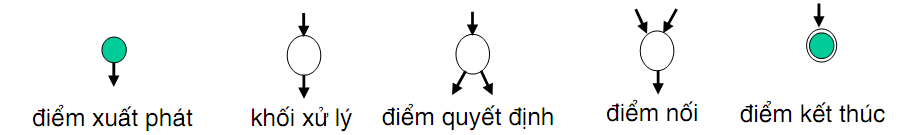
1. Kỷ thuật kiểm thử tĩnh
2. Kỹ thuật kiểm thử động
3. Kỉ thuật kiểm thử hộp trắng

Các phương pháp kiểm thử hộp trắng.

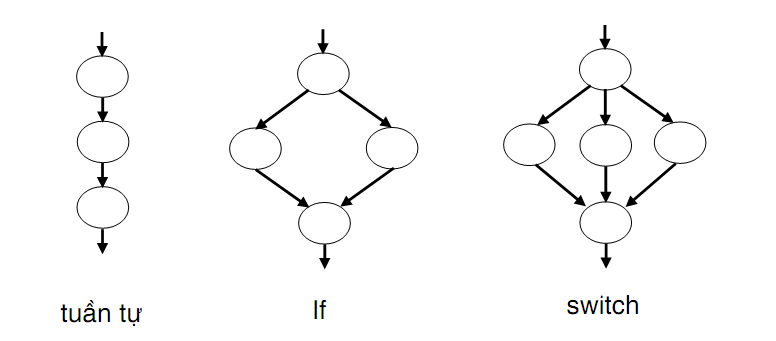
* Kiểm thử luồng điều khiển: tập trung kiểm thử giải thuật chức năng

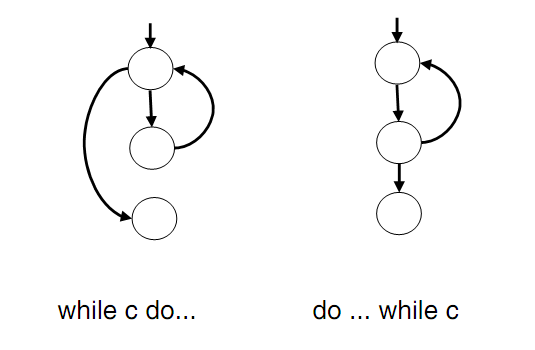
1. Phủ cấp 0 : kiểm thử những gì có thể kiểm thử được, phần còn lại để người dùng phát hiện và báo lại sau. => (thiếu trách nhiệm)
2. Phủ cấp 1: Kiểm thử sao cho mỗi lênh được thực hiện ít nhất 1 lần
3. Phủ cấp 2: Kiểm thử sao cho mỗi điểm quyết định luận lí đều được thực hiện ít nhất 1 lần trong các trường hợp.( mức phủ nhánh)
4. Phủ cấp 3: Kiểm thủ sao cho mỗi điều kiện luận lí con của từng điểm quyết định đều được thực hiện ít nhất 1 lần cho các trường hợp TRUE lẩn FALSE (phủ điều kiện con)
5. Phủ cấp 4: Kiểm thử sao cho mỗi điều kiện luận lí con của từng điểm quyết định đều được thực hiện ít nhất 1 lần cho trường hợp TRUE lẩn FALSE. Và điểm quyết định cũng được kiểm thử cho cả hai nhánh TRUE và FALSE. ( mức kiểm thử nhánh và các điều kiện con)

* Đồ thị dòng điều khiển

Phương pháp trực quan cho chúng ta thấy dể dàng các thành phần của thuật giải. Gồm 2 loại thành phần, các node và các cung kết nối giữa chúng.

* Mô tả cấu trúc điều khiển phổ dụng





* Độ phức tạp

V(G) = E – N + 2 ( E: số cung, N là số đỉnh)

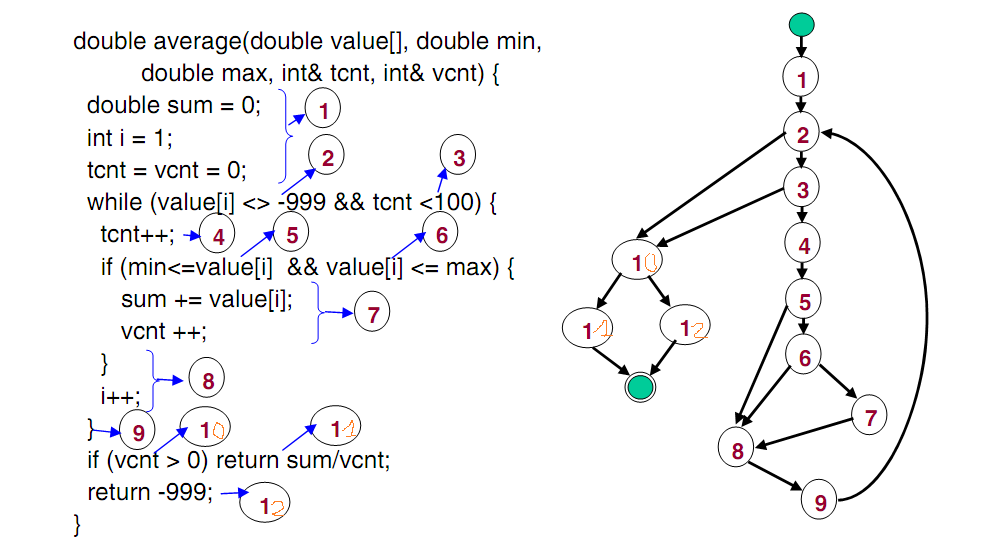
Nếu trong dòng điều khiển nhị phân

V(G) = P + 1 (P : số nút quyết định)

Quy trình kiểm thử hộp trắng:

* Bước 1: chuyển code thành đồ thị dòng điều khiển cơ bản
* Bước 2: Tính độ phức tạp C = P + 1
* Bước 3: xác định C đường tuyến tính độc lập cơ bản cần kiểm thử
* Bước 4 : tạo từng testcase cho mỗi đường tuyến tính thực thi cơ bản

Ví dụ:



Đồ thị trên có 5 nút quyết định nhị phân nên có độ phức tạp C = 5 +1 =6

6 đường thi hành tuyến tính cơ bản là

1. 1-2-10-11
2. 1-2-3-10-11
3. 1-2-10-12
4. 1-2-3-4-5-8-9
5. 1-2-3-4-5-6-8-9
6. 1-2-3-4-5-6-7-8-9

* Test case cho trường hợp 1: value(k)<>-999, với 1<k<i

Value(i) = -999 với 2<=i<=100

Kết quả kì vọng: 1) trung bình = giá trị trung bình của i=1 giá trị. 2)tcnt=i-1. 3) vcnt=i-1

* Test case cho trườn hợp 2: value(k)<>-999 với mọi k < i, i>100

Kết quả kì vọng: 1) trung bình = giá trị trung bình của 100 giá trị hợp lệ. 2)tcnt=100. 3) vcnt=100

* Test case cho trườn hợp 4: value(1)==-999 với mọi k < i, i>100

Kết quả kì vọng: 1) trung bình = -999. 2)tcnt=0. 3) vcnt=0

* Test case cho trườn hợp 5: value(k)<>-999 với mọi i<=100 và value(k)< min với k<i

Kết quả kì vọng: 1) giá trị trung bình của n giá trị hợp lệ. 2)tcnt=100. 3) vcnt=n

* Test case cho trườn hợp 6: value(k)<>-999 với mọi i<=100 và value(k)> max với k<=i

Kết quả kì vọng: 1) giá trị trung bình của n giá trị hợp lệ. 2)tcnt=100. 3) vcnt=n

* Test case cho trườn hợp 3: value(i)<>-999 ,min<=value(i)<=mã với mọi i<=100

Kết quả kì vọng: 1) giá trị trung bình của n giá trị hợp lệ. 2)tcnt=100. 3) vcnt=100

* Kiểm thủ dòng dữ liệu: kiểm thử đời sống của từng biến dữ liệu được dùng trong thuật giải

Mục đích là kiểm thử đời sống của từng biến dữ liệu có “ tốt lành” trong từng luồng thi hành của chương trình.

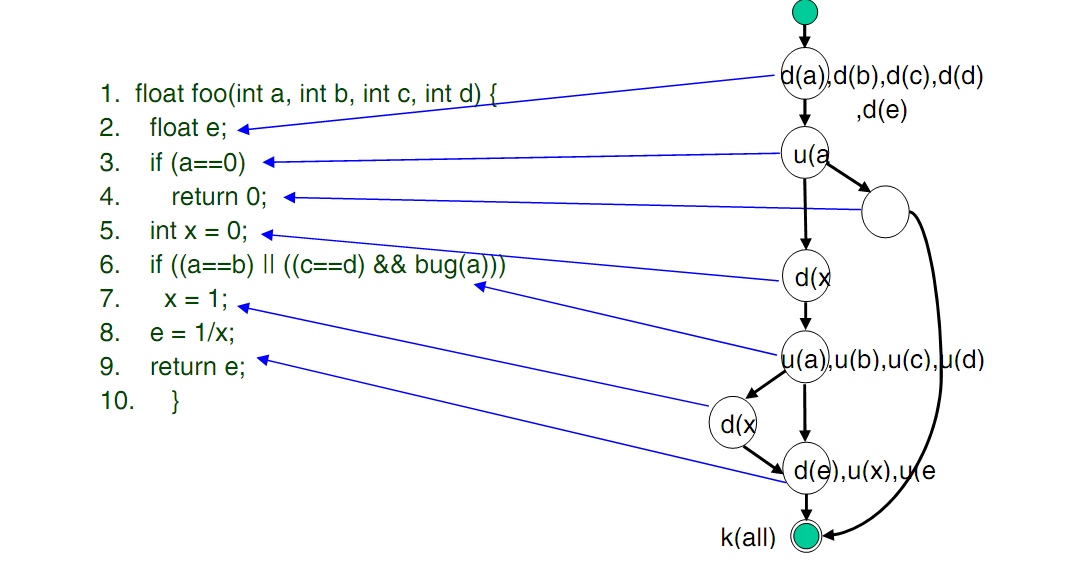
* (d) : định nghĩa biến, gán giá trị cho biến, hay nhập dữ liệu vào biến
* (u) : tham khảo giá trị của biến ( thường thông qua biểu thức)
* (k) : hủy (xóa bỏ) biến đi.
* ~d : biến chưa tồn tại được định nghĩa với giá trị xác định
* ~u :biến chưa tồn tại được dùng ngay
* ~k : biến chưa tồn tại bị hủy

3 hoạt động xử lí biến khác nhau kết hợp lại tạo thành tổ hợp 9 cặp đôi hoạt động xử lí biến theo thứ tự.

* (dd): biến đinh nghĩa rồi được định nghĩa lại (chấp nhận được, nhưng có thể là lỗi lập trình)
* (du) : biến định nghĩa rồi và được dùng (okie)
* (dk) : biến được định nghĩa rồi được định nghĩa lại (chấp nhận được, nhưng có thể là lỗi lập trình)
* (ud) : biến được định nghĩa rồi định nghĩa giá trị mới(okie)
* (uu) : biến được dùng rồi thì dùng tiếp (okie)
* (uk) : biến được dùng rồi bị hủy (okie)
* (kd) : biến bị xóa bỏ rồi được định nghĩa lại (okie)
* (ku) : biến bị xóa bỏ rồi được dùng ( error)
* (kk) : biến bị xóa bỏ rồi bị xóa bỏ nữa ( có thể là error lập trình)
* Quy trình kiểm thử dòng dữ liệu

1. Từ mã chương tình cần kiểm thử, xây dựng đồ thị dòng điều khiển tương ứng, rồi chuyển chuyển thành đồ thị dòng điều khiển nhị phân, rồi chuyển thành đồ thị dòng dữ liệu.
2. Tính độ phức tạp của đồ thị (C= P+1)
3. Xác định C đường thi hành tuyến tính độc lập cơ bản cần kiểm thử
4. Lặp kiểm thử đới sống từng biến dữ liệu

* Mỗi biến có tối đa C kịch bản đời sống khác nhau
* Trong từng kịch bản đời sống của 1 biến, kiểm thử xem có tồn tại cặp đôi hoạt động không bình thường nào không ?
* Ví dụ:

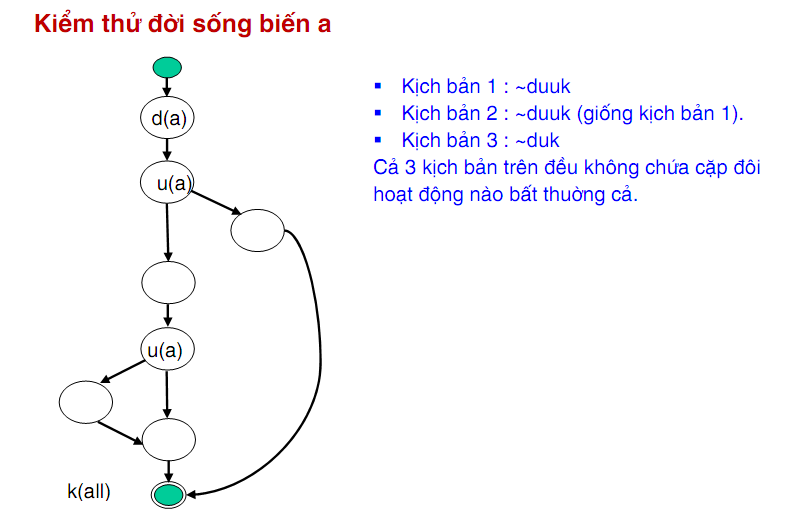


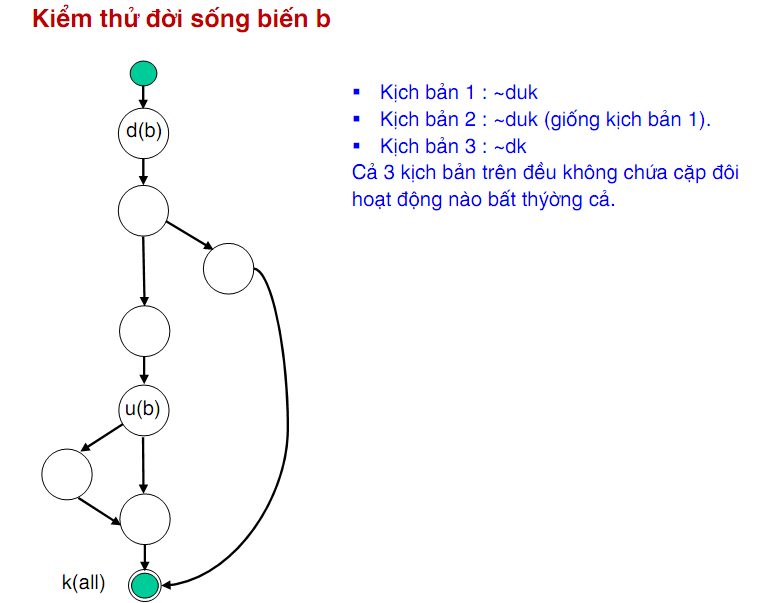
Hình trên có 2 nút quyết định nên ta có

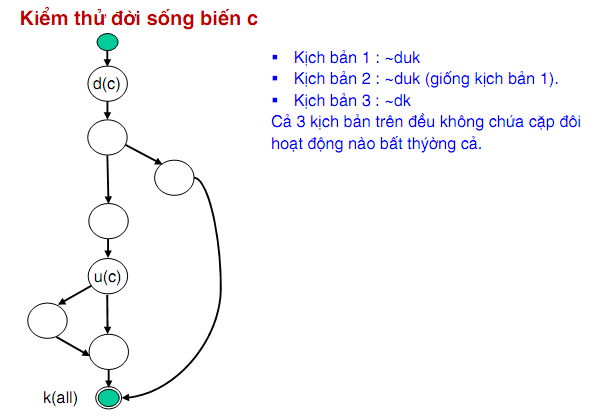
C= P+1= 2 +1 = 3

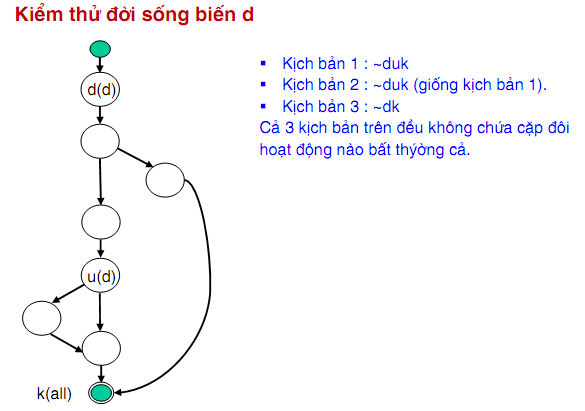
Nó có 4 biến là tham số và 2 biến cục bộ.

* Ta kiểm thử đời sống từng biến a,b,c,d,e,x như sau:

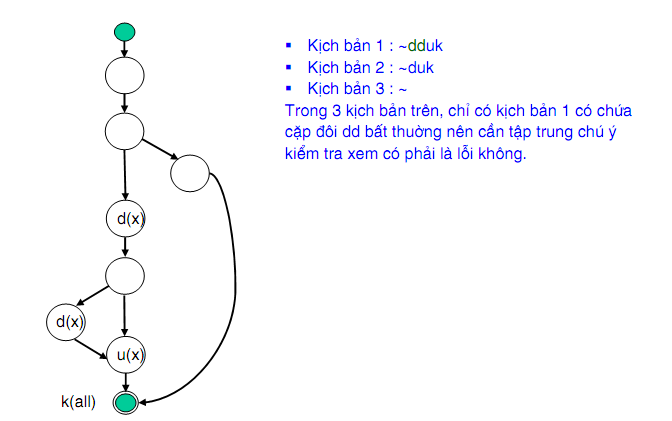
­­­





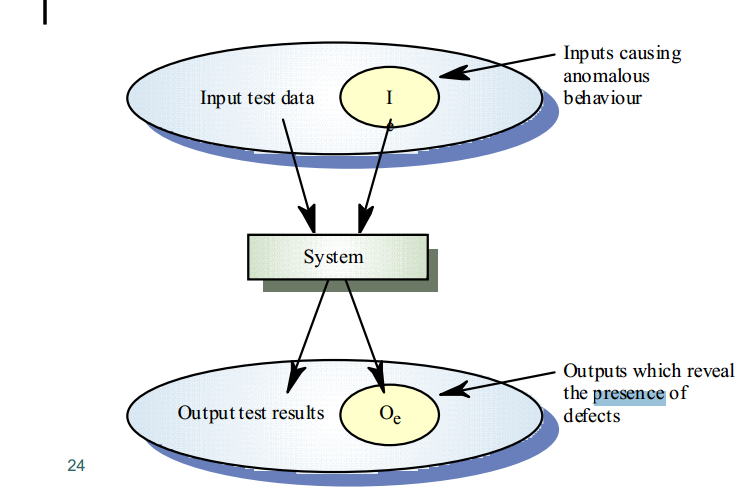






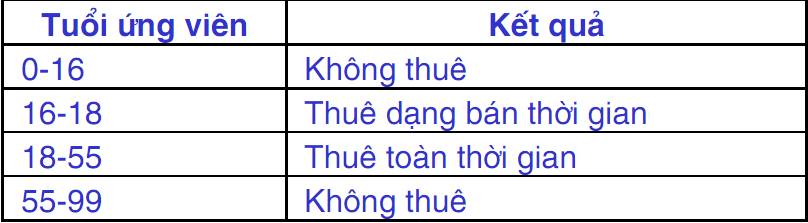
1. Kỉ thuật kiêm thử hộp đen

Chiến lược kiểm thử dựa vào thông tin duy nhất là các đặc tả về yêu cầu chức năng của phần mềm.



1. Kỹ thuật phân lớp tương đương( boundary value analysis)

* Tư tưởng của kỹ thuật này là phân các test case thành các nhóm họ khác nhau, các test case cùng mỗi họ sẻ thực hiện cùng 1 hành vi. Mỗi nhóm test case thỏa mản tiêu chuẩn trên gọi là 1 lớp tương đương, ta chỉ cần xác định 1 testcase đại diện cho nhóm và dùng test case nà để kiểm thử.
* Ví dụ ta cần xây dựng một phần mềm quản lí nguồn nhân lực, phần mềm sẻ ra quyết định dựa vào bảng tuổi ứng viên như sau

Ta thấy có 4 lớp tương đương, 1 lớp ứng với một chế độ xử lí. Bao gồm không thuê vì quá trẻ, thuê bán thời gian, thê full time, không thuê vì quá già.

Testcase 1: {Input : 2 tuổi, Output : không thuê}

Testcase 2: {Input : 17 tuổi, Output : thuê bán thời gian}

Testcase 3: {Input : 35 tuổi , Output: thê full time}

Testcase 44: {Input : 70 tuổi, Output : không thuê

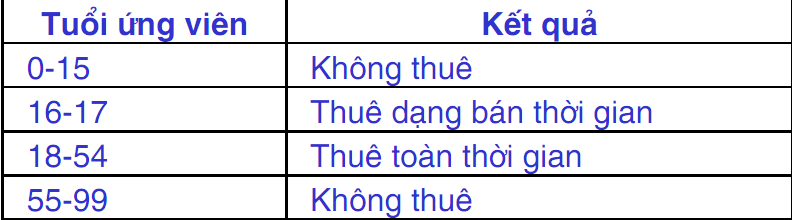
* Ví dụ: nếu dữ liệu nhập là số thực liên tục, ta có thể chọn 1 test case đại diện cho giá trị nhập hợp lệ, 12test case đại diện cho giá trị nhập không hợp lệ nằm phía dưới và phía trên.

1. Kỹ thuật phân tích các giá trị ở biên ( equivalence class testing)

* Kinh nghiệm cho ta thấy rằng phần lớn lỗi lập trình thường nằm ở các giá trị biên của một khoảng liên tục nào đó. Kỹ thuật này thích hợp với lớp tương đương có những giá trị liên tục.
* Phương pháp tiến hành:

1. Nhận dạng các giá trị biên của mỗi lớp tương đương
2. Nhận dạng hai biên của mỗi lớp tương đương
3. Tạp test case cho mỗi biên của mỗi lớp tương đương

* 1 testcase cho giá trị biên
* 1 testcase cho giá trị dưới biên
* 1 testcase cho giá trị trên biên
* Ví dụ đặc tả của phần mềm “quản lí nguồn nhân lực”



Ta sẻ định nghĩa các test case tương ứng như sau: {-1,0,1},{14,15,16}, {15,16,17}, {16,17,18}, {17,18,19}, {53,54,55}, {54,55,56}, {98,99,100}.

Nếu loại bỏ các testcase cùng thì ta còn:

{-1,0,1,14,15,16,17,18,19,53,54,55,56,98,99,100}

1. Kỹ thuật dùng bảng quyết định (decision tables)

* Công cụ rất hữu ích để miêu tả các yêu cầu phần mềm hoặc các bảng thiết kế hệ thống phần mềm. Nó miêu tả nghiệp vụ phức tạp mà phần mềm phải thực hiện dưới dạng dễ đọc và dễ kiểm soát.

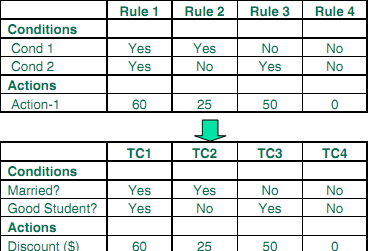
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Rule 1 | Rule 2 | ………. | Rule p |
| **Condition** |  |  |  |  |
| Condition-1 |  |  |  |  |
| …… |  |  |  |  |
| Condition-m |  |  |  |  |
| **Actions** |  |  |  |  |
| Action-1 |  |  |  |  |
| …….. |  |  |  |  |
| Action-n |  |  |  |  |

Số rule = 2 ^ condition

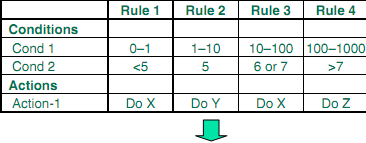
* Quy trình thực hiện kiểm thửu

1. Tìm bảng quyết định dựa vào yêu cầu chức năng của phần mềm hay từ bảng thiết kế. Nếu chưa có thì phải xây dựng dựa vào đặc tả chức năng hay yêu cầu thiết kế phần mềm.
2. Từ bảng quyết đinh chuyển thành bảng các testcase trong đó mỗi cột miêu tả 1 luật được chuyển thành từ 1 đến n cột miều tả các testcase tương ứng với luật đó.

* Nếu điều kiện là luận lí thì mỗi cột được chuyển thành 1 testcase
* Nếu điều kiện nhập là một lớp tương đương nhiều giá trị liên tục thì mỗi cột luật sẻ được chuyển thành nhiều testcase dựa trên kĩ thuật lớp tương đương hay kĩ thuật giá trị biên.
* Ví dụ 1:



* Ví dụ 2





1. Kỹ thuật kiểm thử các bộ n thần kỳ (pairwise)
2. Kỉ thuật chuyển trạng thái (State transition )
3. Kỉ thuật phân tích vùng miền (domain analysis)
4. Kỹ thuật dựa trên đặc tả Use Case (Use case)
5. Kỹ thuật lược đồ quan hệ nhân quả ( Cause-effect diagram)