

# PASCAL BR

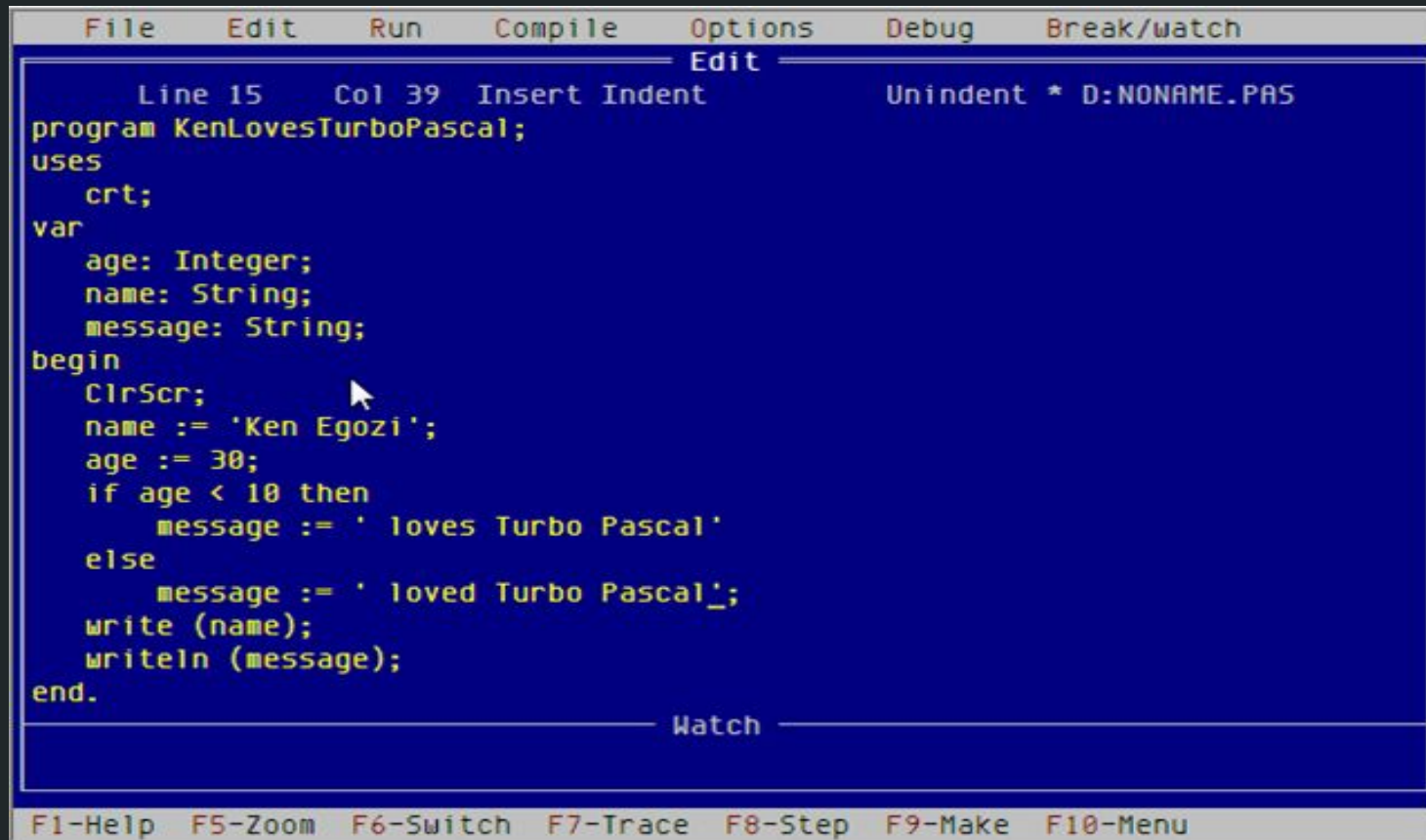
---

**Alunos:** Gabriel, Hudson e Widimarque

# Tema

- **Compilador:** Pascal BR
- **Linguagem para Desenvolvimento:** JAVA
- **Bibliotecas:** Jcup , Jflex

# Pascal



The screenshot shows the Turbo Pascal IDE interface. The menu bar at the top includes File, Edit, Run, Compile, Options, Debug, and Break/watch. The status bar at the bottom shows function key shortcuts: F1-Help, F5-Zoom, F6-Switch, F7-Trace, F8-Step, F9-Make, and F10-Menu. The main editing area has a blue background with yellow text. It shows a Pascal program named 'KenLovesTurboPascal'. The code includes variable declarations for 'age' (Integer), 'name' (String), and 'message' (String). The 'begin' block contains a 'ClrScr;' statement, followed by assignments for 'name' and 'age', and an 'if' statement that checks if 'age' is less than 10. Depending on the condition, it assigns a message to 'message'. Finally, it uses 'write' and 'writeln' to output the name and message. The cursor is positioned on the 'ClrScr;' line. The status bar at the top right indicates 'Line 15 Col 39 Insert Indent Unindent \* D:NONAME.PAS'.

```
File Edit Run Compile Options Debug Break/watch
Line 15 Col 39 Insert Indent Unindent * D:NONAME.PAS
program KenLovesTurboPascal;
uses
  crt;
var
  age: Integer;
  name: String;
  message: String;
begin
  ClrScr;
  name := 'Ken Egozi';
  age := 30;
  if age < 10 then
    message := ' loves Turbo Pascal'
  else
    message := ' loved Turbo Pascal';
  write (name);
  writeln (message);
end.
```

Watch

F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F10-Menu

# Tabela de Tokens

Token	Lexema	Descrição
algoritmo	INICIO_PROGRAMA	Palavra reservada ao nome do programa
inicio	INICIO	Palavra reservada begin
fim	FIM	Palavra reservada end
var	var	Palavra reservada var
declaravari	VARIAVEL	Declara Variavel
atribui	:=	Símbolo de atribuição
entrada	ENTRADA	Entrada primitiva de entrada
abrepar	(	Abre Parênteses
fechapar	)	Fecha Parênteses
igual	=	Símbolo reservado igual
relaciona	>, <, >=, <=, ==, !=	Tokens Relacionais
operaciona	+, -, *, /, ^	Tokens Aritméticos
valor	(0-9)*	Constantes Numéricas
quebradelinha	\n	Símbolo reservado \n

<u>fimdelinha</u>	;	Símbolo reservado ;
pontofinal	.	Símbolo reservado .
se	IF	Palavra reservada if
entao	THEN	Palavra reservada then
para	FOR	Estrutura de repetição
enquanto	WHILE	Estrutura de repetição
faca	DO	Estrutura de repetição
senao	ELSE	Palavra reservada else
ate	UNTIL	Palavra reservada until
caso	CASE	Palavra reservada case
saida	WRITELN	Atribuição primitiva de saída
repita	REPEAT	Palavra reservada repeat
p_int	INTEIRO	Declarar variável inteira
p_str	STRING	Declarar variável string
p_real	REAL	Declarar variável real
p_bool	BOOLEAN	Declarar variável booleana
p_delesq	[	Delimitador para iniciar entrada do valor de uma variável
p_delest	]	Delimitador para finalizar entrada do valor de uma variável
virgula	,	Símbolo reservado ,
<u>doispontos</u>	:	Símbolo reservado :



# Trechos do Código

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package AnaliseLexica;

/**
 *
 */
public enum Token {
    algoritmo, inicio, fim, var, atribui, entrada, declaravari,
    abrepar, fechapar, relaciona, operaciona, valor, quebradelinha,
    fimdelinha, pontofinal, se, entao, para, enquanto, faca, senao,
    ate, caso, saida, repita, p_int, p_str, p_real, p_bool, p_delesq,
    p_delest, igual, virgula, doispontos, erro;
}
```



## Trechos do Código

```
/* Palavras reservadas */
("INICIO_PROGRAMA") {lexeme = yytext(); return algoritmo;}
("INICIO") {lexeme = yytext(); return inicio;}
("FIM") {lexeme = yytext(); return fim;}
("VARIAVEL") {lexeme = yytext(); return declaravari;}
({letras}*) {lexeme = yytext(); return var;}
({numeros}*) {lexeme = yytext(); return valor;}
(":=") {lexeme = yytext(); return atribui;}
("ENTRADA") {lexeme = yytext(); return entrada;}
("(") {lexeme = yytext(); return abrepar;}
(")") {lexeme = yytext(); return fechapar;}
("=") {lexeme = yytext(); return igual;}
("+") {lexeme = yytext(); return operaciona;}
("-") {lexeme = yytext(); return operaciona;}
("*") {lexeme = yytext(); return operaciona;}
("/") {lexeme = yytext(); return operaciona;}
("^") {lexeme = yytext(); return operaciona;}
("<") {lexeme = yytext(); return relaciona;}
(">") {lexeme = yytext(); return relaciona;}
("==") {lexeme = yytext(); return relaciona;}
("!=") {lexeme = yytext(); return relaciona;}
(">=") {lexeme = yytext(); return relaciona;}
("<=") {lexeme = yytext(); return relaciona;}
("\n") {lexeme = yytext(); return quebradelinha;}
(";") {lexeme = yytext(); return fimdelinha;}
(".") {lexeme = yytext(); return pontofinal;}
("SE") {lexeme = yytext(); return se;}
("ENTAO") {lexeme = yytext(); return entao;}
("PARA") {lexeme = yytext(); return para;}
("ENQUANTO") {lexeme = yytext(); return enquanto;}
```

# Exemplo:

## Programa para calcular fatorial.

```
INICIO_PROGRAMA fatorial ;
```

```
VARIAVEL res , num : INTEIRO;
```

```
INICIO
```

```
res := 1;
```

```
ENTRADA(num);
```

```
ENQUANTO num > 1 FACA
```

```
INICIO
```

```
res := res * num ;
```

```
num := num -1
```

```
FIM;
```

```
SAIDA(res)
```

```
FIM.
```





## Resposta

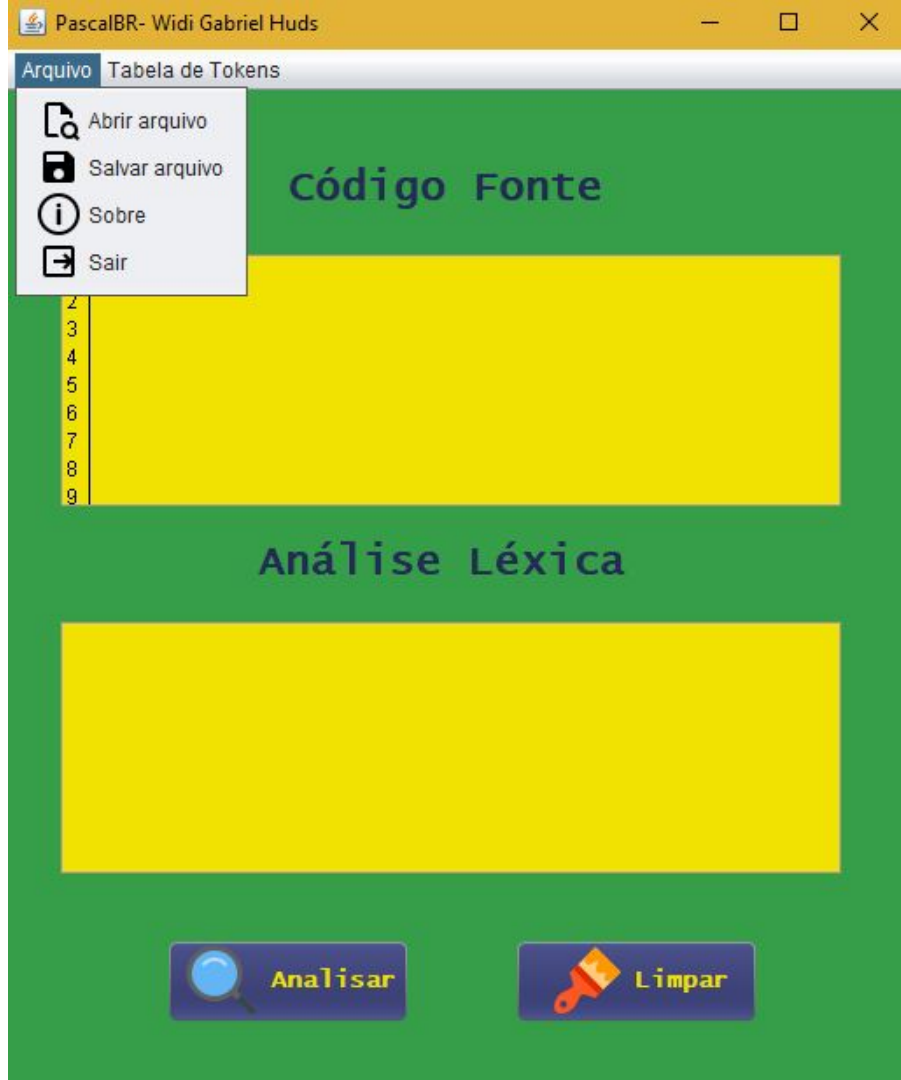
Linha 1: Token: algoritmo / Lexema: INICIO\_PROGRAMA  
Linha 1: Token: var / Lexema: fatorial  
Linha 1: Token: fimdelinha / Lexema: ;  
Linha 3: Token: declaravari / Lexema: VARIABEL  
Linha 3: Token: var / Lexema: res  
Linha 3: Token: virgula / Lexema: ,  
Linha 3: Token: var / Lexema: num  
Linha 3: Token: doisPontos / Lexema: :  
Linha 3: Token: p\_int / Lexema: INTEIRO  
Linha 3: Token: fimdelinha / Lexema: ;  
Linha 5: Token: inicio / Lexema: INICIO  
Linha 6: Token: var / Lexema: res  
Linha 6: Token: atribui / Lexema: :=  
Linha 6: Token: valor / Lexema: 1  
Linha 6: Token: fimdelinha / Lexema: ;

Linha 7: Token: entrada / Lexema: ENTRADA  
Linha 7: Token: abrepar / Lexema: (  
Linha 7: Token: var / Lexema: num  
Linha 7: Token: fechapar / Lexema: )  
Linha 7: Token: fimdelinha / Lexema: ;  
Linha 9: Token: enquanto / Lexema: ENQUANTO  
Linha 9: Token: var / Lexema: num  
Linha 9: Token: relaciona / Lexema: >  
Linha 9: Token: valor / Lexema: 1  
Linha 9: Token: faca / Lexema: FAÇA  
Linha 10: Token: inicio / Lexema: INICIO  
Linha 11: Token: var / Lexema: res  
Linha 11: Token: atribui / Lexema: :=  
Linha 11: Token: var / Lexema: res  
Linha 11: Token: operacional / Lexema: \*  
Linha 11: Token: var / Lexema: num  
Linha 11: Token: fimdelinha / Lexema: ;  
Linha 12: Token: var / Lexema: num  
Linha 12: Token: atribui / Lexema: :=

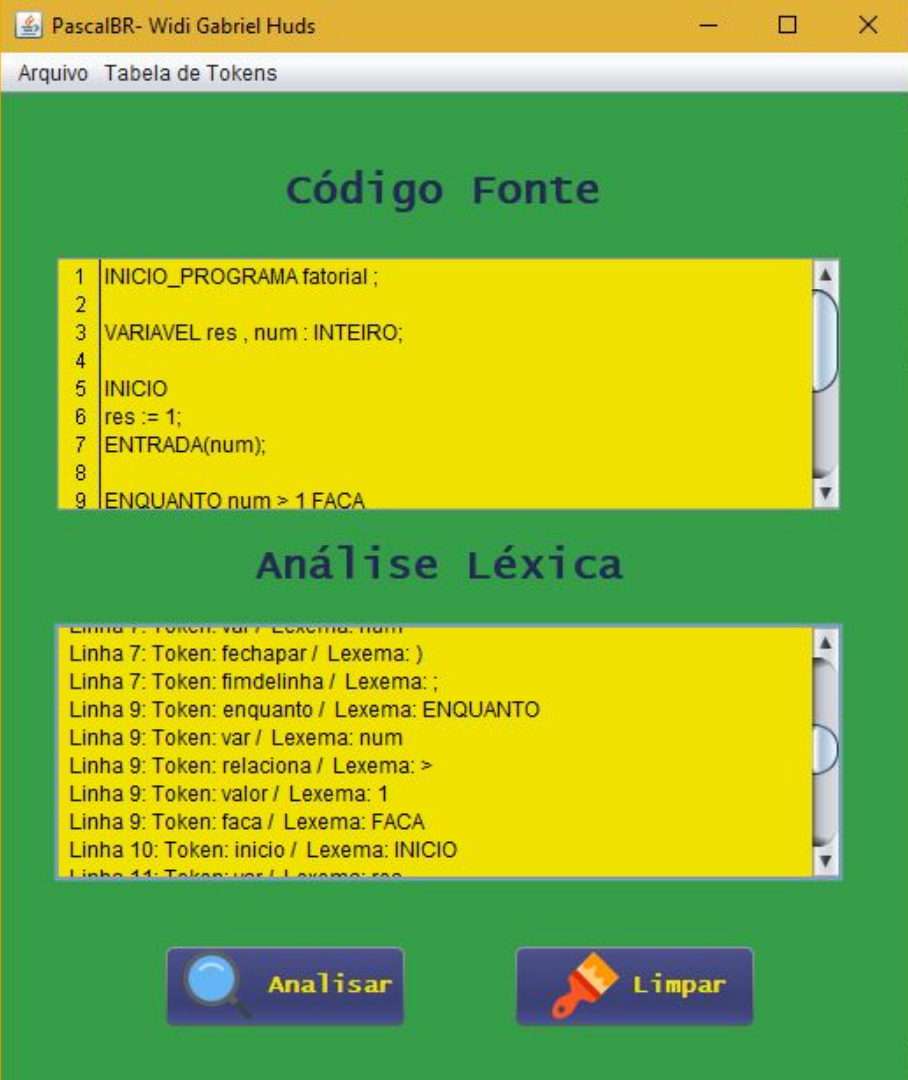
Linha 12: Token: var / Lexema: num  
Linha 12: Token: operacional / Lexema: -  
Linha 12: Token: valor / Lexema: 1  
Linha 13: Token: fim / Lexema: FIM  
Linha 13: Token: fimdelinha / Lexema: ;  
Linha 15: Token: saida / Lexema: SAIDA  
Linha 15: Token: abrepar / Lexema: (  
Linha 15: Token: var / Lexema: res  
Linha 15: Token: fecharpar / Lexema: )  
Linha 16: Token: fim / Lexema: FIM  
Linha 16: Token: pontofinal / Lexema: .



# Tela Inicial do Compilador



Menu de opções  
do compilador



Compilador em  
Execução

A photograph of three young men in a swimming pool, overlaid with a dark semi-transparent filter. One man is in the foreground, another slightly behind him, and a third further back. They are all smiling and appear to be enjoying the water.

# Obrigado!