Software Testing Continued

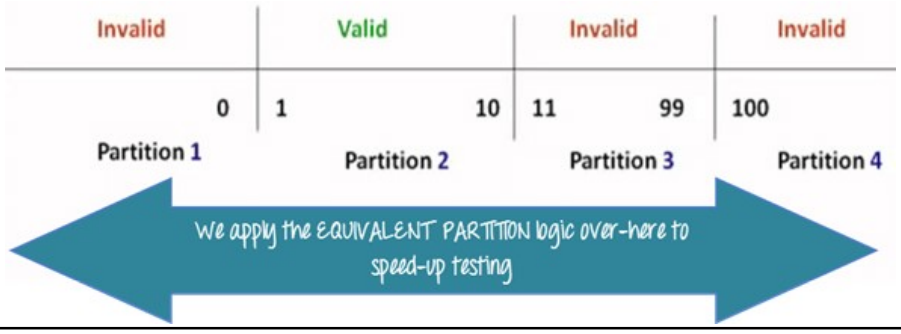
CSCI 3300/5300

Constructing Unit Tests: Equivalence Class Partitioning

- Inputs to the system are divided into groups that are expected to exhibit similar behavior
- Select one input from each group to test the behavior
- Consider valid and invalid classes

Pizza Example

- Online pizza ordering system
  - Can order 1 to 10 pizzas
  - Valid input is 1 and 2 digit positive numbers
- Equivalence classes

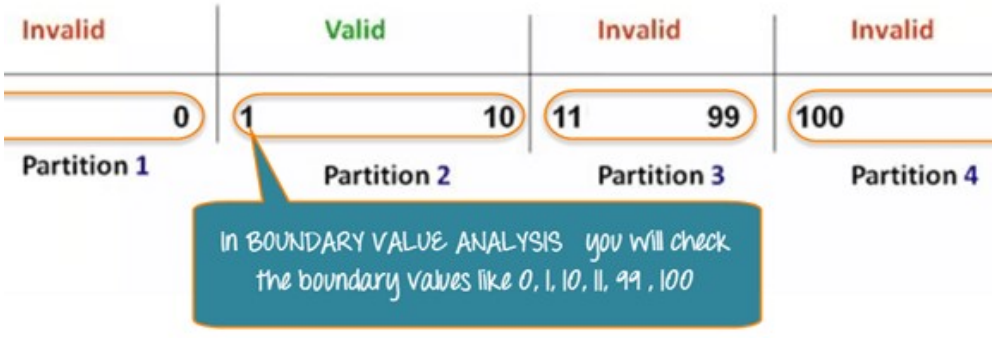


What are the equivalence classes for the triangle testing challenge from last time?

Constructing Unit Tests: Boundary Value Analysis

- Testing between extreme ends or boundaries between partitions of the input values:
  - Start-End
  - Lower-Upper
  - Minimum-Maximum
- For each equivalence class, select a value at
  - Minimum
  - Just above the minimum
  - A nominal value
  - Just below the maximum
  - Maximum

Pizza Example



What are the boundary values for the triangle testing challenge?

Run unit tests with each build

- Frameworks can help automate unit tests:
  - gtest – Google test framework for C++
  - unittes – python unit test framework
  - JUnit – Java unit test framework
- Example from my repo

Isolating issues and Adding Unit tests

- Bugs may be discovered at any time (not just during unit testing)
- Try to isolate the issue in a unit test environment
- Build a test that replicates the issue
  - Will help you debug the problem
  - Will add to regression testing
- Fix the problem and rerun unit test

Example issues and tests from my project implementation

Clock Overflow

- Clock was getting incremented to unexpected values when using InputThread, FromCSV, and Clock
- Frist two increments were fine, third increment was wrong
- Added ClockTest
  - Test one clock increment of 1 second – no issues
  - Test multiple increments of 1 second – replicated the issue
- Cause - overflow
  - Some functions in the Clock class worked on int types instead of uint64_t

Seg Fault Reading Data Rates

- FromCSV – class that reads data from CSV file
- FromCSV::getNextRate() – returns the next 1-second rate
- Seg fault in running simulation
  - Narrowed down the issue to being in FromCSV::getNextRate()
- Cause:
  - FromCSV had `queue<uint64> rates`
  - Called `rates.pop()` when queue was empty
- Added Unit test: FromCSV - ReadPastEnd

Another Seg Fault

- Seg fault running a simulation
- A message retrieved from Buffer was corrupt
- Cause:
  - Buffer object was used by multiple threads
  - Buffer object was not thread safe
- Unit test?
  - Probably not
  - What is the next testing level in the test pyramid?

Integration Testing

- Testing to expose defects in the *interfaces* (or interactions) between components
- Approaches:
  - Big Bang – all or most modules are combined together and tested
  - Top Down – top most modules tested first
    - Test Stubs – substitutes for lower level units – are needed
  - Bottom Up – bottom (simplest) modules are combined
    - Test Drivers simulate higher level modules
  - Sandwich/Hybrid – combines Top Down and Bottom Up

Two Unit Tests Zero Integration