LAB3 ROSE

ex1

21307099 李英骏

目录

1 正确的 Oberon-0 源程序 **2**

2 错误程序说明 **3**

2.1 词法错误 (Lexical Error) 3

2.2 语法错误 (Syntax Error) 3

2.3 语义错误 (Semantic Error) 3

3 Oberon-0 语言特点 **4**

3.1 保留字与关键字 4

3.2 表达式语法规则特点 4

4 Oberon-0 的文法二义性 **5**

1 正确的 Oberon-0 源程序

```
MODULE calculate;
    (* 计算阶乘 *)
    PROCEDURE Factorial;
    VAR
        res: INTEGER; N: INTEGER;
    BEGIN
        Read(N);
        res := 1;
        IF N = 0 THEN res := 1 END;
        WHILE N > 1 DO
            res := N * res;
            N := N - 1
        END;
        Write(res); WriteLn
    END Factorial;
    (* 计算数组元素之和 *)
    PROCEDURE CalculateArraySum;
    VAR
        arr: ARRAY 5 OF INTEGER;
        sum: INTEGER; n: INTEGER; k: INTEGER;
    BEGIN
        (* 初始化数组 *)
        arr[0] := 1; arr[1] := 2; arr[2] := 3;
        arr[3] := 4; arr[4] := 5;
        sum := 0; n := 0;
        WHILE n < 5 DO
            sum := sum + arr[n];
            n := n + 1;
        END;
        Write(sum); WriteLn
    END CalculateArraySum;
    PROCEDURE call;
    BEGIN
        Factorial();
        CalculateArraySum();
    END call;
END calculate.
```

2 错误程序说明

提供了如下的错误程序, 详情见错例中的注释, 均已写明.

2.1 词法错误 (Lexical Error)

- IllegalOctalException (001)
- IllegalIntegerRangeException (002)
- IllegalIdentifierLengthException (003)
- IllegalSymbolException (004)

2.2 语法错误 (Syntax Error)

- MissingOperatorException (005)
- MissingOperandException (006)
- MissingRightParenthesisException (007)
- MissingLeftParenthesisException (008)

2.3 语义错误 (Semantic Error)

- VariableNotDeclaredException (009)
- TypeMismatchedException (010)

3 Oberon-0 语言特点

3.1 保留字与关键字

以下内容摘自 *Wikipedia*:

In a computer language, a reserved word (also known as a reserved identifier) is a word that cannot be used as an identifier, such as the name of a variable, function, or label –it is "reserved from use". This is a syntactic definition, and a reserved word may have no user-defined meaning.

A closely related and often conflated notion is a keyword, which is a word with special meaning in a particular context. This is a semantic definition. By contrast, names in a standard library but not built into a language are not considered reserved words or keywords. The terms "reserved word" and "keyword" are often used interchangeably –one may say that a reserved word is "reserved for use as a keyword" –and formal use varies from language to language. For this article, we distinguish as above.

也就是说，保留字是一个语法上的定义. 它指的是是不能用作标识符的词汇 (即不能用作变量名、函数名或标签名), 比如文档中提及的 IF、THEN、ELSIF, 它们都是用来表示条件控制结构的保留字, 在代码中有特定的语法功能, 不能用作变量名或函数名.

而关键字是一个语义上的定义 (尽管它在文档中看上去是词法定义), 它是在特定上下文中具有特殊意义的词汇, 正因如此, 它通常表示数据类型、标准库函数或其他内置功能. 例如, 'INTEGER' 表示一种数据类型,'WRITE' 和 'WRITELN' 表示标准输出函数. Wiki 中指出, 在某些语言中关键字可以用作标识符或重新定义, 在 Oberon-0 中这似乎是不支持的.

3.2 表达式语法规则特点

- Oberon-0 仅支持布尔(BOOLEAN)和整数(INTEGER)两种数据类型,而 C/C++,JAVA 还支持字符类型和类型转换, 同时 Oberon-0 不支持指针类型。
- Oberon-0 具有复杂数据类型，包括数组类型 (ARRAY) 和记录类型 (RECORD)。
- 在程序结构方面，Oberon-0 不支持 FOR 和 DO...WHILE 循环类型，而 C/C++ 支持. Oberon-0 使用 END 语句作为语句段的结束 (类似 Pascal Fortran 和 Matlab 等语言)，而 C/C++ 使用大括号来划分程序结构。
- 在函数和过程方面，C++ 支持函数的返回值，而 Oberon-0 通常通过按引用传递参数来返回数据

- Oberon-0 的源程序可以不含有主过程，而 C/C++ 通常需要一个 ‘main‘ 函数

4 Oberon-0 的文法二义性

Oberon-0 文法定义没有二义性.

一般的一些高级语言中，常见的二义性问题有以下几种:

1. **if-else 悬挂问题 (Dangling Else Problem)**: 这是最常见的二义性问题之一，主要出现在条件语句嵌套中，特别是当没有明确的块结构来划分 else 语句时，容易造成解析上的歧义。

```
if (condition1)
    if (condition2)
        statement1;
else
    statement2; // 这个 else 是匹配哪个 if ?
```

2. **空语句问题 (Empty Statement Problem)**: 在某些语言中，空语句可能导致解析上的歧义。

```
if (condition);  // 这里的分号表示一个空语句
    statement;
```

这里的 ‘statement‘ 是总是执行的还是仅在 ‘condition‘ 为真时执行，可能会存在歧义。

3. **块结构问题 (Block Structure Problem)**: 当块结构不明确或没有块结构时，语句的范围可能导致歧义。

```
if (condition)
    statement1;
    statement2;
```

这里的 ‘statement2‘ 是不是 ‘if‘ 语句的一部分，可能存在歧义。

在 Oberon-0 中, 通过以下的语法定义避免了以上三个问题:

```
if_statement          =  "IF" expression "THEN"
                            statement_sequence
                         {"ELSIF" expression "THEN"
                            statement_sequence}
                         ["ELSE"
                            statement_sequence]
                         "END" ;
```

图 1: IF THEN ELSIF

可以看出在每一个 IF 语句后面结束的时候都要使用 END, 或者使用 ELSIF, 这样就是一一匹配的了.

4. **表达式优先级问题（Expression Precedence Problem）**：当表达式中包含多个运算符时，不同运算符的优先级可能导致解析上的歧义。

```
a + b * c
```

5. **运算符结合性问题（Operator Associativity Problem）**：结合性问题也会导致解析上的歧义，特别是当同一优先级的运算符连用时。

```
a - b - c
```

```
simple_expression     =  ["+" | "-"] term {("+" | "-" | "OR") term} ;

term                  =  factor {("*" | "DIV" | "MOD" | "&")
factor} ;
```

图 2: Enter Caption

Oberon-0 用上图中的语法定义避免了上面两个问题 (隐含了优先级和结合性).(还有一些运算符定义没截图, 同理)

6. **类型推断问题（Type Inference Problem）**：在一些支持类型推断的语言中，变量的类型可能会导致歧义。

```
var x = y + z;
```

这里的 ‘x‘ 的类型取决于 ‘y‘ 和 ‘z‘ 的类型，如果类型推断规则不明确，可能导致解析上的歧义. 这个问题显然不存在, 因为 Oberon-0 语言不支持类型推断.

4 OBERON-0 的文法二义性