

Universidad Rafael Landívar

Facultad de Ingeniería

Ingeniería en Informática y Sistemas

Lenguajes formales y automatas. Sec: 1

Ing. Julio Duarte



“PROYECTO – MAQUINA DE TURING”

Nahomy Mariángel Chitay Duarte 1211523

Guatemala, 10 de noviembre de 2025.

INTRODUCCION

El proyecto es un simulador visual de Máquina de Turing desarrollado en Python con Tkinter. Permite introducir una cadena de entrada, visualizar la cinta, el cabezal y el estado actual, y ejecutar la simulación paso a paso o de forma automática. La aplicación incluye un panel para probar expresiones regulares con la cadena actual (usando `re.fullmatch`), lo que facilita contrastar el resultado de una MT concreta (por ejemplo para los lenguajes 0^*1^* y $(ab)^*$) con la aceptación o rechazo definido por un patrón regular.

En la interfaz se muestran:

- Cadena de entrada: texto que se coloca en la cinta.
- Selector de máquina: MT de ejemplo (0^*1^* y $(ab)^*$).
- Botones Cargar Máquina, Reset, Paso (Step) y Auto (con control de velocidad).
- Sección de estado y resultado (ACEPTADA / RECHAZADA).
- Cinta con celdas y el cabezal indicado por un triángulo.
- Log de eventos (pasos, cambios de estado, pruebas de regex).
- Panel de Expresiones Regulares con 10 patrones y su descripción breve.

Con este simulador el estudiante puede observar el comportamiento operativo de una MT (transiciones, movimientos del cabezal y decisión final) y, en paralelo, verificar cadenas con expresiones regulares típicas en teoría de lenguajes.

INSTRUCCIONES INSTALACION Y EJECUCION

Requisitos

- Python 3.8+ instalado.
- Tener Visual studio code
- Tkinter viene incluido en la mayoría de instalaciones estándar de Python.

Descarga del proyecto

- Archivo (MaquinaTuring.py)

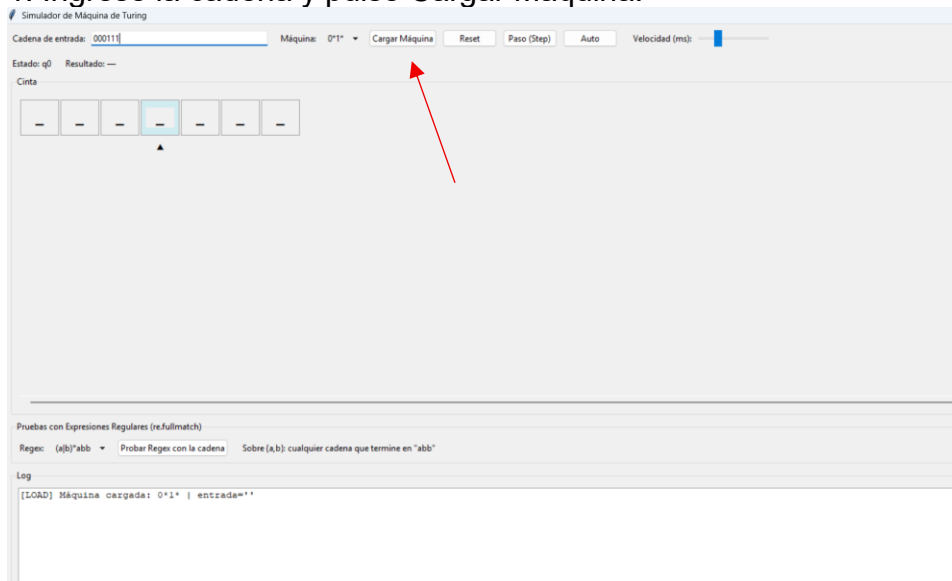
Ejecución

Abrir en visual studio el archivo .py

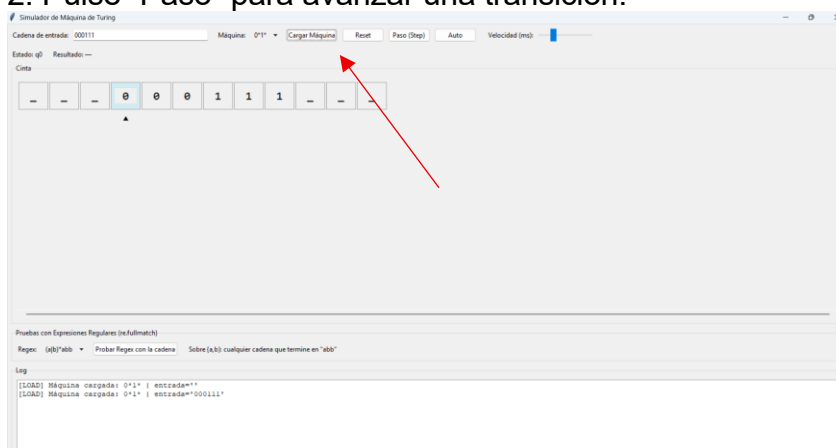
Darle en “run” y ya se mostrará el programa funcionando

FUNCIONAMIENTO

1. Ingrese la cadena y pulse Cargar Máquina.



2. Pulse “Paso” para avanzar una transición.



3. Observe el Estado, el cabezal y la cinta.
4. Si desea, pulse “Auto” para que avance solo ajuste la Velocidad.

Simulador de Máquina de Turing

Cadena de entrada: 000111 Máquina: 0*1* Cargar Máquina Reset Paso (Step) Auto Velocidad (ms):

Estado: q_a Resultado: ACEPTADA

Cinta

-	-	-	0	0	0	1	1	1	-	-	-
---	---	---	---	---	---	---	---	---	---	---	---

▲

Resultado

ACEPTADA

OK

Pruebas con Expresiones Regulares (re.fullmatch)

Regex: (a|b)*abb Probar Regex con la cadena Sobre {a,b}: cualquier cadena que termine en "abb"

Log

```
[STEP] estado=q0 head=1 status=running
[STEP] estado=q0 head=2 status=running
[STEP] estado=q0 head=3 status=running
[STEP] estado=q1 head=4 status=running
[STEP] estado=q1 head=5 status=running
[STEP] estado=q1 head=6 status=running
[STEP] estado=qa head=6 status=accept
```

Mensajes de resultado

ACEPTADA: la máquina llegó a un estado de aceptación.

RECHAZADA: se detuvo sin transición válida o llegó a un estado de rechazo.

Simulador de Máquina de Turing

Cadena de entrada: 1100 Máquina: 0*1* Cargar Máquina Reset Paso (Step) Auto Velocidad (ms):

Estado: q_r Resultado: RECHAZADA

Cinta

-	-	-	1	1	0	0	-	-	-
---	---	---	---	---	---	---	---	---	---

▲

Resultado

RECHAZADA

OK

Pruebas con Expresiones Regulares (re.fullmatch)

Regex: (a|b)*abb Probar Regex con la cadena Sobre {a,b}: cualquier cadena que termine en "abb"

Log

```
[STEP] estado=q0 head=3 status=running
[STEP] estado=q1 head=4 status=running
[STEP] estado=q1 head=5 status=running
[STEP] estado=q1 head=6 status=running
[STEP] estado=qa head=6 status=accept
[RESET] Entrada="000111"
[LOAD] Máquina cargada: 0*1* | entrada="1100"
```

EXPRESIONES REGULARES UTILIZADAS

Se aplican sobre alfabetos sencillos ($\{a,b\}$ o $\{0,1\}$) y se evalúan con coincidencia completa.

1. $(a|b)^*abb$: Cualquier cadena sobre $\{a,b\}$ que termine en abb .
Acepta: $abb, aabb, bbabb, aababb, babb\dots$
Rechaza: $ab, abba, aba, aababa$ (no termina en abb).
2. 0^*1^* : Descripción: Cero o más 0 seguidos de cero o más 1. El bloque de 0 va antes que el de 1. El vacío también es válido.
Acepta: $\epsilon, 0, 00, 1, 11, 00111$.
Rechaza: $10, 101, 1100$ (aparece algún 0 después de un 1).
3. $(ab)^*$: Repeticiones del bloque ab . Incluye la cadena vacía.
Acepta: $\epsilon, ab, abab, ababab$.
Rechaza: $a, aba, abb, baba$ (no está compuesto solo por bloques ab).
4. $1(01)^*0$: Empieza con 1, luego cero o más repeticiones de 01, y termina en 0.
Acepta: $10, 1010, 101010, 10101010\dots$
Rechaza: $1, 100, 110, 101$ (no termina en 0 o se rompe el patrón 01).
5. $(a|b)^*a(a|b)^*$: Cadenas sobre $\{a,b\}$ que contienen al menos una a en alguna posición.
Acepta: $a, ba, ab, babab, aaab$.
Rechaza: Solo b^* (cualquier cantidad de b sin a), por ejemplo $\epsilon, b, bb, bbbb$.
6. $a^*b^*a^*$: Cero o más a , luego un solo b , luego cero o más a . Siempre hay exactamente un b y va en medio.
Acepta: $b, ab, aab, ba, baa, aaabaaa$.
Rechaza: aa
7. $(0|1)^+00(0|1)^*$: Cadenas binarias que contienen 00 en algún lugar.
Acepta: $100, 1100, 001, 101001$.
Rechaza: $\epsilon, 0, 1, 10101$
8. $(aa|bb)^*$: Concatenaciones de pares iguales. La longitud es par y cada par es aa o bb .
Acepta: $\epsilon, aa, bb, aabb, bbaa, aabbbaa$.
Rechaza: $a, b, ab, aba, aabb\ b$
9. $(a|b)\{3,\}$: Cadenas sobre $\{a,b\}$ de longitud al menos 3.
Acepta: $aaa, aab, bba, abab, bbbba$.
Rechaza: ϵ, a, bb .
10. $([ab])\backslash 1$: Un par de símbolos iguales consecutivos
Acepta: aa, bb .
Rechaza: ab, ba, a, b, aab .