2023 年秋编译原理上机实验

实验一

以课程主页上的供参考的 PL/0 实验（2017 版）为基础，扩展 PL/0 语言，添加 **指针和数组**，完成相关的**词法、语法、语义检查、代码生成及解释执行等编译实现（包含但不限于）**：

1 指针和数组的变量声明；
2 指针变量赋值和访问；
3 数组元素读写的一般形式；
4 数组元素读写的指针化表示；
5 输出语句 print。
… …

示例 1（文本形式仅供参考）：

```
var m;   // 普通变量
var *p;   // 一级指针变量
var **q;  // 二级指针变量
var ***r;      // 三级指针变量
var * a[10] ;  // 一维数组，元素为一级指针
begin
p := &m;     // 取普通变量地址并赋值给一级指针变量
q := &p;     // 取一级指针变量地址并赋值给二级指针变量
r := &q;           // 取二级指针变量地址并赋值给三级指针变量

***r := 100; // 访问三级指针变量
a[0] := &m; // 给一维数组元素赋值
a[1] := p;
print(* a[0], *a[1], m); // 输出三个 100
end.
```

示例 2（文本形式仅供参考）：

```
var brr[20][20][20];          // 三维数组，元素为普通变量
begin
brr[1][2][0] := 99;      // 三维数组元素赋值的一般形式
print( *(*(*(brr + 1) + 2)) ) ; // 三维数组元素访问的指针化表示； 输出 99
end.
```

实验二

以课程主页上的供参考的 PL/0 实验（2017 版）为基础，扩展 PL/0 语言，添加类似于 C++ 语言的**作用域算符 ::** 用来访问在过程（procedure）嵌套声明中的外层过程所声明的同名变量。完成相关的**词法、语法、语义检查、代码生成及解释执行等编译实现（包含但不限于），能处理类似于示例 3 中的程序文本并得到相应输出结果。**

示例 3（文本形式仅供参考）：

```
var i;   // 主程序 main 中声明的变量 i
procedure p1; //嵌套声明在主程序 main 中
    var i ;   //子程序 p1 中声明的局部变量 i
    procedure p2; //嵌套声明在过程 p1 中
        var i;     // 子程序 p2 中声明的局部变量 i
        procedure p3; //嵌套声明在过程 p2 中
            var i;       // 子程序 p3 中声明的局部变量 i
        begin//子程序 p3 的过程体
            i := 3;
            print( ::i, p1::i, p2::i, i ); // 输出 0 1 2 3
            print( ::i, ::p1::i, ::p1::p2::i, ::p1::p2::p3::i ); // 输出 0 1 2 3
        end;
    begin // 子程序 p2 的过程体
        i := 2;
        call p3;
    end;
begin //子程序 p1 的过程体
    i := 1;
    call p2;
end;
begin // 主程序 main 的过程体
    i := 0;
    call p1;
end.
```

过程 p3 中两个 print 语句中待输出变量的说明：
::i 表示最外层主程序 main 中声明的 i
p1::i 表示外层过程 p1 中声明的 i
p2::i 表示外层过程 p2 中声明的 i
i 表示当前过程 p3 中声明的局部变量 i
::p1:i 表示 最外层声明的过程 p1 中的局部变量 i
::p1::p2::i 表示 最外层声明的过程 p1 中所声明的过程 p2 里声明的局部变量 i
::p1::p2::p3::i 表示 最外层声明的过程 p1 中所声明的过程 p2 里声明的过程 p3 所声明的局部变量 i