### **Phần 1: Lý thuyết**

#### **1. Bảng quyết định trong Black-Box Testing**

**a. Bảng quyết định là gì? Giải thích cách sử dụng bảng quyết định để kiểm thử các điều kiện logic phức tạp.**

* **Bảng quyết định** là một công cụ trong kỹ thuật kiểm thử Black-Box, được sử dụng để mô tả và kiểm tra các tổ hợp điều kiện logic phức tạp của hệ thống. Nó liệt kê tất cả các trường hợp có thể xảy ra của các điều kiện đầu vào và kết quả mong đợi tương ứng.
* **Cách sử dụng**:
  1. **Xác định điều kiện**: Liệt kê các điều kiện logic ảnh hưởng đến hành vi của hệ thống (ví dụ: độ tuổi, thời gian, trạng thái).
  2. **Liệt kê tổ hợp**: Tạo bảng với các cột đại diện cho điều kiện và hành động/kết quả. Mỗi hàng là một tổ hợp điều kiện.
  3. **Xác định kết quả**: Ghi rõ kết quả mong đợi cho mỗi tổ hợp.
  4. **Thiết kế test case**: Dựa trên bảng, tạo các test case để kiểm tra từng tổ hợp, đảm bảo tất cả trường hợp được phủ. Bảng quyết định giúp giảm số lượng test case cần thiết bằng cách tập trung vào các tổ hợp quan trọng và loại bỏ các trường hợp dư thừa.

**b. Ví dụ về hệ thống có điều kiện logic phức tạp**

* **Hệ thống**: Tính giá vé dựa trên độ tuổi (Trẻ em: <12, Người lớn: 12-60, Người cao tuổi: >60) và thời gian (Giờ thường: trước 17:00, Giờ cao điểm: sau 17:00).
* **Tạo bảng quyết định**:
  + **Điều kiện**:
    1. Độ tuổi: Trẻ em, Người lớn, Người cao tuổi.
    2. Thời gian: Giờ thường, Giờ cao điểm.
  + **Kết quả**: Giá vé (ví dụ: Trẻ em giờ thường: 5$, giờ cao điểm: 7$; Người lớn giờ thường: 10$, giờ cao điểm: 12$; Người cao tuổi giờ thường: 7$, giờ cao điểm: 9$).
  + Bảng quyết định:

| Độ tuổi | Thời gian | Giá vé |
| --- | --- | --- |
| Trẻ em | Giờ thường | 5$ |
| Trẻ em | Giờ cao điểm | 7$ |
| Người lớn | Giờ thường | 10$ |
| Người lớn | Giờ cao điểm | 12$ |
| Người cao tuổi | Giờ thường | 7$ |
| Người cao tuổi | Giờ cao điểm | 9$ |

* + Ứng dụng: Dựa trên bảng, thiết kế test case để kiểm tra từng hàng, đảm bảo tất cả tổ hợp được kiểm tra.

#### **2. Sơ đồ chuyển trạng thái**

**a. Sơ đồ chuyển trạng thái là gì? Giải thích cách sử dụng.**

* **Sơ đồ chuyển trạng thái** là một biểu đồ mô tả các trạng thái của hệ thống và các chuyển đổi giữa chúng, thường được sử dụng trong kiểm thử Black-Box để kiểm tra hành vi hệ thống qua các trạng thái.
* **Cách sử dụng**:
  1. **Xác định trạng thái**: Liệt kê các trạng thái chính của hệ thống (ví dụ: Giỏ hàng trống, Đang thanh toán).
  2. **Xác định chuyển đổi**: Xác định các sự kiện hoặc hành động dẫn đến thay đổi trạng thái (ví dụ: Thêm sản phẩm, Hủy đơn hàng).
  3. **Vẽ sơ đồ**: Minh họa các trạng thái bằng vòng tròn và các chuyển đổi bằng mũi tên có nhãn.
  4. **Thiết kế test case**: Tạo test case để kiểm tra từng chuyển đổi và đảm bảo hệ thống hoạt động đúng ở mọi trạng thái. Sơ đồ này giúp kiểm tra hành vi động của hệ thống, đặc biệt trong các ứng dụng có luồng công việc phức tạp.

**b. Ví dụ về hệ thống cần kiểm thử chuyển trạng thái**

* **Hệ thống**: Quy trình đặt hàng trực tuyến.
* **Trạng thái chính**:
  + **Giỏ hàng trống**: Không có sản phẩm trong giỏ.
  + **Đã thêm sản phẩm**: Có ít nhất một sản phẩm trong giỏ.
  + **Đang thanh toán**: Người dùng đang nhập thông tin thanh toán.
  + **Đã xác nhận**: Đơn hàng được xác nhận và xử lý.
  + **Đã hủy**: Đơn hàng bị hủy bởi người dùng hoặc hệ thống.
* **Chuyển đổi**:
  + Từ "Giỏ hàng trống" → "Đã thêm sản phẩm" (Thêm sản phẩm).
  + Từ "Đã thêm sản phẩm" → "Đang thanh toán" (Chọn thanh toán).
  + Từ "Đang thanh toán" → "Đã xác nhận" (Hoàn tất thanh toán).
  + Từ "Đã thêm sản phẩm" hoặc "Đang thanh toán" → "Đã hủy" (Hủy đơn hàng).

#### **3. Statement và Branch Coverage trong White-Box Testing**

**a. Định nghĩa và sự khác biệt**

* **Statement Coverage**: Đo lường tỷ lệ các dòng mã (statements) được thực thi trong quá trình kiểm thử. Mục tiêu là đảm bảo mỗi dòng mã được thực thi ít nhất một lần.
* **Branch Coverage**: Đo lường tỷ lệ các nhánh (branches) trong các cấu trúc điều khiển (if, else, switch) được thực thi. Mỗi nhánh phải được kiểm tra cho cả kết quả true và false.
* **Khác biệt**:
  + Statement coverage chỉ đảm bảo rằng mã được chạy qua, nhưng không đảm bảo kiểm tra tất cả các trường hợp logic (ví dụ: không kiểm tra nhánh "else" nếu nó không được thực thi).
  + Branch coverage nghiêm ngặt hơn, yêu cầu kiểm tra tất cả các nhánh quyết định, do đó bao phủ nhiều trường hợp logic hơn.

**b. Tại sao 100% branch coverage đảm bảo chất lượng tốt hơn?**

* Branch coverage đảm bảo rằng mọi quyết định logic (true/false) trong mã được kiểm tra, phát hiện các lỗi liên quan đến điều kiện bị bỏ qua hoặc xử lý sai. Trong khi đó, 100% statement coverage có thể đạt được mà không kiểm tra tất cả các nhánh, dẫn đến bỏ sót lỗi trong các trường hợp không được thực thi.

#### **4. Phân tích mã để xác định điểm kiểm thử**

**a. Quy trình phân tích mã**

1. **Đọc và hiểu mã**: Phân tích cấu trúc mã, xác định các cấu trúc điều khiển (if, else, vòng lặp).
2. **Xác định điểm kiểm thử**:
   * **Statement coverage**: Đảm bảo mỗi dòng mã được thực thi.
   * **Branch coverage**: Xác định tất cả các nhánh (true/false) trong cấu trúc điều khiển.
   * **Path coverage**: Xem xét các đường dẫn logic độc lập trong mã.
3. **Xác định giá trị đầu vào**: Chọn các giá trị đầu vào để kích hoạt từng nhánh hoặc đường dẫn.
4. **Kiểm tra ranh giới**: Tập trung vào các giá trị biên (boundary values) như số âm, số 0, hoặc giá trị tối đa.
5. **Thiết kế test case**: Tạo test case dựa trên các điểm kiểm thử đã xác định.

**b. Ví dụ lỗi tiềm ẩn**

* **Mã ví dụ**:

def divide(a, b):

return a / b

* **Lỗi tiềm ẩn**: Không kiểm tra trường hợp b == 0, dẫn đến lỗi chia cho 0.
* **Phát hiện qua White-Box**: Phân tích mã cho thấy thiếu kiểm tra điều kiện if b == 0. Test case với b = 0 sẽ phát hiện lỗi này.

### **Phần 2: Bài tập thực hành**

#### **1. Tạo bảng quyết định**

**Hệ thống**: Tính giá vé xem phim dựa trên độ tuổi và thời gian.

* **Điều kiện**:
  + Độ tuổi: Trẻ em (<12), Người lớn (12-60), Người cao tuổi (>60).
  + Thời gian: Giờ thường (trước 17:00), Giờ cao điểm (sau 17:00).
* **Kết quả** (giả định giá vé):
  + Trẻ em: Giờ thường (5$), Giờ cao điểm (7$).
  + Người lớn: Giờ thường (10$), Giờ cao điểm (12$).
  + Người cao tuổi: Giờ thường (7$), Giờ cao điểm (9$).

**a. Bảng quyết định**

| Độ tuổi | Thời gian | Giá vé |
| --- | --- | --- |
| Trẻ em | Giờ thường | 5$ |
| Trẻ em | Giờ cao điểm | 7$ |
| Người lớn | Giờ thường | 10$ |
| Người lớn | Giờ cao điểm | 12$ |
| Người cao tuổi | Giờ thường | 7$ |
| Người cao tuổi | Giờ cao điểm | 9$ |

**b. Thiết kế 4 test case**

1. **Test Case 1:**
   * Mô tả: Kiểm tra giá vé cho trẻ em vào giờ thường.
   * Dữ liệu đầu vào: Độ tuổi = 10, Thời gian = 15:00.
   * Kết quả mong đợi: Giá vé = 5$.
2. **Test Case 2:**
   * Mô tả: Kiểm tra giá vé cho người lớn vào giờ cao điểm.
   * Dữ liệu đầu vào: Độ tuổi = 30, Thời gian = 18:00.
   * Kết quả mong đợi: Giá vé = 12$.
3. **Test Case 3:**
   * Mô tả: Kiểm tra giá vé cho người cao tuổi vào giờ thường.
   * Dữ liệu đầu vào: Độ tuổi = 65, Thời gian = 14:00.
   * Kết quả mong đợi: Giá vé = 7$.
4. **Test Case 4:**
   * Mô tả: Kiểm tra giá vé cho trẻ em vào giờ cao điểm.
   * Dữ liệu đầu vào: Độ tuổi = 8, Thời gian = 19:00.
   * Kết quả mong đợi: Giá vé = 7$.

#### **2. Thiết kế sơ đồ chuyển trạng thái**

**Hệ thống:** Ứng dụng đặt hàng trực tuyến.

* **Trạng thái:**
  1. Giỏ hàng trống
  2. Đã thêm sản phẩm
  3. Đang thanh toán
  4. Đã xác nhận
  5. Đã hủy

**a. Sơ đồ chuyển trạng thái (mô tả bằng văn bản)**

* Giỏ hàng trống → Đã thêm sản phẩm (Hành động: Thêm sản phẩm).
* Đã thêm sản phẩm → Giỏ hàng trống (Hành động: Xóa toàn bộ sản phẩm).
* Đã thêm sản phẩm → Đang thanh toán (Hành động: Chọn thanh toán).
* Đang thanh toán → Đã xác nhận (Hành động: Hoàn tất thanh toán).
* Đang thanh toán → Đã hủy (Hành động: Hủy thanh toán).
* Đã thêm sản phẩm → Đã hủy (Hành động: Hủy đơn hàng).

**b. Thiết kế 3 test case**

1. **Test Case 1:**
   * Mô tả: Kiểm tra chuyển từ Giỏ hàng trống sang Đã thêm sản phẩm.
   * Hành động: Thêm một sản phẩm vào giỏ hàng.
   * Kết quả mong đợi: Trạng thái chuyển thành "Đã thêm sản phẩm", sản phẩm hiển thị trong giỏ.
2. **Test Case 2:**
   * Mô tả: Kiểm tra chuyển từ Đã thêm sản phẩm sang Đang thanh toán, rồi sang Đã xác nhận.
   * Hành động: Thêm sản phẩm → Chọn thanh toán → Hoàn tất thanh toán.
   * Kết quả mong đợi: Trạng thái chuyển thành "Đã xác nhận", đơn hàng được ghi nhận.
3. **Test Case 3:**
   * Mô tả: Kiểm tra chuyển từ Đang thanh toán sang Đã hủy.
   * Hành động: Thêm sản phẩm → Chọn thanh toán → Hủy thanh toán.
   * Kết quả mong đợi: Trạng thái chuyển thành "Đã hủy", đơn hàng bị hủy.

#### **3. Phân tích mã và xác định điểm kiểm thử**

**Mã Python:**

def calculate\_ticket\_price(age):

if age < 12:

price = 5

elif age <= 60:

price = 10

else:

price = 7

return price

**a. Phân tích mã**

* Statement Coverage: Cần thực thi tất cả các dòng mã:
  + Dòng if age < 12: price = 5.
  + Dòng elif age <= 60: price = 10.
  + Dòng else: price = 7.
  + Dòng return price.
* Branch Coverage: Cần kiểm tra tất cả các nhánh:
  + Nhánh 1: age < 12 (true).
  + Nhánh 2: age >= 12 và age <= 60 (true).
  + Nhánh 3: age > 60 (true).
* Điểm kiểm thử:
  + Tuổi < 12 (ví dụ: 10).
  + Tuổi từ 12 đến 60 (ví dụ: 30).
  + Tuổi > 60 (ví dụ: 65).
  + Kiểm tra giá trị biên: 12, 60.

**b. Thiết kế 4 test case**

1. **Test Case 1:**
   * Mô tả: Kiểm tra giá vé cho trẻ em.
   * Dữ liệu đầu vào: age = 10.
   * Kết quả mong đợi: price = 5.
2. **Test Case 2:**
   * Mô tả: Kiểm tra giá vé cho người lớn.
   * Dữ liệu đầu vào: age = 30.
   * Kết quả mong đợi: price = 10.
3. **Test Case 3:**
   * Mô tả: Kiểm tra giá vé cho người cao tuổi.
   * Dữ liệu đầu vào: age = 65.
   * Kết quả mong đợi: price = 7.
4. **Test Case 4:**
   * Mô tả: Kiểm tra giá trị biên tại tuổi 12.
   * Dữ liệu đầu vào: age = 12.
   * Kết quả mong đợi: price = 10.

**c. Lỗi tiềm ẩn và cách sửa**

* Lỗi tiềm ẩn: Mã không xử lý trường hợp tuổi âm (age < 0), có thể dẫn đến giá vé không hợp lý (ví dụ: age = -1 trả về price = 5).
* Đề xuất sửa đổi:

def calculate\_ticket\_price(age):

if age < 0:

raise ValueError("Tuổi không thể âm")

if age < 12:

price = 5

elif age <= 60:

price = 10

else:

price = 7

return price

* Giải thích: Thêm kiểm tra age < 0 để ném ngoại lệ, đảm bảo tính hợp lệ của đầu vào.