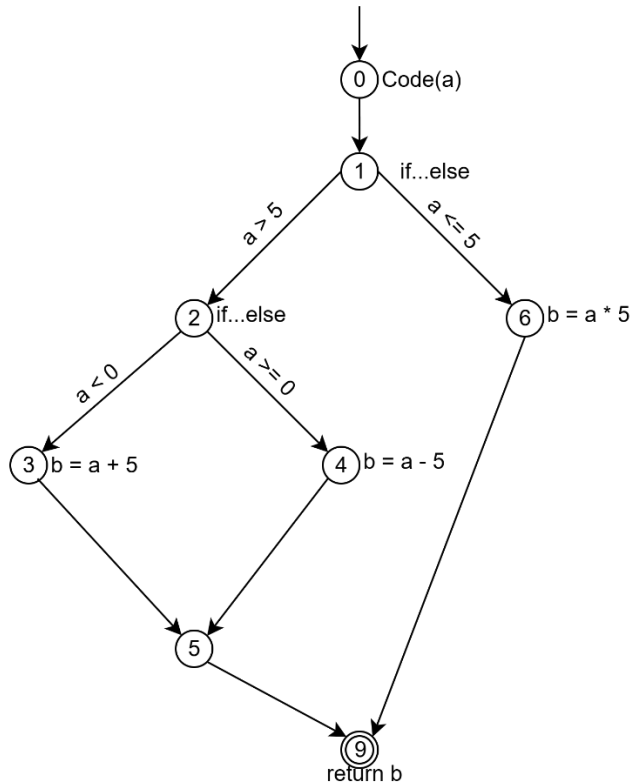


TIÊU CHÍ BAO PHỦ MÃ NGUỒN CODE COVERAGE

$$\text{Tỷ lệ bao phủ} = \frac{\text{Số thành phần được kiểm tra}}{\text{Tổng số thành phần có trong mã nguồn}}$$

Ví dụ: Với Data test $a = 10$ thực thi lộ trình 012457



Có tất cả 6 lệnh: if...else, if...else, $b = a + 5$, $b = a - 5$, $b = a * 5$, return b.

1. Tỷ lệ bao phủ lệnh:

Số thành phần được kiểm tra: if...else, if...else, $b = a - 5$, return b

➔ Tỷ lệ bao phủ lệnh = $4/6$

2. Tỷ lệ bao phủ rẽ nhánh:

Số đỉnh quyết định: 2 (đỉnh 1, đỉnh 2)

Số nhánh: 4 (12, 16, 23, 24)

Với lộ trình 012457 thì chỉ đi qua nhánh 12 và 24

➔ Tỷ lệ bao phủ rẽ nhánh = $2/4$

3. Tỷ lệ bao phủ điều kiện:

Các điều kiện: $a > 5$, $a \leq 5$, $a < 0$ và $a \geq 0$

Với lộ trình 012457 thì chỉ đi qua $a > 5$ và $a \geq 0$

→ Tỷ lệ bao phủ điều kiện = $2/4$

CÁC TIÊU CHÍ BAO PHỦ MÃ NGUỒN

I. Một số tiêu chí bao phủ mã nguồn:

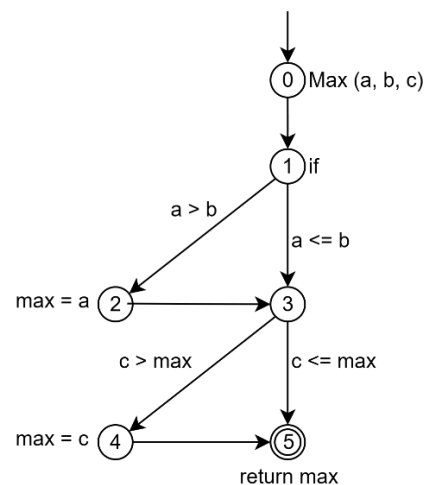
- Tiêu chí bao phủ đỉnh – Node/Statement Coverage
- Tiêu chí bao phủ nhánh – Decision/Edge/Branch Coverage
- Tiêu chí bao phủ điều kiện – Condition Coverage
- Tiêu chí bao phủ điều kiện và rẽ nhánh – Decision/Condition Coverage
- Tiêu chí bao phủ đa điều kiện – Multiple Condition Coverage
- Tiêu chí bao phủ lộ trình – Path Coverage

II. Tiêu chí bao phủ đỉnh – Node/Statement Coverage

Thiết kế Testcase sao cho đảm bảo các Node được thực hiện ít nhất 1 lần (Chọn lộ trình khả thi tối thiểu bao phủ tất cả các Node trong đồ thị)

Ví dụ 1: Hàm tìm max

```
int Max (int a, int b, int c){  
    int max;  
    if(a > b)  
        max = a;  
    if(c > max)  
        max = c;  
    return max;  
}
```



Paths = {012345, 0135, 01345, 01235}

Theo tiêu chí 1: lộ trình 012345 là lộ trình khả thi tối thiểu bao phủ được các đỉnh trong CFG

Giải thích:

Lộ trình 012345	Data test
$a > b$ $\text{max} = a$	$a = 5$ $b = 2$

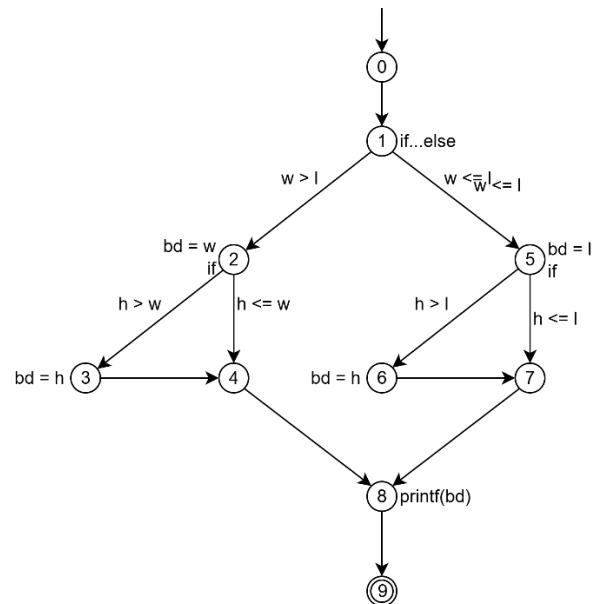
$c > \max$ $\max = c$ $\text{return } \max$ $\rightarrow c > a > b$	$c = 10$
Lộ trình khả thi	

Ví dụ 2:

```

if(w > l){
    bd = w;
    if(h > w)
        bd = h;
}
else{
    bd = l;
    if(h > l)
        bd = h;
}
printf("%d", bd);

```



Paths = {0123489, 012489, 0156789, 015789}

Theo tiêu chí 1: lộ trình 0123489 và 0156789 là lộ trình khả thi tối thiểu bao phủ được các đỉnh trong CFG

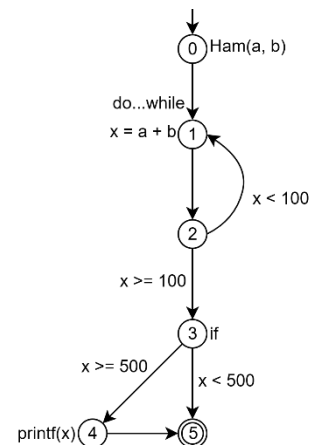
Giải thích:

Lộ trình 0123489	Data test
$w > l$ $bd = w$ $h > w$ $bd = h$ $\text{printf}(bd)$ $\rightarrow h > w > l$	$h = 10$ $w = 5$ $l = 3$
Lộ trình khả thi	
Lộ trình 0156789	Data test
$w \leq l$ $bd = l$	$h = 20$ $l = 5$

$h > l$ $bd = h$ $\text{printf}(bd)$ $\rightarrow h > l \geq w$	$w = 5$
Lộ trình khả thi	

Ví dụ 3:

```
void Ham (int a, int b){
    do
        x = a + b;
    while (x < 100);
    if(x >= 500)
        printf("%d", x);
}
```



Paths = {012345, 01235,...}

Theo tiêu chí 1: lộ trình 012345 là lộ trình khả thi tối thiểu bao phủ được các đỉnh trong CFG

Giải thích:

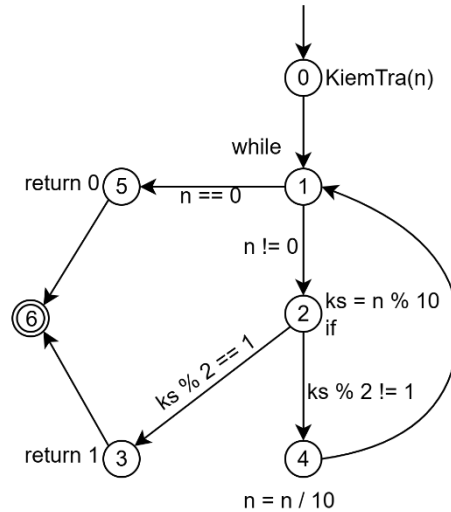
Lộ trình 012345	Data test
$x = a + b$ $x \geq 100$ $x \geq 500$ $\text{printf}(x)$ $\rightarrow x \geq 500$	$a = 100$ $b = 400$
Lộ trình khả thi	

Ví dụ 4:

```

char KiemTra (int n){
    char ks;
    while(n != 0){
        ks = n % 10;
        if(ks % 2 == 1)
            return 1;
        n = n / 10;
    }
    return 0;
}

```



Paths = {01236, 0(124)*156,...}

Theo tiêu chí 1: lộ trình 01236 và 0124156 là lộ trình khả thi tối thiểu bao phủ được các đỉnh trong CFG

Giải thích:

Lộ trình 01236	Data test
n != 0 ks = n % 10 ks % 2 == 1 return 1	n = 5 ks = 5 % 10 = 5 5 % 2 == 1
Lộ trình khả thi	
Lộ trình 0124156	Data test
n != 0 ks = n % 10 ks % 2 != 1 n == 0 return 0	n = 2 ks = 2 % 10 = 2 2 % 2 != 1 n = 2 / 10 = 0 0 == 0
Lộ trình khả thi	