

Feb 14, 17 13:01

Name: put your name here

Page 1/1

Summary file for decaf-daniel-byers

Name: put your name here

Student number: put your student number here

Lab class: put the day and time of your Software Architectures lab class here

\*\*\*\*\*

## PARSER TESTING

\*\*\* Legal tests \*\*\*

15/15 correct parser test cases

\*\*\* Legal tests with debug \*\*\*

\*\*\* Illegal tests \*\*\*

illegal-01 generated expected error

illegal-02 generated expected error

illegal-03 generated error:

line 2:8 missing INTLITERAL at ']' ✓

illegal-04 generated expected error

illegal-05 generated expected error

illegal-06 generated expected error

illegal-07 generated error:

line 3:8 mismatched input 'callout' expecting IDENTIFIER ✓

line 3:15 mismatched input ';' expecting '('

illegal-08 generated error:

line 4:14 mismatched input '=' expecting {';', '+', '-', '/', '\*', '%', '==', '!' ✓

=, '&lt;', '&gt;', '&lt;=', '&gt;=', '&amp;&amp;', '||'}

illegal-09 generated expected error

illegal-10 generated error:

line 6:2 extraneous input 'void' expecting {'for', 'break', 'if', 'callout', 're ✓

turn', 'continue', '{', '}', IDENTIFIER}

illegal-11 generated error:

line 7:7 missing IDENTIFIER at ';' ✓

illegal-12 generated expected error

illegal-13 generated expected error

illegal-14 generated error:

line 2:11 extraneous input 'a' expecting {'boolean', 'int', 'void', '}' ✓

illegal-15 generated error:

line 3:8 extraneous input 'int' expecting {'callout', '(', '-', NOT, INTLIT ✓

ERAL, BOOLEANLITERAL, CHARLITERAL, STRINGLITERAL, IDENTIFIER}

illegal-16 generated error:

line 3:8 mismatched input ')' expecting {'callout', '(', '-', NOT, INTLIT, B ✓

OOLEANLITERAL, CHARLITERAL, STRINGLITERAL, IDENTIFIER}

illegal-17 generated error:

line 3:4 mismatched input '0xafe' expecting {'boolean', 'for', 'break', 'if', ' ✓

callout', 'int', 'return', 'continue', 'void', '{', '}', IDENTIFIER}

line 5:0 extraneous input '}' expecting &lt;EOF&gt;

illegal-18 generated error:

line 3:12 mismatched input '5' expecting STRINGLITERAL

illegal-19 generated error:

line 5:11 no viable alternative at input 'forpari' ✓

line 5:16 mismatched input ',' expecting {';', '+', '-', '/', '\*', '%', '==', '!' ✓

=, '&lt;', '&gt;', '&lt;=', '&gt;=', '&amp;&amp;', '||'}

illegal-20 generated error:

line 5:11 no viable alternative at input 'forpari' ✓

line 5:16 mismatched input ',' expecting {';', '+', '-', '/', '\*', '%', '==', '!' ✓

=, '&lt;', '&gt;', '&lt;=', '&gt;=', '&amp;&amp;', '||'}

20 /20 correctly detected parser errors

Please make sure that you only  
give extra output if the -debug  
flag is used (otherwise crashes auto-  
mated testing).

```

Feb 14, 17 11:31      DecafParser.g4      Page 1/2

/**
 * @author Daniel Byers | 13121312
 *
 * This code builds on examples provided by the following book:
 * Parr, Terence (2012). The Definitive ANTLR 4 Reference. USA: The Pragmatic Bo
 * okshelf. 322.
 */

parser grammar DecafParser;
options { tokenVocab = DecafLexer; }

// top level. The context of the whole Decaf application.
program: CLASS IDENTIFIER LCURLY fieldDecl* methodDecl* RCURLY EOF;

// global scope. Arrays are only allowed there!
// int a; int a[10]; int a, b; int a[], b, c;
fieldDecl: type (IDENTIFIER | arrayDecl) (COMMA (IDENTIFIER | arrayDecl))* EOL;

// This is a "named identifier" so a method is generated to access it through
// the FieldDeclContext object.
arrayDecl: IDENTIFIER LBRACE INTLITERAL RBRACE;

methodDecl:
    (type | VOID) methodName LPAREN ((type IDENTIFIER) (COMMA type IDENTIFIER))*?
    RPAREN block;

// This is a "named identifier" so a method is generated to access it through
// the MethodDeclContext object.
methodName: IDENTIFIER;

// Variables have to be declared first.
block: LCURLY varDecl* statement* RCURLY;

// local scope
varDecl: type IDENTIFIER (COMMA IDENTIFIER)* EOL;

// basic types.
type: (INT | BOOLEAN | VOID);

statement:
    (
        location assignOp expr EOL
        methodCall EOL
        IF LPAREN expr RPAREN block (ELSE block)?
        FOR IDENTIFIER ASSIGNMENT expr COMMA expr block
        RETURN expr? EOL
        BREAK EOL
        CONTINUE EOL
        block
    );

assignOp: (ASSIGNMENT | ASSIGNMENTP | ASSIGNMENTS);

methodCall:
    (
        methodName LPAREN (expr (COMMA expr))*? RPAREN
        | CALLOUT LPAREN STRINGLITERAL (COMMA calloutArg (COMMA calloutArg))*
    )? RPAREN;

calloutArg: (expr | STRINGLITERAL);

location: (IDENTIFIER | IDENTIFIER LBRACE expr RBRACE);

// MINUS causes an ambiguity check to be raised because it's not clear

```

```

Feb 14, 17 11:31      DecafParser.g4      Page

// if "x-foo()" is "- foo()" or "-foo()".
// ANTLR has a "longest-match" rule that is used in these situations.
expr: MINUS expr
    | NOT expr
    | expr (MULTIPLY | DIVISION | MODULO) expr
    | expr (ADDITION | MINUS) expr
    | expr (LESSTHAN | GREATERTHAN | LSSTHNEQTO | GRTHNEQTO) expr
    | expr (EQUAL | NOTEQUAL) expr
    | expr AND expr
    | expr OR expr
    | location
    | methodCall
    | (INTLITERAL | CHARLITERAL | BOOLEANLITERAL | STRINGLITERAL)
    | LPAREN expr RPAREN
;

```

↑  
NO!

But otherwise, excellent work.

```
class Program {
    boolean b;

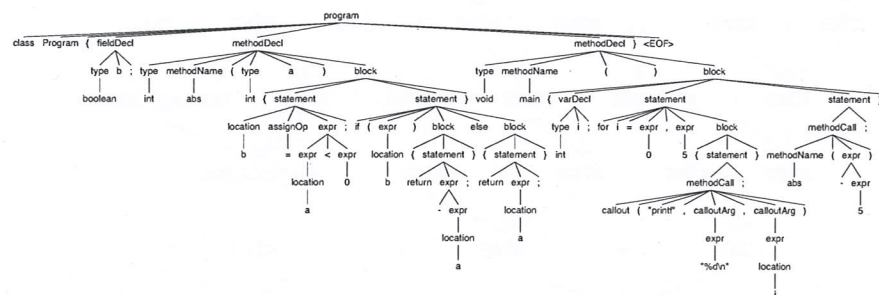
    int abs(int a) {
        b = a < 0;

        if (b) {
            return -a;
        }
        else {
            return a;
        }
    }

    void main() {
        int i;

        for i = 0, 5 {
            callout("printf", "%d\n", i);
        }

        abs(-5);
    }
}
```



```

graph TD
    program --> class
    program --> Program
    program --> methodDecl
    program --> EOF["<EOF>"]
    methodDecl --> type
    methodDecl --> methodName
    methodDecl --> parens["("]
    methodDecl --> block
    type --> void
    methodName --> main
    block --> statement
    statement --> location
    statement --> assignOp
    statement --> semicolon[";"]
    location --> c
    location --> lbrack["["]
    location --> expr1["expr"]
    location --> rbrack["]"]
    assignOp --> eq["="]
    semicolon --> plus["+"]
    semicolon --> expr2["expr"]
    semicolon --> slash["/"]
    semicolon --> expr3["expr"]
    expr1 --> methodCall
    methodCall --> methodName2["methodName"]
    methodCall --> parens2["("]
    methodCall --> expr4["expr"]
    methodCall --> parens3[")"]
    methodName2 --> md
    expr4 --> expr5["expr"]
    expr4 --> plus2["+"]
    expr4 --> expr6["expr"]
    expr5 --> location2["location"]
    location2 --> c2["c"]
    expr6 --> 3
    expr2 --> location3["location"]
    location3 --> a["a"]
    expr2 --> plus3["+"]
    plus3 --> expr7["expr"]
    plus3 --> expr8["expr"]
    expr7 --> parens4["("]
    parens4 --> expr9["expr"]
    parens4 --> rparen4[")"]
    expr9 --> minus["-"]
    expr9 --> expr10["expr"]
    expr10 --> 3
    expr8 --> 6
    expr3 --> 6
    expr3 --> star["*"]
    expr3 --> expr11["expr"]
    expr11 --> 11
    expr3 --> expr12["expr"]
    expr12 --> 6
    expr3 --> expr13["expr"]
    expr13 --> 11
    expr3 --> expr14["expr"]
    expr14 --> 11
    expr3 --> expr15["expr"]
    expr15 --> 11
    expr3 --> expr16["expr"]
    expr16 --> 11
    expr3 --> expr17["expr"]
    expr17 --> 11
    expr3 --> expr18["expr"]
    expr18 --> 11
    expr3 --> expr19["expr"]
    expr19 --> 11
    expr3 --> expr20["expr"]
    expr20 --> 11
    expr3 --> expr21["expr"]
    expr21 --> 11
    expr3 --> expr22["expr"]
    expr22 --> 11
    expr3 --> expr23["expr"]
    expr23 --> 11
    expr3 --> expr24["expr"]
    expr24 --> 11
    expr3 --> expr25["expr"]
    expr25 --> 11
    expr3 --> expr26["expr"]
    expr26 --> 11
    expr3 --> expr27["expr"]
    expr27 --> 11
    expr3 --> expr28["expr"]
    expr28 --> 11
    expr3 --> expr29["expr"]
    expr29 --> 11
    expr3 --> expr30["expr"]
    expr30 --> 11
    expr3 --> expr31["expr"]
    expr31 --> 11
    expr3 --> expr32["expr"]
    expr32 --> 11
    expr3 --> expr33["expr"]
    expr33 --> 11
    expr3 --> expr34["expr"]
    expr34 --> 11
    expr3 --> expr35["expr"]
    expr35 --> 11
    expr3 --> expr36["expr"]
    expr36 --> 11
    expr3 --> expr37["expr"]
    expr37 --> 11
    expr3 --> expr38["expr"]
    expr38 --> 11
    expr3 --> expr39["expr"]
    expr39 --> 11
    expr3 --> expr40["expr"]
    expr40 --> 11
    expr3 --> expr41["expr"]
    expr41 --> 11
    expr3 --> expr42["expr"]
    expr42 --> 11
    expr3 --> expr43["expr"]
    expr43 --> 11
    expr3 --> expr44["expr"]
    expr44 --> 11
    expr3 --> expr45["expr"]
    expr45 --> 11
    expr3 --> expr46["expr"]
    expr46 --> 11
    expr3 --> expr47["expr"]
    expr47 --> 11
    expr3 --> expr48["expr"]
    expr48 --> 11
    expr3 --> expr49["expr"]
    expr49 --> 11
    expr3 --> expr50["expr"]
    expr50 --> 11
    expr3 --> expr51["expr"]
    expr51 --> 11
    expr3 --> expr52["expr"]
    expr52 --> 11
    expr3 --> expr53["expr"]
    expr53 --> 11
    expr3 --> expr54["expr"]
    expr54 --> 11
    expr3 --> expr55["expr"]
    expr55 --> 11
    expr3 --> expr56["expr"]
    expr56 --> 11
    expr3 --> expr57["expr"]
    expr57 --> 11
    expr3 --> expr58["expr"]
    expr58 --> 11
    expr3 --> expr59["expr"]
    expr59 --> 11
    expr3 --> expr60["expr"]
    expr60 --> 11
    expr3 --> expr61["expr"]
    expr61 --> 11
    expr3 --> expr62["expr"]
    expr62 --> 11
    expr3 --> expr63["expr"]
    expr63 --> 11
    expr3 --> expr64["expr"]
    expr64 --> 11
    expr3 --> expr65["expr"]
    expr65 --> 11
    expr3 --> expr66["expr"]
    expr66 --> 11
    expr3 --> expr67["expr"]
    expr67 --> 11
    expr3 --> expr68["expr"]
    expr68 --> 11
    expr3 --> expr69["expr"]
    expr69 --> 11
    expr3 --> expr70["expr"]
    expr70 --> 11
    expr3 --> expr71["expr"]
    expr71 --> 11
    expr3 --> expr72["expr"]
    expr72 --> 11
    expr3 --> expr73["expr"]
    expr73 --> 11
    expr3 --> expr74["expr"]
    expr74 --> 11
    expr3 --> expr75["expr"]
    expr75 --> 11
    expr3 --> expr76["expr"]
    expr76 --> 11
    expr3 --> expr77["expr"]
    expr77 --> 11
    expr3 --> expr78["expr"]
    expr78 --> 11
    expr3 --> expr79["expr"]
    expr79 --> 11
    expr3 --> expr80["expr"]
    expr80 --> 11
    expr3 --> expr81["expr"]
    expr81 --> 11
    expr3 --> expr82["expr"]
    expr82 --> 11
    expr3 --> expr83["expr"]
    expr83 --> 11
    expr3 --> expr84["expr"]
    expr84 --> 11
    expr3 --> expr85["expr"]
    expr85 --> 11
    expr3 --> expr86["expr"]
    expr86 --> 11
    expr3 --> expr87["expr"]
    expr87 --> 11
    expr3 --> expr88["expr"]
    expr88 --> 11
    expr3 --> expr89["expr"]
    expr89 --> 11
    expr3 --> expr90["expr"]
    expr90 --> 11
    expr3 --> expr91["expr"]
    expr91 --> 11
    expr3 --> expr92["expr"]
    expr92 --> 11
    expr3 --> expr93["expr"]
    expr93 --> 11
    expr3 --> expr94["expr"]
    expr94 --> 11
    expr3 --> expr95["expr"]
    expr95 --> 11
    expr3 --> expr96["expr"]
    expr96 --> 11
    expr3 --> expr97["expr"]
    expr97 --> 11
    expr3 --> expr98["expr"]
    expr98 --> 11
    expr3 --> expr99["expr"]
    expr99 --> 11
    expr3 --> expr100["expr"]
    expr100 --> 11
    expr3 --> expr101["expr"]
    expr101 --> 11
    expr3 --> expr102["expr"]
    expr102 --> 11
    expr3 --> expr103["expr"]
    expr103 --> 11
    expr3 --> expr104["expr"]
    expr104 --> 11
    expr3 --> expr105["expr"]
    expr105 --> 11
    expr3 --> expr106["expr"]
    expr106 --> 11
    expr3 --> expr107["expr"]
    expr107 --> 11
    expr3 --> expr108["expr"]
    expr108 --> 11
    expr3 --> expr109["expr"]
    expr109 --> 11
    expr3 --> expr110["expr"]
    expr110 --> 11
    expr3 --> expr111["expr"]
    expr111 --> 11
    expr3 --> expr112["expr"]
    expr112 --> 11
    expr3 --> expr113["expr"]
    expr113 --> 11
    expr3 --> expr114["expr"]
    expr114 --> 11
    expr3 --> expr115["expr"]
    expr115 --> 11
    expr3 --> expr116["expr"]
    expr116 --> 11
    expr3 --> expr117["expr"]
    expr117 --> 11
    expr3 --> expr118["expr"]
    expr118 --> 11
    expr3 --> expr119["expr"]
    expr119 --> 11
    expr3 --> expr120["expr"]
    expr120 --> 11
    expr3 --> expr121["expr"]
    expr121 --> 11
    expr3 --> expr122["expr"]
    expr122 --> 11
    expr3 --> expr123["expr"]
    expr123 --> 11
    expr3 --> expr124["expr"]
    expr124 --> 11
    expr3 --> expr125["expr"]
    expr125 --> 11
    expr3 --> expr126["expr"]
    expr126 --> 11
    expr3 --> expr127["expr"]
    expr127 --> 11
    expr3 --> expr128["expr"]
    expr128 --> 11
    expr3 --> expr129["expr"]
    expr129 --> 11
    expr3 --> expr130["expr"]
    expr130 --> 11
    expr3 --> expr131["expr"]
    expr131 --> 11
    expr3 --> expr132["expr"]
    expr132 --> 11
    expr3 --> expr133["expr"]
    expr133 --> 11
    expr3 --> expr134["expr"]
    expr134 --> 11
    expr3 --> expr135["expr"]
    expr135 --> 11
    expr3 --> expr136["expr"]
    expr136 --> 11
    expr3 --> expr137["expr"]
    expr137 --> 11
    expr3 --> expr138["expr"]
    expr138 --> 11
    expr3 --> expr139["expr"]
    expr139 --> 11
    expr3 --> expr140["expr"]
    expr140 --> 11
    expr3 --> expr141["expr"]
    expr141 --> 11
    expr3 --> expr142["expr"]
    expr142 --> 11
    expr3 --> expr14
```

Well done, very few generated  
correct tree here.