Ira A. Fulton Schools of Engineering
Arizona State University

CSE 565: Software Verification and Validation

Specification-Based Testing Assignment

Purpose

This project will help you reinforce the topics discussed in the class including categories of testing, equivalence partitioning, boundary value, and cause and effect testing.

Objectives

Learners will be able to:

- Apply equivalence partitioning, boundary value and cause-effect testing techniques.

Technology Requirements

- Java 17 or above

Project Description

In this project, you will gain hands-on experience in software testing methodologies. You will focus on three essential techniques:

- Equivalence Partitioning: Learn how to group inputs into classes and design test cases that cover each class efficiently.
- Boundary Value Analysis: Explore the edges of input ranges to uncover potential defects and create precise test cases.
- Cause and Effect Testing: Identify input conditions and their corresponding effects to build targeted test scenarios.

Throughout the project, you will apply these techniques to a provided software program, developing a set of well-structured test cases. By executing these tests, you will validate the program's functionality and ensure it meets the desired specifications.

To complete this project, you will use equivalence partitioning, boundary value, and cause/effect analysis techniques to develop a set of test cases for the provided program and execute the tests.

Directions

Download the provided files located in assignment description page, then carefully review the directions before starting your work:

- SpecificationBasedTesting_Test Case Spreadsheet.xlsx
- Jar Files and Project1Script.zip

In this assignment, we have developed a database to gather information and provide discounts about a product and its users depending on a variety of factors given. The jar file with the program is given to you, but contains at least 10 seeded defects. Using equivalence partitioning, boundary value, and cause/effect analysis techniques discussed in the lecture, please develop a set of test cases for this program and execute the tests following the instructions below:

Running Jar Files

There are three options to run your test cases for black box testing. Please make sure to have java installed on your computer to run the jar file.

Using the GUI

- project1GUI.jar opens up the GUI where you can manually input your test cases and view the output

- To run: Open in Command Prompt or Terminal and enter “java -jar project1GUI.jar” and the GUI will open up

Using Terminal or Command Prompt

- CSE565P1.jar contains the program that runs your test cases

- The jar file takes in arguments corresponding to <name> <age> <user status> <reward member status> <season bought> <product category> <rating of the product>

    - i.e.: java -jar <PATH to jar file> 24 Returning Silver Spring Electronics 5

    - The only values that will be accepted for User Status, Rewards Member Status, Season Bought, Product Category are below.
        - User Status

            - New or Returning
        - Rewards Member Status
            - Bronze, Silver or Gold
        - Season Bought
            - Winter, Spring, Summer or Fall
        - Product Category
            - Unknown or Electronics

    - To run: Open in Command Prompt or Terminal and enter “java -jar CSE565P1.jar Peter 24 Returning Silver Spring Electronics 5” and the output will be given

Running tests automatically (Linux/ Bash for Windows/Mac)

- CSE565P1.jar contains the program that runs your test cases.

- Project1Script is a bash script where you can input your test cases, and Project1Script will automatically run them.

- To add a test case and run:

    a. Open Project1Script in vim or an editor

    b. Add in a test case by doing: java -jar <PATH to jar file> Peter 24 Returning Silver Spring Electronics 5

    c. Exit out of vim or the editor

    d. If needed, open in Bash or Terminal and make Project1Script an executable by running “chmod u+x Project1Script”

    e. In the Bash or Terminal, run “./Project1Script” and the output for each test case will be given

Requirements

Input requirements

1. User Name (string)
    1.1. Can be 5-10 characters long
    1.2. Only contain characters (no numbers)
    1.3. Cannot contain a hyphen (-) or underscore (_)

2. Age (Integer)
    2.1. Must be 18 or over

3. User Status (do not test for invalids)
    3.1. New or Returning

4. Rewards Member Status (do not test for invalids)
    4.1. Bronze, Silver or Gold

5. Season Bought (do not test for invalids)
    5.1. Winter, Spring, Summer or Fall

6. Product Category (do not test for invalids)
    6.1. Unknown or Electronics

7. Rating of Product (Integer)
    7.1. Rating must be between 1 – 10

Output requirements

8. Error messages for incorrect username, age, and rating.

9. Discounts Given based on:
    9.1. New users cannot be given any discounts regardless of reward status, product, and season

    9.2. If product category is unknown, user gets no discount

    9.3. If bought in Summer:
        9.3.1. Returning users who are gold members who bought Electronics get a 15% discount voucher

    9.4. If bought in Spring:
        9.4.1. Returning users who are bronze members who bought Electronics get a 10% discount voucher

    9.5. If bought in Winter:
        9.5.1. Returning users who are gold members who bought Electronics get a 25% discount voucher

    9.6. If bought in Fall:
        9.6.1. Returning users who are silver members who bought Electronics get a 15% discount voucher

9.7. Any other combination of returning user, reward status, product, and season does not get a discount.

Remember in equivalence partitioning, you can only test one (1) invalid partition **at a time**.

Preparing the Deliverables

Follow the steps to accurately complete this assignment:

1. Complete test case matrix (Tab 1 at the bottom of the spreadsheet) from **SpecificationBasedTesting_Test Case Spreadsheet.xls**
2. Execute test cases following one of the three options detailed above
3. Report results in test case matrix tab (Pass / Fail). Highlight Failed test cases with RED color.
4. Complete the defect tracking sheet (Tab 2 at the bottom of the spreadsheet) containing the defect number, a brief description of each defect, and the requirement that the defect maps to.
    - i.e.: If a user was able to input a rating > 10, the defect tracking spreadsheet entry would look like this:

Defect Number	Defect Description	Requirement Mapping	Test Cases Mapping
1	User allowed to input rating > 10	7	10

Submission Directions for Project Deliverables

This assignment requires submission of **one (1)** deliverable: Submit the Excel file containing the **SpecificationBasedTesting_Test Case Spreadsheet**.

When ready, title your file as **yourlastname_firstname_CSE 565_SpecificationBasedTestingAssignment** and submit to the assignment submission page.

Submission Guidelines

You may submit your deliverable as many times as needed. However, only the most recent submission will be graded.

You must submit your assignment file in the designated submission space. Learners may **not** email or use other means to submit any project for review, including feedback, and grading.

Evaluation

Your submission will be evaluated based on the criteria (**worth 80 points total**):

1. **Set of complete test cases (40 points)**: included all test cases (passing and failing test cases as per the requirements)
2. **Defects found and described (30 points)**: identified at least 10 defects and described them
3. **Requirements and test cases mapped to defects (10 points)**: mapped 10 defects