

# RDP Code Generator

## CZ3007 Compiler Techniques Lab Project

Student: Lim Pek Cheng, David Samuel    Supervisor: Dr Huang Shell Ying

### RDP Code Generator

Produces Recursive Descent Parser (RDP) code from a Context-Free Grammar (CFG) of your own design.  
Available in C++, Java, and C#.

Enter your CFG

C++  
 Java  
 C#

Include Doxygen Comments  
 Include Header File (for C++)

Guidelines

- The Non-Terminal on the Left of the First Rule is the Start Terminal.
- The arrow '->' is used to separate the LHS and RHS of a Production Rule.
- Production Rules belonging to a Non-Terminal can be separated by the OR '|' symbol, or by new lines.  
For example :  
A-> a | b  
and  
A-> a  
A-> b  
are both equally valid.
- An Epsilon-only Production Rule is indicated by an empty Rule.

Robust CFG Processing and RDP Code Generation engines.

Support for various languages (C++, Java, and C#).

Produces detailed reports that explain the RDP functionalities.

Demonstrates concepts taught in the CZ3007 course, such as LL(1) Predictive Parsing.

### Project Objectives:

The RDP Code Generator web application was developed as a supplementary tool for the CZ3007 Compiler Techniques course. It aims to help students gain a better understanding of the topics of Context-Free Grammars (CFGs) and Recursive Descent Parsers (RDPs) as taught in the course. Students can submit CFGs of their own design, and the application produces RDP code based on the CFG specification. Students can then download the RDP code to test on their own devices. Additionally, the application can also generate detailed reports that provide an in-depth explanation of the RDP functionalities.

By allowing students to engage in CFG design and RDP analysis in a hands-on manner, putting into practice concepts taught in the course, the application strives to fuel students' interest and knowledge of Compiler Techniques.

### CFG to RDP

Enter your CFG

S -> ABC  
A -> a |  
B -> b |  
C -> c



```
public static boolean parse_S()
{
    if (IsMemberOfPredictSet(peek(), "S", 1))
    {
        return parse_A() && parse_B() && parse_C();
    }
    else
    {
        System.out.println("Parse Error in parse_S() : Expected 'a' OR 'b' OR 'c'. Instead, found " + peek() + ".\n");
        return false;
    }
}

public static boolean parse_A()
{
    if (IsMemberOfPredictSet(peek(), "A", 2))
    {
        return match("a");
    }
    else
    {
        return true;
    }
}
```