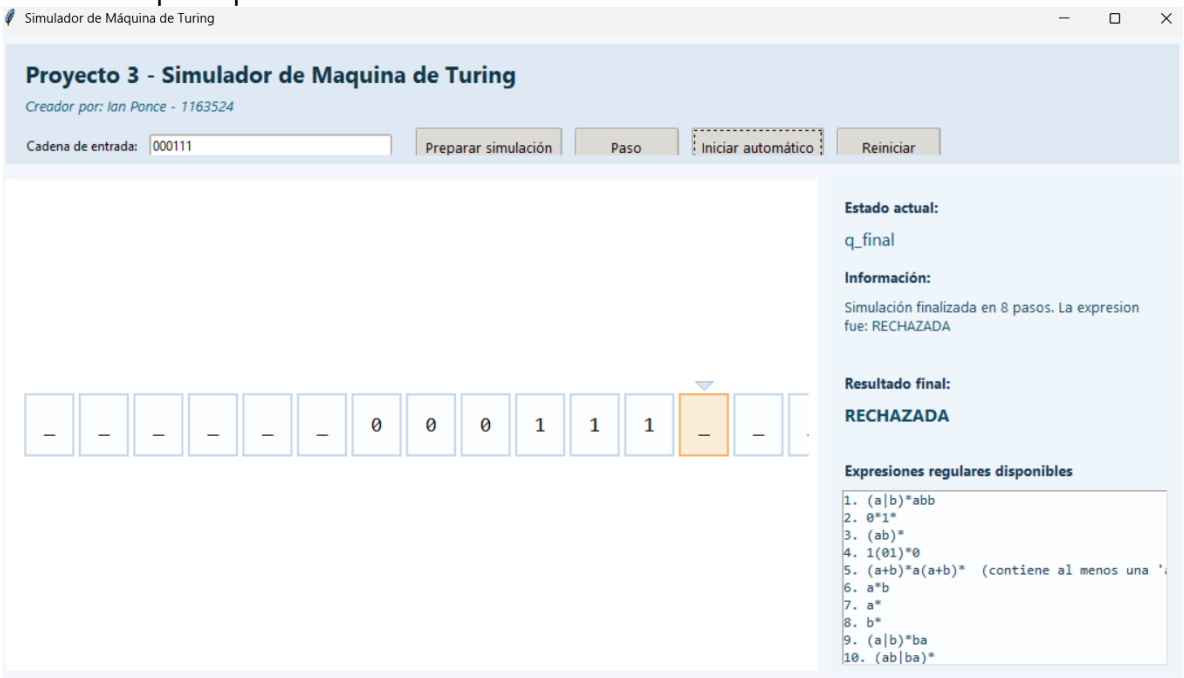
3. Al ejecutar el proyecto se abrirá automáticamente la interfaz gráfica



4. Ingresar la expresión a analizar y seleccionar en preparar simulación

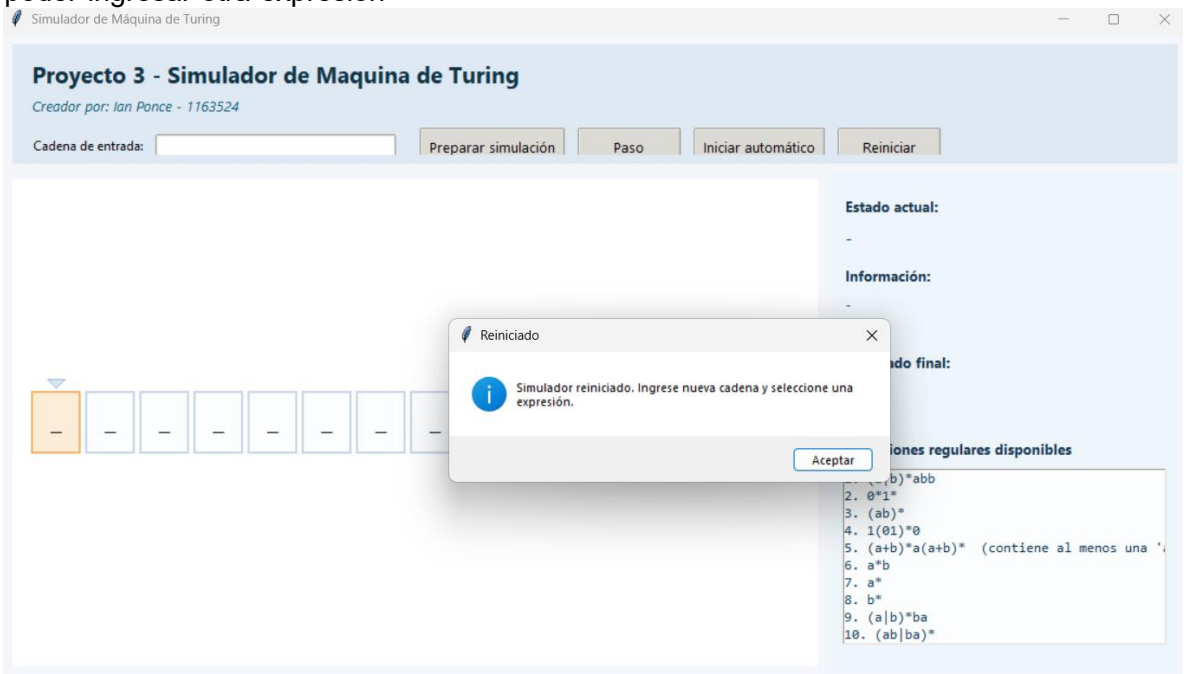


5. Seleccionar en la opción paso si se quiere una simulación paso a paso o en iniciar automático para que la simulación sea automática



Nota: Una vez finalizada la ejecución se desplegará el mensaje si la cadena fue aceptada o rechazada

6. Una vez finalizada la ejecución seleccionar Reiniciar para limpiar la pantalla y poder ingresar otra expresión



Ejemplo con una cadena aceptada