```c
// 文件: semantics.c
// 内容: 语义计算所需的辅助程序（与人机界面无关）
// 作者: WXQ#2018

#include <stdlib.h>
#include <math.h> // cos(), sin(), ...

#include "semantics.h"
#include "ui.h"

// 解释器内部的数据：绘图参数
double
    Origin_x =0, Origin_y =0,   // 横、纵平移距离，缺省不平移
    Scale_x  =1, Scale_y  =1,   // 横、纵比例因子，缺省不缩放
    rot_angle=0;                // 旋转角度，缺省不旋转


// 计算表达式的值
double GetExprValue(ExprNode_Ptr root)
{
    if (root == NULL) return 0.0;
    switch (root -> OpCode)
    {
    case PLUS  :
        return GetExprValue(root->content.CaseOperator.left ) + GetExprValue(root->content.CaseOperator.right) ;
    case MINUS :
        return GetExprValue(root->content.CaseOperator.left ) - GetExprValue(root->content.CaseOperator.right) ;
    case MUL   :
        return GetExprValue(root->content.CaseOperator.left ) * GetExprValue(root->content.CaseOperator.right) ;
    case DIV   :
        return GetExprValue(root->content.CaseOperator.left ) / GetExprValue(root->content.CaseOperator.right) ;
    case POWER :
        return pow(GetExprValue(root->content.CaseOperator.left ), GetExprValue(root->content.CaseOperator.right) );
    case FUNC  :  // 调用指定的函数，如 cos(x)
        return (* (root->content.CaseFunc.MathFuncPtr)) (GetExprValue(root->content.CaseFunc.child) );
    case CONST_ID :
        return root->content.CaseConst ;
    case T  :
        return *(root->content.CaseParmPtr);
    default :
        return 0.0 ;
    }
}

// 计算被绘制点的坐标
void CalcCoord(ExprNode_Ptr x_tree,
               ExprNode_Ptr y_tree,
               double * ptr_x_value,
               double * ptr_y_value)
{
    double x_val, x_temp, y_val;

    // 计算表达式的值，得到点的原始坐标
    x_val = GetExprValue(x_tree);
    y_val = GetExprValue(y_tree);

    // 比例变换
```

```
58      x_val *= Scale_x ;
59      y_val *= Scale_y ;
60
61      // 旋转变换
62      x_temp = x_val*cos(rot_angle) + y_val*sin(rot_angle);
63      y_val  = y_val*cos(rot_angle) - x_val*sin(rot_angle);
64      x_val  = x_temp;
65
66      // 平移变换
67      x_val += Origin_x;
68      y_val += Origin_y;
69
70      // 返回变换后点的坐标
71      if(NULL != ptr_x_value) *ptr_x_value = x_val;
72      if(NULL != ptr_y_value) *ptr_y_value = y_val;
73  }
74
75
76
77  void setOrigin(ExprNode_Ptr x_tree, ExprNode_Ptr y_tree)
78  {
79      Origin_x = GetExprValue(x_tree);  // 根据语法树计算横坐标的平移距离
80      Origin_y = GetExprValue(y_tree);  // 根据语法树计算纵坐标的平移距离
81  }
82
83  void setScale(ExprNode_Ptr x_tree, ExprNode_Ptr y_tree)
84  {
85      Scale_x = GetExprValue(x_tree);   // 根据语法树计算横坐标的比例因子
86      Scale_y = GetExprValue(y_tree);   // 根据语法树计算纵坐标的比例因子
87  }
88
89  void setRotate(ExprNode_Ptr tree)
90  {
91      rot_angle = GetExprValue(tree);  // 根据语法树计算旋转角度
92  }
93
94
95  //  循环绘制点坐标
96  void DrawLoop(  ExprNode_Ptr   start_tree,
97                  ExprNode_Ptr   end_tree,
98                  ExprNode_Ptr   step_tree,
99                  ExprNode_Ptr   x_tree,
100                 ExprNode_Ptr   y_tree)
101 {
102     double x_val, y_val;
103     double start_val, end_val, step_val;        // 绘图起点、终点、步长
104     double * p_T_value = getTmemory() ; // parser manager it
105
106     start_val = GetExprValue(start_tree);   // 计算起点表达式的值
107     end_val   = GetExprValue(end_tree);     // 计算终点表达式的值
108     step_val  = GetExprValue(step_tree);    // 计算步长表达式的值
109
110     for(*p_T_value = start_val; *p_T_value <= end_val; *p_T_value += step_val)
111     {
112         CalcCoord(x_tree, y_tree, &x_val, &y_val);
113 //      y_val = 460-y_val;            // 修改显示屏的直角坐标系
114         DrawPixel((unsigned long)x_val, (unsigned long)y_val, 0); // in ui.h
115     }
116 }
117
```