EDUCACIÓN
SECRETARÍA DE EDUCACIÓN

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLOGICO DE PACHUCA

Instituto Tecnológico de Pachuca

NOMBRE DE LA CARRERA:

INGENIERÍA EN SISTEMAS COMPUTACIONALES

GRUPO: B

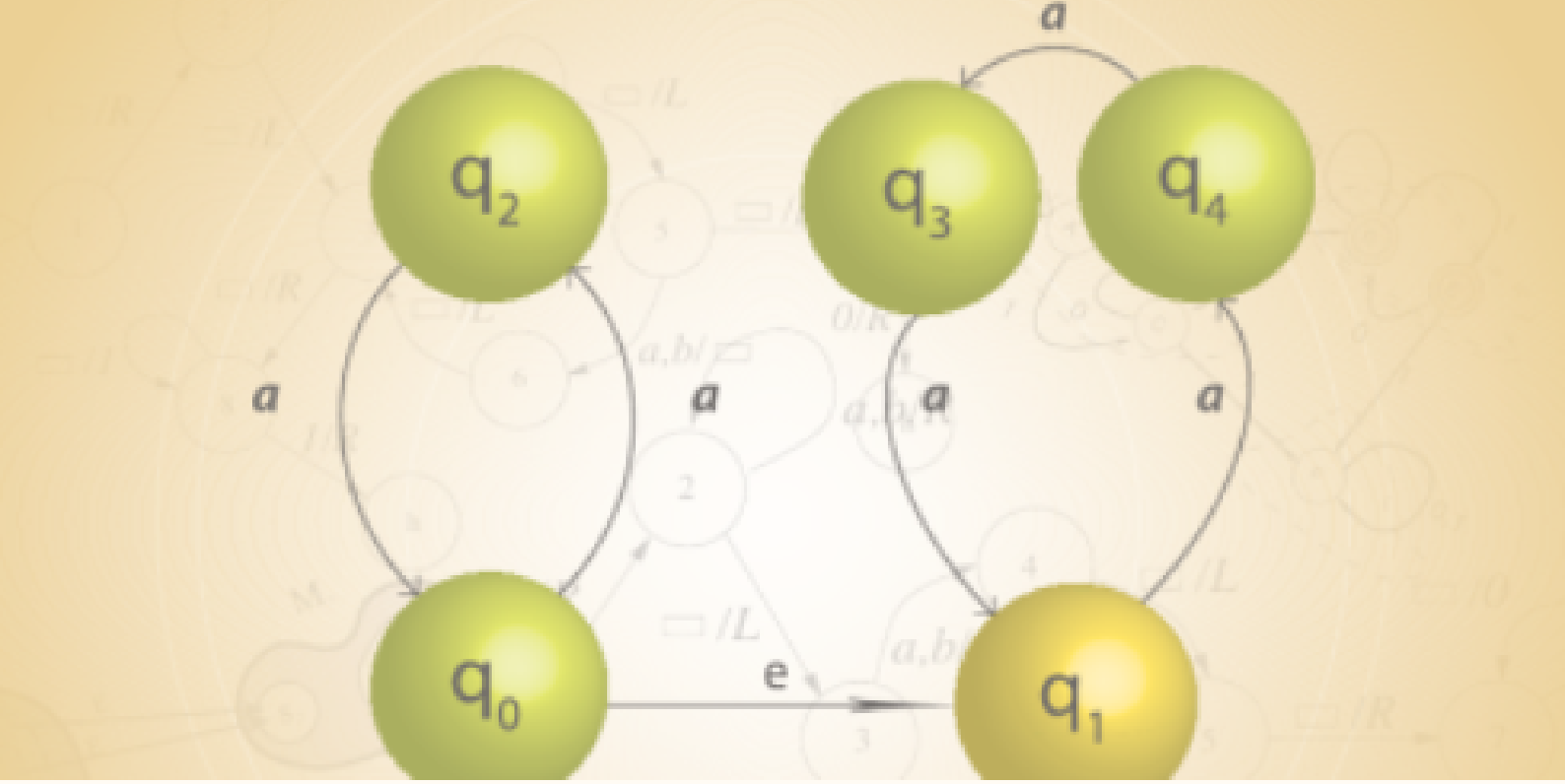
DOCUMENTACION

INTEGRANTES:

KEVIN HANS JUÁREZ PÉREZ
EMANUEL TOLENTINO SANTANDER
MARLY GALARZA ACOSTA
CITLALI MARTINEZ SANCHEZ

Mayo del 2024

PERIODO: ENERO - JUNIO 2024



INTRODUCCIÓN

El proceso de escribir las reglas de análisis es fundamental en el desarrollo de un analizador léxico eficiente. Antes de comenzar, es imprescindible tener una tabla de tokens bien definida, que enumere todos los elementos que el analizador debe reconocer en el texto de entrada. Estos tokens pueden abarcar desde palabras clave y operadores hasta símbolos especiales y números.
Estas reglas establecen cómo se identifican los tokens en el texto de entrada, y suelen estar expresadas mediante expresiones regulares o patrones de coincidencia proporcionados por la herramienta o librería seleccionada.

Es crucial también considerar el manejo de casos especiales o ambigüedades en las reglas de análisis. Por ejemplo, cuando un mismo patrón puede corresponder a múltiples tokens, es necesario definir reglas que especifiquen cómo resolver estas ambigüedades para garantizar la precisión y coherencia del analizador léxico.
Todo esto ya mencionado lo podremos ver más detallado en el siguiente documento.

Tabla de operadores

	VALOR	DESCRIPCIÓN
+	Suma	Suma dos o más operandos
-	Resta	Resta dos o más operandos
*	Multiplicación	Multiplica dos o más operandos
/	División	Divide dos o más operandos
^	Potencia	Potencia de un numero o variable
%	Porcentaje	Porcentaje de un numero o variable
||	Uno u otro	Para condicionar entre uno u otro
!	Diferente de	Para condicionar que los valores sean diferentes
=	Igual	Para condicionar que los valores sean iguales

	VALOR	DESCRIPCIÓN
>	Mayor que	Para condicionar que un valor sea mayo o igual que el otro
>=	Mayor o Igual	Para condicionar que un operador sea igual o mayor que el otro
<	Menor que	Para condicionar que un valor sea menor que el otro
<=	Menor o Igual	Para condicionar que un valor sea menor o igual que el otro
&	Y también	Para condicionar que un valor cumpla con otro criterio

Tabla de palabras clave

	VALOR	DESCRIPCIÓN
MOD	Método	Palabra reservada para declarar un método
OPERA	Librería	Palabra reservada para declarar una operación
IN	Importar	Palabra reservada para importar librerías o extensiones en el lenguaje
IMP	Imprimir	Palabra reservada para imprimir cadenas o caracteres
FOR	Ciclo For	Palabra reservada para inicializar el ciclo for
IF	Condicional	Palabra reservada para inicializar el condicional if
ELSE	Condicional con dos casos o mas	Palabra reservada utilizada para declarar otro condicional

	VALOR	DESCRIPCIÓN
WHILE	Ciclo while	Palabra reservada para declarar un ciclo while
SWITCH	Condicional	Palabra reservada para declarar un condicional switch
DO	Ciclo con condicional	Palabra reservada para declarar un Do while, el cual es un ciclo, pero con condicional
CASE	Condicional switch con dos casos o mas	Palabra reservada para declarar otro condicional en un switch
PI	Valor de pi	Palabra reservada para el valor de pi (3.1416)
RAD	Radianes	Palabra reservada para convertir grados a radianes
SEN	Seno	Palabra reservada para ángulo seno
COS	Coseno	Palabra reservada para el ángulo coseno
TAN	Tangente	Palabra reservada para el ángulo tangente

Tabla de delimitadores

	VALOR	DESCRIPCIÓN
()	Agrupar sentencias	Delimitador para dígitos
{ }	Encapsular parte de código	Delimitador para sentencias de código
[ ]	Determinar valores	Delimitador para valores dentro de los corchetes

Reglas de análisis

Importar librerías

Para importar una librería será necesario colocar la palabra reservada “in” y colocar punto y coma “;” al final de la librería para que esta sea validada

Ejemplo:

```
In Opera;
```

Operaciones trigonométricas

Para realizar operaciones trigonométricas como seno, coseno, tangente y radianes será necesario colocar paréntesis ( ) después de la palabra reservada correspondiente, para que el programa pueda realizar la operación de la misma

Ejemplo:

```
Sen ( )
Cos ( )
Tan ( )
Rad ( )
```

Imprimir

Para imprimir un carácter basta con colocar la palabra reservada “imp”, pero en el caso de que se desee imprimir una cadena, esta deberá ser escrita entre comillas y con punto y coma al final de esta en ambos casos.

Ejemplo:

```
Imp X;
Imp “Hola Mundo”;
```

Sentencia if

Para la sentencia if, será necesario colocar la condición entre comillas y será necesario utilizar llaves “{ }” para colocar el código correspondiente a la condición.

Ejemplo:

```
If “x > y“{
Imp “X es mayor que Y”;
}
```

Reglas de análisis

Ciclo for

Para la sentencia for será necesario colocar los argumentos correspondientes entre comillas seguido de las llaves “{ }” para colocar el código correspondiente.

Ejemplo:

```
For “i=0, i=10, i++”{
Imp “Numero: “+i;
}
```

Métodos

Para declarar un método usamos la palabra reservada “Mod”, seguido de su nombre que se le asigne con llaves “{ }” para colocar el código correspondiente.

Ejemplo:

```
Mod Suma{

}
```

Declarar Variables

Para declarar variables será necesario colocar el símbolo de asignación “=” con punto y coma “;” al final de la asignación, no existe ninguna palabra reservada para declarar variables, pero debe cumplir con esta estructura para que sea validada.

Ejemplo:

```
X = 23;
```

Pi

Para usar el valor de pi basta con usar la palabra reservada “PI” esto debido a que el lenguaje esta centrado en realizar operaciones aritméticas complejas, si se declara la variable pi, el lenguaje lo determinara como error ya que es una palabra propia del lenguaje.

Ejemplo:

```
4 * Pi
```

Reglas de análisis

Switch

Switch es un condicional muy similar al if, solo cambiamos su estructura y palabra reservada. Para hacer uso de este basta con colar la palabra reservada “Switch”, en seguida se coloca el condicional entre comillas y paréntesis para colocar el código correspondiente.

Ejemplo:

```
Switch “X < Y”(
Imp “X es menor”;
)
```

Asi como el if, switch puede tener mas condicionales, en este caso se usa la palabra reservada “Case” y se realiza exactamente lo mismo.

Ejemplo:

```
Switch “X < Y”(
Imp “X es menor”;
) case (
Imp “Y es menor”;
)
```

Expresiones regulares

01

Switch "X < Y"(Imp "X es menor";)

^(\d+(\.\d+)?)(?:\s*<\s*)(\d+(\.\d+)?)$

02

If "x > y"{Imp "X es mayor que Y";}

^(\d+(\.\d+)?)(?:\s*>\s*)(\d+(\.\d+)?)$

03

Imp X;

^\d{1,3}(\.\d{1,3})?$

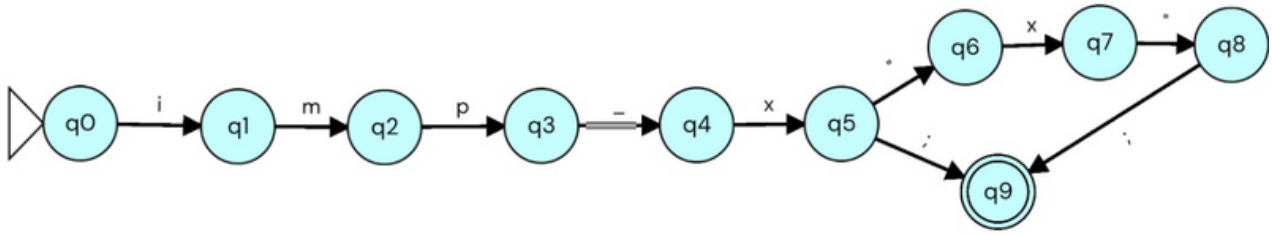
04

Imp "Hola Mundo";

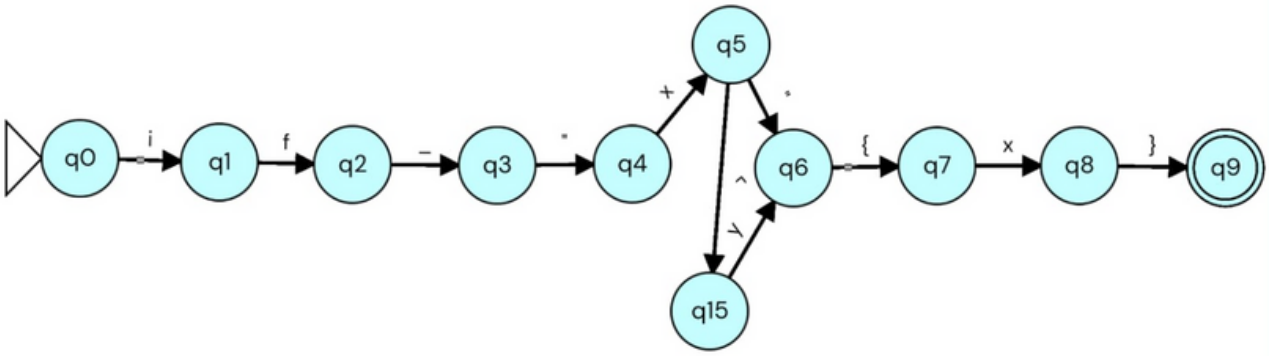
^Hola Mundo$

Autómatas finitos

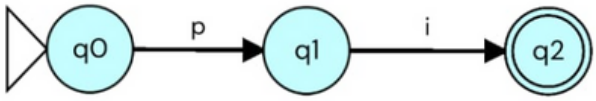
Imprimir



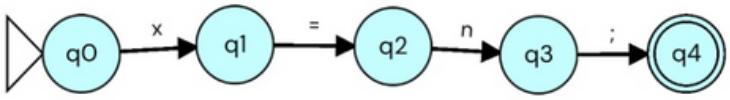
Sentencia if



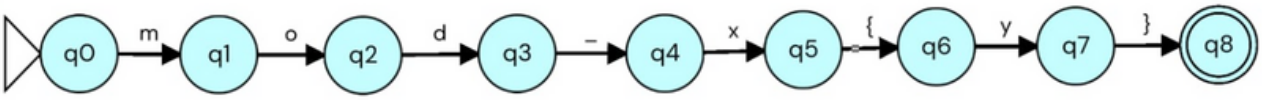
Pi

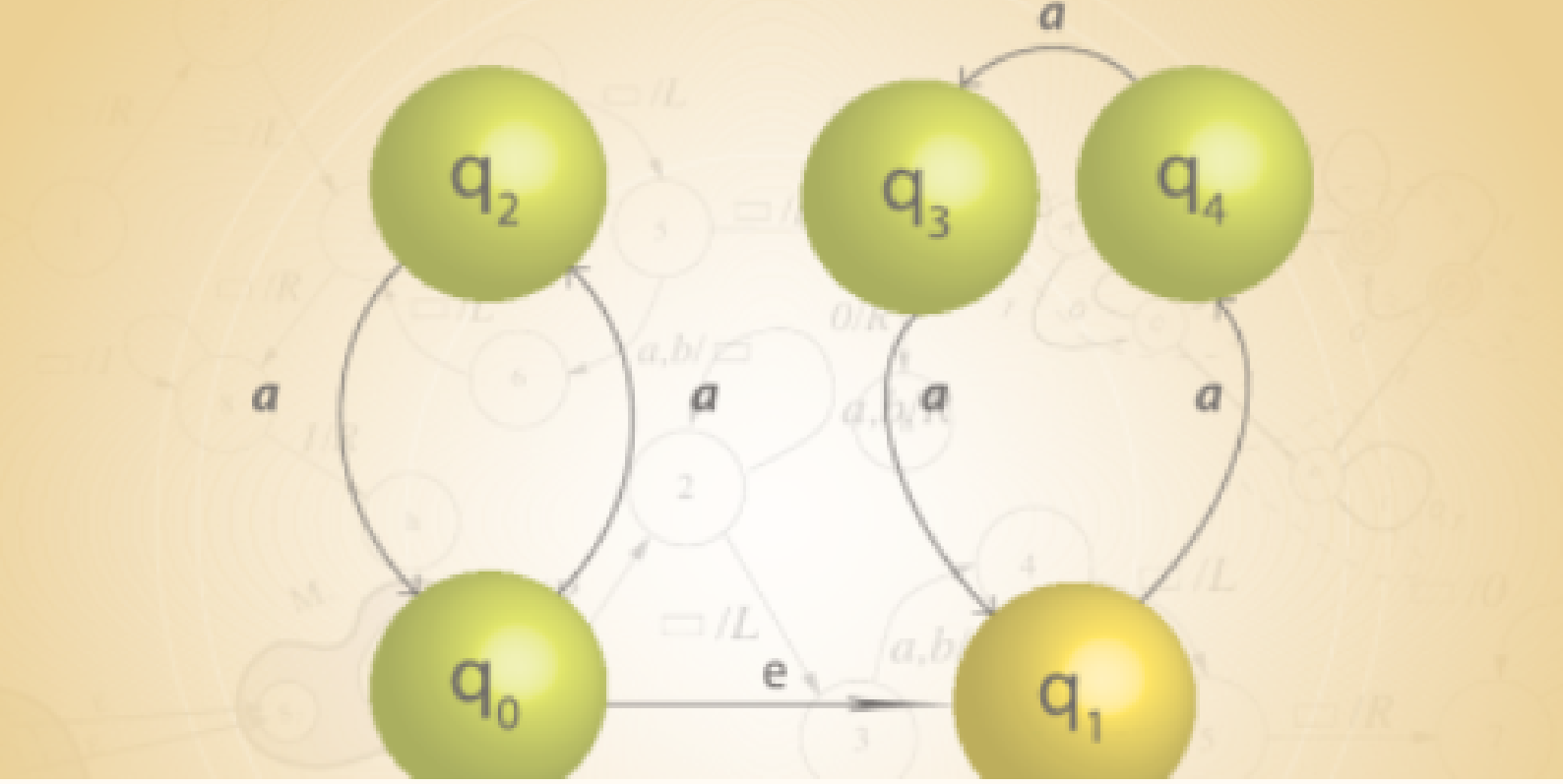


Declarar Variables



Métodos





CONCLUSIÓN

Las reglas de coincidencia, junto con la asociación de acciones y el manejo de casos especiales, son elementos esenciales en la construcción de analizadores léxicos eficientes y precisos. Al utilizar expresiones regulares o reglas de coincidencia de patrones, podemos definir cómo se reconocen los tokens en el texto de entrada, lo que permite una identificación precisa de los elementos del lenguaje.

Al escribir reglas de coincidencia, asociar acciones y manejar casos especiales, podemos construir analizadores léxicos que sean capaces de reconocer y procesar de manera efectiva el texto de entrada, contribuyendo así al desarrollo y la implementación exitosa de sistemas de procesamiento de lenguaje natural y compiladores.