

Pontifícia Universidade Católica do Paraná

Avaliativa - RA1

BRUNO FELICIANO, MICAEL FONE, PEDRO NAZARIO, THIAGO ROSA

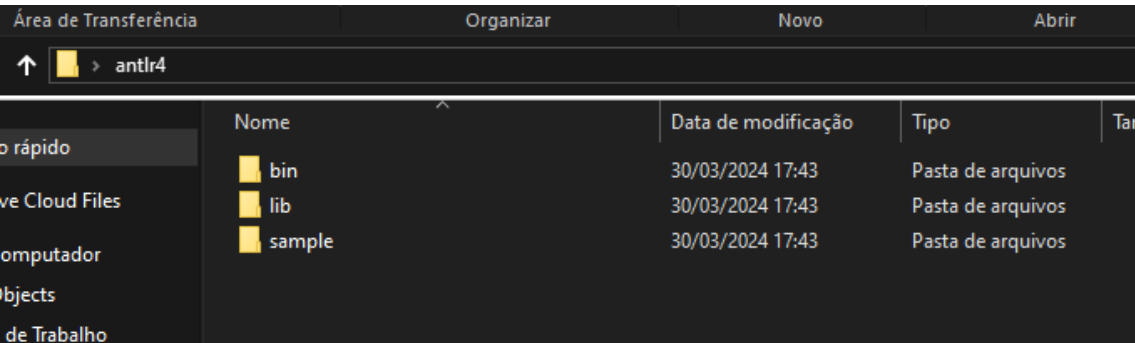
CURITIBA

2024

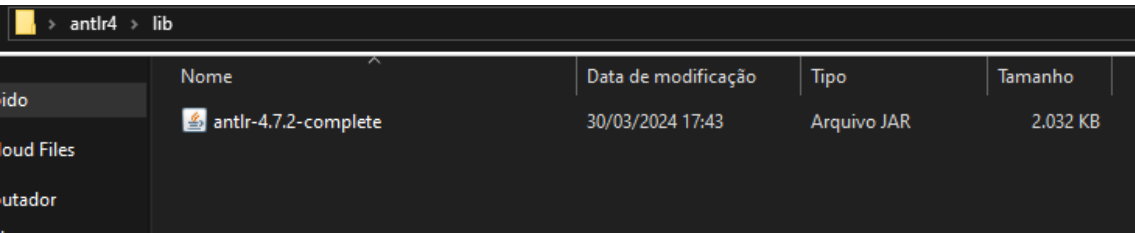
INTRODUÇÃO:

Neste trabalho, abordamos a criação de analisadores léxicos e sintáticos para uma calculadora utilizando a ferramenta Antlr. Nosso objetivo é validar arquivos de teste fornecidos pela equipe, demonstrando a conformidade dos mesmos com a linguagem desenvolvida.

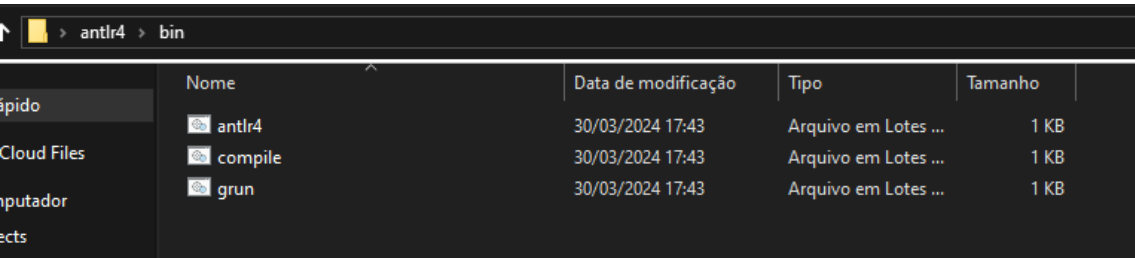
Para desenvolver os analisadores léxicos, a equipe utilizou a ferramenta Antlr para a criação da gramática que irá interpretar as expressões matemáticas. No projeto, utilizamos um número de shellscrips organizados para que possamos trabalhar com o Antlr.



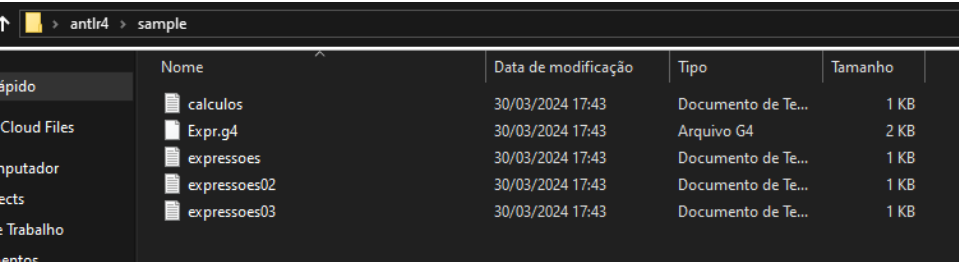
Dentro do diretório, temos três subdiretórios, o primeiro nomeado como lib possui o arquivo jar para os binários do Antlr, todas as classes da ferramenta estão empacotadas nesse arquivo jar.



O segundo arquivo, intitulado bin, contem os arquivos shellscrips que permitem trabalhar com o Antlr, ele possui mais 3 arquivos, o primeiro arquivo (antlr4.bat) permite compilar a gramática e gerar alguns arquivos java que serão compilados pelo segundo arquivo (compile.bat), o terceiro e ultimo arquivo (grun.bat) nos permite testar a gramática que foi criada por nós.



O último diretório que contém na pasta do Antlr é o sample, nele fica a nossa gramática com os nossos arquivos que contém os cálculos que a calculadora executará, a gramática foi definida para atuar semelhante a calculadora que foi pedida pelo TDE-Calculadora.



	Nome	Data de modificação	Tipo	Tamanho
	calculos	30/03/2024 17:43	Documento de Te...	1 KB
	Expr.g4	30/03/2024 17:43	Arquivo G4	2 KB
	expressoes	30/03/2024 17:43	Documento de Te...	1 KB
	expressoes02	30/03/2024 17:43	Documento de Te...	1 KB
	expressoes03	30/03/2024 17:43	Documento de Te...	1 KB

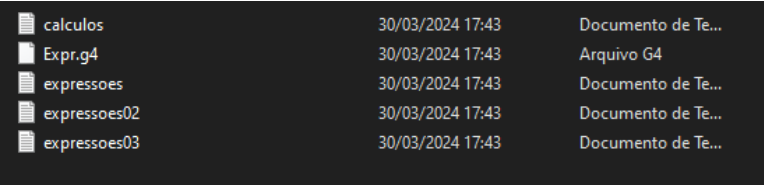
Para testar a nossa calculadora, devemos criar duas variáveis de ambiente para que possamos testar a execução, uma variável para a pasta do Antlr e outra dentro da variável path.

Variável	Valor
ANTLR4_HOME	C:\antlr4
C:\Users\Bruno\AppData\Roaming\npm	
C:\Program Files\Azure Data Studio\bin	
C:\Users\Bruno\AppData\Local\GitHubDesktop\bin	
C:\Users\Bruno\AppData\Local\Programs\mongosh\	
%ANTLR4_HOME%\bin	

Depois disso podemos abrir o prompt de comando e ir até o caminho da pasta sample, então digitamos o comando “antlr4 nome\_arquivo\_gramatica.g4”, após finalizar esse comando, ele gerará alguns arquivos que irão possibilitar a execução da calculadora.

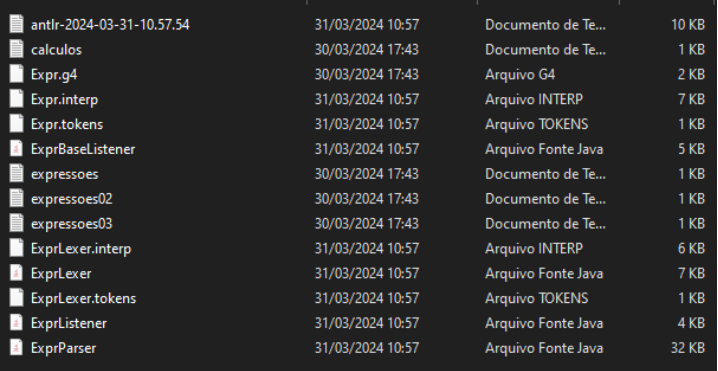
```
C:\antlr4\sample>antlr4 Expr.g4
wrote ./antlr-2024-03-31-10.57.54.log
```

- Antes:































calculos	30/03/2024 17:43	Documento de Te...
Expr.g4	30/03/2024 17:43	Arquivo G4
expressoes	30/03/2024 17:43	Documento de Te...
expressoes02	30/03/2024 17:43	Documento de Te...
expressoes03	30/03/2024 17:43	Documento de Te...

- Depois:



antlr-2024-03-31-10.57.54	31/03/2024 10:57	Documento de Te...	10 KB
calculos	30/03/2024 17:43	Documento de Te...	1 KB
Expr.g4	30/03/2024 17:43	Arquivo G4	2 KB
Expr.interp	31/03/2024 10:57	Arquivo INTERP	7 KB
Expr.tokens	31/03/2024 10:57	Arquivo TOKENS	1 KB
ExprBaseListener	31/03/2024 10:57	Arquivo Fonte Java	5 KB
expressoes	30/03/2024 17:43	Documento de Te...	1 KB
expressoes02	30/03/2024 17:43	Documento de Te...	1 KB
expressoes03	30/03/2024 17:43	Documento de Te...	1 KB
ExprLexer.interp	31/03/2024 10:57	Arquivo INTERP	6 KB
ExprLexer	31/03/2024 10:57	Arquivo Fonte Java	7 KB
ExprLexer.tokens	31/03/2024 10:57	Arquivo TOKENS	1 KB
ExprListener	31/03/2024 10:57	Arquivo Fonte Java	4 KB
ExprParser	31/03/2024 10:57	Arquivo Fonte Java	32 KB

```
C:\antlr4\sample>compile Expr*.java
```

 antr-2024-03-31-10.57.54 Documento de Texto 9,55 KB	 calculos Documento de Texto 57 bytes	 Expr.g4 Arquivo G4 1,93 KB	 Expr.interp Arquivo INTERP 6,81 KB
 Expr.tokens Arquivo TOKENS 327 bytes	 ExprBaseListener.class Arquivo CLASS 2,80 KB	 ExprBaseListener Arquivo Fonte Java 4,39 KB	 expressoes Documento de Texto 246 bytes
 expressoes02 Documento de Texto 172 bytes	 expressoes03 Documento de Texto 220 bytes	 ExprLexer.class Arquivo CLASS 5,61 KB	 ExprLexer.interp Arquivo INTERP 5,14 KB
 ExprLexer Arquivo Fonte Java 6,75 KB	 ExprLexer.tokens Arquivo TOKENS 327 bytes	 ExprListener.class Arquivo CLASS 1,78 KB	 ExprListener Arquivo Fonte Java 3,69 KB
 ExprParser\$ExprContext.class Arquivo CLASS 2,69 KB	 ExprParser\$Numero_float_semzer oContext.class Arquivo CLASS	 ExprParser\$Numero_floatContext.c lass Arquivo CLASS	 ExprParser\$Numero_inteiro_positiv oContext.class Arquivo CLASS
 ExprParser\$Numero_inteiro_semzer oContext.class Arquivo CLASS	 ExprParser\$Numero_inteiroContext .class Arquivo CLASS	 ExprParser\$Numero_realContext.cl ass Arquivo CLASS	 ExprParser\$Numero_zeroContext.cl ass Arquivo CLASS
 ExprParser\$OperadorContext.class Arquivo CLASS 755 bytes	 ExprParser\$ProgContext.class Arquivo CLASS 1,55 KB	 ExprParser.class Arquivo CLASS 15,7 KB	 ExprParser Arquivo Fonte Java 31,5 KB

- Exemplo de parse tree gerada de um dos arquivos teste:

