实验3

> 19335019 陈泷

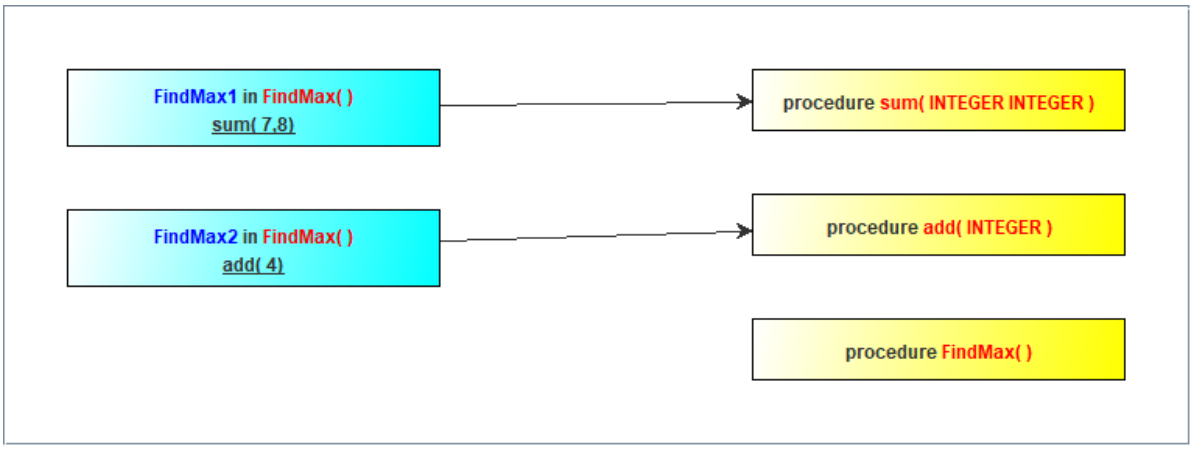
实验步骤 3.3 生成 Oberon-0 语法分析和语法制导翻译程序

根据题目要求编写相应的源程序，并以此为输入运行JavaCUP，运行命令见gen.bat，运行结果如下：

```
D:\实验5\19335019陈泷\ex3\src>java -jar ../javacup/java-cup-11b.jar oberon.cup
------- CUP v0.11b 20160615 (GIT 4ac7450) Parser Generation Summary -------
  0 errors and 0 warnings
  50 terminals, 39 non-terminals, and 113 productions declared,
  producing 200 unique parse states.
  0 terminals declared but not used.
  0 non-terminals declared but not used.
  0 productions never reduced.
  0 conflicts detected (0 expected).
  Code written to "parser.java", and "sym.java".
--------------------------------------------------- (CUP v0.11b 20160615 (GIT 4ac7450))
```

修改 oberon.flex 的返回值，编写 Main.java 文件，之后编译相关文件，运行命令见 build.bat。之后再简单修改 ex1 中的 accout.obr ，增加函数调用(因为原来的源代码函数调用较少)，之后在 accout.obr 和各类错误文件上运行源程序，运行命令分别见 run.bat 和 test.bat。结果如下：

accout.obr 的调用图：



各类错误文件的执行结果：

```
D:\ [redacted] \实验5\19335019陈泷\ex3\bin>java -classpath ../bin Main ../src/testcases/account.00*
..\src\testcases\account.001:Error position : Line 17  Column 3exceptions.IllegalSymbolException: LexicalException :
Illegal Symbol.
Illegal Symbol.

..\src\testcases\account.002:Error position : Line 35  Column 13callgraph.TargetNotFoundException: Target procedure does
 not exist when adding a new edge.

..\src\testcases\account.003:Error position : Line 17  Column 7exceptions.IllegalOctalException: LexicalException :
Illegal Octal number.
Illegal Octal number.

..\src\testcases\account.004:Syntax error at character 17 of input
instead expected token classes are []
Couldn't repair and continue parse at character 17 of input
Error position : Line 18  Column 18java.lang.Exception: Can't recover from previous error(s)

..\src\testcases\account.005:Syntax error at character 17 of input
instead expected token classes are []
Couldn't repair and continue parse at character 17 of input
Error position : Line 18  Column 15java.lang.Exception: Can't recover from previous error(s)

..\src\testcases\account.006:Error position : Line 24  Column 25exceptions.TypeMismatchedException: Semantic Exception:
Type Mismatched Exception.

..\src\testcases\account.007:Error position : Line 16  Column 7exceptions.IllegalIntegerRangeException: LexicalException
 :
Illegal IntegerRange: more than 12.
Illegal IntegerRange: more than 12.

..\src\testcases\account.008:Error position : Line 12  Column 12exceptions.IllegalIdentifierLengthException: LexicalExce
ption :
Illegal Identifier Length: more than 24.
Illegal Identifier Length: more than 24.
```

各个文件对应的错误类型：

1. Illegal Symbol Exception：识别单词时遇到不合法的输入符号(譬如@、 $等符号)则抛出该异常。
   eg.`sal$ary: INTEGER`
2. Parameter Mismatched Exception：当调用的函数的实际参数数目与过程声明的形式参数数目不一致时则抛出 该异常。
   line 32和33之间添加：

```
PROCEDURE call();
    BEGIN
        Findmax(5);
    END call;
```

3. Illegal Octal Exception：当 0 开头的整数常量中含有 0~7 之外的符号(包括 8 和 9或其他字母)时抛出该异常。
   eg.`i := 08;`
4. Missing Right Parenthesis Exception：当左右圆括号不匹配、且缺少右括号时则抛出该异常。
   eg.`if (accounts[i].salary >= x THEN`
5. Missing Right Parenthesis Exception：当左右圆括号不匹配、且缺少左括号时则抛出该异常。
   eg.`if accounts[i].salary >= x) THEN`
6. Type Mismatched Exception：当表达式、赋值语句、或参数传递等构造中出现类型不兼容错误时则抛出该异常。
   eg.`x := ~x`
7. 含有不合法的常量：常量中数字的个数超出限制
   eg.`i := 10000000000000000000000000000001;`
8. 标识符长度不允许超过 24 个字符
   eg.`VAR x, n, izxcvbnmasdfghjklqwert: INTEGER;`

实验步骤 3.4、 讨论不同生成工具的差异

3.4.1 JavaCUP 和 GNU Bison的差异主要有以下几个方面：

1. 使用的词法分析工具不同：JavaCUP 使用 jflex，GNU Bison使用 lex。
2. 优先级定义的语法规则不同：JavaCUP的语法规则是 `precedence right/left non-terminal-name`，
   GNU Bison 的语法规则是 `%left/right non-terminal-name` 。
3. 表达式相加的语义规则不同：JavaCUP中的语法规则是 `RESULT = new Integer (e1.intValue() + e2.intValue())` ，GNU Bison中的语法规则是 `expression: expression'+' expression { $$ = $1 + $3; }` 。
4. 返回结果的语法规则不同：JavaCUP的结果通过result返回，GNU Bison的结果通过$返回。

3.4.2 JavaCC 与 JavaCUP的核心区别

JavaCC 与 JavaCUP的核心区别：JavaCC采用的是TOP-DOWN的方式进行分析，而JavaCUP采用的是BOTTOM-UP的方式进行分析。由此带来了语法规则等多个方面的差异。

实验心得与体会

实验三内容相比实验一和二更多。通过实验三，我熟悉了 JavaCUP 的使用，对于自底向上的语法分析过程以及调用图的生成有了更深入的理解与思考。另外，通过与JavaCC、GNU Bison的比较，也加深了对这几类工具的认识与辩证性思考。总之，在这一过程中有一定的收获。