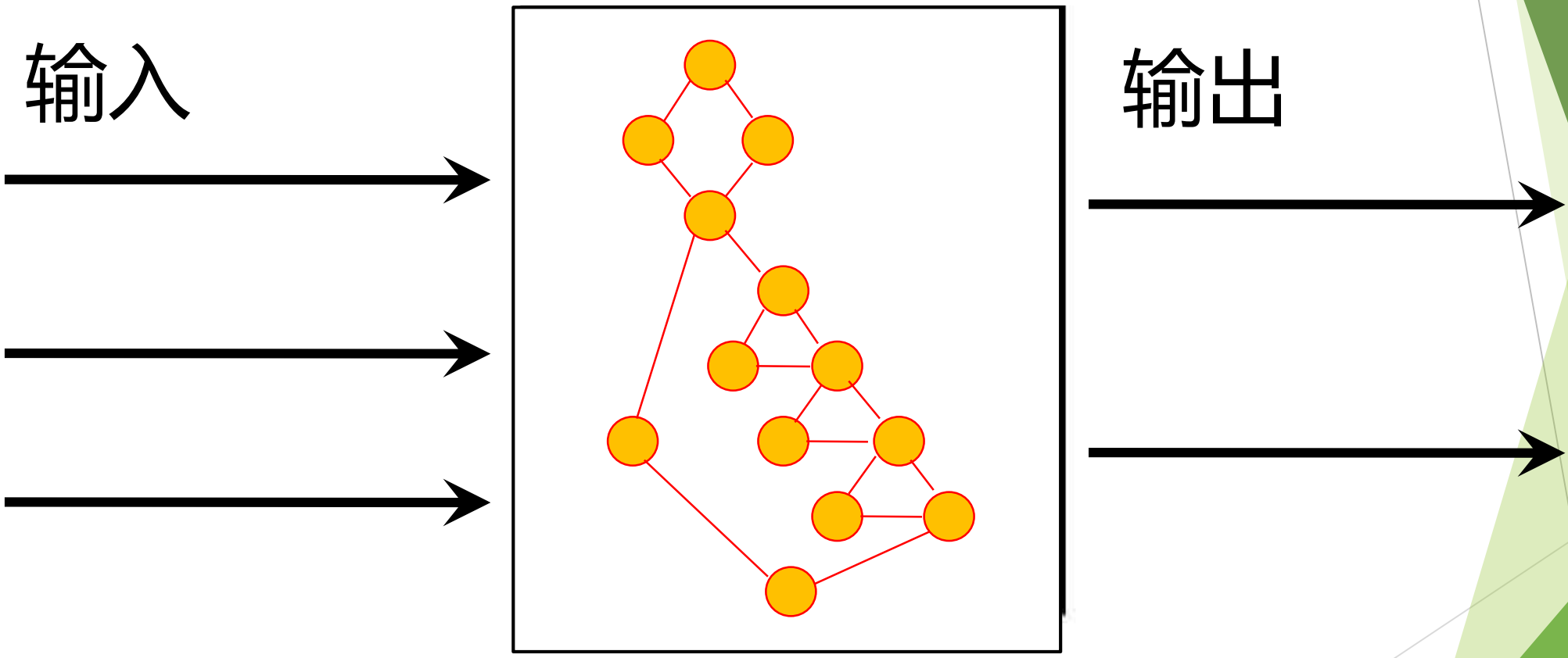
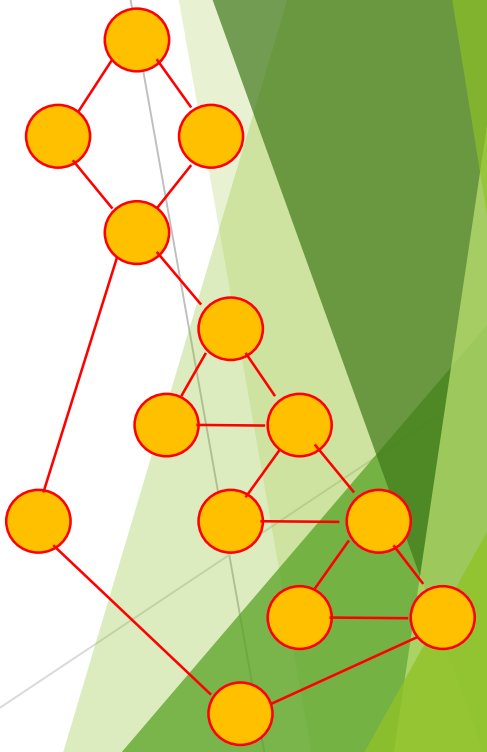
控制流分析技术

白盒测试的基本原理



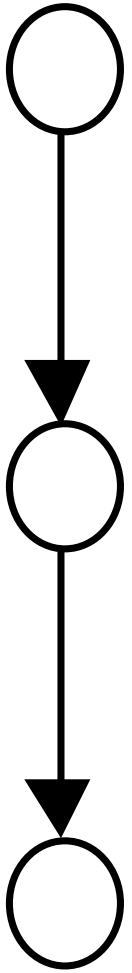
控制流分析要解决的问题

- 什么因素导致程序结构变得复杂？
- 如何衡量程序结构的复杂程度？
- 控制程序执行流程发生变化的主要因素是什么？
- 如何测试这些因素，并确保测试的效率？



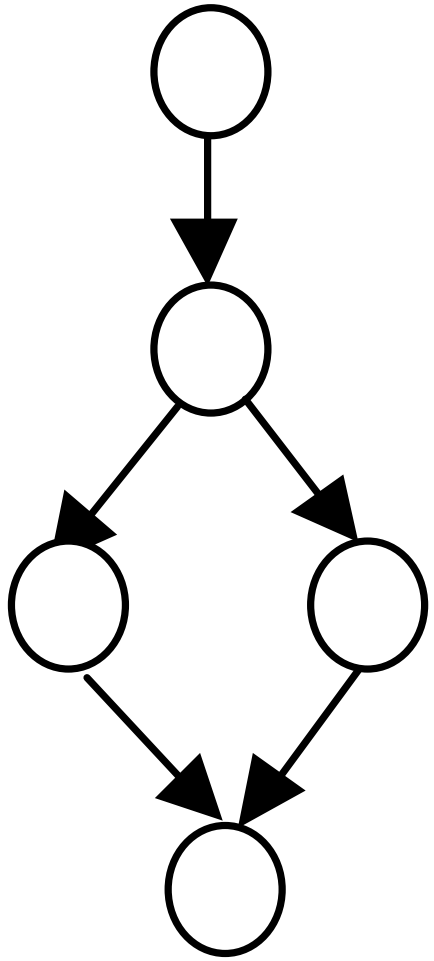
线性结构

```
void Func1( int a )
{
    int b;
    b = a + 1;
    printf( “a = %d, b = %d\n”, a, b );
}
```



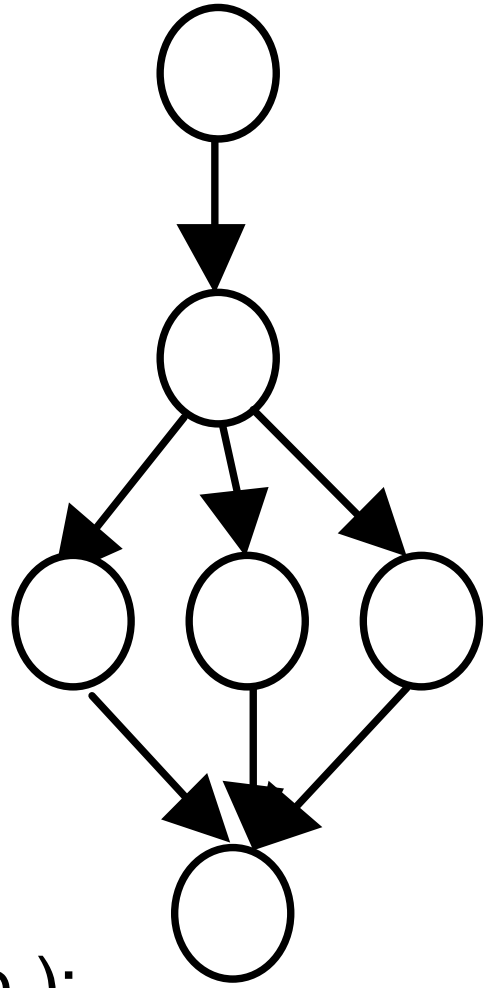
条件判定结构

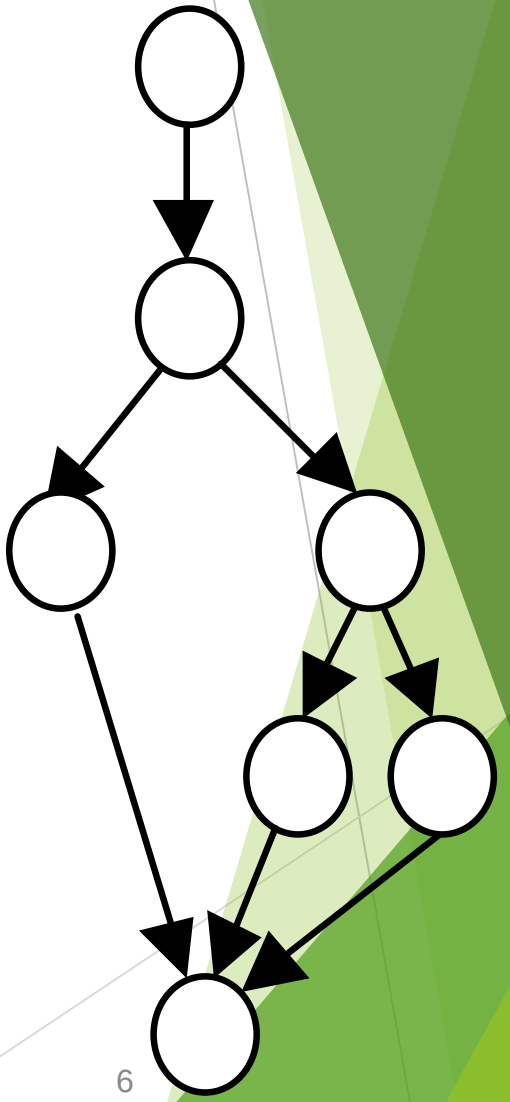
```
void Func2( int a )
{
    int b = 0;
    if( a > 1 )
       b = a + 1;
    else
       b = a – 1;
    printf( “a = %d, b = %d\n”, a, b );
}
```



条件判定结构

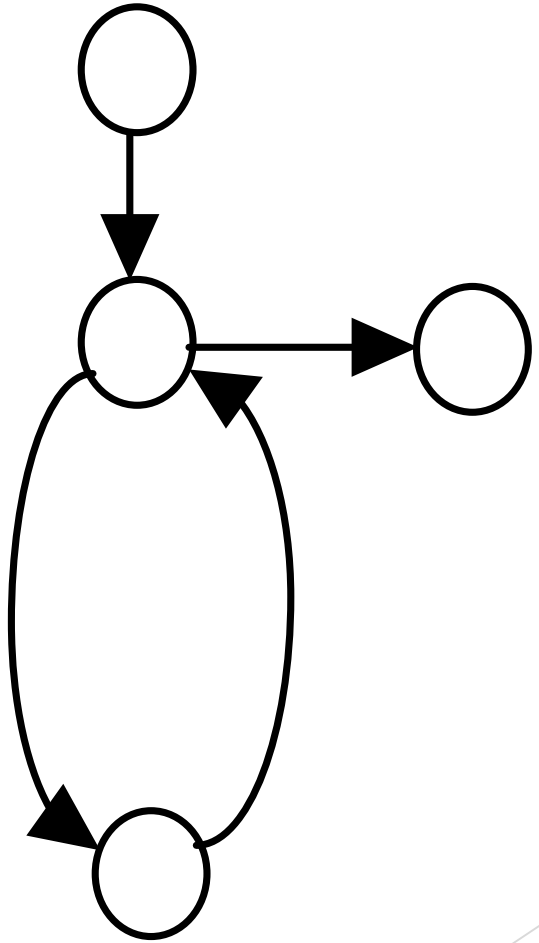
```
void Func3( int a )
{
    int b = 0;
    switch( a ){
        case 0: b = a; break;
        case 1: b = a * 2; break;
        case 2: b = a * 3; break;
        default: break;
    }
    printf( “a = %d, b = %d\n”, a, b );
}
```





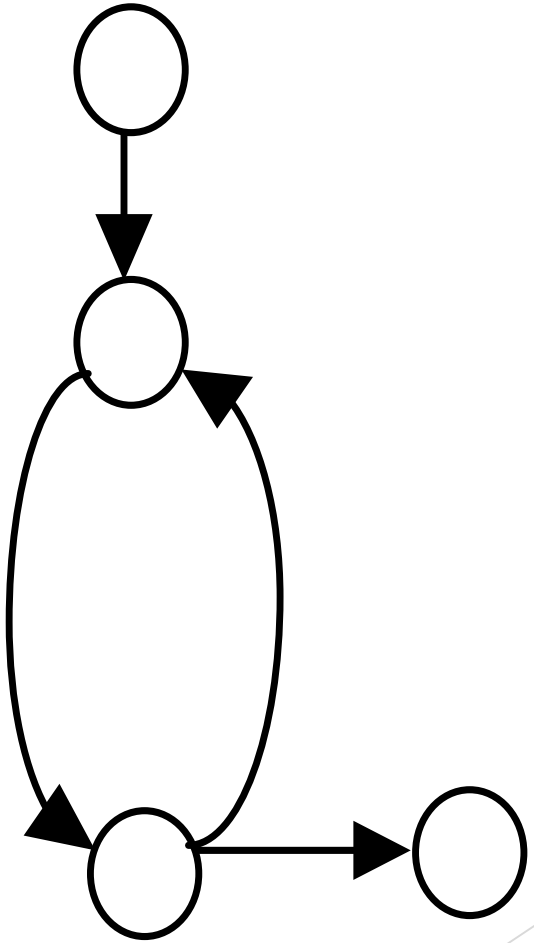
While-do循环结构

```
void Func4( int a )
{
    int b = 0;
    int i = 1;
    while( i < 10 ){
        b = b + a*i;
        i ++;
    }
    printf( “a = %d, b = %d\n”, a, b );
}
```

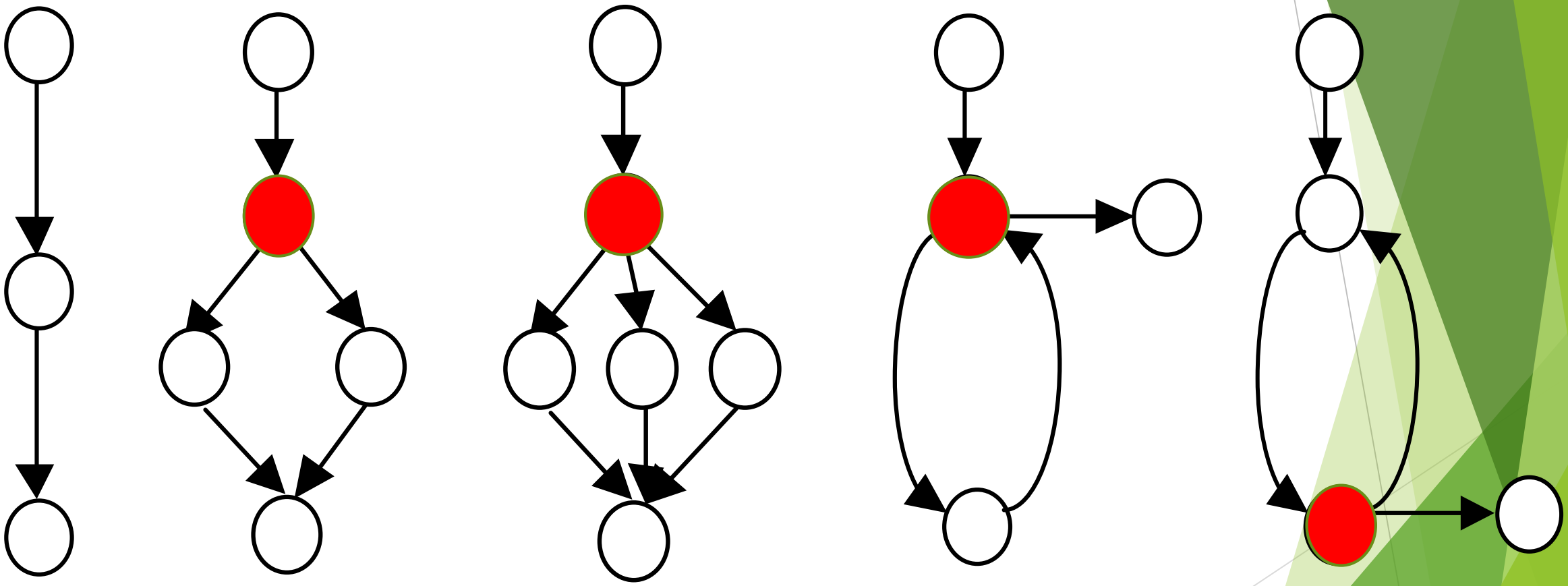


Do-while循环结构

```
void Func5( int a )
{
    int b = 0;
    int i = 1;
    do{
       b = b + b*i;
       i ++;
    }while( i < 10 );
    printf( “a = %d, b = %d\n”, a, b );
}
```

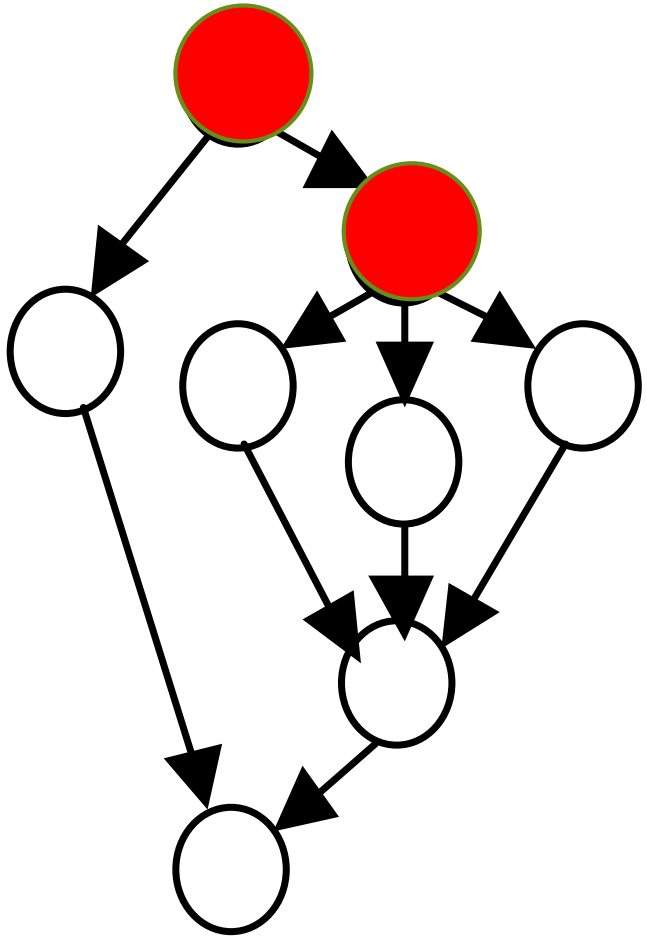


常见的程序结构：判定节点导致结构的复杂

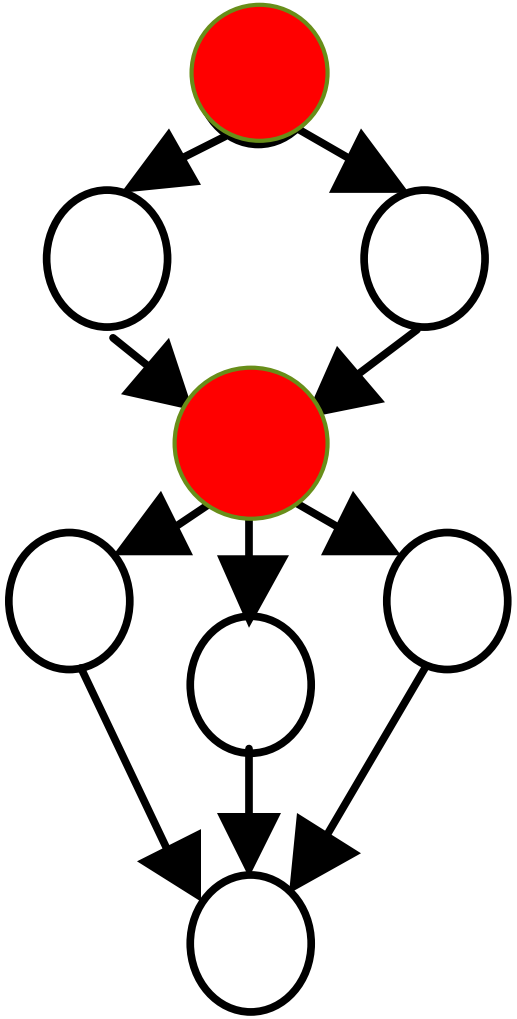


风险：小 → 大

嵌套

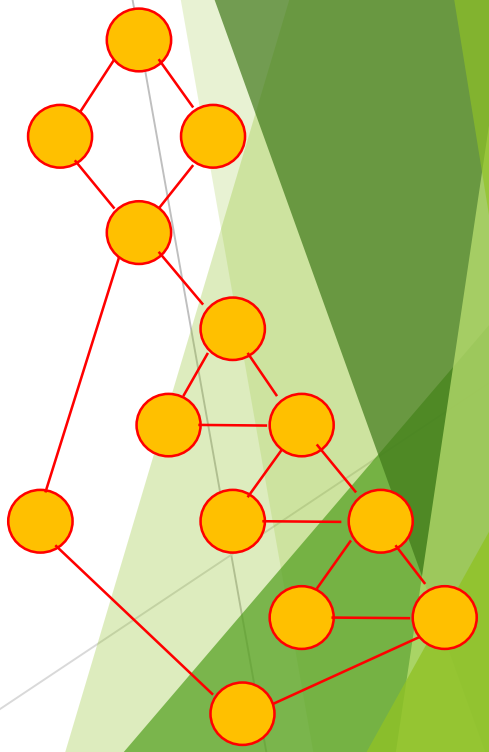


串联



控制流分析要解决的问题

- 什么因素导致程序结构变得复杂？
  - 判定节点
- 控制程序执行流程发生变化的主要因素是什么？
  - 判定节点
- 如何衡量程序结构的复杂程度？
- 如何测试这些因素，并确保测试的效率？



控制流分析的内容

- 关注判定节点固有的复杂性
  - 焦点：判定表达式
  - 方法：逻辑覆盖测试
- 关注判定结构与循环结构对执行路径产生的影响
  - 焦点：路径
  - 方法：独立路径测试
- 关注循环结构本身的复杂性
  - 焦点：循环体
  - 方法：基于数据的静态分析