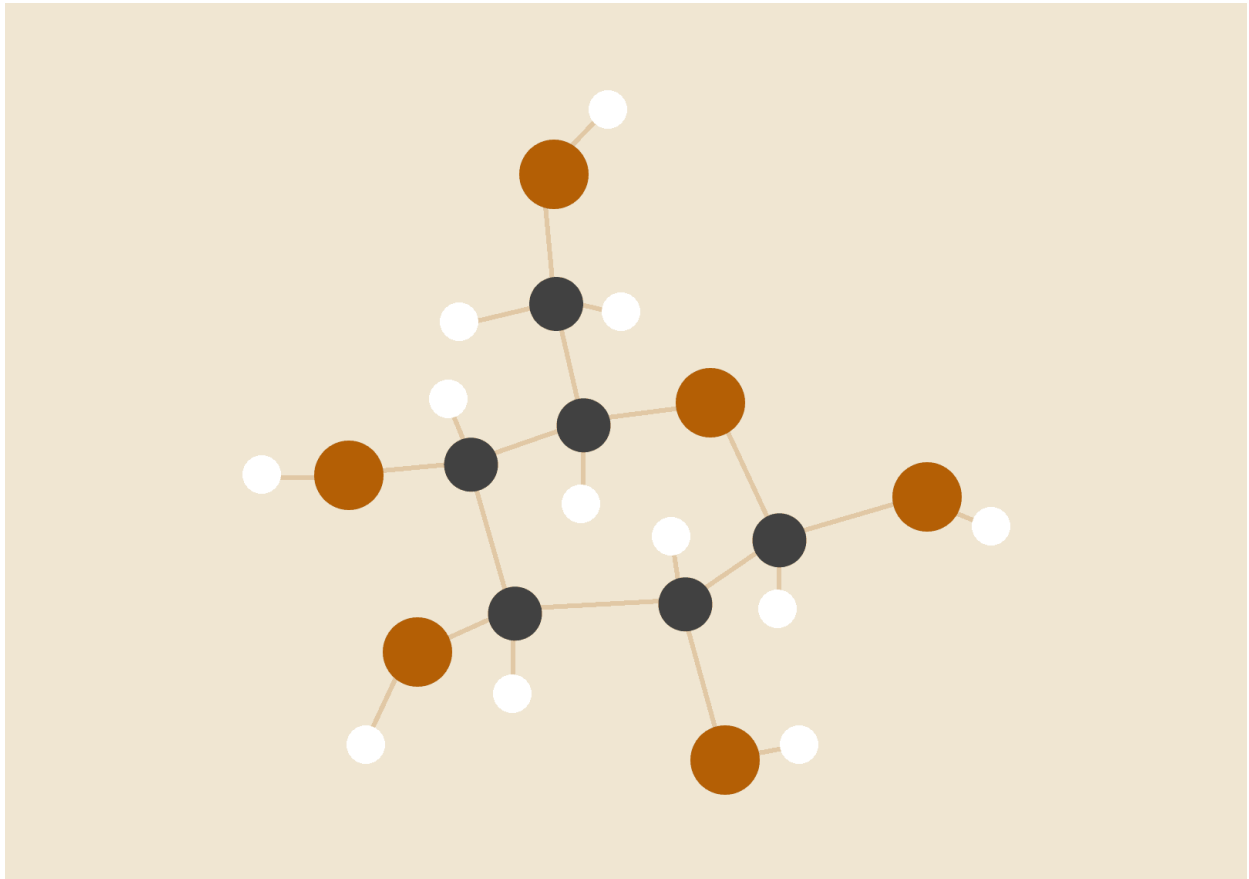


# Máquina de turing

Act.2.2

*Diseñar un algoritmo de máquina de turing que al inicio de la entrada tenga dos números binarios y realice la suma de  $n$  números.*

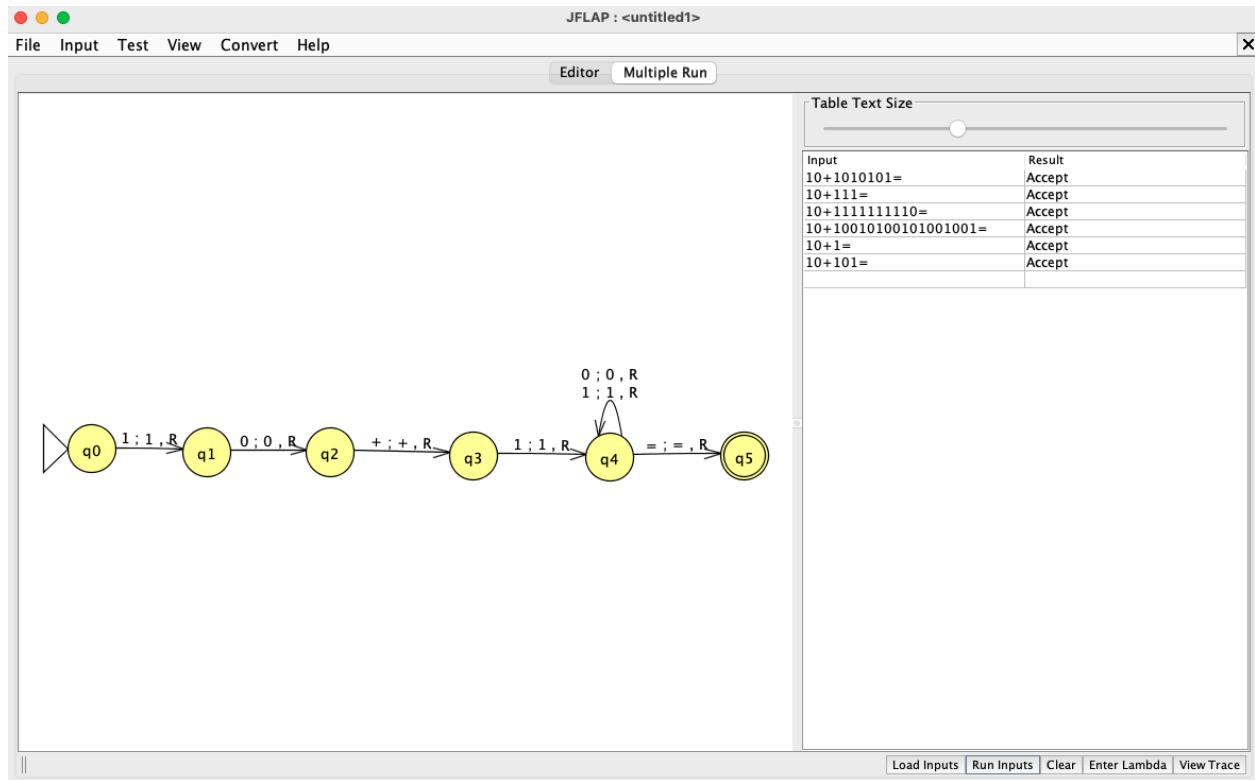


**Emiliano Nataren Del Rivero**

28/10/2024

7C

## INTRODUCCIÓN



Una máquina de Turing es un modelo computacional fundamental en la teoría de la computación, propuesto por Alan Turing en 1936. Consiste en un sistema abstracto que manipula símbolos en una cinta infinita siguiendo un conjunto de reglas. La máquina de Turing es importante porque proporciona una definición formal de algoritmo y de computabilidad. En teoría, puede resolver cualquier problema que sea computacionalmente solucionable, dado el tiempo y la memoria necesarios.

En este proyecto, hemos desarrollado una máquina de Turing capaz de realizar la suma en binario de una cantidad fija (representada como 10 o 2 en decimal) con un número binario proporcionado por el usuario. La entrada debe seguir el formato 10+[binary\_number], en el que 10+ es un prefijo constante que simboliza el número 2 y binary\_number representa un número en binario. Esta máquina de Turing valida la entrada, ejecuta la suma y muestra el resultado en binario en una interfaz gráfica.

## Definición formal

La máquina de Turing implementada puede ser definida formalmente como una séxtupla:

$$MT=(Q,\Sigma,\Gamma,\delta,q_0,q_f)$$

donde:

- $Q$ : Es el conjunto de estados, que en esta implementación contiene  $q_0, q_1, q_2, q_3, q_4$ , y  $q_f$  (estado final).
- $\Sigma$  Es el conjunto de símbolos de entrada, en este caso,  $\{0, 1, +\}$ .
- $\Gamma$ : Es el alfabeto de la cinta, que incluye  $\{0, 1, +, [\text{blank}] \}$ .
- $\delta$ : Es la función de transición, que mapea un par (estado, símbolo leído) a un trío (nuevo estado, símbolo a escribir, dirección de movimiento).
- $q_0$ : Es el estado inicial, que es  $q_0$ .
- $q_f$ : Es el estado de aceptación o final  $q_f$ .

**Tabla de Transiciones** La función de transición  $\delta$  está definida en el código mediante un diccionario `transitions`, el cual especifica cómo cambia el estado, el símbolo a escribir y la dirección de movimiento (izquierda o derecha) para cada combinación de estado y símbolo.

## Diseño de la interfaz gráfica

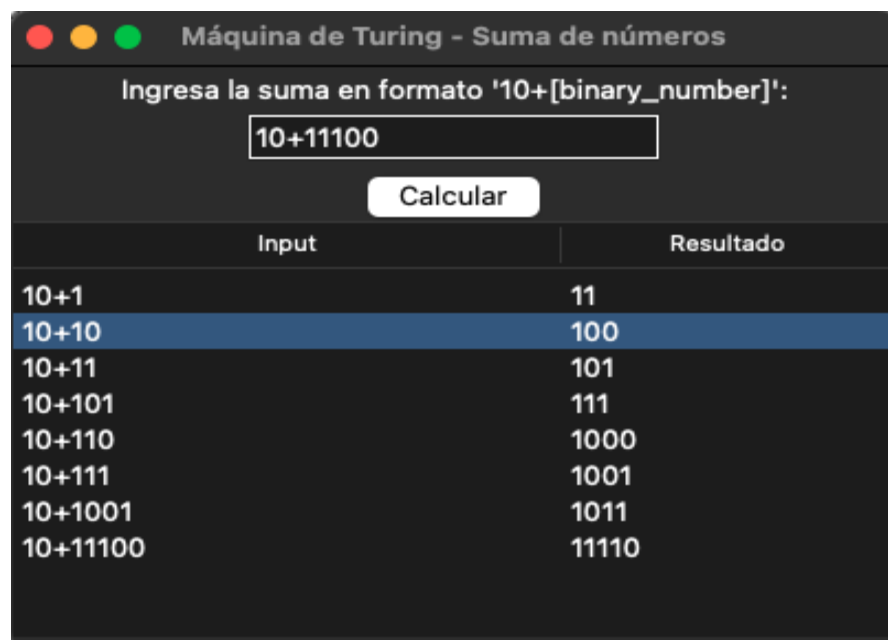
Para interactuar con la máquina de Turing, hemos creado una interfaz gráfica sencilla con **Tkinter**. La interfaz permite al usuario ingresar la expresión binaria y ver el resultado de cada operación en una tabla de historial.

### Elementos de la Interfaz:

- **Etiqueta de entrada (`input_label`)**: Proporciona una descripción para guiar al usuario a ingresar una expresión de suma en el formato adecuado (`10+[binary_number]`).
- **Campo de texto (`input_field`)**: Este campo permite al usuario ingresar la

expresión binaria que desea sumar con 10.

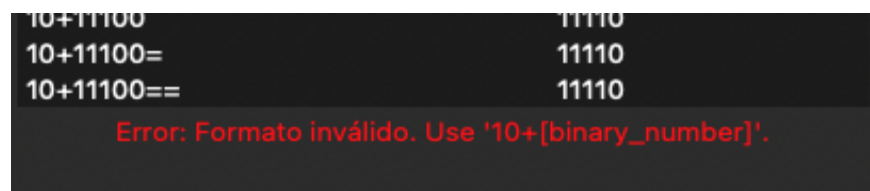
- **Botón "Calcular" (`run_button`)**: Al hacer clic en este botón, se ejecuta la función `run_turing_machine()`, la cual:
  - Valida que el formato de entrada sea correcto.
  - Ejecuta la máquina de Turing para realizar la suma en binario.
  - Muestra el resultado en una tabla y mantiene un registro de todas las operaciones realizadas.
- **Tabla de resultados (`results_table`)**: Implementada con un `Treeview` de Tkinter, muestra una tabla con dos columnas:
  - **Input**: Muestra la expresión de entrada proporcionada por el usuario.
  - **Resultado**: Muestra el resultado binario de la operación de suma.



Input	Resultado
10+1	11
10+10	100
10+11	101
10+101	111
10+110	1000
10+111	1001
10+1001	1011
10+11100	11110

- Interfaz tkinter con calculos previos

- **Etiqueta de error (`result_label`)**: Si el usuario ingresa un formato incorrecto, esta etiqueta muestra un mensaje de error en rojo.



10+11100	11110
10+11100=	11110
10+11100==	11110

Error: Formato inválido. Use '10+[binary\_number]'.

- 

- Mensaje de error

## Funcionamiento:

- El usuario ingresa una expresión en el campo de texto y hace clic en "Calcular".
- La función `run_turing_machine()` verifica si el formato de la entrada es válido (es decir, si comienza con `10+`).
  - Si la entrada es válida, se convierte en una lista de caracteres que representa la cinta de la máquina.
  - Luego, la función `turing_machine()` se encarga de realizar la suma en binario utilizando las transiciones definidas, moviéndose a la derecha o izquierda en la cinta y escribiendo los valores según corresponda.
  - Una vez completada la suma, el resultado se agrega a la tabla de resultados, junto con el input inicial.
  - Si el formato es incorrecto, se muestra un mensaje de error en la etiqueta `result_label`.

## Cadenas generadas con JFLAP

Para validar la precisión del comportamiento de la Máquina de Turing, las cadenas se generaron inicialmente utilizando **JFLAP**. JFLAP es una herramienta educativa que permite visualizar y simular autómatas y Máquinas de Turing, lo que nos permitió verificar de manera precisa que las cadenas introducidas cumplen con las especificaciones del lenguaje  $\{(ab^2)^n \mid n > 0\}$ . Esta generación automática de cadenas permitió realizar pruebas exhaustivas sobre la implementación, asegurando que la Máquina de Turing respondiera de acuerdo a los parámetros definidos.

## Código de la Máquina de Turing

```
transitions = {
    ("q0", "1"): ("q0", "1", "R"),
    ("q0", "0"): ("q1", "0", "R"),
    ("q1", "+"): ("q2", "+", "R"),
    ("q2", "0"): ("q2", "0", "R"),
    ("q2", "1"): ("q2", "1", "R"),
    ("q2", " "): ("q3", " ", "L"),
    ("q3", "+"): ("q4", "+", "R"),
}

def turing_machine(tape, state):
    head_position = 0
    buffer = []

    while state != "qf":
        symbol = tape[head_position] if head_position < len(tape) else " "
        if (state, symbol) not in transitions:
            break
        new_state, new_symbol, direction = transitions[(state, symbol)]

        if head_position < len(tape):
            tape[head_position] = new_symbol
        else:
            tape.append(new_symbol)

        if state == "q2" and symbol in ["0", "1"]:
            buffer.append(symbol)

        state = new_state
        head_position += 1 if direction == "R" else -1

        if head_position < 0:
            tape.insert(0, " ")
            head_position = 0

    number = int("".join(buffer), 2)
    result = bin(2 + number)[2:]

    return result
```

El **código central** de la Máquina de Turing implementado en este proyecto encapsula la lógica fundamental del autómata, donde se definen los **estados**, **transiciones** y el proceso de ejecución de la máquina. Cada estado está configurado para guiar la lectura y escritura de la cinta, asegurando que la máquina siga el comportamiento esperado según las reglas del lenguaje definido.

Este núcleo de código no solo gestiona las transiciones entre estados de manera eficiente, sino que también maneja correctamente la detención de la máquina al alcanzar un estado de aceptación o rechazo. La implementación fue diseñada teniendo en cuenta tanto la **eficiencia** como la **claridad**, lo que facilita su comprensión y modificación futura para adaptarse a otros lenguajes o funcionalidades adicionales.

## Conclusión

La creación de esta máquina de Turing para realizar sumas en binario nos permite profundizar en el funcionamiento de un modelo computacional básico y en su implementación práctica. A través de este proyecto, se ha comprendido cómo se aplican los conceptos teóricos de una máquina de Turing, como la función de transición, los estados y la cinta, en un programa que realiza operaciones computacionales concretas.

Asimismo, la implementación de una interfaz gráfica ha permitido experimentar con el desarrollo de aplicaciones interactivas en Python utilizando **Tkinter**, aprendiendo a organizar y mostrar información en tablas y a manejar eventos de usuario. Este proyecto proporciona una base sólida para el estudio de conceptos avanzados en teoría de la computación y en el diseño de interfaces gráficas.