Fase 2.1 del Proyecto de asignatura

Especificación formal de los tokens de un nuevo lenguaje de programación

**Semestre académico:** II semestre académico de 2020

**Profesor:** Claudia Elena Quiceno Restrepo.

## Objetivo

*Especificar los tokens de un lenguaje de programación creado por los estudiantes, mediante expresiones regulares y autómatas finitos deterministas*

## Instrucciones:

Antes de iniciar la actividad, se sugiere:

1. Lea este documento completo antes de iniciar.
2. Tenga a la mano la tabla ASCII
3. Puede realizar la actividad hasta dos personas como máximo.
4. Emplee su creatividad y conocimientos para desarrollar esta primera parte de la fase 2. Sin embargo, tenga en cuenta que el producto que usted desarrolle, debe ser muy **intuitivo** **y nemotécnico** para que su lenguaje pueda ser usado con facilidad en el siguiente semestre.
5. Para desarrollar la actividad, puede completar y desarrollar las actividades del siguiente formato o puede realizar la actividad en el cuaderno y pegar las fotos dejando la evidencia en la sesión de Desarrollo (puede hacer uso del JFLAB para los Autómatas, pero deben ser pegados como imagen).
6. **Entrega:** 
   1. Debe subirlo al classroom.
   2. Si usted trabajo con otro compañero, tan solo uno de los dos estudiantes debe subirlo y debe especificar los nombres completos de los integrantes del grupo en este documento.
   3. La actividad tiene una duración de 2 horas. Por lo tanto, se publicará un poco antes de la hora de inicio de clase, para que usted pueda leerlo con antelación y se recibirá hasta 30 minutos después de pasada las 2 horas de iniciada la presente actividad.
   4. Si usted se pasa de este tiempo, la nota para la fase 2 rebajará 0,5.
   5. Sin embargo, por cada 2 horas de atraso, se rebajará una unidad (1.0) en la nota obtenida de la fase 2.
   6. Tenga en cuenta que esta actividad no es calificable, pero es necesaria, para realizar el desarrollo de la fase 2. Este documento se denominará el Contrato entre el profesor y el estudiante(s).

NOTA: Si se encuentran tokens iguales a otros ó trabajos similares, que se pueda ver como una copia o plagio (de otros trabajos de grupos, así sea en diferentes jornadas o con trabajos de semestres anteriores) el desarrollo de la segunda fase quedará anulada; para todos los grupos donde se encuentre la copia o plagio y su nota será de 0.0, debido a que los estudiantes se quedarán sin insumos para desarrollar esta fase.

## Convenciones

Para la especificación de los tokens del lenguaje, utilice las siguientes convenciones:

L: Letra mayúscula o minúscula.

M: Letra mayúscula

N: Letra minúscula

D: Dígito; 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

C.S: Cualquier símbolo excepto los delimitadores de cadena.

## Actividades para desarrollar:

**1)** Elija un nombre para el nuevo lenguaje de programación que usted va a diseñar. ¿Explique cuál es la razón de este nombre?

Trueno: Debido a que este lenguaje busca optimizar el tiempo de ejecución de los algoritmos.

**2)** Escriba la expresión regular para cada uno de los siguientes tokens del nuevo lenguaje:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Categoría | Tokens | Expresión Regular o pegue la foto de su trabajo al final | Ejemplos, ó lo que se espera que haga: | Ejemplo de las palabras de su lenguaje |
| Operadores: | Ejemplo:  Operadores aritméticos | **( + U – U \* U / )** | **+ - \* /** (mínimo los 4 básicos) | **+ - \* /** |
| Operadores relacionales | **(? U ¿ U .. U ¡)** | (Mínimo >,<, ==, !=) | ? ¿ .. ¡ |
| Operadores lógicos | **(Y U O U ,,)** | (3 operadores) | Y O ,, |
| Operador(es) de asignación | **(->)** | = += -= (los que necesite) | -> |
| Símbolos de abrir | **(\_ -)** | (los que necesite) | \_- |
| Símbolos de cerrar | **(- \_ )** | (los que necesite) | -\_ |
| Terminal o inicial | **($)** | (al menos 1, en java es ;) | $ |
| Separadores de sentencias |  | (si los requieren, en Java ,) |  |
| Valor de Asignación | Enteros | **(/ DD\* /)** | 9801 es un valor valido en JAVA | /534/ |
| Reales | **(% D , D\* %)** | 0.34 es un valor valido en JAVA | %1,22% |
| Cadenas de caracteres | **°( LL\* U DD\* U C.S C.S\* )°** | “ todo lo que este aquí” | °CadeNa**°** |
| Caracteres | **#(LL\* U DD\* U C.S)** | ‘a’ valido en JAVA | #h |
| Palabras reservadas:  Tipo de dato | Palabra para los enteros | **( \_ entero)** | reservar un tipo de dato int en java | \_entero -> /8\ |
| Palabra para los reales | **( \_ decimal)** | float ó doublé | \_decimal -> /8,2\ |
| Palabra para cadenas de caracteres | **(\_ cadena)** | String | \_cadena -> °CadeNa |
| Palabra para los caracteres | **(\_caracter)** | char | \_caracter -> #h |
| Palabras reservadas:  Diferentes usos | Palabra reservada para bucle | **&( i t e r a r )&** |  | &iterar& i -> 4 |
| Palabra para decisión, | **$(siEs)** |  | $siEs (4 -> 4) |
| Palabra para método principal. | **(Principal)** | main | Principal metodoX |
|  |  |  |  |
| Palabras reservadas:  Identificadores | Los nombres de variables | **(M L\* )** |  | **nameVariable** |
| Los nombres de método, | **(N L\* )** |  | **NameMethod** |
| Nombres de clase | **(^^ L\*)** | **(si lo requiere)** | **^^class** |

**3)** Elabore los diagramas de transición o los Autómata finito determinista para cada una de las categorías mencionadas con los tokens, (solo 1 por cada categoría, por lo tanto, son 5 AFD). Puede omitir los ‘sumideros de rechazo’. Esto por simplicidad, visibilidad y tiempo, ya que no prestan mayor utilidad en este tipo de proyectos.

## Ejemplos para desarrollar la actividad:

Todos los tokens excepto los operadores aritméticos, deben ser diferentes a los de Java. Además, deben ser de alguna manera nemotécnicos, por ejemplo:

* Un operador de asignación no podría ser = ni += (incorrecto: es igual a JAVA)
* Un operador de asignación no podría ser %%%&&& (incorrecto: no es nemotécnico)
* Un operador de asignación sí podría ser <<<<

A continuación, se muestra una instrucción de asignación en Java y la misma instrucción con el operador de asignación del ejemplo

|  |  |
| --- | --- |
| En Java | a **=** b + c; |
| Con el operador de asignación del ejemplo | a **<<<<** b + c |

Por ejemplo, supongamos que vamos a especificar los operadores lógicos, la solución podría quedar de la siguiente manera:

j) Operadores lógicos

$y$ $o$ $no$

A continuación, se muestra la primera línea de un if en lenguaje Java y cómo quedaría la línea utilizando el operador AND del ejemplo.

|  |  |
| --- | --- |
| En Java | if ( a > b && b > c) |
| Con el operador AND del ejemplo | if ( a > b $y$ b > c) |

NOTA: recuerde que las palabras en tlf no necesariamente debe tener significado en el español o en otro idioma, pueden ser símbolos, letras, y dígitos.

Solución de la Fase 2.1 del Proyecto de asignatura

Especificación formal de los tokens de un nuevo lenguaje de programación

**Semestre académico:** II semestre académico de 2020

**Presentado por:**

Nombre: Kevin Edilson Valencia Romero

Nombre: Jeffrey Alexander Vargas Gonzalez

**Fecha:** 20 / 04 / 2020

**Presentado al profesor:** Claudia Elena Quiceno Restrepo.

Resumen:

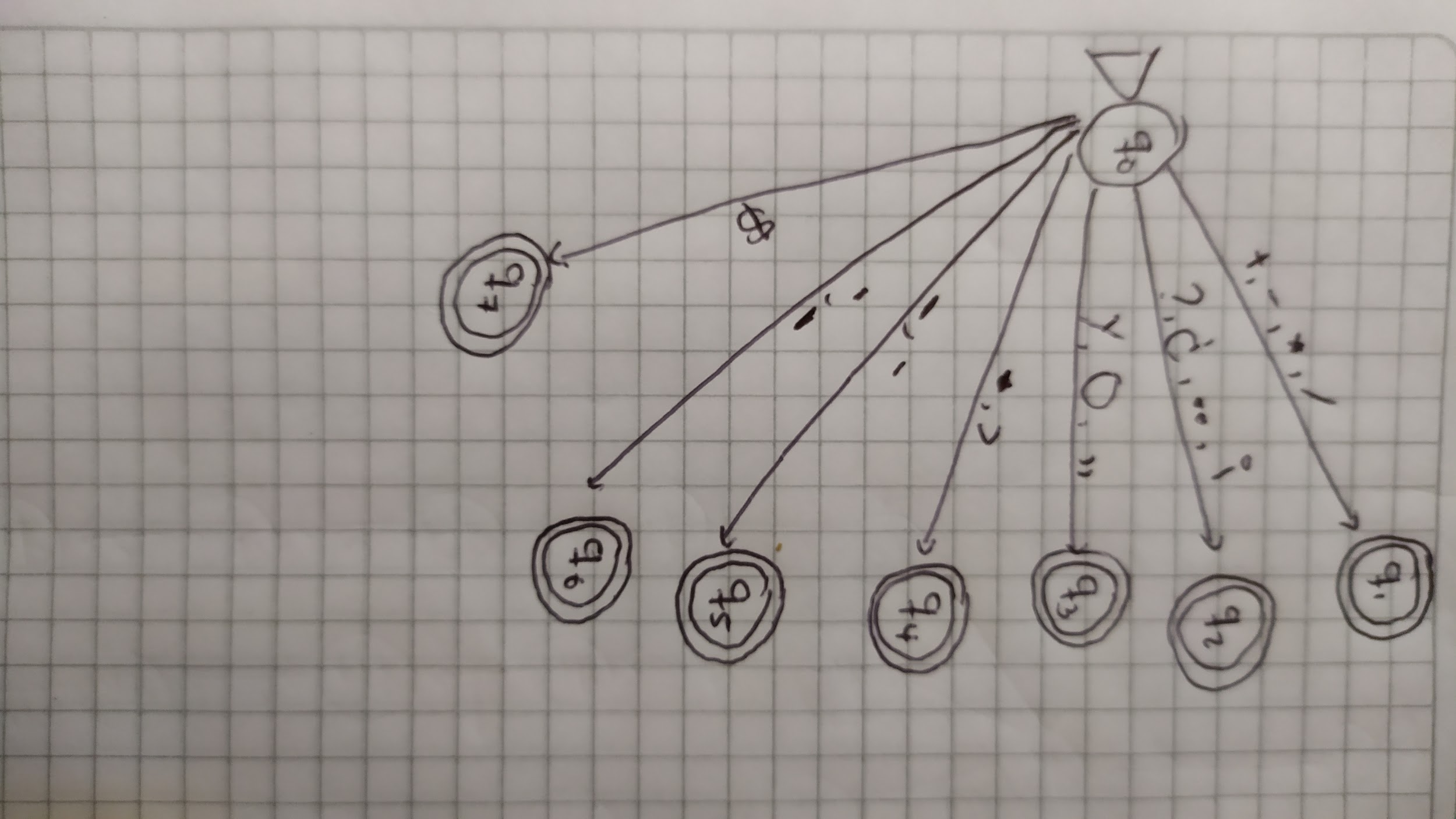
23 expresiones regulares y 5 AFD por categoría

***Desarrollo***

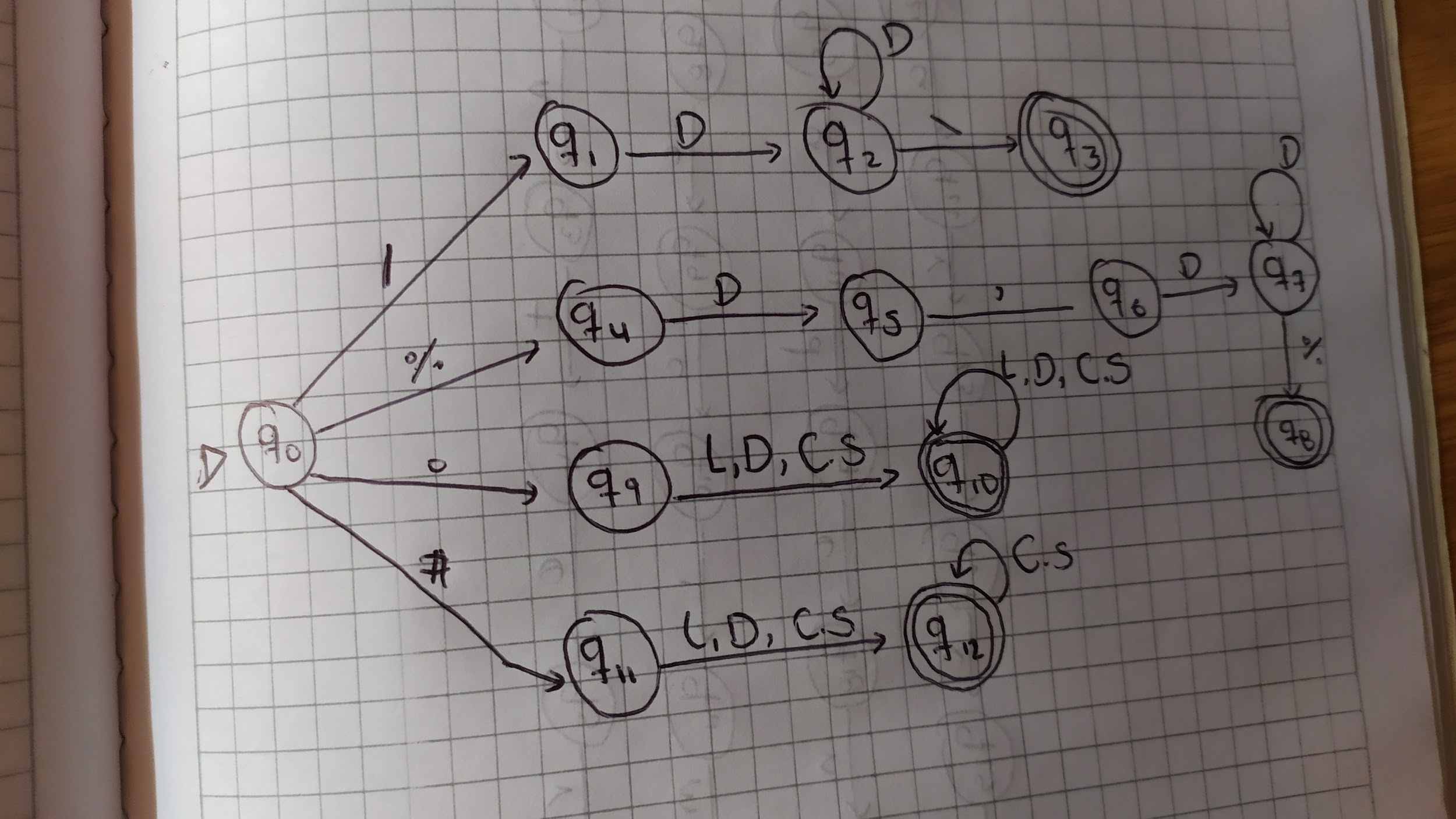
Desde aquí puede pegar sus evidencias de trabajo

1. Fotos nítidas. (que se pueda identificar lo que dice)

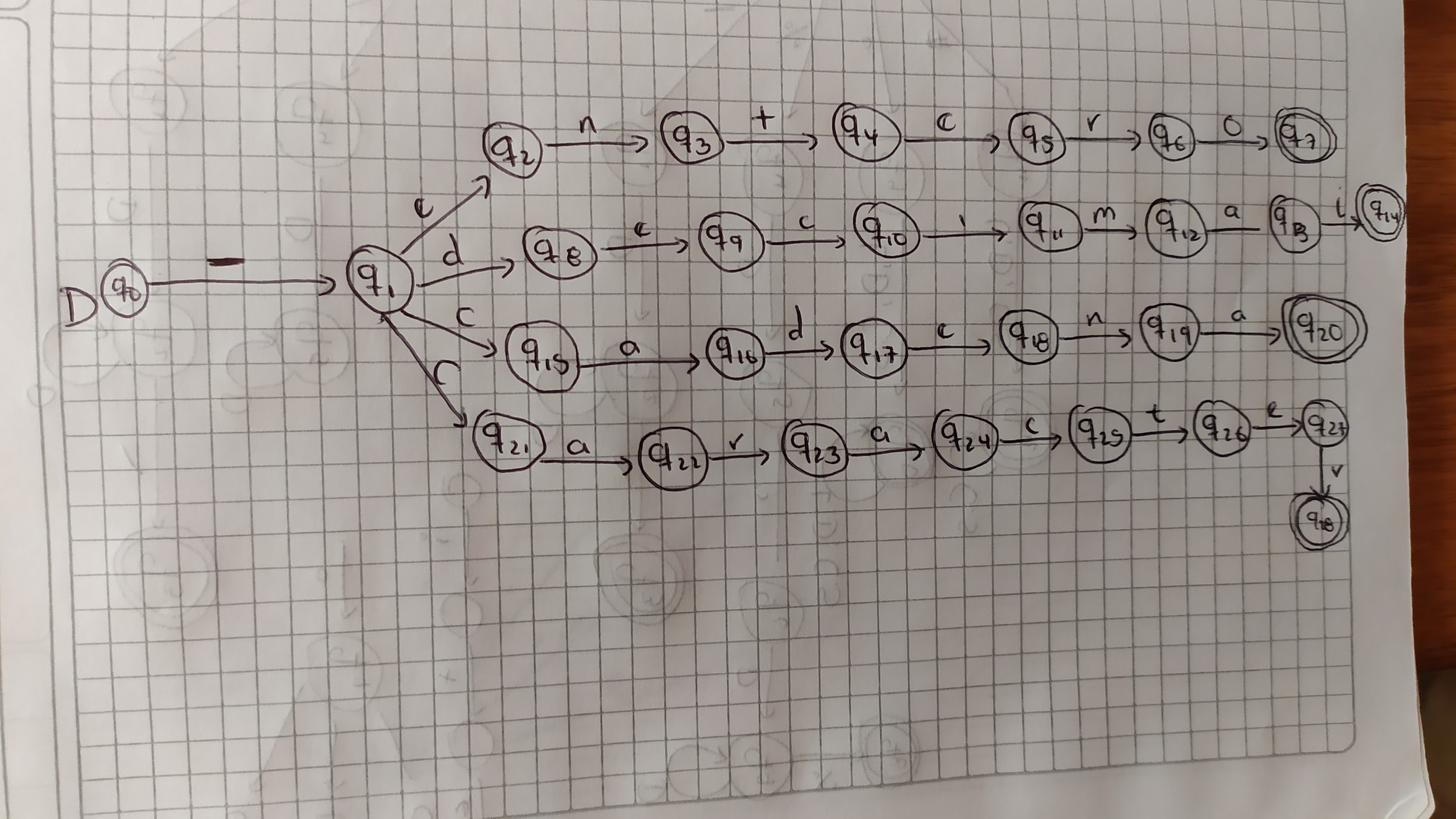
Categoría: **Operadores**



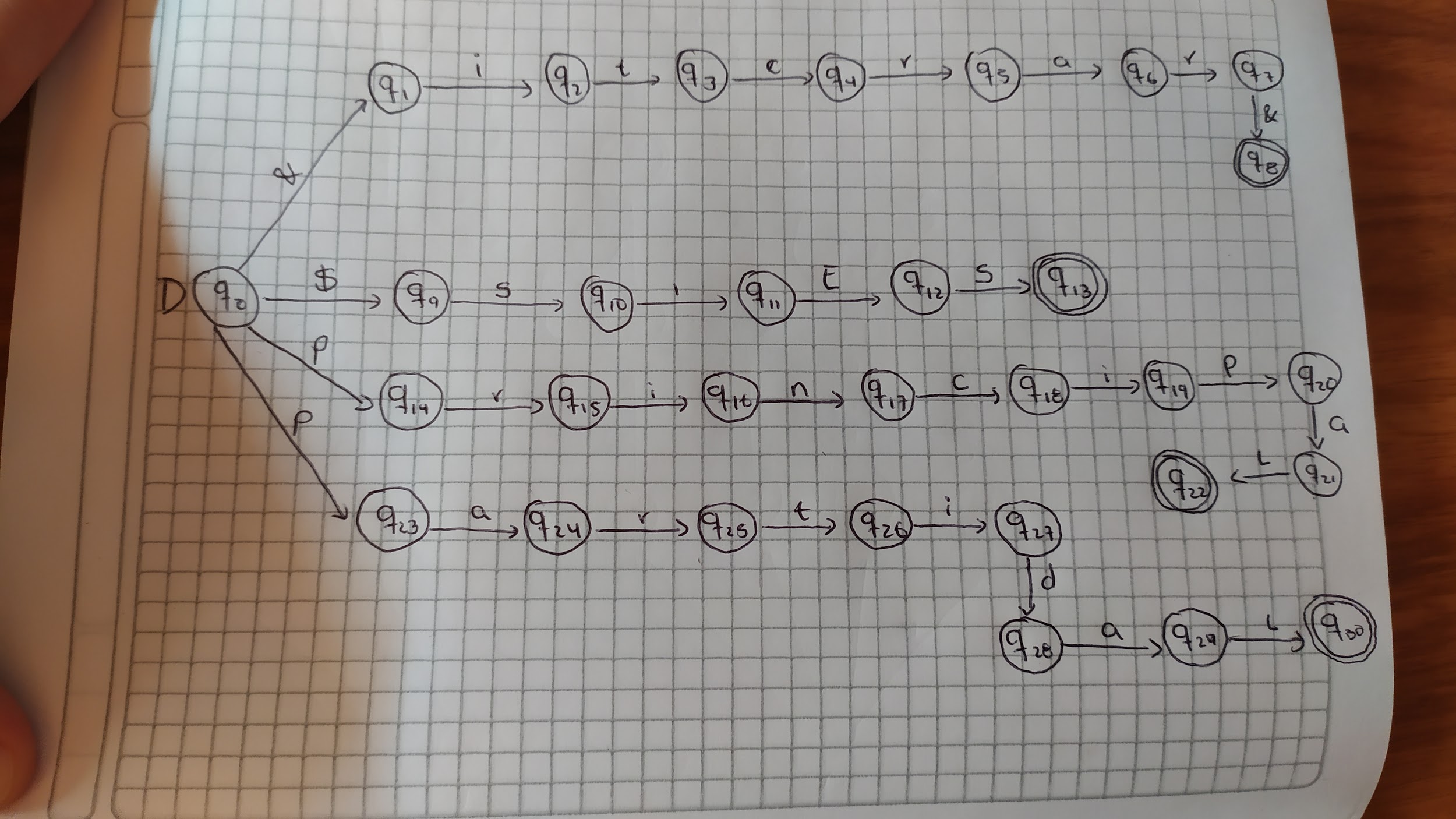
Categoría: **Valor de Asignación**



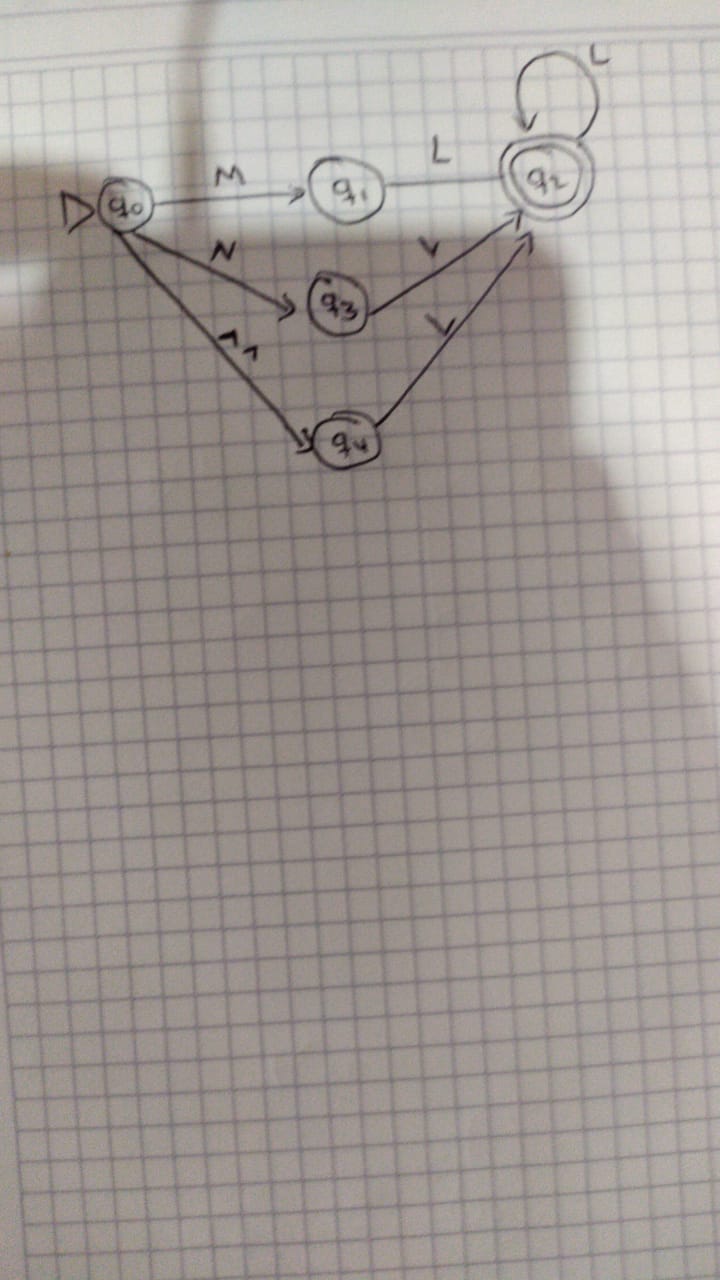
Categoría: **Palabras reservadas: Tipo de dato**



Categoría: **Palabras reservadas:** **Diferentes usos**



Categoría **:Palabras reservadas: Identificadores**

****