LEXER Y
PARSER

HAILTON AMAYA
MONICA CARRANZA

LEXER

La clase incluye métodos para dividir el código fuente en componentes básicos, identificar y clasificar cada componente según su tipo (como paréntesis, operadores, números, identificadores, palabras clave, comentarios, etc.), y almacenar estos tokens en una lista.

También incluye funciones auxiliares para ayudar con la tokenización, como verificar si una cadena es un número, un alfanumérico, o si debe ser ignorada (como los espacios en blanco). El lexer puede imprimir los tokens resultantes y proporcionar acceso a la lista de tokens generados.

```cpp
#include "Lexer.h"

Lexer::Lexer(const std::string &sourceCode) : source(

void Lexer::tokenize() {
    std::vector<std::string> src = splitString(source

    while (!src.empty()) {
        if (src.front() == "(") {
            tokens.push_back(createToken(shift(src),
        } else if (src.front() == ")") {
            tokens.push_back(createToken(shift(src),
        } else if (src.front() == "=") {
            tokens.push_back(createToken(shift(src),
        } else if (src.front() == "+" || src.front()
            tokens.push_back(createToken(shift(src),
        } else if (src.front() == ",") {
            tokens.push_back(createToken(shift(src),
        } else if (src.front()[0] == '"') {
            std::string stringLiteral;
            if (src.front().size() == 3 && src.front(
                stringLiteral = shift(src);
            } else {
                do {
                    stringLiteral += shift(src);
                } while (!src.empty() && src.front()
```

PARSER

Objetivo n° 1

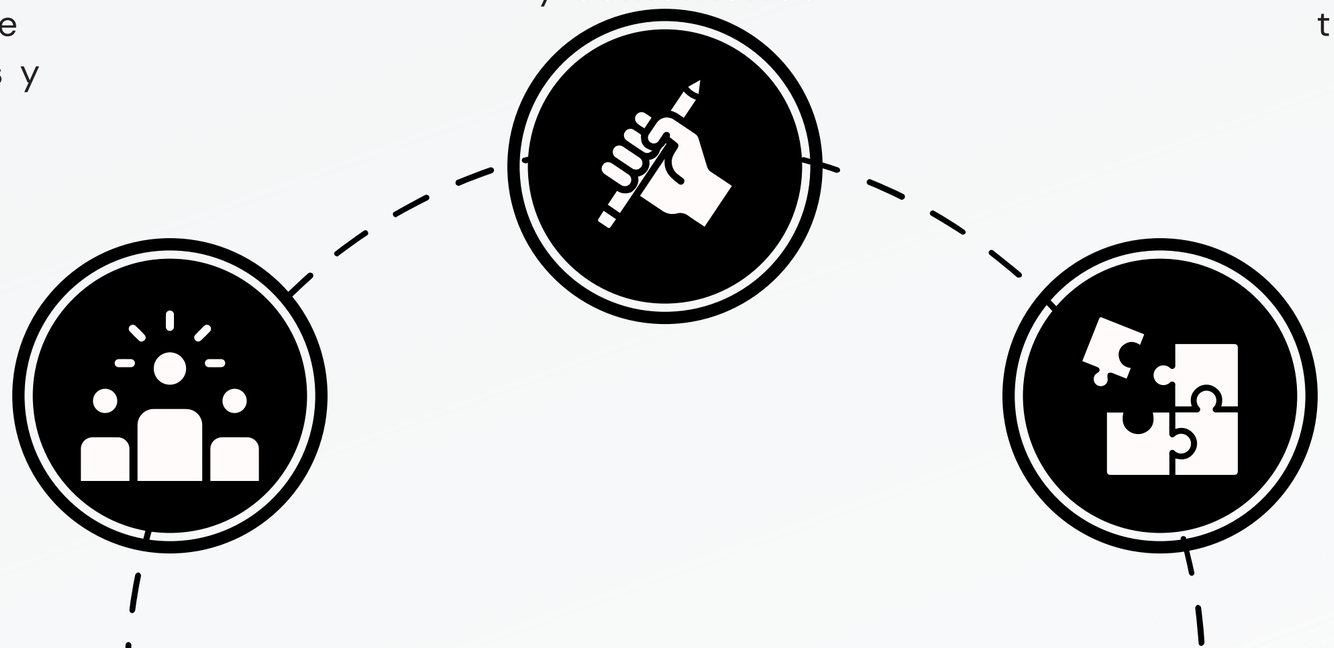
El parser verifica que el programa comience con "program" y termine con "end program", manejando correctamente identificadores y secciones.

Objetivo n° 2

Divide el código en secciones de declaración y ejecución, asegurando "implicit none" y gestionando declaraciones, bucles **do**, sentencias **print**, y comentarios.

Objetivo n° 3

Utiliza funciones auxiliares para validar expresiones y tipos de tokens, y métodos para manejar y examinar tokens de manera ordenada.



DEMOSTRACIÓN PRÁCTICA

GRACIAS