UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA
SEDE BOCA DEL MONTE
ING. SISTEMAS DE INFORMACIÓN Y CIENCIAS DE LA COMPUTACIÓN
CURSO: AUTÓMATAS Y LENGUAJES FORMALES
SECCIÓN: “B”

MANUAL TÉCNICO

MÁQUINA DE TURING

CATEDRÁTICO:
EZEQUIEL URIZAR

EDRAS FERNANDO TATUACA ALVARADO
CARNÉ: 7690-22-11542

ANGEL ENRIQUE IBAÑEZ LINARES
CARNÉ: 7690-22-19119

INTRODUCCIÓN

Este manual contiene los requisitos necesarios para que un usuario pueda ejecutar correctamente el programa desarrollado por Ángel Ibañez y mi persona, Edras Tatuaca que consiste en ejecutar una máquina de Turing de tal manera que el usuario pueda expresarse en cualquier lenguaje informático y que el programa lo pueda interpretar a su manera.

Se mencionan las especificaciones que el usuario necesita para poder ejecutarlo e incluye un análisis a nivel código de como está comprendido el programa para poder entender los procesos que realiza.

REQUERIMIENTOS TÉCNICOS

Entorno de desarrollo:	Apache NetBeans 22
Sistema Operativo:	Windows

ANÁLISIS DEL CÓDIGO:

Este código implementa una Máquina de Turing en Java. Una Máquina de Turing es un modelo matemático que simula el funcionamiento de una computadora mediante una cinta de datos, un cabezal de lectura/escritura, y una serie de estados y transiciones.

Clase Máquina Turing: Es la clase principal que simula el funcionamiento de la Máquina de Turing.

Atributos:

- cinta: Representa la cinta de datos, donde se almacenan los símbolos.
- posicionCabezal: Indica la posición actual del cabezal en la cinta.
- estadoActual y estadoInicial: Representan el estado actual e inicial de la máquina.
- estadoAceptacion: Estado de aceptación, que indica cuándo la máquina acepta la cadena de entrada.
- tablaTransiciones: Mapa que guarda las transiciones de la máquina, indicando el próximo estado, el símbolo a escribir, y el movimiento del cabezal.

Métodos:

- agregarTransicion: Permite agregar transiciones entre estados según un símbolo leído.
- configurarEntrada: Coloca una cadena de entrada en la cinta y posiciona el cabezal en el centro.
- ejecutar: Ejecuta la máquina de Turing paso a paso, aplicando las transiciones según el símbolo en la cinta y actualizando el estado y posición del cabezal. También registra cada paso de ejecución.
- mostrarCinta: Muestra el contenido actual de la cinta en consola.

Clase interna Transicion:

- siguienteEstado: El siguiente estado al que se mueve la máquina.
- simboloEscribir: Símbolo que la máquina escribe en la cinta.
- direccionMovimiento: Dirección en la que se mueve el cabezal (derecha o izquierda).

Clase Ingresar Permite al usuario ingresar los datos necesarios para definir un autómata, visualizarlos en la interfaz y guardarlos en un archivo de texto.

Atributos principales:

- datos: una lista para almacenar los datos ingresados,
- textField1Y2Guardados: booleano para controlar si ya se guardaron el estado inicial y final.

Componentes y funciones:

- **Botón "Ingresar"** (jButton2): guarda los datos de los campos de texto en la lista datos.
- **Botón "Guardar"** (jButton3): guarda los datos de la lista datos en un archivo archivo.txt en el formato especificado (incluye estado actual, símbolo leído, nuevo estado, símbolo escrito y dirección).
- **Botón "Regresar"** (jButton1): regresa a la ventana de menú principal.

Clase Validar: Define una interfaz gráfica en Java, usando Swing, para cargar y validar las transiciones de una máquina de Turing.

Cargar Transiciones de Archivo:

Con el botón "Cargar", se leen los datos desde un archivo archivo.txt. Las primeras dos líneas contienen los estados inicial y final, los cuales se muestran en los campos de texto correspondientes, y las líneas siguientes, con las transiciones, se cargan en una tabla.

Validar Cadena:

El botón "Validar" permite ingresar una cadena para verificar su aceptación en la máquina de Turing. donde:

- Se configuran las transiciones cargadas de la tabla.
- Se establece la cadena de entrada.
- Se ejecuta la máquina, validando si la cadena es aceptada o rechazada, y se muestra el proceso paso a paso.

Regresar al Menú:

Con el botón "Regresar" se cierra la ventana actual y se abre la ventana de menú.

Clase menú: Funciona como un menú principal en la aplicación

Navegación entre Ventanas:

- **Botón "Ingresar"**: Abre una ventana Ingresar, que sirve para ingresar los datos del lenguaje que el usuario desee.
- **Botón "Validar"**: Abre la ventana Validar, que permite cargar y validar las transiciones de la máquina de Turing.

Interfaz Gráfica:

- Incluye un mensaje de bienvenida y etiquetas que describen las funcionalidades de cada botón (Ingresar Tabla y Validar).

CÓDIGO DE LA CLASE MAIN:

```
package com.mycompany.final_automatas;

import java.util.HashMap;

import java.util.List;

import java.util.Map;


public class MaquinaTuring {


    // Definición de atributos de la máquina de Turing

    private String[] cinta;  // Representación de la cinta de la máquina, donde se almacenan los datos

    private int posicionCabezal;  // Posición actual del cabezal de lectura/escritura en la cinta

    private String estadoActual;  // Estado actual de la máquina de Turing

    private String estadoInicial;  // Estado inicial de la máquina de Turing

    private String estadoAceptacion;  // Estado de aceptación (final) de la máquina

    private Map<String, Transicion> tablaTransiciones;  // Tabla de transiciones que define el comportamiento de la máquina

    // Constructor de la máquina de Turing

    public MaquinaTuring(String estadoInicial, String estadoAceptacion) {

        this.cinta = new String[100];  // Inicializa la cinta con un tamaño de 100 celdas

        this.posicionCabezal = 50;  // Coloca el cabezal en el centro de la cinta

        this.estadoInicial = estadoInicial;

        this.estadoActual = estadoInicial;

        this.estadoAceptacion = estadoAceptacion;

        this.tablaTransiciones = new HashMap<>();  // Inicializa la tabla de transiciones

    }
```

```
    // Método para agregar una nueva transición a la tabla de transiciones
    public void agregarTransicion(String estado, char simboloLectura, String siguienteEstado, char simboloEscribir, char movimiento) {
        int direccionMovimiento;
        // Define la dirección del movimiento: derecha (R) o izquierda (L)
        if (movimiento == 'R') {
            direccionMovimiento = 1;
        } else if (movimiento == 'L') {
            direccionMovimiento = -1;
        } else {
            throw new IllegalArgumentException("Movimiento inválido. Use 'R' para derecha y 'L' para izquierda.");
        }
        // Crea una clave única para la transición (estado y símbolo de lectura)
        String claveTransicion = estado + "_" + simboloLectura;
        // Agrega la transición a la tabla de transiciones
        tablaTransiciones.put(claveTransicion, new Transicion(siguienteEstado, simboloEscribir, direccionMovimiento));
    }
    // Método para configurar la entrada en la cinta
    public void configurarEntrada(String entrada) {
        this.cinta = new String[100];  // Reinicia la cinta a su tamaño inicial
        this.posicionCabezal = 50;  // Reinicia la posición del cabezal al centro
        for (int i = 0; i < entrada.length(); i++) {
            // Llena la cinta con la entrada, empezando desde la posición del cabezal
            cinta[posicionCabezal + i] = String.valueOf(entrada.charAt(i));
        }
        this.estadoActual = this.estadoInicial;  // Restablece el estado actual al inicial
```

```
    // Método para ejecutar la máquina de Turing y registrar cada paso
    public boolean ejecutar(StringBuilder registroPasos, List<String> estadosList) {
        while (!estadoActual.equals(estadoAceptacion)) {
            // Obtiene el símbolo de la cinta en la posición del cabezal o un espacio en blanco si es nulo
            char simboloLectura = cinta[posicionCabezal] != null ? cinta[posicionCabezal].charAt(0) : '_';
            String claveTransicion = estadoActual + "_" + simboloLectura;

            // Registrar el contenido de la cinta y el estado actual
            StringBuilder cintaActual = new StringBuilder();
            cintaActual.append("Cinta: ");
            for (int i = 0; i < cinta.length; i++) {
                if (i == posicionCabezal) {
                    cintaActual.append("[ ").append(cinta[i] != null ? cinta[i] : "_").append(" ] ");
                } else {
                    cintaActual.append(cinta[i] != null ? cinta[i] : "_").append(" ");
            cintaActual.append("\n");
            registroPasos.append("Estado actual: ").append(estadoActual).append("\n");
            // Si no existe una transición para el estado actual y el símbolo de lectura, rechaza la cadena
            if (!tablaTransiciones.containsKey(claveTransicion)) {
                registroPasos.append(cintaActual).append("No hay otra transición: cadena RECHAZADA.\n");
                return false;
            }
```

```
        // Obtiene la transición y realiza las operaciones correspondientes
        Transicion transicion = tablaTransiciones.get(claveTransicion);
        cinta[posicionCabezal] = String.valueOf(transicion.simboloEscribir); // Escribe el nuevo símbolo en la cinta
        // Registrar la transición
        String movimiento = transicion.direccionMovimiento == 1 ? "Derecha →" : "Izquierda ←";
        cintaActual.append("Escribiendo: ").append(transicion.simboloEscribir)
                .append(", Movimiento: ").append(movimiento).append("\n");
        registroPasos.append(cintaActual);  // Agrega la información de la cinta al registro de pasos

        // Actualiza la posición del cabezal y el estado actual
        posicionCabezal += transicion.direccionMovimiento;
        estadoActual = transicion.siguienteEstado;      }
    // Registrar el estado final y el contenido final de la cinta
    registroPasos.append("Estado actual: ").append(estadoActual).append("\n");
    StringBuilder cintaFinal = new StringBuilder();
    cintaFinal.append("Cinta: ");
    for (int i = 0; i < cinta.length; i++) {
        if (i == posicionCabezal) {
            cintaFinal.append("[ ").append(cinta[i] != null ? cinta[i] : "_").append(" ] ");
        } else {
            cintaFinal.append(cinta[i] != null ? cinta[i] : "_").append(" ");
        }
    }
    registroPasos.append(cintaFinal).append("\n");
```

```
        // Indica si la cadena es aceptada
        if (estadoActual.equals(estadoAceptacion)) {
            registroPasos.append("CADENA ACEPTADA.\n");
        }
        return estadoActual.equals(estadoAceptacion);
    }
    // Método para mostrar el contenido de la cinta en consola
    public void mostrarCinta() {
        for (String celda : cinta) {
            System.out.print(celda != null ? celda : "_");  // Imprime el contenido de cada
celda o un espacio en blanco si es nula
        }
        System.out.println();
    }
}
// Clase interna que representa una transición de la máquina de Turing
class Transicion {
    String siguienteEstado;  // Estado al que se transita
    char simboloEscribir;  // Símbolo que se escribe en la cinta
    int direccionMovimiento;  // Dirección del movimiento del cabezal (1 para derecha,
-1 para izquierda)

    // Constructor para crear una transición
    Transicion(String siguienteEstado, char simboloEscribir, int direccionMovimiento)
{
        this.siguienteEstado = siguienteEstado;
        this.simboloEscribir = simboloEscribir;
        this.direccionMovimiento = direccionMovimiento`}
```