MONASH University

FIT2107
Software Quality and Testing

Lecture 3 - Black-Box Testing - I

- What is Black-Box Testing?
- Random Testing
- Equivalence Testing
- Category-Partition Testing

Dr Najam Nazar

Announcements

- Assignment 1 ready and published
- Time to form groups.
  - This will be done in this week's tutorial.
  - Tutors will form groups randomly.

Recap

- Software Testing
- Testing Objective
- Testing Pyramids and Cones
- Some definitions such as Error, Failure etc.

Blackbox Testing

- Software testing methods that analyses the functionality of a software or an application without
  - Knowing the internals (code or structure or design).
  - Input and output values compared.

Black-Box Testing Strategies

1. Random Testing
2. Equivalence Testing
3. Category-Partition Testing
4. Boundary Value Testing
5. Combinatorial Testing / Pairwise Testing

Random Testing

- Testing by generating random inputs.
- Usually through a random generator
- And comparing output with specifications
- Fuzzing is one way of doing it.
- Alternatively, any random generator program in any languages can be used
- Some industrial application.

Absolute number (Implement in seconds)

```
1   int myAbs(int x) {
2       if (x > 0) {
3           return x;
4       }
5       else {
6           return x; // bug: should be '-x'
7       }
8   }
```

>inputs -> {123, 36, -35, 48, 0}
> output -> {123, 36, -35, 48, 0}

So what is the problem. -35 should return 35.

Partitioning (Classes)

- In blackbox testing we aim at devising set of inputs to test the program.
- A program may behave differently under different conditions.
- Programs are complex and can’t be tested with one input.
- Need to devise set of input that tackles one part (or partition) of the program.
- Either the system performs as expected on every input within the partition or unexpectedly on every input in the partition.
- How??
  - Disjoint classes: no two partitions represent the same behavior.
  - Can easily verify if the behavior is correct or not.
- Pretty much all software testing that isn't random testing can be viewed as a type of partitioning testing

Partitioning - Example

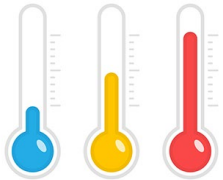
- **Leap Year**
  - **The year is divisible by 4**
  - **The year is not divisible by 100**
  - **Exceptions are years divisible by 400, that are leap years.**
- Partitions?
  - Divisible by 4, not by 100 = leap year, TRUE. (2000)
  - Divisible by 4, 100,400 = leap year, TRUE (2040)
  - Not divisible by 4 = not leap year, FALSE (2013)
  - Divisible by 4,100 but not by 400 = not leap year, FALSE (1900)

Equivalence Partitioning

- If the program behaves **correctly** for one given input, it will work correctly for all other **inputs** from that **partition**.
- Each equivalence classes = subset of input
- Let's look at the leap year example discussed earlier
  - Partition 1: {2016, (divisible by 4, not by 100)}
  - Partition 2: {2000, (divisible by 4,100,400)}
  - Partition 3: {39, not divisible by 4}
  - Partition 4: {1900, (divisible by 4,100, not by 400)}
- Every input in a given partition will give same result so we end up with 4 test cases.

Equivalence Partitioning

- Let’s look at another example
- Temperature monitoring
  - If the patient's temperature is below 36.5 degrees, display a blue light.
  - If the patient's temperature is between 36.5 and 37.5 degrees (inclusive) display a yellow light.
  - If the patient's temperature is above 37.5 degrees, display a red light.
- Partitions
  - Partition 1: {<36.5} will always display a blue light.
  - Partition2: {36.5,37.5} will always display a yellow light.
  - Partition3: {>37.5} will always display a red light.



- We have used our experience and specifications to identify partitions.
- How to find classes?
  - Art not science
  - Some guidelines in the reading materials.

Category Partitioning

- It is a systematic way of identifying test cases using the characteristics of the input values.
- Categories – attributes that you'd divide up the input space
- Choices – the groups of values that a category might take.
- It minimizes the test cases to feasible amounts.
- How it works? Steps
  - Identify the parameters
  - Understand and identify the characteristics of each parameter.
  - Add constraints to minimize the test suits
  - Generate combinations (that are the test cases).

Category Partitioning

- **Scenario**: The system should give 25 discount on the **raw** amount of the cart when it is **Christmas**. The method has **two** input parameters: the **total price** of the products in the cart and the **date**. When it is **not Christmas** it just returns the original price, otherwise it applies the discount.
- How it works? Steps
  - Identify the parameters:
    - Current date, total price
  - Understand and identify the characteristics of each parameter:
    - date can be Christmas or not Christmas
    - Price can be positive, 0 or negative [exceptional condition]
  - Add constraints to minimize the test suits:
    - Exceptional case is the only constraint in the above scenario
  - Generate combinations (that are the test cases).
    - Christmas {+,0, -}
    - Not Christmas {+,0,-}
    - We end up with 5 cases. The not Christmas with negative number will be discarded.

- Positive price at Christmas
- Positive price not at Christmas
- Price of 0 at Christmas
- Price of 0 not at Christmas
- Negative price at Christmas

Summary

- In blackbox testing an input is provided to the system to determine the outputs and system is tested based on the output.
- Random testing where we use randomly generated inputs to test a system
- Equivalence Testing where we create partitions of equivalent size where set of inputs in each partition gives same (equal) result.
- Category partition where we identify categories and their characteristics. Then we use both properties to generate test partitions and test cases.

Next week

- More Black box testing
  - Boundary Value Analysis
  - Pair Wise Testing
- Decision making between different types

Questions

MONASH University