UNIVERSIDAD AUTÓNOMA DE CHIAPAS
FACULTAD DE CONTADURIA Y ADMINISTRACION
CAMPUS - I

LICENCIATURA EN INGENIERÍA EN DESARROLLO Y TECNOLOGÍAS DE SOFTWARE

ASIGNATURA:

COMPILADORES

DOCENTE:

DR. LUIS GUTIERREZ ALFARO

ACTIVIDAD 1:

INVESTIGACIÓN Y EJEMPLOS

ELABORADO POR:

JUAN ANTONIO MENDEZ MARROQUIN

6 - N 26/01/2024

TUXTLA GUTIERREZ, CHIAPAS

Definir los siguientes Conceptos y ejemplos de cada uno de los Inicios de I, II, III.

Definir el concepto de expresión regular

En el amplio mundo de la programación, las expresiones regulares son una herramienta poderosa y muy versátil que permiten buscar y manipular patrones de texto de manera muy eficiente. Seguramente hayas escuchado hablar de ello y no tengas muy claro en qué consisten exactamente. Para que te hagas una idea, la expresión regular es algo esencial para cualquier programador o desarrollador que trabaje con manipulación de texto. Esta también se conoce como regex o regexp, y es una secuencia de caracteres que define un patrón de búsqueda. Además, estos patrones pueden ser utilizados para realizar tareas como la validación de datos, la búsqueda y reemplazo de texto, y la extracción de información específica de un conjunto de datos.

Tipos de expresiones regulares

Existen varios tipos de expresiones regulares que se adaptan a diferentes necesidades:

1. **Coincidencia Literal**

   La coincidencia literal es el tipo más básico de expresión regular. Se compone de caracteres literales que deben coincidir exactamente con la cadena de texto. Por ejemplo, la expresión regular "hola" coincidiría solo con la cadena "hola".

2. **Caracteres Especiales**

   Los caracteres especiales son metacaracteres que tienen significados especiales en las expresiones regulares. Algunos ejemplos comunes son:

   - . (punto): Este símbolo coincide con cualquier carácter excepto el salto de línea.
   - * (asterisco): En este caso, coincide con cero o más repeticiones del carácter anterior.
   - + (más): Coincide con una o más repeticiones del carácter anterior.
   - ? (signo de interrogación): Coincide con cero o una repetición del carácter anterior.

- [] (corchetes): Define un conjunto de caracteres posibles para una posición en la cadena.
- \ (barra invertida): Escapa un carácter especial para tratarlo como literal.

3. **Grupos y Alternativas**

   Los grupos y las alternativas permiten agrupar y buscar patrones específicos. Los paréntesis () se utilizan para agrupar partes de la expresión regular. Por ejemplo, (abc)+ coincidiría con "abc", "abcabc", etc. El símbolo | se utiliza para alternativas. Por ejemplo, (hola|María) coincidiría con "hola" o "María".

4. **Cuantificadores**

   Los cuantificadores definen cuántas veces debe aparecer un carácter o un grupo en la cadena. Algunos ejemplos son:
   - {n}: Coincide con exactamente n repeticiones.
   - {n, m}: Coincide con al menos n y como máximo m repeticiones.

I.- Explicar los tipos de operadores de expresiones regulares

.
Los operadores de expresiones regulares son símbolos o caracteres especiales que modifican el significado o el comportamiento de una expresión regular. Existen diferentes tipos de operadores, como los siguientes:

- **Operadores de concatenación**: son los que unen dos o más subexpresiones para formar una expresión más compleja. **Por ejemplo, la expresión regular `ab` significa que se busca la letra `a` seguida de la letra `b`.**

- **Operadores de alternancia:** son los que permiten elegir entre varias opciones o posibilidades. Se representan con el símbolo `|`. **Por ejemplo, la expresión regular `a|b` significa que se busca la letra `a` o la letra `b`.**

- **Operadores de repetición:** son los que indican cuántas veces se debe repetir una subexpresión. Se representan con los símbolos `*`, `+`, `?` y `{}`. **Por ejemplo, la expresión regular `a*` significa que se busca cero o más veces la letra `a`.**

- **Operadores de agrupamiento:** son los que agrupan subexpresiones para aplicarles a otros operadores o para capturar su valor. Se representan con los símbolos `()` y `[]`. **Por ejemplo, la expresión regular `(ab)*` significa que se busca cero o más veces la secuencia `ab`.**

- **Operadores de posición:** son los que indican dónde debe aparecer una subexpresión dentro de una cadena de texto. Se representan con los símbolos `^`, `$`, `\b` y `\B`. **Por ejemplo, la expresión regular `^a` significa que se busca la letra `a` al inicio de la cadena.**

II.- Explicar el proceso de conversión de DFA a expresiones regulares.

El proceso de conversión de un autómata finito determinista (DFA) a una expresión regular se basa en el algoritmo de eliminación de estados. Este algoritmo consiste en eliminar sucesivamente los estados del DFA, excepto el inicial y el final, y sustituir las transiciones que pasan por ellos por expresiones regulares que las equivalen. El resultado final es una expresión regular que acepta el mismo lenguaje que el DFA original.

Consideremos el siguiente DFA:

1. **DFA**
   - Estados: Q = {q0, q1}
   - Alfabeto: Σ = {0, 1}
   - Estado inicial: q0
   - Estados de aceptación: {q1}
   - Transiciones:
     - δ(q0, 0) = q1
     - δ(q0, 1) = q0
     - δ(q1, 0) = q0
     - (q1, 1) = q1

2. **Paso 1: Eliminar estados intermedios:**
   - No hay estados intermedios para eliminar en este caso.

3. **Paso 2: Expresar transiciones como expresiones regulares:**
   - Expresamos cada transición como una expresión regular.

     - De q0 a q1 con 0: 0
     - De q0 a q0 con 1: 1
     - De q1 a q0 con 0: 0
     - De q1 a q1 con 1: 1

4. **Expresión Regular Final:**
   - Combinamos todas las expresiones regulares usando la unión (|)
     - 0 + 1(0 + 1)*
   - La expresión regular final para el lenguaje aceptado por el DFA es "0 + 1(0 + 1)*".

Este es un ejemplo más sencillo donde hemos convertido un DFA con dos estados a la expresión regular "0 + 1(0 + 1)*".

III.- Las expresiones regulares son una forma de representar patrones de texto mediante símbolos y operaciones.

Las expresiones regulares son patrones utilizados para encontrar una determinada combinación de caracteres dentro de una cadena de texto. Se componen de símbolos y operadores que permiten definir estos patrones de búsqueda de manera flexible y precisa.

Algunas de estas leyes son:

- **Ley de idempotencia:** A + A = A. Esto significa que la unión de una expresión regular consigo misma es igual a la expresión original.

- **Ley de asociatividad:** (A + B) + C = A + (B + C) y (AB)C = A(BC). Esto significa que el orden en que se agrupan las expresiones regulares no afecta al resultado final.

- **Ley de conmutatividad**: A + B = B + A. Esto significa que el orden en que se escriben las expresiones regulares no afecta al resultado final.

- **Ley de distributividad:** A(B + C) = AB + AC y (A + B)C = AC + BC. Esto significa que se puede aplicar la operación de concatenación a cada término de una unión, y viceversa.

- **Ley de absorción**: A + AB = A y A(A + B) = A. Esto significa que se puede eliminar una expresión regular que contenga a otra como subexpresión.

Expresión Regular para Direcciones de Correo Electrónico:

- **Expresión Regular:** ^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$
- Significado: Representa direcciones de correo electrónico.
    - ^: Inicio de la cadena.
    - [a-zA-Z0-9._%+-]+: Usuario con letras, números, y algunos caracteres especiales.
    - @: Símbolo de arroba.
    - [a-zA-Z0-9.-]+: Nombre del dominio.
    - \.: Punto que separa el nombre de dominio de la extensión.
    - [a-zA-Z]{2,}: Extensión del dominio con al menos dos caracteres alfabéticos.
    - $: Fin de la cadena.

BIBLIOGRAFÍA

- de Formación Profesional Superior, E. C. (2023, agosto 29). ¿Qué es una expresión regular y qué tipos existen? Esic.edu; ESIC. https://www.esic.edu/rethink/tecnologia/que-es-una-expresion-regular-que-tipos-existen-c
- DFA A Expresion Regular. (s/f). Scribd. Recuperado el 27 de enero de 2024, de https://es.scribd.com/doc/12929632/DFA-a-Expresion-Regular
- IBM Documentation. (2021, febrero 28). Ibm.com. https://www.ibm.com/docs/es/iis/11.5?topic=checks-matches-regular-expression-check
- Lenguaje de expresiones regulares - Referencia rápida. (s/f). Microsoft.com. Recuperado el 27 de enero de 2024, de https://learn.microsoft.com/es-es/dotnet/standard/base-types/regular-expression-language-quick-reference
- (S/f). Researchgate.net. Recuperado el 27 de enero de 2024, de https://www.researchgate.net/figure/Figura-123Proceso-de-conversion-de-AFD-a-expresion-regular_fig4_324114613