

Compiladores

CIÊNCIA DA COMPUTAÇÃO

Prof. Dr. João Paulo Aramuni

Laboratório

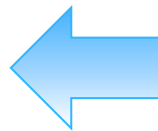
- * **Laboratório**
- * **Analizador Sintático (Parser)**
- * Implementação de analisadores sintáticos através da linguagem de programação C++.

Laboratório

- * Laboratório
- * Analisador Sintático (Parser)

- * Arquivos:

- * `Parser1.cpp`
- * `Parser2.cpp`

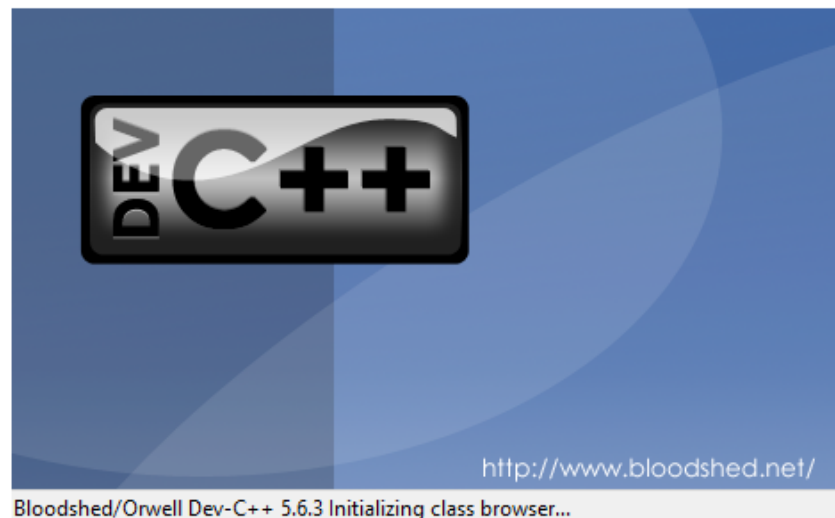
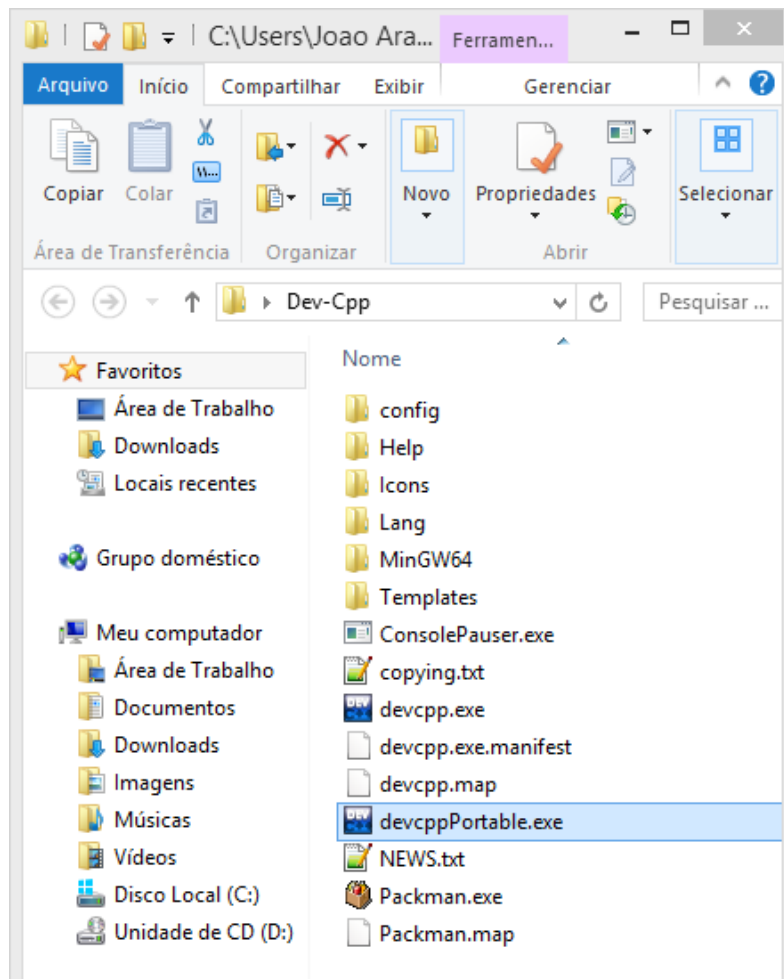


Disponível no SINEF
em Material Didático

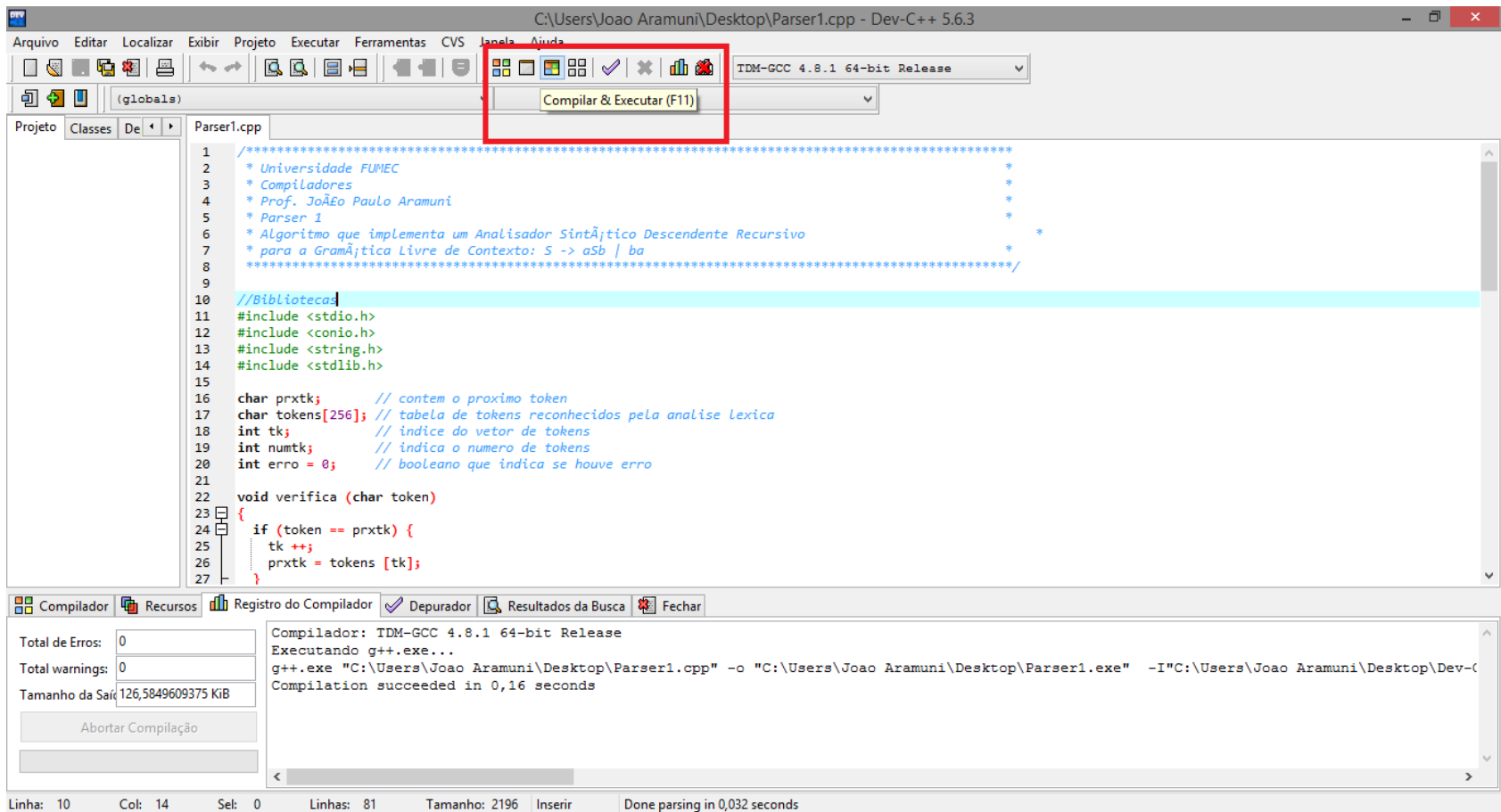
Laboratório

- * **Laboratório**
- * **Analizador Sintático (Parser)**
- * **Ferramentas:**
 - * Dev - Cpp Portable (Padrão adotado)
 - * Disponível no Correio
 - * Ou
 - * Eclipse Cpp + MinGW

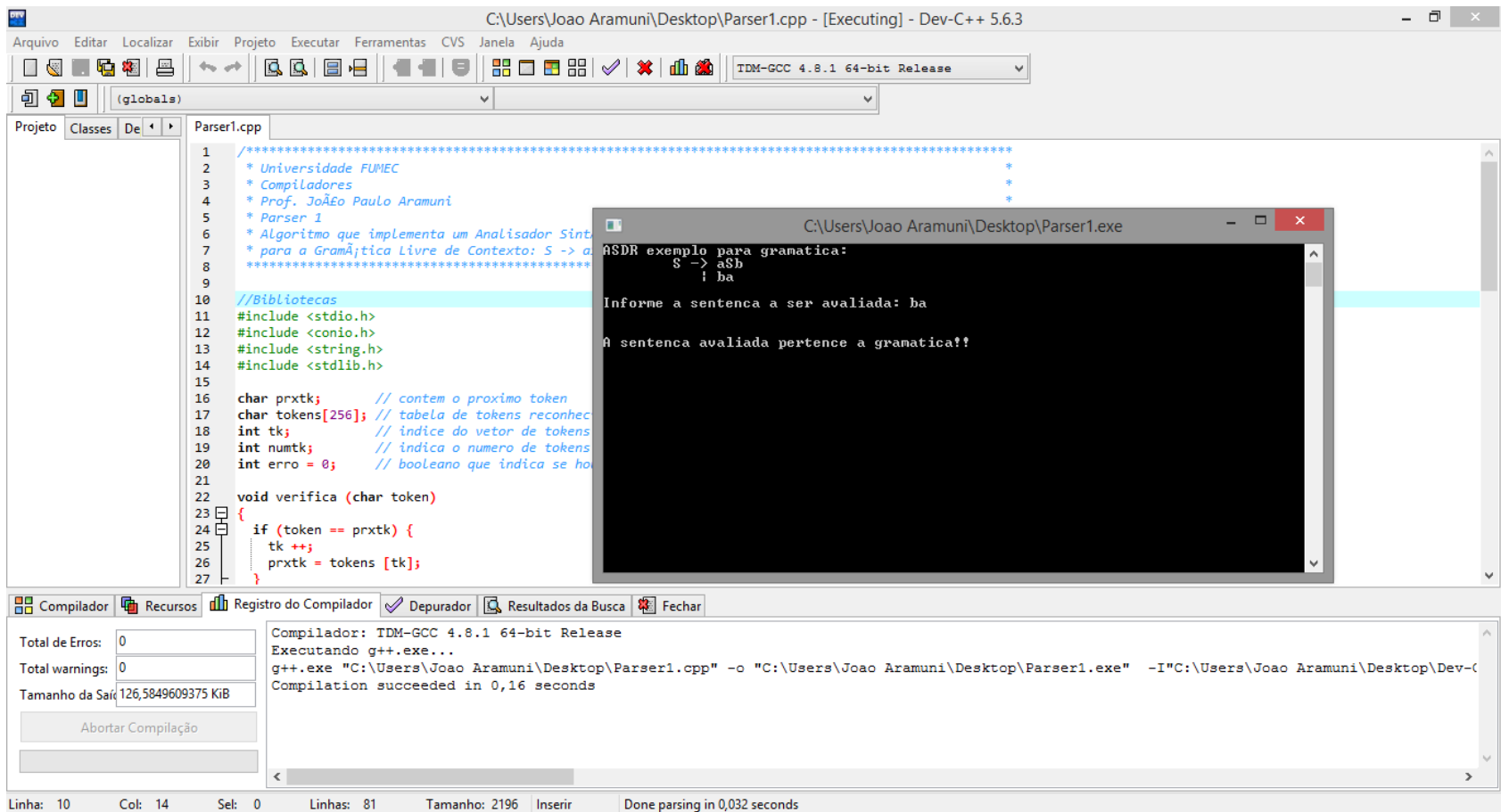
* Dev Cpp Portable:



* Dev Cpp Portable:



* Dev Cpp Portable:



The screenshot shows the Dev-C++ 5.6.3 IDE with the file `C:\Users\Joao Aramuni\Desktop\Parser1.cpp` open and being executed. The code is a C++ program for a simple grammar parser. The output window shows the program running and displaying the input sentence 'ba' and the result 'A sentença avaliada pertence a gramática!!'.

```
1  /* Universidade FUMEC
2  * Compiladores
3  * Prof. João Paulo Aramuni
4  * Parser 1
5  * Algoritmo que implementa um Analisador Sintático
6  * para a Gramática Livre de Contexto: S -> aSb
7  *
8  *
9  */
10 //Bibliotecas
11 #include <stdio.h>
12 #include <conio.h>
13 #include <string.h>
14 #include <stdlib.h>
15
16 char prxtk; // contem o proximo token
17 char tokens[256]; // tabela de tokens reconhecidos
18 int tk; // indice do vetor de tokens
19 int numtk; // indica o numero de tokens
20 int erro = 0; // booleano que indica se houve erro
21
22 void verifica (char token)
23 {
24     if (token == prxtk) {
25         tk++;
26         prxtk = tokens[tk];
27     }
```

ASDR exemplo para gramatica:
S -> aSb
! ba

Informe a sentença a ser avaliada: ba

A sentença avaliada pertence a gramática!!

Compilador: TDM-GCC 4.8.1 64-bit Release
Executando g++.exe...
g++.exe "C:\Users\Joao Aramuni\Desktop\Parser1.cpp" -o "C:\Users\Joao Aramuni\Desktop\Parser1.exe" -I"C:\Users\Joao Aramuni\Desktop\Dev-C++\include" -lstdc++
Compilation succeeded in 0,16 seconds

Total de Erros: 0
Total warnings: 0
Tamanho da Saída: 126,5849609375 KiB

Abortar Compilação

Linha: 10 Col: 14 Sel: 0 Linhas: 81 Tamanho: 2196 Inserir Done parsing in 0,032 seconds

Laboratório

- * **Laboratório**
- * **Analizador Sintático (Parser)**
- * Primeira parte da prática:
 - * 1) Examine cada um dos analisadores sintáticos.
 - * 2) Identifique as diferenças entre eles.
 - * 3) Compile e execute o código.
 - * 4) Realize alguns testes para validar o funcionamento.

Laboratório

- * **Laboratório**
- * **Analizador Sintático (Parser)**
- * Segunda parte da prática:
 - * 5) Modifique o código para reconhecer outras linguagens geradas por outras gramáticas livres de contexto.
 - * 6) Comente o código com a massa de testes realizada ou salve em um arquivo txt (Ex.: Testes_Parser1.txt).
 - * 7) Realize melhorias de sua preferência no analisador sintático.
 - * 8) Pesquise outros analisadores sintáticos e compare com os seus.

Laboratório

- * **Laboratório**
- * **Analizador Sintático (Parser)**
- * Terceira parte da prática (Extraclasse):
 - * 9) Pesquise sobre geradores de analisadores sintáticos.
 - * 10) Gere um analisador sintático utilizando o *Yacc Parser Generator* ou o *Javacc*.

* YACC

YACC

Yacc: Yet Another Compiler-Compiler

Stephen C. Johnson

Computer program input generally has some structure; in fact, every computer program that does input is as complex as a programming language, or as simple as a sequence of numbers. Unfortunately, usual input

Yacc provides a general tool for describing the input to a computer program. The Yacc user specifies the grammar, and Yacc turns such a specification into a subroutine that handles the input process; frequently, it is convenient as

- [Online Manual](#)
- [PostScript](#)
- [Yacc Manual Page](#)

* <http://dinosaur.compilertools.net/#yacc>

* JAVACC



JavaCC™

The Java Parser Generator

Java Compiler Compiler™ (JavaCC™) is the most popular parser generator for use with Java™ applications. A parser generator is a tool that reads a grammar specification and converts it to a Java program that can recognize matches to the grammar. In addition to the parser generator itself, JavaCC provides other standard capabilities related to parser generation such as tree building (via a tool called JJTree included with JavaCC), actions, debugging, etc.

* <https://javacc.org/>

Obrigado.

joapauloaramuni@gmail.com
joapauloaramuni@fumec.br