LFP SECCIÓN P (Vacaciones de junio 2025)
AUX. ELIAN SAUL ESTRADA URBINA
FERNANDO ANDHRE GONZALEZ ESPINOZA

PRACTICA 1:
MANUAL TÉCNICO

INTRODUCCIÓN

La finalidad de todo manual técnico es la de proporcionar al lector las pautas de configuración y la lógica con la que se ha desarrollado una aplicación, la cual se sabe que es propia de cada programador; por lo que se considera necesario ser documentada.

TypeScript

TypeScript es un lenguaje de programación de tipado estático que se construye sobre JavaScript. En esencia, es un superconjunto de JavaScript que añade tipos opcionales y características avanzadas para mejorar la seguridad y el mantenimiento del código.

Importaciones

Cabe resaltar que el import permite agregar a nuestro proyecto una o varias clases (paquete) según lo necesitemos.

```
import { Router } from "express";
import { analyze, home } from "../controllers/analyze.controller";
```

Variables

Son contenedores que se utilizan para almacenar y manipular datos en un programa. Una variable tiene un tipo de dato que define qué tipo de valor puede contener y un nombre que se utiliza para hacer referencia a ella en el código.

```
const analyzeRouter = Router();
```

Métodos

Los métodos son bloques de código que se utilizan para realizar tareas específicas. Cada método tiene un nombre que lo identifica y puede recibir una serie de parámetros, los cuales son

variables que proporcionan información necesaria para que el método realice su tarea.

```
//Funcion de retorno
private clean(){
    this.state = 0;
    this.auxChar ='';
}

private addToken(type: Type, lexeme: string, row: number, column: number){
    this.tokenList.push(new Token(type, lexeme, row, column));
}

private addError(type: Type, lexeme: string, row: number, column: number){
    this.errorList.push(new Token(type, lexeme, row, column));
}

getErroList(){
    return this.errorList;
}
```

CICLOS

Los ciclos o también conocidos como bucles, son una estructura de control de total importancia para el proceso de creación de un programa. Java y prácticamente todos los lenguajes más populares de la actualidad nos permiten hacer uso de estas estructuras y muchas más.

Ciclo for

Permite ejecutar una instrucción en Java varias veces y se utiliza principalmente cuando se conoce el número total de pasadas antes de la ejecución. Las instrucciones necesarias se colocan dentro del cuerpo del bucle

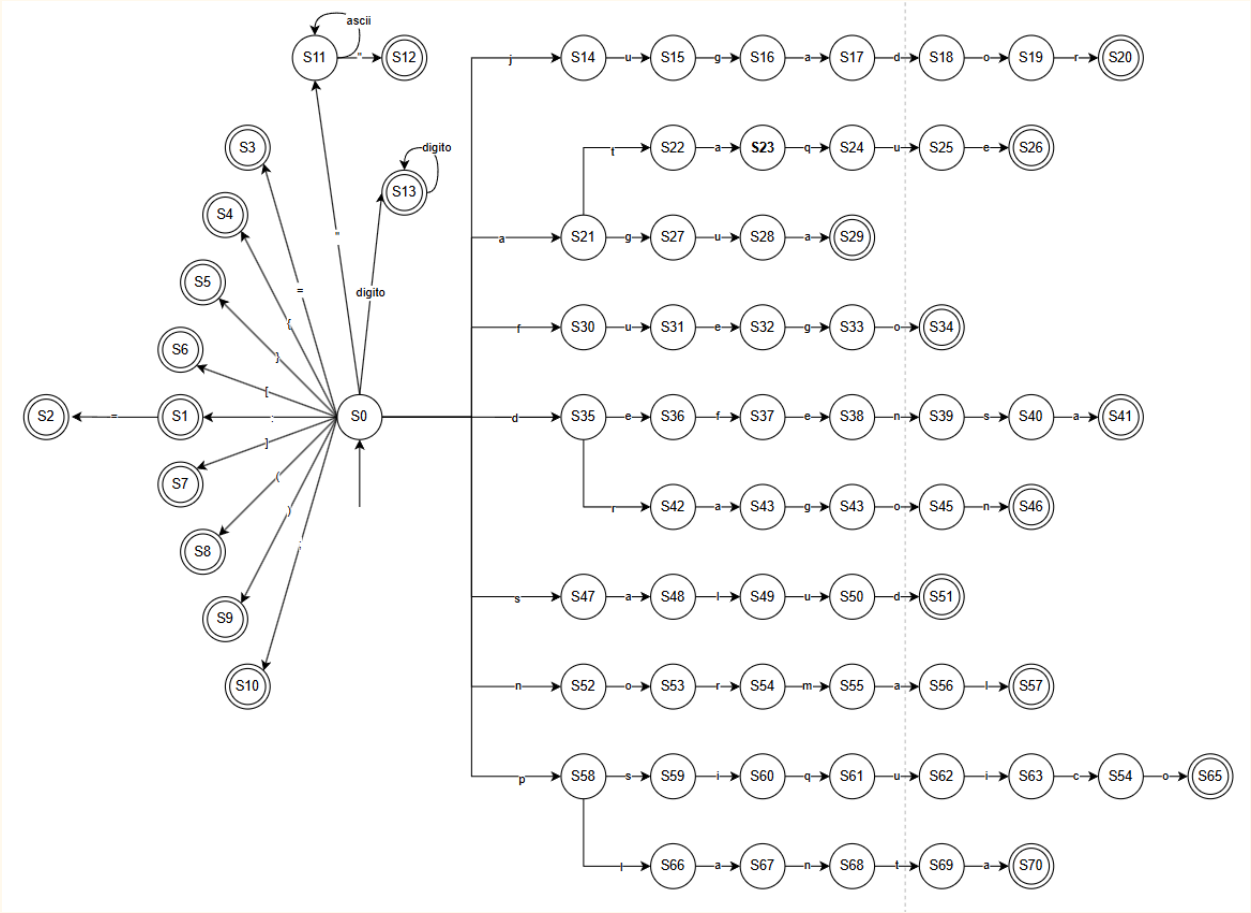
```
for (let i = 0; i < imagenes.length; i++) {

    let pokemon = imagenes[i].getAttribute('id');
    console.log(pokemon);

    getPokemon(pokemon, imagenes[i]);
}
```

Autómata Finito Determinista (AFD)

AFD

Un Autómata Finito Determinista (AFD) es un modelo matemático que simula el funcionamiento de un sistema que puede estar en un número finito de estados y que, para cada estado y cada entrada, tiene una única transición definida. Es decir, es determinista porque la siguiente transición está completamente determinada por el estado actual y la entrada actual, sin posibilidad de múltiples caminos.

Para esta práctica usamos el siguiente afd:



Para poder realizar el AFD, primero identificamos los tokens que usamos en nuestro lenguaje.

```
enum Type {
    UNKNOW,
    BRACK_OPEN,
    BRACK_CLOSE,
    COLON,
    BRACE_OPEN,
    BRACE_CLOSE,
    PAR_OPEN,
    PAR_CLOSE,
    SEMICOLON,
    EQUAL,
    RESERVERD_WORD,
    NUMBER,
    STRING,
    ASSIGN
}

class Token {

    private row: number;
    private column: number;
    private lexeme: string;
    private typeToken: Type;
    private typeTokenString: string;

    constructor(typeToken: Type, lexeme: string, row: number, column: number){
        this.typeToken = typeToken;
        this.typeTokenString = Type[typeToken];
        this.lexeme = lexeme;
        this.row = row;
        this.column = column;
    }
}

export { Token, Type}

```

Ahora que ya identificamos los tokens procedemos a crear el analizador léxico, ya que este nos dará la cadena de tokens (lexemas). Para poder crear el analizador léxico, en este caso usaremos el ciclo for y la sentencia switch case ya que este es un mecanismo de control de flujo que permite ejecutar diferentes bloques de código dependiendo del valor de una variable o expresión.

```
26          for (let i: number = 0; i < input.length; i++){
27
28              char = input[i];
29
30              switch(this.state){
31                  case 0:
32                      switch(char){
33                          case ';':
34                              this.state = 10;
35                              this.addCharacter(char);
36                              break;
37                          case ')':
38                              this.state = 9;
39                              this.addCharacter(char);
40                              break;
41                          case '(':
42                              this.state = 8;
43                              this.addCharacter(char);
44                              break;
45                          case ']':
46                              this.state = 7;
47                              this.addCharacter(char);
48                              break;
49                          case ':':
50                              this.state = 1;
51                              this.addCharacter(char);
```

```
52                              break;
53                          case '[':
54                              this.state = 6;
55                              this.addCharacter(char);
56                              break;
57                          case '}':
58                              this.state = 5;
59                              this.addCharacter(char);
60                              break;
61                          case '{':
62                              this.state = 4;
63                              this.addCharacter(char);
64                              break;
65                          case '=':
66                              this.state = 3;
67                              this.addCharacter(char);
68                              break;
69                          case '"':
70                              this.state = 11;
71                              this.addCharacter(char);
72                              break;
73                          case 'j':
74                              this.state = 14;
75                              this.addCharacter(char);
76                              break;
77                          case 'a':
78                              this.state = 21;
```

```
                this.addCharacter(char);
                break;
            case 'f':
                this.state = 30;
                this.addCharacter(char);
                break;
            case 'd':
                this.state = 35;
                this.addCharacter(char);
                break;
            case 's':
                this.state = 47;
                this.addCharacter(char);
                break;
            case 'n':
                this.state = 52;
                this.addCharacter(char);
                break;
            case 'p':
                this.state = 58;
                this.addCharacter(char);
                break;
            case ' ':
                this.column++;
                break;
            case '\n':
            case '\r':
```

```
            case '\r':
                this.row++;
                this.column = 1;
                break;
            case '\t':
                this.column += 4;
                break;
            default:
                if(/\d/.test(char)){
                    //es un digito
                    this.state = 13;
                    this.addCharacter(char);

                }else if (char == '#' && i == input.length - 1){
                    //se termino el analisis
                    console.log("Anakyze finised");
                }else{
                    //error lexico
                    this.addError(Type.UNKNOW, char, this.row, this.column);
                    this.column++;
                }
                break;
        }

```