Relatório Técnico:

Este código implementa um analisador sintático para a linguagem MicroPascal. Ele utiliza um analisador léxico para obter os tokens do código-fonte e, em seguida, verifica se a sequência de tokens está de acordo com as regras da gramática.

Structs:

- **Token:** Representa um token léxico, contendo seu nome (tipo), lexema (valor), linha e coluna onde foi encontrado.
- **Symbol:** Representa um símbolo na tabela de símbolos, contendo seu lexema e tipo (identificador ou palavra reservada).

Funções:

- **casaToken(char *tokenEsperado):** Verifica se o token atual corresponde ao token esperado. Se corresponder, avança para o próximo token; caso contrário, exibe uma mensagem de erro e encerra o programa.
- **programa(), bloco(), parteDeclaracoesVariaveis(), declaracaoVariaveis(), listaIdentificadores(), tipo(), comandoComposto(), comando(), atribuicao(), comandoCondicional(), comandoRepetitivo(), expressao(), relacao(), expressaoSimples(), termo(), fator(), variavel():** Cada uma dessas funções implementa uma regra da gramática do MicroPascal. Elas verificam a sequência de tokens e chamam outras funções recursivamente para analisar as partes da regra.

Testes:

Programas Corretos:

```
program teste1;

var

 x : integer;

begin

 x := 10;

end


program teste2;

var
```

```
 x, y : integer;
begin
 if x > y then
   x := y
 else
   y := x;
end
```

```
program teste3;
var
 x : integer;
begin
 x := 0;
 while x < 10 do
   x := x + 1;
end
```

Programas Errados:

```
program teste4
var
 x : integer;
begin
 x := 10
end.
```

Saída: 4:token nao esperado [end.].

```
program teste5;
var
```

```
 x : string;
begin
 x := 10;
end.
```

Saída: 3:token nao esperado [string].

```
program teste6;
var
 x : integer;
begin
 x ;= 10 + 5 * 2;
end.
```

Saída: 5:token nao esperado [;].