控制流图（Control Flow Graph）是以有向图表示程序逻辑结构的一种形式，能够直观展示程序的分支、跳转、复杂度等特性，是编译优化、程序分析、软件测试等领域的一种基本程序中间表示。

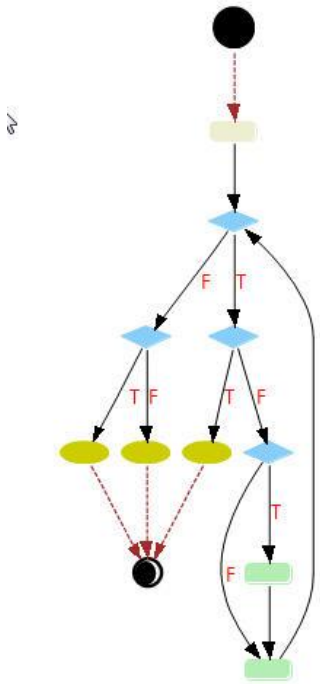
圈复杂度是用于评价函数复杂度的一种指标，一般认为圈复杂度越高，说明函数越复杂、越难以维护或测试。圈复杂度的一种计算方法是：

圈复杂度 = 控制流图中分支节点个数+1

分支节点是指含 if，else if，for，while，switch 语句的节点，即在控制流图中有多个后继节点的节点。

下图所示为一个 C 语言函数 func 和其对应的控制流图。

源代码	对应的控制流图

int function(int from) {
int i;
int count = 0;
for (i = from; i < 100; i++) {
if (i < 0) {
return 0;
}
if (i % 7 == 0) {
count += i;
}
}
if (count % 2 == 0) {
return 1;
}
return 2;
}
	

理解上述概念，编写一个函数 int calculateMcCabe(GraphNode root)：给定函数 func 的控制流图的根节点 root，计算其对应的圈复杂度。

参考数据结构如下，可自行补充成员：

JAVA

```
class GraphNode{
    String data; //节点数据
    Collection<GraphNode> nextList; //后继节点列表
}
```

C 语言

```
typedef struct Node{
    unsigned int data;   //节点数据
    unsigned int nextSize;//后继节点数量
    struct Node **       next; //保存后继节点指针的数组
} *GraphNode;
```