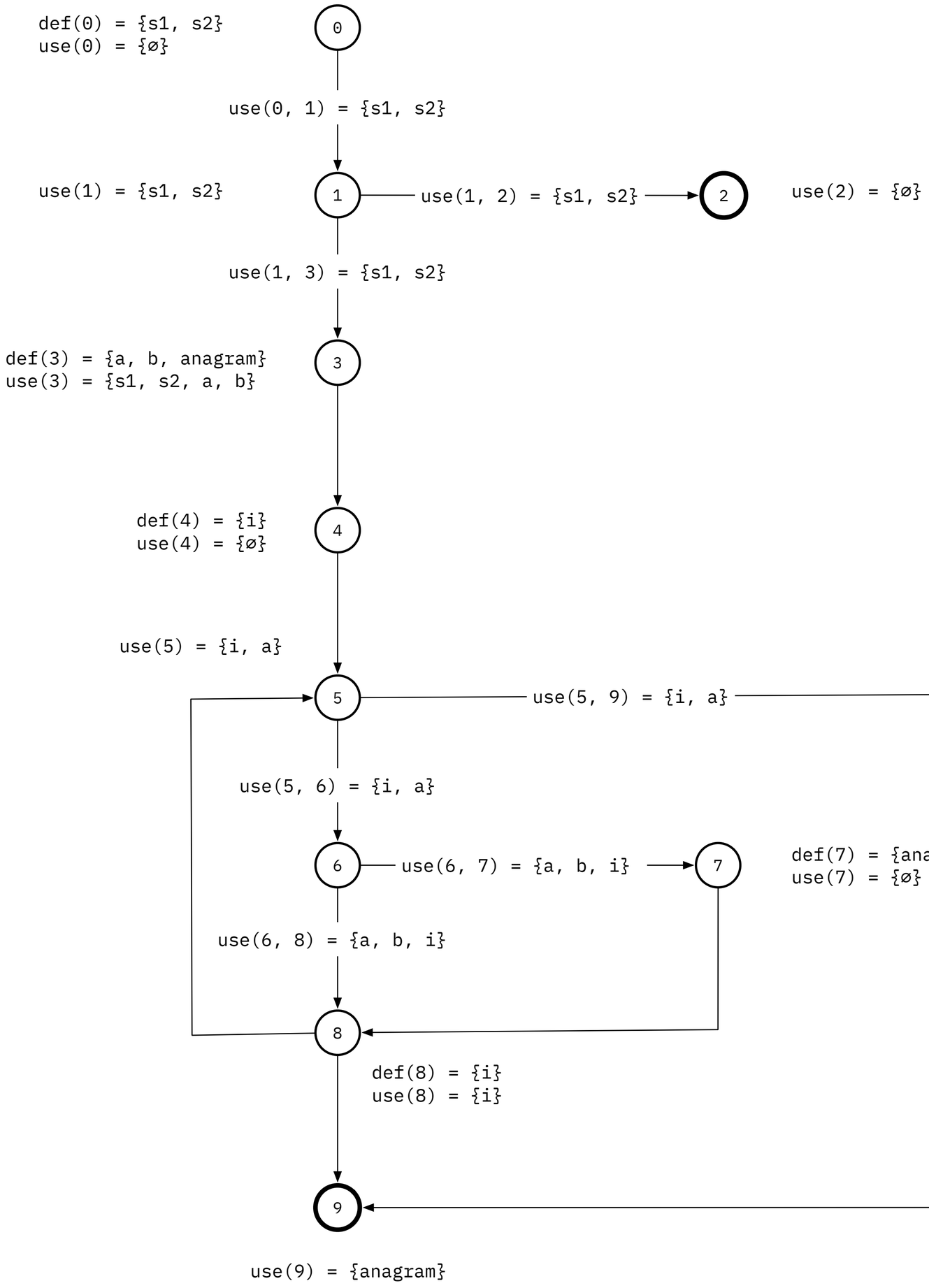
```
8  @  0  static boolean isAnagram(String s1, String s2) {
9        1   if (s1.length() != s2.length()) {
10       2       return false;
11           }
12
13           char[] a = s1.toLowerCase().toCharArray();
14           char[] b = s2.toLowerCase().toCharArray();
15       3   boolean anagram = true;
16           Arrays.sort(a);
17           Arrays.sort(b);
18                 4          5          8
19           for(int i = 0; i < a.length; i++) {
20       6       if(a[i] != b[i]) {
21       7           anagram = false;
22               }
23           }
24       9   return anagram;
25   }
```

Data Flow Diagram



CFG 如图所示。

程序定义的所有变量有：`s1`，`s2`，`a`，`b`，`i`，`anagram`。

对其分别构造 DU Path 如下：

- du(0, 3, s1) = {(0, 1, 3)} -- Path A
- du(0, 2, s2) = {(0, 1, 2)} -- Path B
- du(3, 6, a) = {(3, 4, 5, 6)} -- Path C
- du(3, 6, b) = {(3, 4, 5, 6)} -- Path D
- du(4, 9, i) = {(4, 5, 6, 8, 9)} -- Path E
- du(4, 9, anagram) = {(4, 5, 6, 7, 8, 9)} -- Path F

为了满足 ADC，需要分别对每个 du path 构造测试用例。于是构造如下：

输入	**预期输出**	**覆盖 Path**
s1 = "abc", s2 = "ac"	false	B
s1 = "abc", s2 = "abd"	false	A, C, D, E, F

对每个定义节点的 DU Path，该测试样例均能够覆盖，于是满足了 ADC。