Kiểm chứng phần mềm

White-box testing

Kiểm thự hộp trắng

- Structural/Clear box/Glass box testing
- Thiết kế các trường hợp kiểm thử dựa vào cấu trúc của thủ tục để suy dẫn các trường hợp cần kiểm thử
- Nguyên tắc
  - Thực hiện mọi đường dẫn độc lập ít nhất một lần
  - Thực hiện mọi điều kiện logic trên True/False
  - Thực hiện mọi vòng lặp tại các biên và trong phạm vi hoạt động
  - Thực hiện mọi cấu trúc dữ liệu bên trong để đảm bảo tính hợp lệ

Kiểm thự hộp trắng

- 2 hướng tiếp cận
  - Kiểm thử đường dẫn cơ sở (Basic path testing)
  - Kiểm thử cấu trúc điều kiển (Control structure testing)

Kiểm thử đường dẫn cơ sở

- Đảm bảo tất cả đường dẫn độc lập (independent path) điều được kiểm thử
- Đường dẫn độc lập là đường dẫn đi từ đầu đến cuối chương trình mà không chứa đường dẫn độc lập khác
- Tập đường dẫn độc lập → tập cơ sở (basic set)

Kiểm thử đường dẫn cơ sở

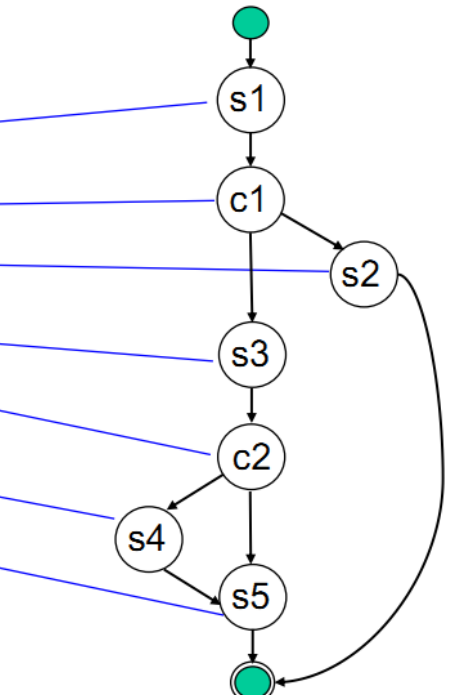
- Các bước thực hiện
  - Bước 1: Vẽ đồ thị lưu trình (flowgraph)
  - Bước 2: Xác định độ phức tạp Cyclomat của đồ thị lưu trình
  - Bước 3: Xác định tập cơ sở các đường dẫn độc lập
  - Bước 4: Thiết kế test case cho mỗi đường dẫn độc lập

Kiểm thử đường dẫn cơ sở

- Bước 1: vẽ đồ thị lưu trình

- Thí dụ :

```
1   float foo(int a, int b, int c, int d) {
2     float e;
3     if (a==0)
4       return 0;
5     int x = 0;
6     if ((a==b) || ((c==d) && bug(a)))
7       x = 1;
8     e = 1/x;
9     return e;
10  }
```



Kiểm thử đường dẫn cơ sở

- Bước 2: Xác định độ phức tạp cyclomat → cho biết số lượng đường dẫn độc lập
  - V(G) = R(số vùng) = 3
  - V(G) = P(số đỉnh điều kiện)+1 = 2+1 = 3
  - V(G) = E(số cạnh)–N(số đỉnh)+2 = 10-9+2 = 3

Kiểm thử đường dẫn cơ sở

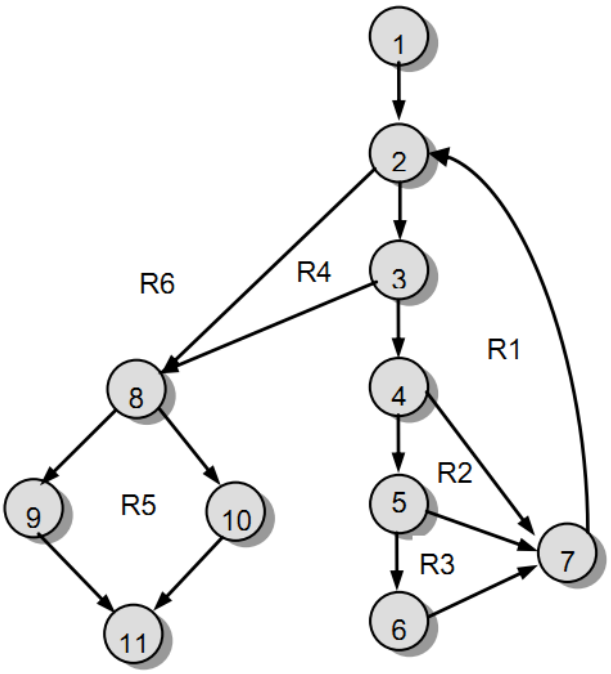
- Bước 3: tìm tập cơ sở các đường dẫn độc lập
  - Tìm 1 đường dẫn từ đầu đến cuối chương trình
  - Tìm đường dẫn mới có đi qua một cạnh mới mà không trùng với các đường dẫn trước đó
  - Làm cho đến khi đủ số lượng đường dẫn
- Ví dụ:
  - Đường dẫn 1: S1→C1→S3→C2→S5
  - Đường dẫn 2: S1→C1→S2
  - Đường dẫn 3: S1→C1→S3→C2→S4→S5

Kiểm thử đường dẫn cơ sở

- Bước 4: thiết kế test case cho từng đường dẫn độc lập
- Ví dụ:
  - Test case cho đường dẫn 1
    - Đầu vào: …
    - Đầu ra mong muốn: …
    - Mục đích: …

Kiểm thử đường dẫn cơ sở

- Bước 1: đồ thị lưu trình
  - Đỉnh
  - Cung
  - Đỉnh điều kiện
  - Vùng



Kiểm thử đường dẫn cơ sở

- Bước 2: Xác định độ phức tạp cyclomat → cho biết số lượng đường dẫn độc lập
  - V(G) = R(số vùng) = 6
  - V(G) = P(số đỉnh điều kiện)+1 = 5+1 = 6
  - V(G) = E(số cạnh)–N(số đỉnh)+2 = 17-13+2 = 6

Kiểm thử đường dẫn cơ sở

- Bước 3: tìm tập cơ sở các đường dẫn độc lập
  - Tìm 1 đường dẫn từ đầu đến cuối chương trình
  - Tìm đường dẫn mới có đi qua một cạnh mới mà không trùng với các đường dẫn trước đó
  - Làm cho đến khi đủ số lượng đường dẫn
- Ví dụ:
  - Đường dẫn 1: 1→2→8→9→11
  - Đường dẫn 2: 1→2→8→10→11
  - Đường dẫn 3: 1→2→3→8→9→11
  - Đường dẫn 4: 1→2→3→4→7→2→…
  - Đường dẫn 5: 1→2→3→4→5→7→2→…
  - Đường dẫn 6: 1→2→3→4→5→6→7→2→…

Kiểm thử đường dẫn cơ sở

- Bước 4: thiết kế test case cho từng đường dẫn độc lập
- Ví dụ:
  - Test case cho đường dẫn 1
    - Đầu vào: …
    - Đầu ra mong muốn: …
    - Mục đích: …

Kiểm thử cấu trúc điều kiện

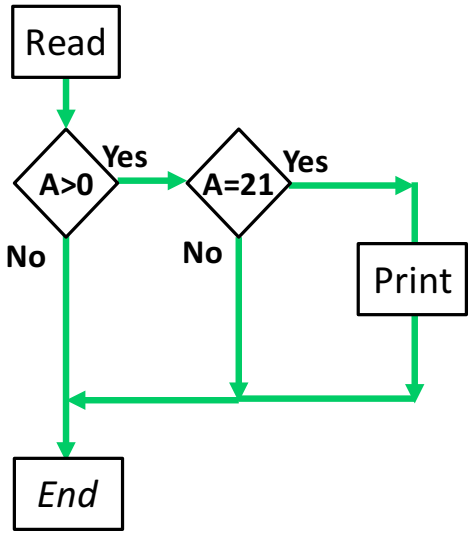
- Kiểm thử dòng điều khiển (Control-flow/Coverage testing)
- Kiểm thử dòng dữ liệu (Data flow testing)
- Kiểm thử vòng lặp (loop testing)

Kiểm thử dòng điều khiển

- Coverage dùng để đánh giá tính phủ của tập test case
  - Statement coverage
  - Decision/branch coverage
  - Condition coverage
  - Path coverage

Ví dụ

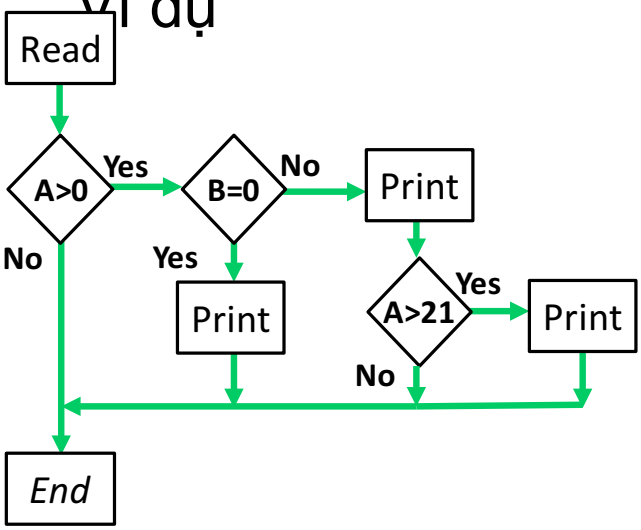
```
Read A
IF A > 0 THEN
    IF A  = 21 THEN
      Print "Key"
    ENDIF
ENDIF
```



- Cyclomatic complexity: 3
- Minimum tests to achieve:
  - Statement coverage: 1
  - Branch coverage: 3

Ví dụ

```
Read A
Read B
IF A > 0 THEN
    IF B  = 0 THEN
     Print “No values”
    ELSE
     Print B
     IF A > 21 THEN
        Print A
     ENDIF
    ENDIF
ENDIF
```



- Cyclomatic complexity: 4
- Minimum tests to achieve:
  - Statement coverage: 2
  - Branch coverage: 4

Kiểm thử dòng dữ liệu

- Một biến (variable)
  - Được xác định (define): được gán hay thay đổi giá trị
  - Được sử dụng (use): tính toán (c-use) hay điều kiện (p-use)
- Def-use path: đường dẫn từ def đến use của một biến
- Dữ liệu test được tạo ra để phủ tất cả các def-use

Kiểm thử dòng dữ liệu

- Ví dụ

```
1       sum = 0                          sum, def
2       read (n),                        n, def
3       i = 1                            i, def
4       while (i <= n)                   i, n p-sue
5             read (number)              number, def
6.            sum = sum + number         sum, def, sum, number, c-use
7             i = i + 1                  i, def, c-use
8       end while
9       print (sum)                      sum, c-use
```


Kiểm thử dòng dữ liệu

Table for sum

pair id	def	use
1	1	6
2	1	9
3	6	6
4	6	9

Table for i

pair id	def	use
1	3	4
2	3	7
3	7	7
4	7	4

Kiểm thử vòng lặp

- Kiểm tra tính hợp hệ của cấu trúc vòng lặp
- Bốn dạng vòng lặp:
  - Lặp đơn (simple loops)
  - Lặp móc nối (concatenated loops)
  - Lặp lồng nhau (nested loops)
  - Lặp không cấu trúc (unstructured loops)