静态结构分析（上）

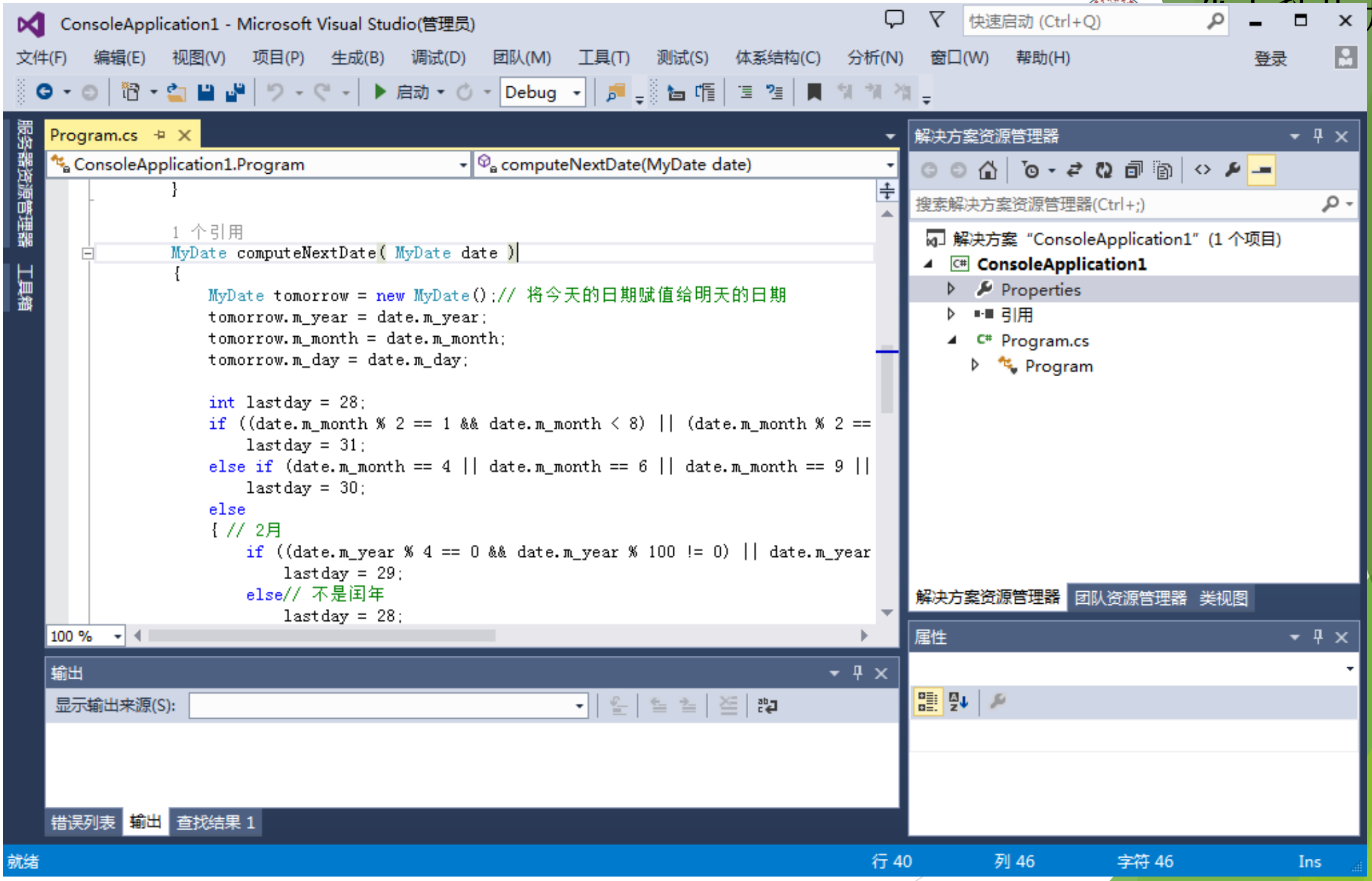
C:\Users\Administrator\Desktop\new 2.cpp - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

new 2.cpp

```
40      MyDate computeNextDate( MyDate date )
41      {
42          MyDate tomorrow = new MyDate();// 将今天的日期赋值给明天的日期
43          tomorrow.m_year = date.m_year;
44          tomorrow.m_month = date.m_month;
45          tomorrow.m_day = date.m_day;
46
47          int lastday = 28;
48          if ((date.m_month % 2 == 1 && date.m_month < 8) || (date.m_month % 2 == 0 && date.m_month >= 8)
49              lastday = 31;
50          else if (date.m_month == 4 || date.m_month == 6 || date.m_month == 9 || date.m_month == 11)
51              lastday = 30;
52          else
53          { // 2月
54              if ((date.m_year % 4 == 0 && date.m_year % 100 != 0) || date.m_year % 400 == 0)// 是闰年
55                  lastday = 29;
56              else// 不是闰年
57                  lastday = 28;
58          }
59
60          // 如果日期是本月最后一天，则下一天是下个月第一天，否则就是普通日期
61          if (date.m_day == lastday)// 月末的日期
62          {
63              tomorrow.m_day = 1;
64              if (date.m_month == 12)// 年末的日期
65              {
66                  tomorrow.m_month = 1;
```

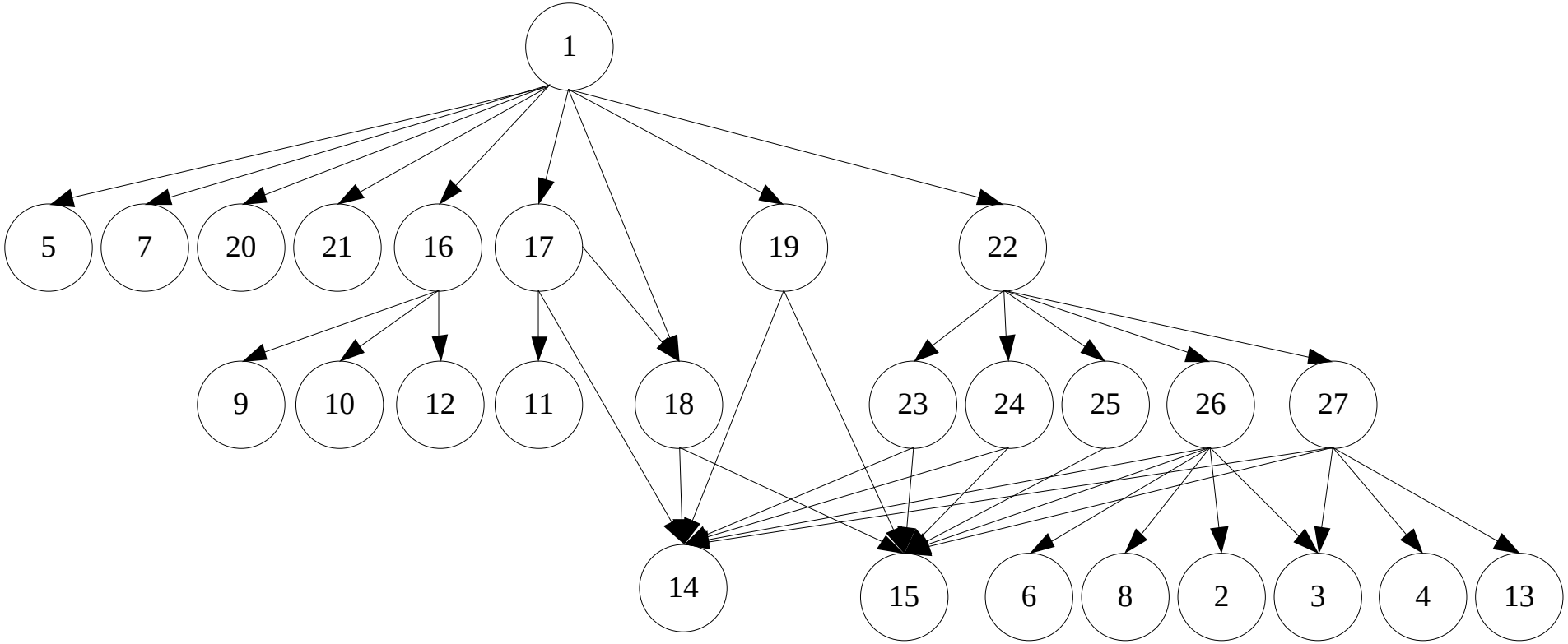
C++ source file | length : 3085 lines : 96 | Ln : 90 Col : 32 Sel : 10 | Dos\Windows | ANSI | INS


ConsoleApplication1 - Microsoft Visual Studio(管理员)
快速启动 (Ctrl+Q)
文件(F) 编辑(E) 视图(V) 项目(P) 生成(B) 调试(D) 团队(M) 工具(T) 测试(S) 体系结构(C) 分析(N) 窗口(W) 帮助(H) 登录
启动 Debug
Program.cs
ConsoleApplication1.Program
computeNextDate(MyDate date)
}
1 个引用
MyDate computeNextDate(MyDate date)
{
MyDate tomorrow = new MyDate();// 将今天的日期赋值给明天的日期
tomorrow.m_year = date.m_year;
tomorrow.m_month = date.m_month;
tomorrow.m_day = date.m_day;
int lastday = 28;
if ((date.m_month % 2 == 1 && date.m_month < 8) || (date.m_month % 2 ==
lastday = 31;
else if (date.m_month == 4 || date.m_month == 6 || date.m_month == 9 ||
lastday = 30;
else
{ // 2月
if ((date.m_year % 4 == 0 && date.m_year % 100 != 0) || date.m_year
lastday = 29;
else// 不是闰年
lastday = 28;
100 %
解决方案资源管理器
搜索解决方案资源管理器(Ctrl+;)
解决方案 "ConsoleApplication1" (1 个项目)
ConsoleApplication1
Properties
引用
Program.cs
Program
解决方案资源管理器 团队资源管理器 类视图
属性
输出
显示输出来源(S):
错误列表 输出 查找结果 1
就绪
行 40 列 46 字符 46 Ins

静态结构分析

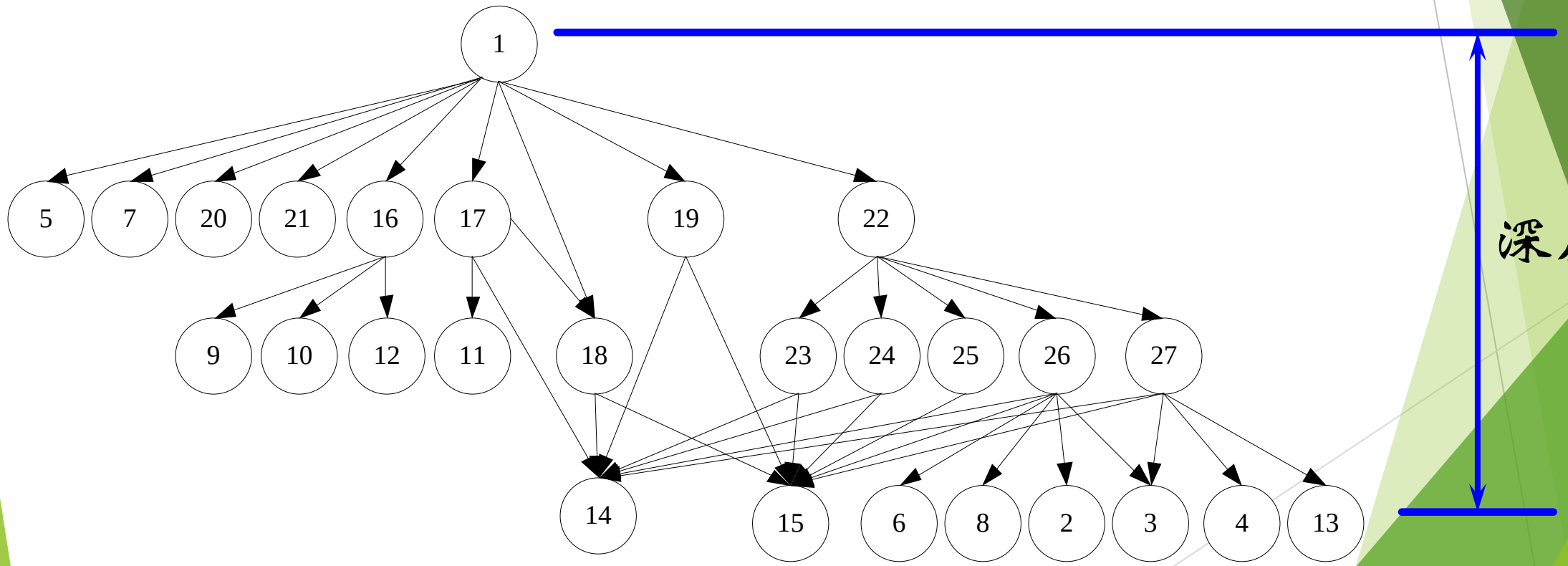
- 通过引入多种形式的图表（如函数调用关系图、模块控制流图等），帮助我们快速了解程序设计和结构，更好地理解源代码，有利于找到程序设计的缺陷和代码优化的方向。

函数调用关系图



1. 函数调用层次

- 层次太深，增大集成测试负担



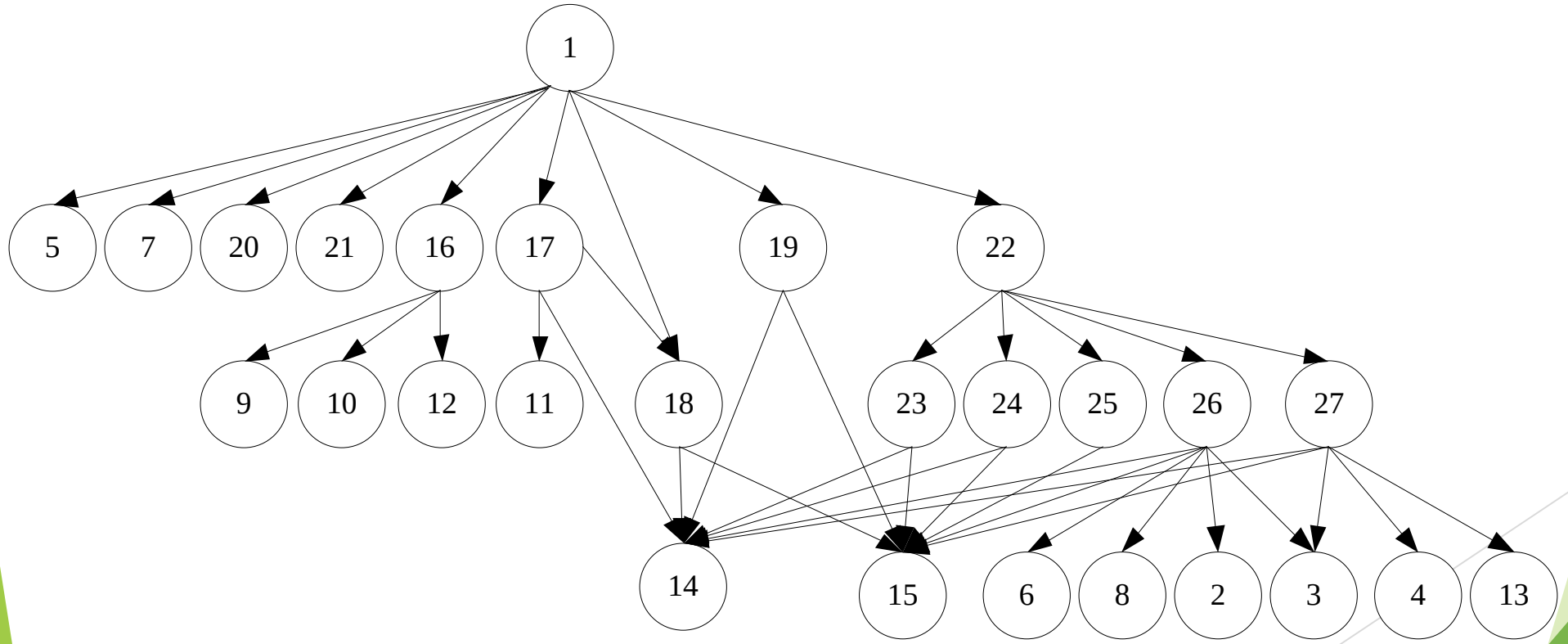
1. 函数调用层次

- 对栈造成压力，容易导致溢出

一个典型的栈帧

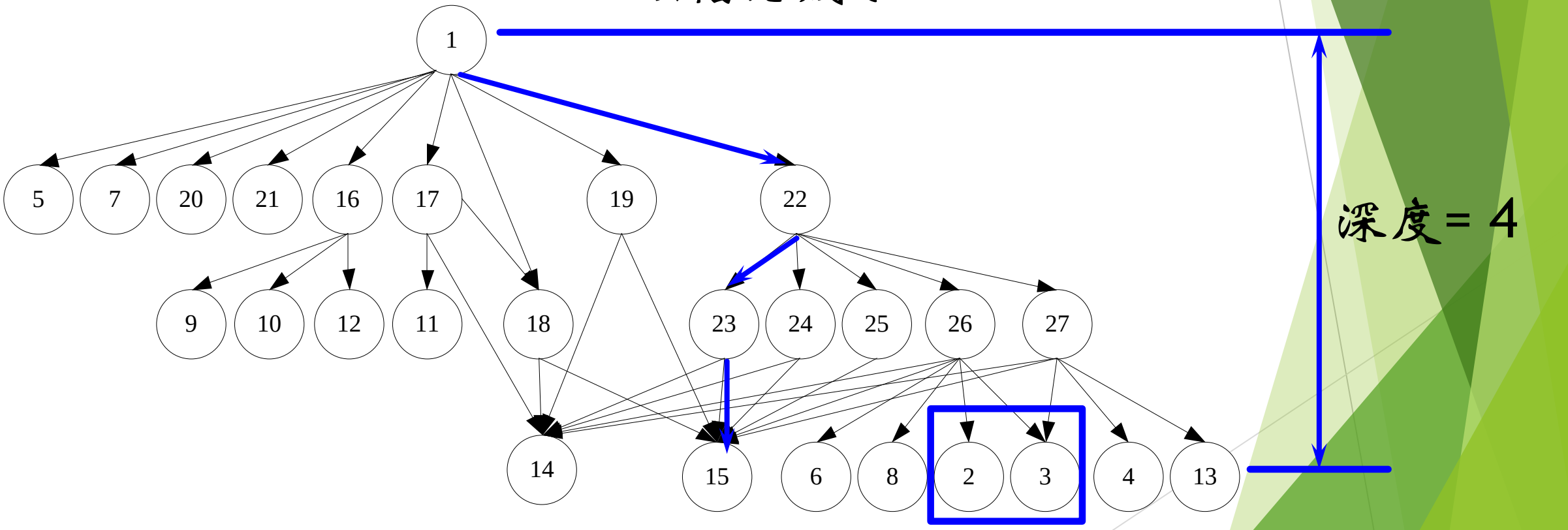
1. 函数调用层次

- 控制单个函数的复杂度



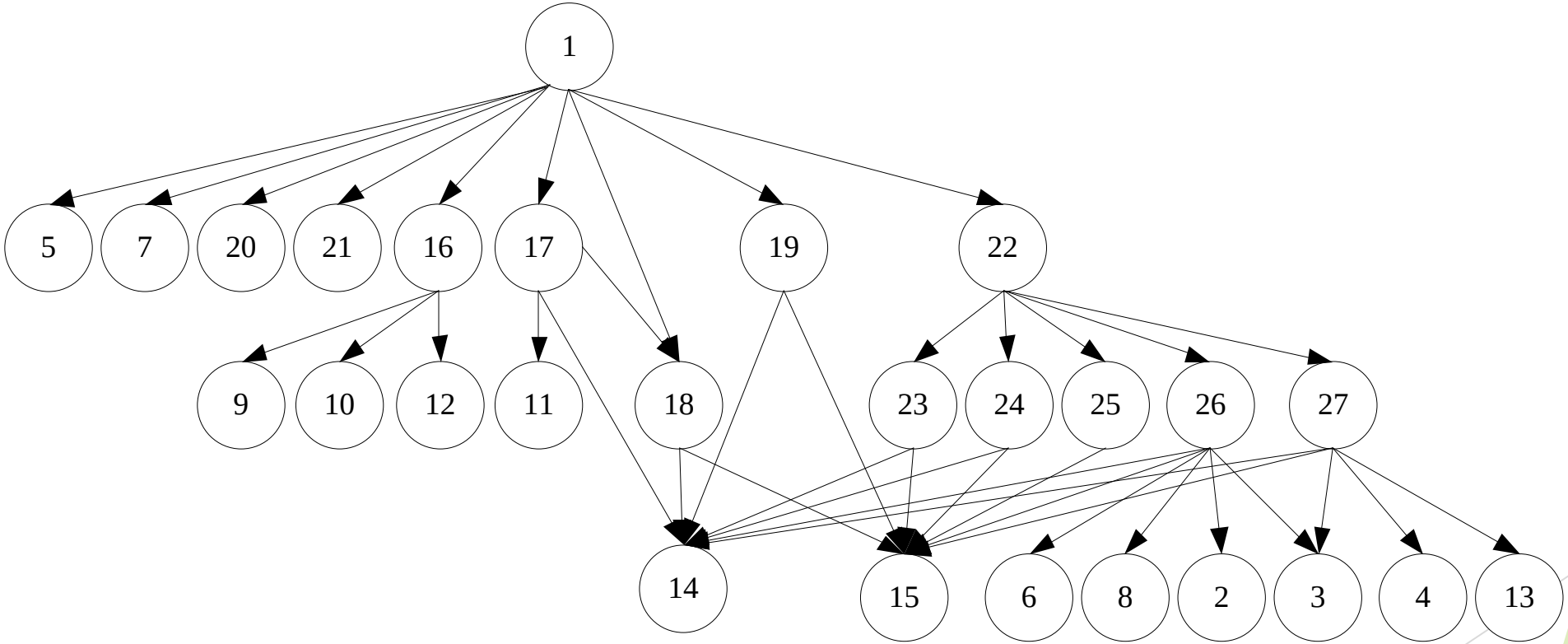
1. 函数调用层次

- 高风险节点
- 调用层次深
- 缺陷隐藏深



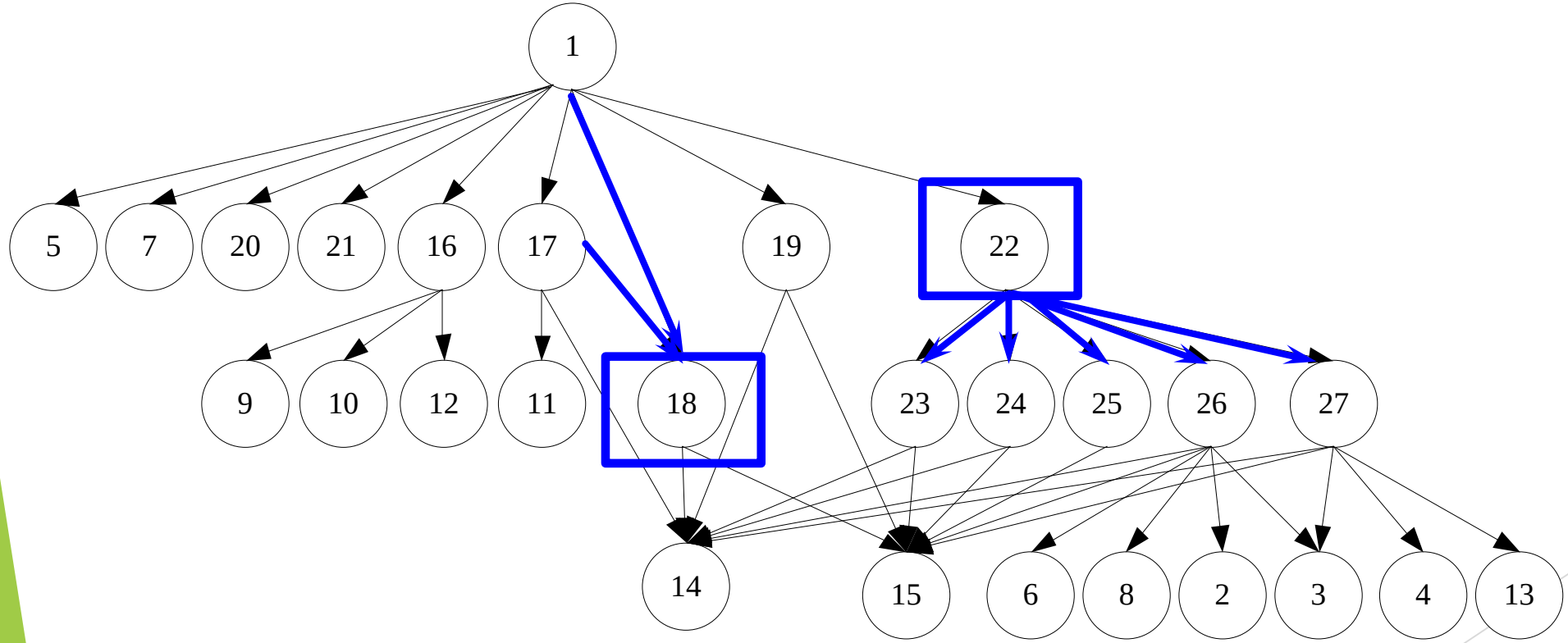
1. 函数调用层次

- 高风险节点
- 根节点

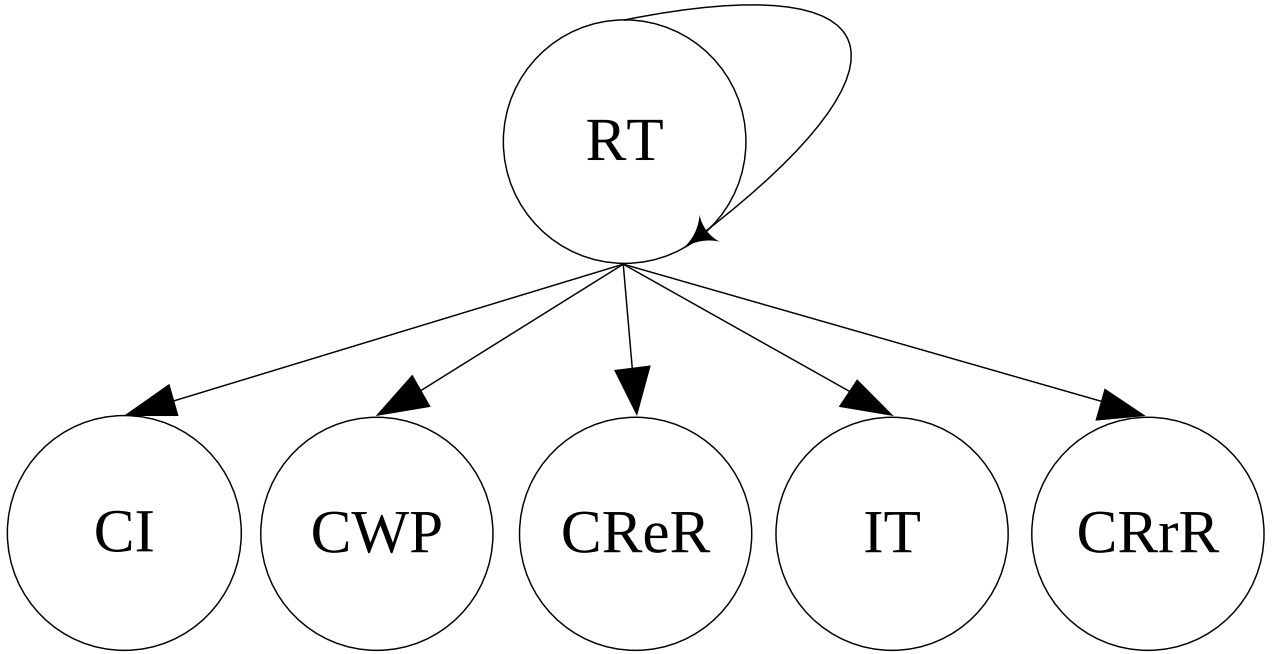


2. 调用关系

- 高风险节点
  - 入度越大，缺陷传播速度越快
  - 出度越大，对缺陷的敏感度越高



3. 递归调用



```
Color RayTracing ( Ray ray, int depth, Color background){// 光线跟踪算法
    Color curColor;
    Point P = CalcIntersection(ray, objects);// 计算光线ray与场景中物体表面最近的交点P
    if( P != NULL ){// 如果存在交点
        Color Ic = CalcIcWithPhong(P);//用Phong模型计算P点的Ic，即光源直接照射引起的反射光亮度
        curColor = Ic;// 将颜色设置为Ic的值
        if( depth < depthThreshold ){// 如果深度小于给定的最大跟踪深度
          Ray reflectionRay = CalcReflectionRay(ray); // 计算ray的反射光线
          Is = RayTracing(reflectionRay, depth+1, background);//将反射光线当做一条新光线，从交点发出
        if( IsTransparent(object) ){// 如果当前具有交点的物体是透明的
            Ray refractionRay = CalcRefractionRay(ray); // 计算ray的折射光线；
            It = RayTracing(refractionRay, depth+1, background);//将折射光线当做一条新的光线，从交点
发出
        }
        curColor = Ic + Is + It;// 根据Whitted整体光亮度模型，设置当前位置的光亮度
        }
    }
    else{
        curColor = backgroud;// 设置当前颜色为背景色
    }
    return curColor;
}
```

4. 孤立节点

- 孤立的函数意味着不执行的场景或路径，代表编码或设计的不合理，应尽量避免。