PROYECTO - *Validacion de correos*
Automata de Estados Finitos

CARRERA: Ingeniería en Computación

NOMBRES: *Efrain Robles Pulido*
Miguel Alejando Lomeli Haro
Marco Aurelio Dominguez Amezcua

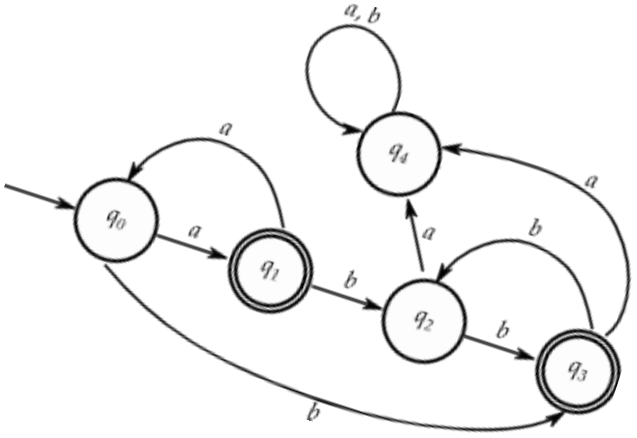
CÓDIGO: 221350095
221350486
216818534

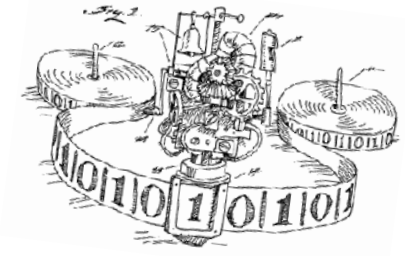
MATERIA: Teoría de la computación

MAESTRA: Abelardo Gomez Andrade

SECCIÓN: D17 **CALENDARIO:** 2023A

UNIVERSIDAD DE GUADALAJARA





ÍNDICE:

Introducción:

Es importante destacar que existen problemas relacionados con las tecnologías de la información en el mundo, y sus soluciones pueden adoptar diversas formas. Una de las principales formas de abordar estos problemas es mediante el uso de la lógica, que es fundamental para establecer bases sólidas en la resolución. La práctica también juega un papel crucial en la solución de problemas.

En el campo de la teoría de la computación, las soluciones pueden presentarse en forma de autómatas finitos, máquinas de Turing, autómatas de pila y máquinas de estado finito. Estas representaciones abstractas de la computación permiten describir con precisión el comportamiento de una computadora a través de transiciones y cambios de estado.

En este proyecto, nuestra prioridad es demostrar cómo es posible resolver un problema común en las páginas de registro de usuarios, utilizando instrucciones lógicas y representaciones que mejoran la comprensión de la importancia de la teoría de la computación. Esto nos permite generar un pensamiento razonado y útil para desarrollar lenguajes que produzcan un autómata finito y viceversa.

Planteamiento del problema:

Nuestra propuesta de proyecto se deriva a partir de que en ocasiones hemos decidido desarrollar formularios para páginas web o en ocasiones para un registro dentro de alguna TDA, en donde es necesario ingresar ciertos datos que tengan ciertos caracteres específicos, entre ellos se encuentran nombre, número de teléfono, dirección, edad, cumpleaños, etcétera. Para este caso, nos vamos a enfocar en el correo electrónico, verificar que se cumplan todos los requisitos para que tenga la validez de un correo, ya que este tiene que tener una estructura algo distinta a los demás datos, por consiguiente, hay total seguridad de que es imposible dar cómo válido un correo electrónico que sea por ejemplo “efrainRobles123.com” o “marco amezcua@”. Como se pudo observar se requiere de unos caracteres especiales y que la mayoría de ellos están en posiciones específicas para aceptar que sea un correo correcto, como puede ser el “@” o el “.”, ya que estos caracteres son importantes para la validación de un correo electrónico y sino se cumplen el registro tendrá que quedar en nulo. Entonces, el equipo propondrá una solución a esta problemática para hacer tener la solución más eficiente, en el que iremos realizando un análisis para demostrar situaciones posibles ante la problemática, además de ir comprobando cada uno de los casos existentes tanto de no aceptación y aceptación de los registros de correos electrónicos.

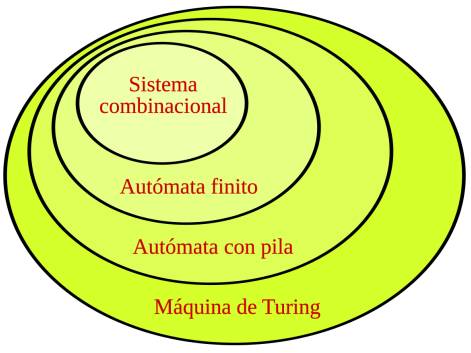
Objetivo:

Nosotros queremos demostrar una posible solución ante esta problemática para verificar que el registro de un correo electrónico sea correctamente escrito. Mediante la utilización de un autómata de estados finitos que nosotros diseñemos para que al recibir caracteres erróneos y/o en diferentes orden no deseado nos rechace esa entrada realizada, por lo que nos permitirá crear un código de programación que sea lo más eficientemente posible. Además, nos interesa comprender las transiciones relevantes que ocurren al cambiar de un estado a otro al recibir caracteres que pueden ser válidos o no. Por lo que deberemos de cumplir con las reglas correctas para la estructura de un correo electrónico, logrando entender con mayor detalle como un programa puede validar un correo electrónico y cómo los cambios de estado influyen en el comportamiento del autómata de estados finitos durante su ejecución.

Justificación:

La validación de correos electrónicos es una tarea crucial en culturizar sistemas que involucran la comunicación por correo electrónico, como sistemas de registros en línea, formularios de contacto y sistemas de verificación de identidad. Si no se logra esta verificación se puede tener graves consecuencias negativas en la eficiencia y seguridad de estos sistemas. Si se utiliza un autómata de estados finitos puede proporcionar una solución bastante alta y eficiente, lo que nos permitirá examinar cada carácter del correo electrónico y determinar si cumple con las reglas de formato adecuadas. Además, el autómata puede detectar errores de entrada y rechazar correos electrónicos que no cumplen con los requisitos mínimos, reduciendo así la posibilidad de errores y mejorando la eficiencia del sistema. Además, su implementación nos da una gran escalabilidad, a medida que los sistemas de registro y verificación de identidad se vuelven más complejos, la cantidad de correos electrónicos que deben ser verificados aumenta, lo que hace que la tarea de validación manual sea cada vez más difícil. Un autómata de estados finitos puede manejar grandes cantidades de correos electrónicos de manera eficiente y rápida, lo que lo convierte en una solución escalable y rentable.

Marco teórico:



Teoría de autómatas.

Para facilitar la comprensión del uso del autómata finito como simulación en nuestro proyecto, revisaremos los conceptos

básicos de los autómatas y destacaremos su importancia para la validación de correos electrónicos. Es relevante mencionar que el término "Autómata" tiene un significado fundamental en la teoría de los autómatas. La palabra "autómata" proviene del griego αὐτόματα, que significa "autorrealización". En este contexto, un autómata se refiere a un dispositivo informático abstracto que sigue una secuencia predefinida de operaciones de forma automática. Podemos decir que sigue una serie de reglas establecidas para alcanzar una posible solución.

1. Componentes: Un autómata finito consta de los siguientes componentes:
    - Conjunto de estados: Es un conjunto finito de estados posibles en los que puede encontrarse el autómata.
    - Alfabeto: Es un conjunto finito de símbolos utilizados como entradas para el autómata.
    - Función de transición: Es una función que especifica cómo el autómata cambia de un estado a otro en respuesta a una entrada. La función de transición mapea un estado y un símbolo de entrada a un nuevo estado.
    - Estado inicial: Es el estado en el que se encuentra el autómata al inicio.
    - Conjunto de estados finales: Es un conjunto de estados que se consideran estados de aceptación. Si el autómata alcanza uno de estos estados después de procesar una entrada, se dice que la entrada es aceptada.
2. Tipos de autómatas finitos: Hay dos tipos principales de autómatas finitos:
    - Autómata finito determinista (AFD): En un AFD, para cada estado y símbolo de entrada dado, hay exactamente una transición definida. Esto significa que el autómata siempre puede determinar el siguiente estado de manera única. Cada estado tiene una única transición para cada símbolo del alfabeto.
    - Autómata finito no determinista (AFND): En un AFND, para un estado y un símbolo de entrada, puede haber varias transiciones posibles o incluso ninguna transición definida. Esto significa que el autómata no puede determinar de manera única el siguiente estado. Puede tener transiciones que lleven a múltiples estados simultáneamente o transiciones nulas (transiciones sin entrada).

3. Aplicaciones: Los autómatas finitos tienen diversas aplicaciones en ciencias de la computación y más allá. Algunas de ellas son:
    - Análisis y diseño de lenguajes formales.
    - Modelado y análisis de sistemas de control.
    - Validación y verificación de programas y protocolos.

- Implementación de analizadores léxicos y sintácticos en compiladores.
- Diseño de circuitos digitales y sistemas secuenciales.

Cuando un autómata tiene un número finito de estados, se le denomina autómata finito (FA) o máquina de estados finitos. La teoría de los autómatas es una rama de las ciencias computacionales que se enfoca en el estudio de dispositivos de cálculo abstractos, es decir, "máquinas". La importancia de la teoría de los autómatas se puede explicar de manera sencilla: describe de forma precisa los límites de lo que una máquina de cálculo puede y no puede hacer, como los cálculos matemáticos que pueden tener una resolución válida o inválida según su funcionamiento en una simulación.

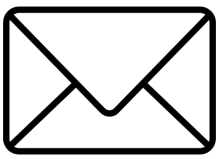
En este caso, la teoría de autómatas nos permite crear un reconocedor de caracteres válidos, lo cual es relevante para validar y demostrar el funcionamiento de nuestro código de una manera más sencilla. La teoría de los autómatas también está estrechamente relacionada con la teoría del lenguaje formal, ya que los autómatas suelen formarse en base a los lenguajes que pueden reconocer.

Considerando todo lo anterior, hemos decidido utilizar un autómata finito para la validación de correos electrónicos en nuestro proyecto. Lo implementaremos con las transiciones y cambios de estado necesarios, los cuales representan una validación correcta de los caracteres que conforman una dirección de correo electrónico.

Para comprender mejor cómo se aplica un autómata finito en nuestro proyecto, es necesario destacar una definición concreta de lo que son los autómatas finitos. Un autómata finito es un modelo matemático de una máquina que acepta cadenas de un lenguaje definido sobre un alfabeto A. Está compuesto por un conjunto finito de estados y un conjunto de transiciones entre esos estados, que dependen de los símbolos de la cadena de entrada. El autómata finito acepta una cadena x si la secuencia de transiciones correspondientes a los símbolos de x lleva al estado inicial a un estado final.

Verificador de una dirección de correo.

Una página web necesita verificar la validez de una dirección de correo electrónico ingresada por el usuario. Antes de determinar si la dirección es correcta, se deben aplicar una serie de reglas. Estas reglas están estrechamente relacionadas con el uso adecuado de los caracteres, su posición y, por último, el dominio del correo electrónico. El dominio comúnmente se identifica mediante un dominio de primer nivel genérico como ".com", pero también pueden existir



otras variaciones de dominio, como ".net", ".mx" o incluso ".ud", entre otros. El correcto formato de la dirección de correo electrónico es importante, ya que determinará si la página web acepta o rechaza la dirección ingresada. Si los caracteres están dispuestos de manera incorrecta, es probable que el ingreso sea rechazado debido a un formato inválido.

Es importante destacar que, como equipo, hemos decidido mostrar un pequeño ejemplo de cómo debe ser formulada una dirección de correo electrónico para que sea aceptada por nuestro software antes de presentar la simulación de nuestro autómata finito y su funcionamiento.

Una dirección de correo electrónico consta de caracteres y partes fundamentales para su validez:

1. El nombre de usuario, que es el identificador de la persona que utilizará y gestionará el correo electrónico. Puede estar compuesto por cualquier combinación de caracteres deseables. gato-MX, j123alejandro, etc.

2. El símbolo "@" (arroba), que siempre debe seguir al nombre de usuario.

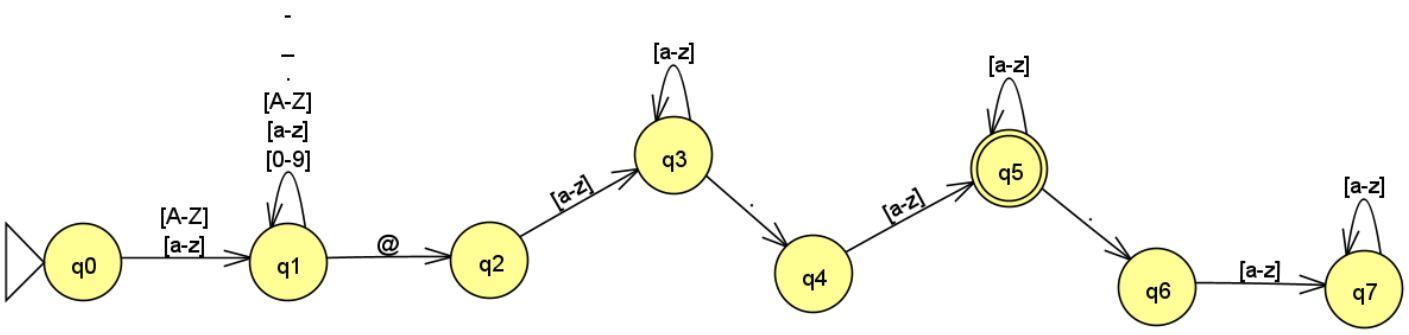
3. El dominio, que consta de dos subgrupos:
    - La organización o nombre del servidor de la organización.
    - El punto ".", que siempre se utiliza en nuestro proyecto.
    - El tipo, que define la funcionalidad del dominio.

Estos elementos son esenciales para que una dirección de correo electrónico sea válida y pueda ser aceptada por nuestro software.

Metodología

Durante este proceso de investigación hemos aprendido que podemos validar y comprobar con muchos ejemplos sobre el tema de los correos electrónicos, por ello es importante tener bien definido y diseñado de manera correcta la estructura de cómo se conforma un correo electrónico. JFLAP es una aplicación desarrollada en JAVA especializada en proveer herramientas necesarias para el diseño de varios diagramas y apegada a la teoría de la computación para dar su aporte práctico, con esta aplicación podemos generar infinidad de diagramas en los cuales podamos comprobar si alguna cadena es válida o no, es por ellos que decidimos diseñar un autómata finito determinista el cual nos permita validar si una

dirección de correo electrónico es válida o no. A continuación, se muestra el autómata finito determinista que usamos:



Como sabemos, los autómatas finitos cuentan con una representación en forma de tupla en la que tienen varios conjuntos y valores que nos indican cómo trabaja, dicha tupla tiene esta forma $AFD = \{\Sigma,\ Q,\ f,\ q_0,\ F\}$, este AF se conforma de la siguiente forma:

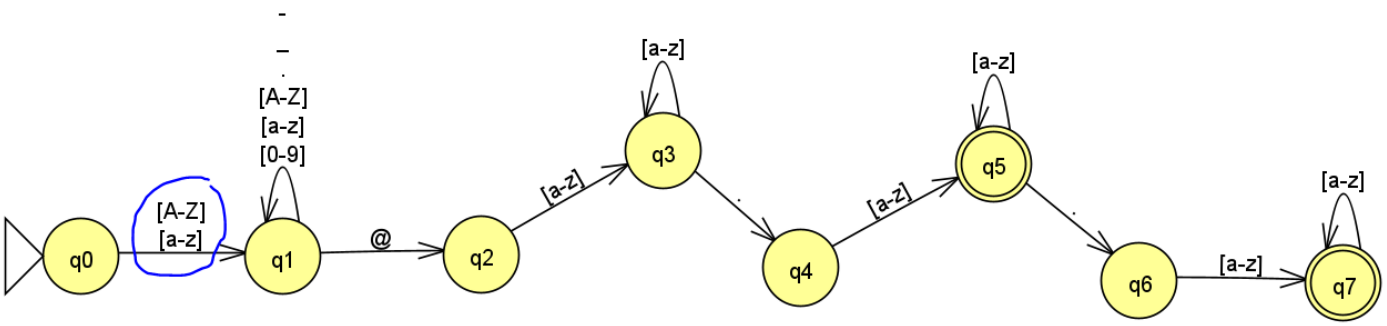
- Σ: Es el alfabeto finito de símbolos: a-z, A-Z, 0-9, @, -, _, .
- Q: Es el conjunto finito de estados $\{q_0,\ q_1,\ q_2,\ q_3,\ q_4,\ q_5,\ q_6,\ q_7\}$
- q_0: Es el estado inicial.
- F: Es el conjunto de estados de aceptación $\{q_5,\ q_7\}$
- f: Es la funcion de transicion de la forma $< q_i\, x\, q_j >$, donde "$q_i\, q_j$", son estados del conjunto Q.

Tabla de estados

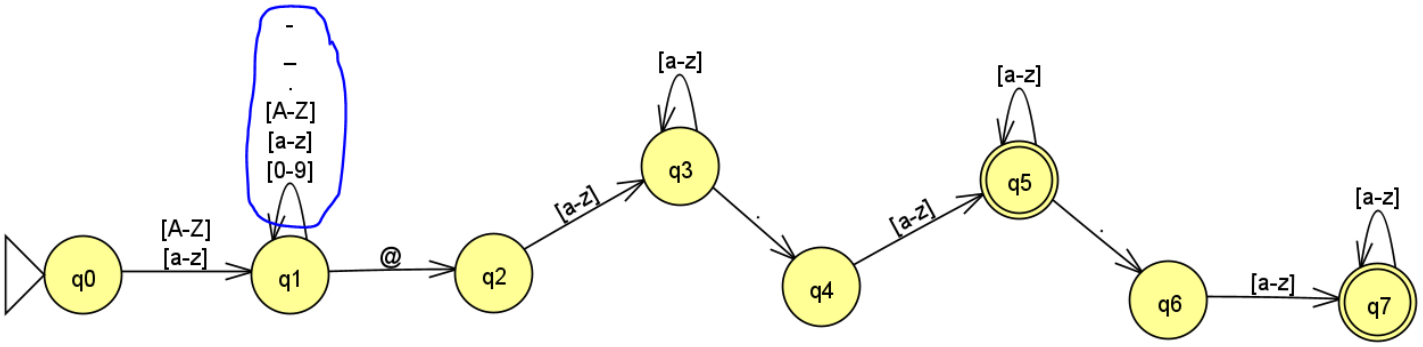
Estado / Alfabeto	a-z	A-Z	0-9	@	.	_	-
$\rightarrow q_0$	q_1	q_1	-	-	-	-	-
q_1	q_1	q_1	q_1	q_2	q_1	q_1	q_1
q_2	q_3	-	-	-	-	-	-
q_3	q_3	-	-	-	q_4	-	-
q_4	q_5	-	-	-	-	-	-

$*q_5$	q_5	-	-	-	q_6	-	-
q_6	q_7	-	-	-	-	-	-
$*q_7$	q_7	-	-	-	-	-	-

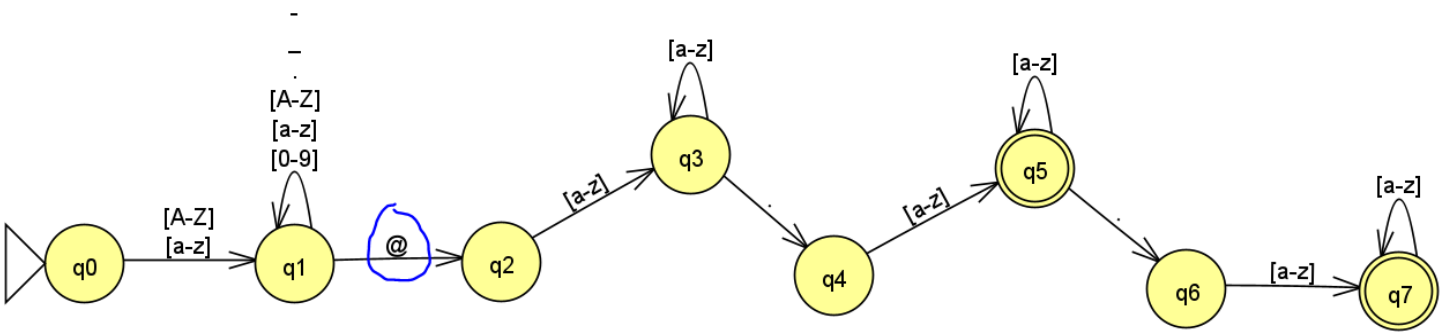
En la tabla de transiciones podemos darnos cuenta cual es el funcionamiento del autómata, se indica que nuestro punto de partida es el estado $q0$ el cual solamente nos permite cambiarnos de estado si es que utilizamos un carácter que se encuentre dentro del rango [a-zA-Z], de esta forma nos aseguramos que por lo menos haya un carácter alfabético en el nombre del usuario:



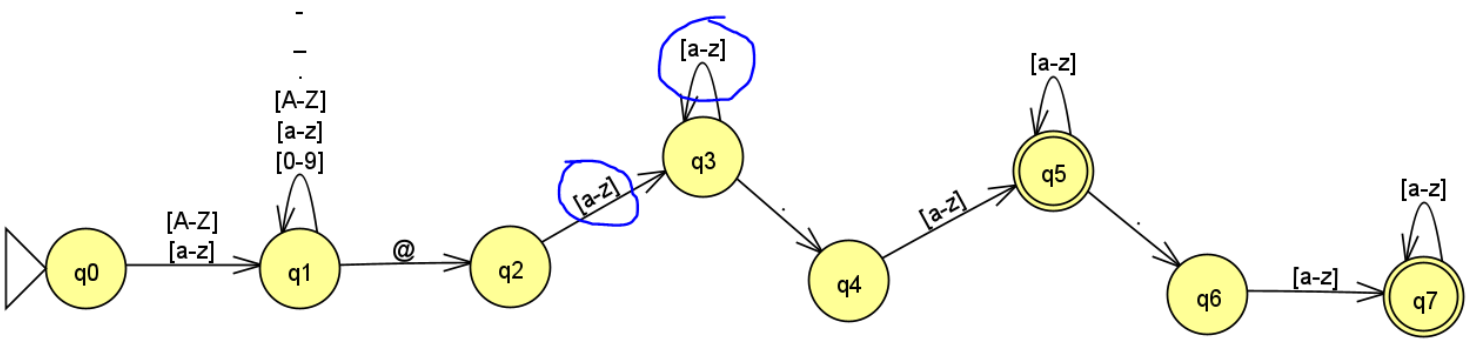
Posteriormente, llegamos al estado $q1$ en el cual podemos permanecer realizando inserciones a la cadena de diferentes símbolos estos pueden ser alfanuméricos, guiones (bajo y medio) y puntos:



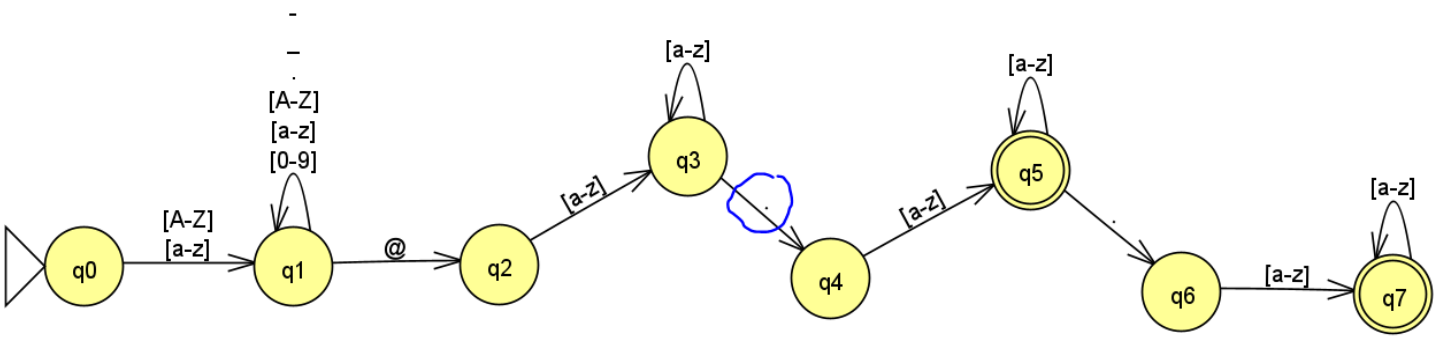
Para realizar un nuevo cambio de estado es obligatorio insertar al menos un arroba (@), para cambiar de campo en nuestro correo electrónico:



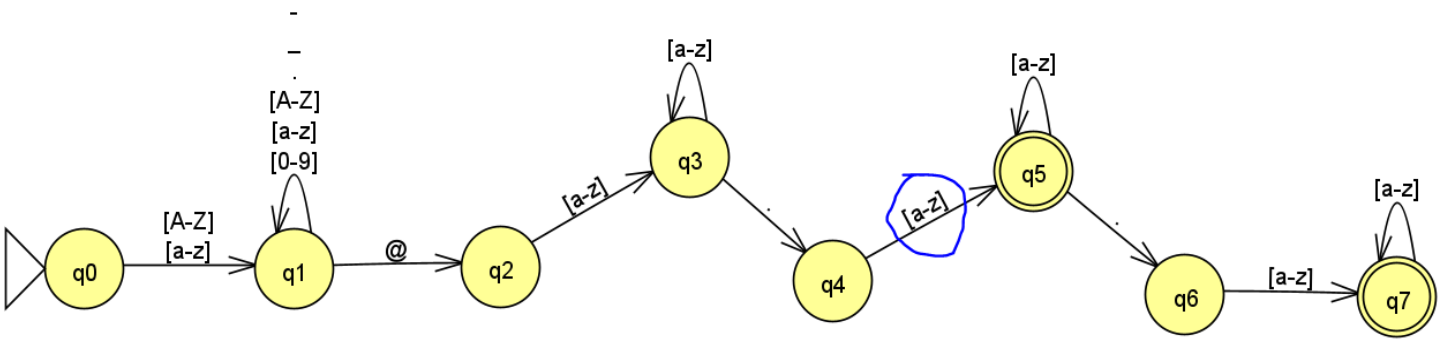
Estando ahora en el estado q2 tenemos que al menos ingresar un carácter alfabético en su versión minúscula para continuar con el dominio, este dominio solamente permite letras y al menos una, de tal forma que estaremos ubicados en el estado q3 hasta decidir cambiar de estado:



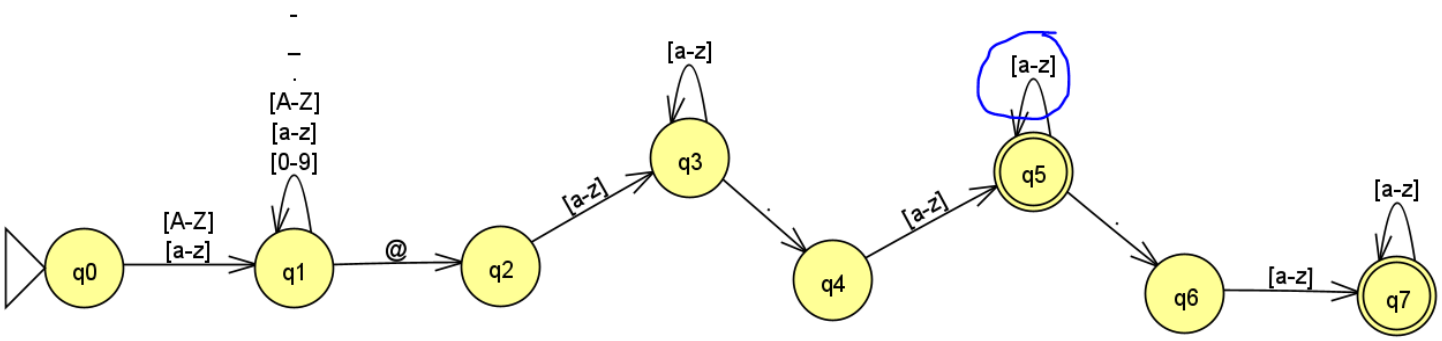
A continuación, requerimos de un punto simple para poder avanzar de q3 a q4:



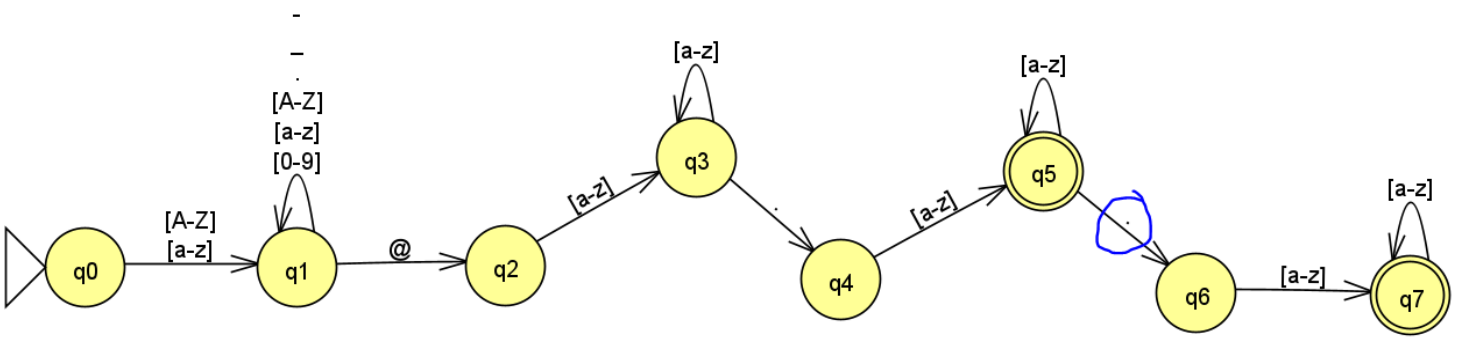
Ya ubicados en el estado actual como q4 de igual manera debemos de insertar por lo menos una letra para poder avanzar a q5 el cual ya es un estado de aceptación:



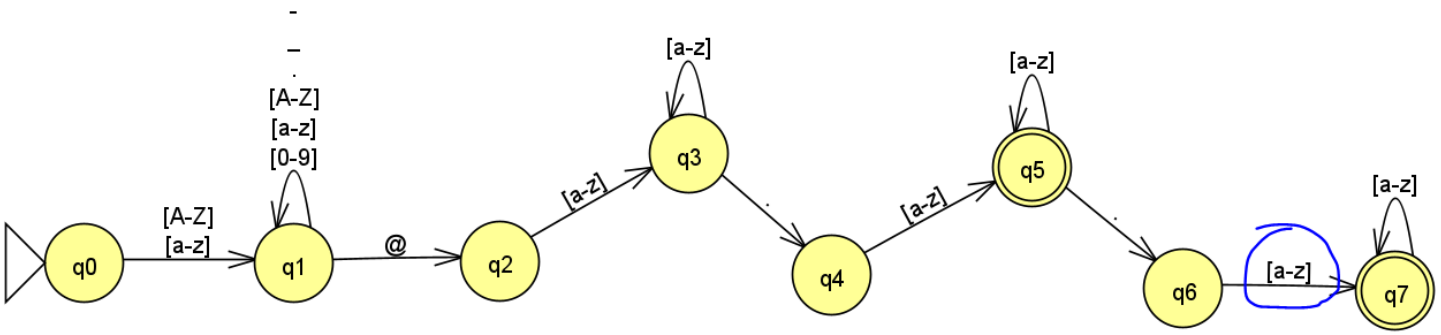
Estando en el estado q5 podemos ingresar una cadena de letras minúsculas para indicar la empresa o el dominio de nuestra dirección de correo electrónico, actualmente q5 es de aceptación, ya que contiene el campo del tipo de empresa, en este caso podemos permanecer en este estado y permanecer en un estado de aceptación, sin embargo, hay algunas direcciones de correo que permiten también un campo más que representa el país de dicha dirección:



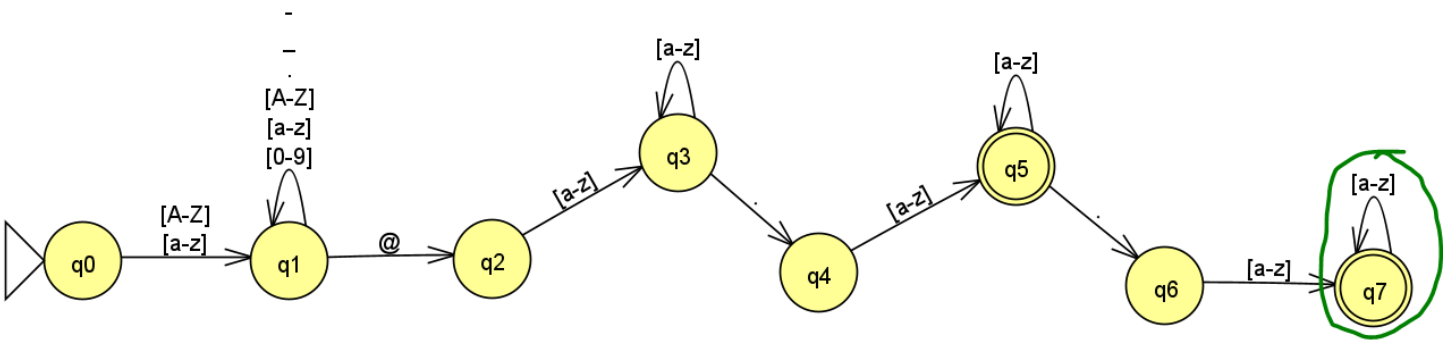
Entonces, estando en q5 debemos de ingresar un '.' para poder avanzar al siguiente estado que es q6:



Ya estando en q6 debemos ingresar por lo menos una letra minúscula para el campo final:



Haciendo esta inserción ahora sí, llegamos al último estado que es q7, y por ende entramos nuevamente en un estado de aceptación.



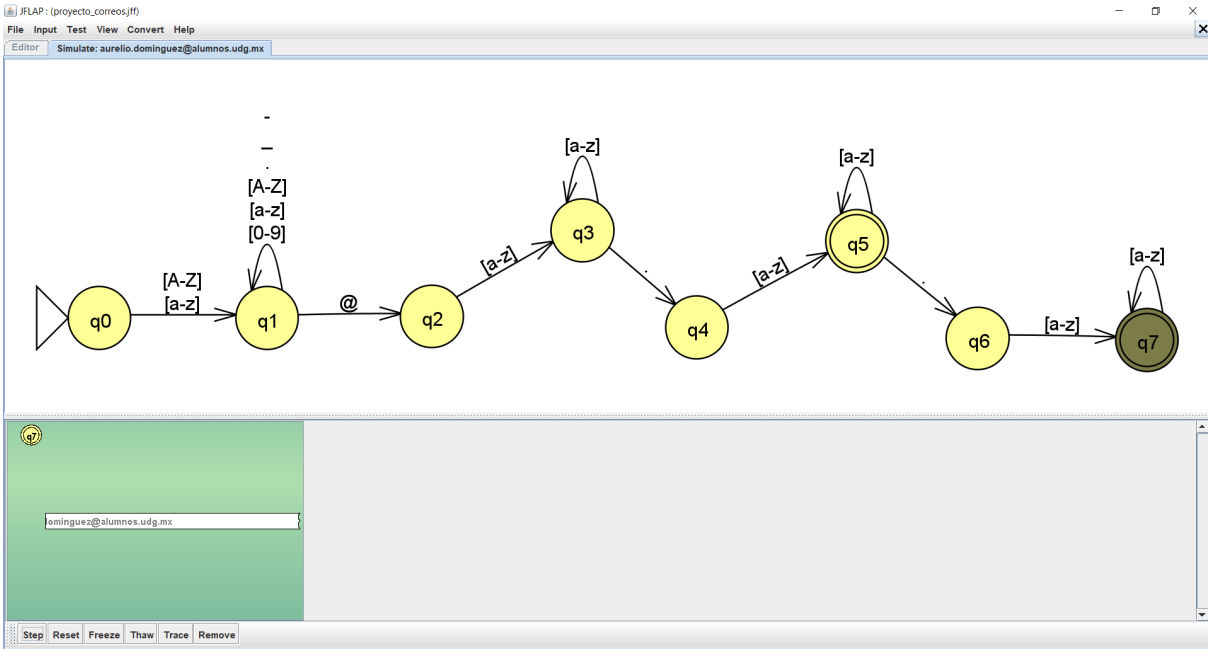
Este autómata está totalmente diseñado para poder permitir diferentes tipos de correos electrónicos, permite tener dominios como por ejemplo .com, .net, .udg, etcétera, siempre y cuando toda la dirección de correo anterior y posterior esté bien escrita y cumpla con todos los requisitos. Más adelante en las pruebas y resultados demostrare que no es válido ingresar espacios en blanco, caracteres especiales no reconocidos o inclusive no ingresar caracteres permitidos obligatorios.

Hemos reconocido lo sencillo que es diseñar un autómata finito en relación a una expresión regular, se puede adaptar a otros usos como asignación de matrículas para autos, códigos de estudiantes, CURP, nombres válidos en sitios web, etcétera.

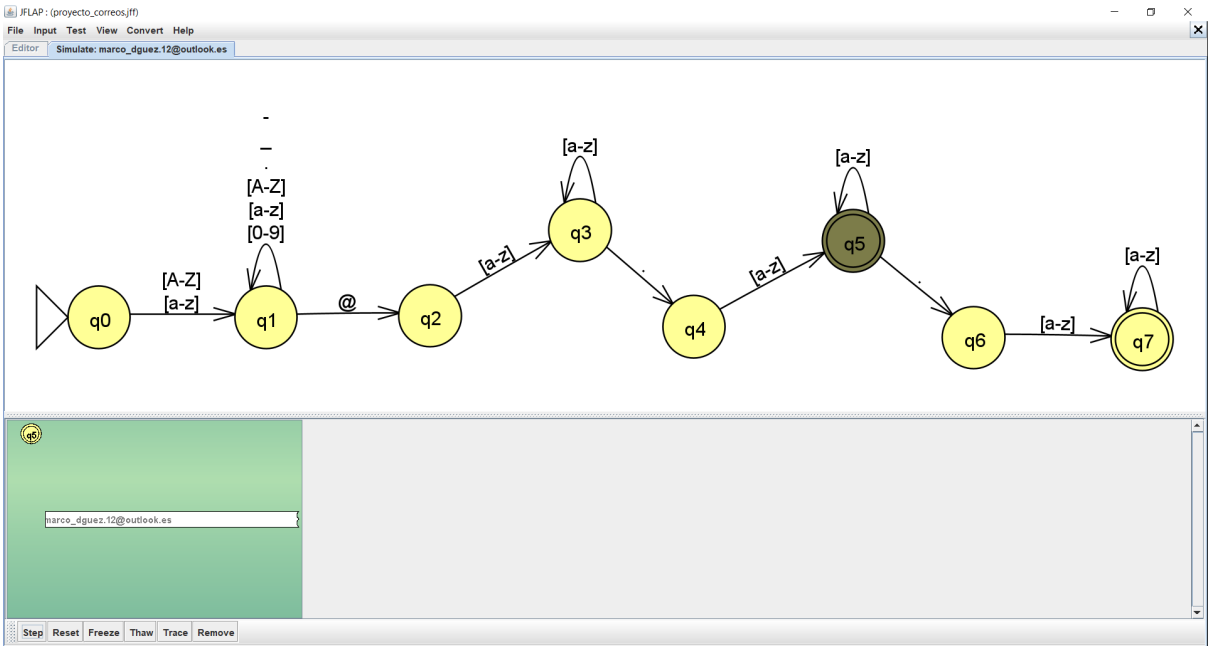
Usar el modelo del autómata finito determinista (AFD) tiene ventajas sobre otros modelos de máquinas y autómatas, el principal es la simplicidad de sus estados y las transiciones. Solamente hicimos uso de siete estados de transición, y en su mayoría tiene lazos para poder ingresar suficientes caracteres que sean necesarios. Además, con el uso de las expresiones regulares en los autómatas finitos podemos recrear modelos para la coincidencia y manipulación de series.

Pruebas y resultados:

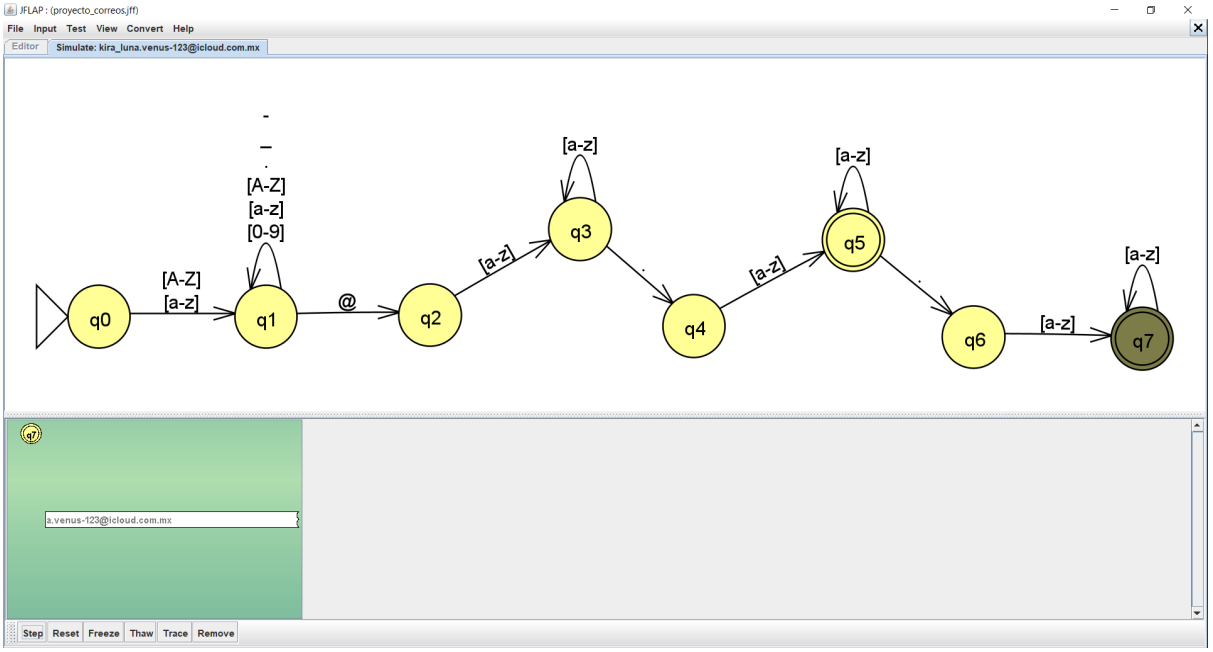
Correos Electrónicos Aceptados:



Este ejemplo es una simulación al correo institucional de uno de los miembros de este equipo, el cual exitosamente, es aceptado en el estado q_7.

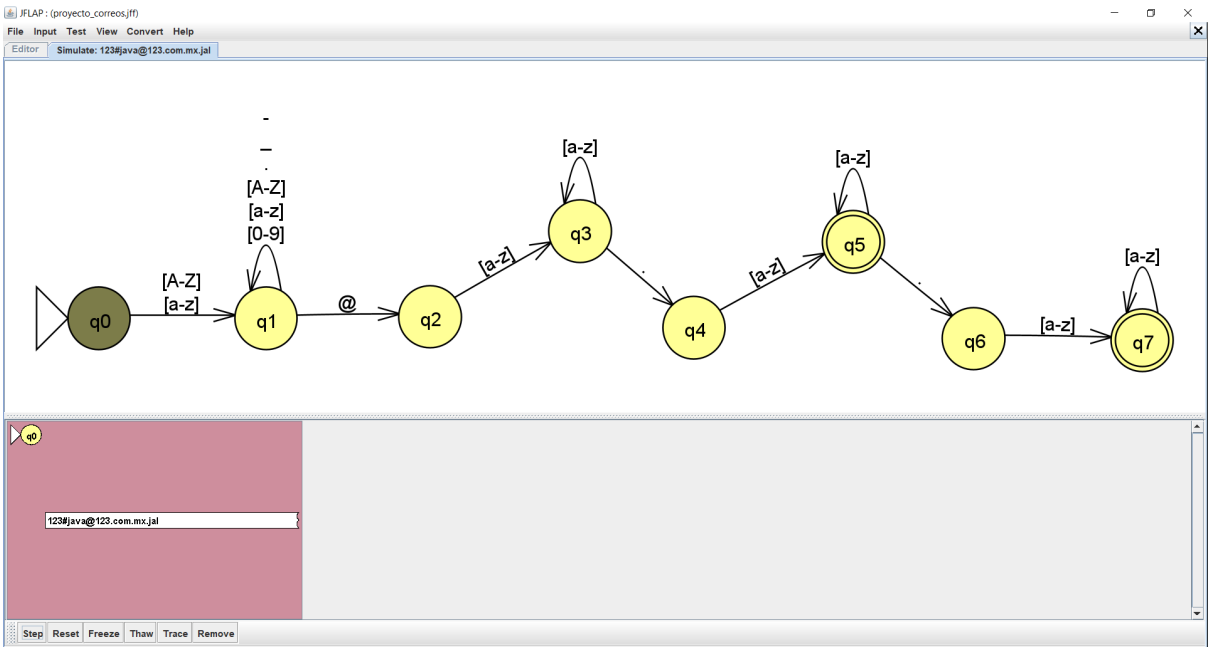


Este ejemplo es un correo electrónico común que la mayoría de personas suele utilizar, el cual es aceptado en el estado q_5.

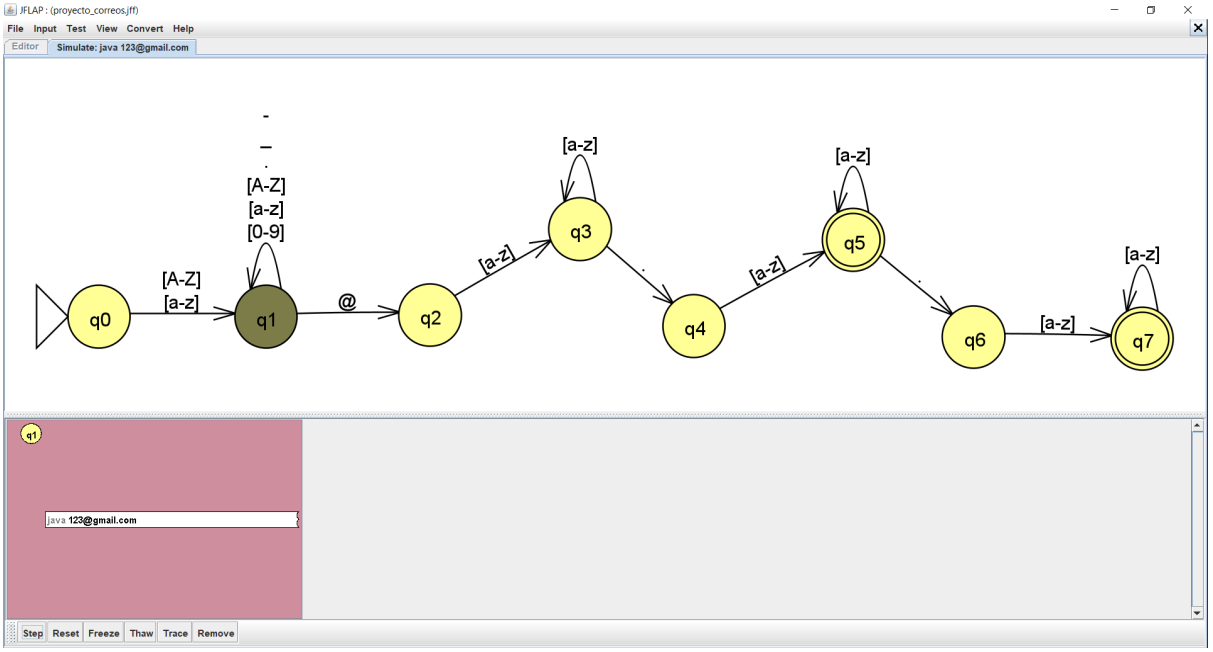


Este ejemplo incluye todos los tipos de datos válidos para la aceptación de un correo electrónico en el campo del nombre del usuario, además que tiene extensiones al final donde se indica el tipo de empresa y el país.

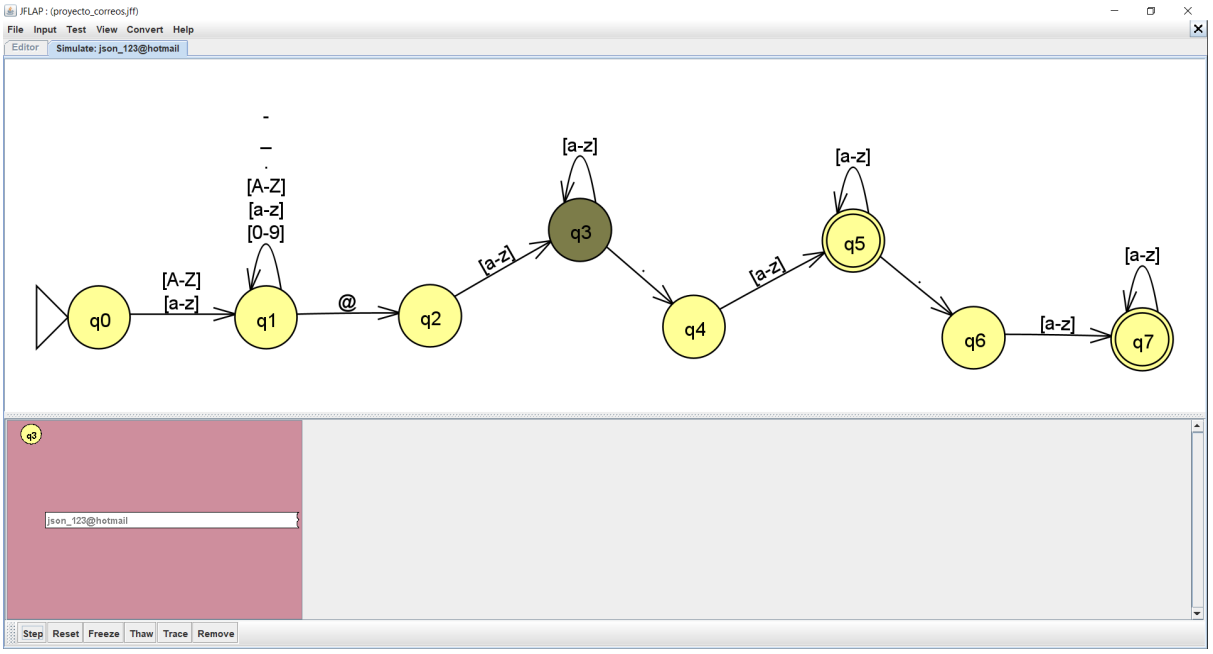
Correos electrónicos no aceptados:



Este ejemplo no es aceptado debido a que un correo electrónico debe iniciar con una letra y no un número.

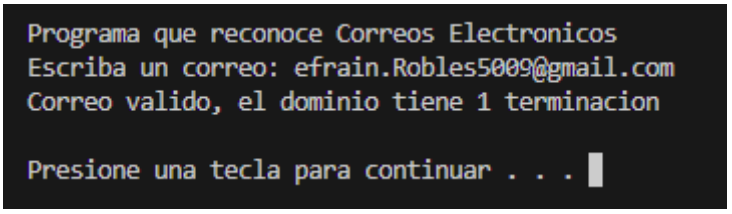


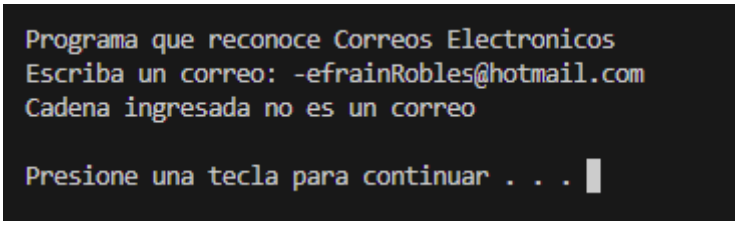
Este ejemplo contiene un error al momento de ingresar un espacio en blanco, ya que este es un símbolo no válido para la cadena.

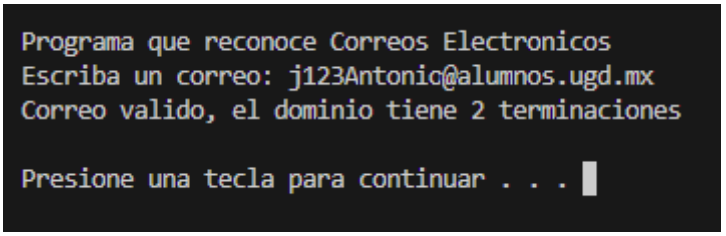


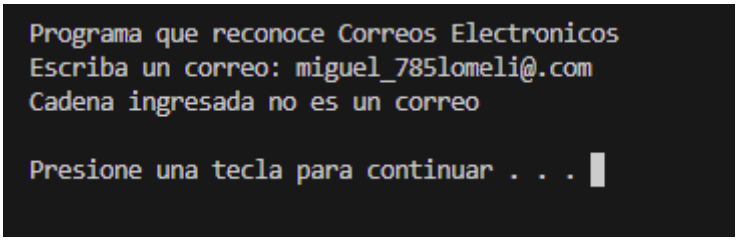
Este ejemplo no es aceptado debido que aunque contiene los primeros cambios de forma correcta, no se tiene por lo menos un punto para indicar el tipo de negocio que es o el país del cual proviene, es por ello que también se rechaza esta cadena.

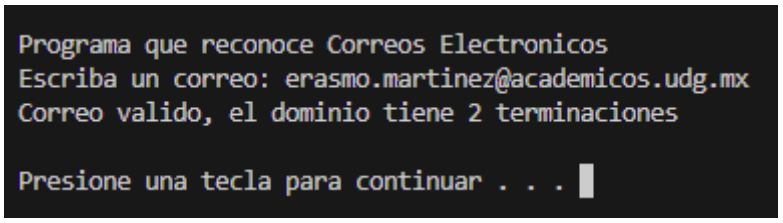
Ejecucion de codigo

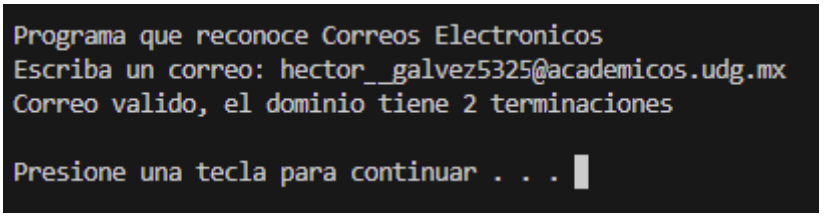












Conclusión:

En este proyecto, nuestro equipo se propuso detallar de manera secuencial los diversos cambios de estado que pueden ocurrir en un autómata finito al recibir textos que deben estar correctamente escritos para formar una dirección de correo electrónico válida.
Es fundamental prestar atención a los aspectos lógicos de un autómata determinista que procesa caracteres, ya que es necesario comprender el contexto en el cual se validarán los textos ingresados. Por ejemplo, si estuviéramos verificando la entrada de un número telefónico, tendríamos que determinar si dicho número pertenece a un país en particular, lo cual implica validar que el código inicial corresponda al país y que los dígitos subsiguientes cumplan con una cantidad exacta para ser aceptados.
Por consiguiente, es importante resaltar que el contexto del problema tiene un impacto directo en la obtención de una solución concreta. El hecho de comprender si los problemas cotidianos pueden ser abordados mediante un proceso computacional nos permite adoptar una perspectiva más lógica y racional. Nuestra propuesta consistio en idear una solución que reproduzca el desarrollo del problema, avanzando hacia la etapa de implementar sistemas formales y máquinas abstractas con un mecanismo preciso y eficiente. Esto nos permite plantear soluciones ideales a través de una máquina de Turing, un autómata finito o, posiblemente, un autómata de pila.

Bibliografía:

Gómez Andrade, Abelardo. (2014). *INTRODUCCIÓN A LA TEORÍA DE AUTÓMATAS Y LENGUAJES FORMALES*. México: TRAUCO EDITORIAL

Hopcroft J. , Motwani R. , & Ullman J. . (2007). Introducción a la teoría de autómatas, lenguajes y computación. Madrid: PEARSON EDUCACIÓN S.A.

J. Vázquez, L. Constable, W. Jornet, B. Meloni, “Enseñanzas de la Implementación de un Analizador Léxico”, CONAIISI 2015, Buenos Aires, Argentina, 2015