CSE 565: Software Verification and Validation

# Specification-Based Testing Assignment

**Purpose**
This project aims to reinforce key testing concepts discussed in the course, including equivalence partitioning, boundary value analysis, and cause-effect testing techniques.

**Objectives**
By completing this project, you will:

- Apply equivalence partitioning, boundary value analysis, and cause-effect testing to design and execute test cases.
- Gain hands-on experience in testing methodologies to validate a program's functionality and identify defects.

**Project Description**
This assignment provides practical exposure to software testing methodologies. You will apply three essential techniques to validate a program designed to calculate discounts based on user and product attributes. These techniques include:

1. **Equivalence Partitioning**: Grouping inputs into equivalence classes to design test cases that efficiently cover each class.
2. **Boundary Value Analysis**: Focusing on edge values in input ranges to identify potential defects.
3. **Cause-Effect Testing**: Defining input conditions and their expected outcomes to build targeted test scenarios.

The goal is to develop well-structured test cases, execute them, and analyze the results to validate the program's functionality against specified requirements. The provided program contains at least 10 seeded defects, offering an opportunity to identify and resolve issues systematically.

**Directions**

1. **Download the Materials**
   Access the files needed for this assignment from the course portal:
   - `SpecificationBasedTesting_TestCaseSpreadsheet.xlsx`
   - `Jar Files and Project1Script.zip`
2. **Understand the Program**
   The program calculates discounts based on a database of user and product attributes. You will use equivalence partitioning, boundary value analysis, and cause-effect testing techniques to develop and execute test cases.
3. **Running the Program**
   There are three methods to execute test cases:
   - **Using the GUI**:

        - Open the `project1GUI.jar` file to manually input test cases and view results.
        - Command: `java -jar project1GUI.jar`
    - **Using the Command Line**:
        - Execute the `CSE565P1.jar` file, providing inputs via command-line arguments.
        - Example:

```
java -jar CSE565P1.jar <Name> <Age> <User Status> <Reward
Status> <Season> <Product Category> <Rating>

java -jar CSE565P1.jar Peter 24 Returning Silver Spring
Electronics 5
```

    - **Using the Script for Batch Testing**:
        - Use the `Project1Script` bash script to automate test case execution.
        - Steps:
            1. Open the script and add test cases using the required command format.
            2. Save and make the script executable with: `chmod u+x Project1Script`.
            3. Run the script: `./Project1Script`.

**nput Requirements**

1. **User Name**:
    - Must be 5–10 characters long.
    - Can only contain alphabetic characters (no numbers or special characters like - or _).
2. **Age**:
    - Must be an integer value of 18 or higher.
3. **User Status**:
    - Must be either "New" or "Returning".
4. **Rewards Member Status**:
    - Must be one of: "Bronze", "Silver", or "Gold".
5. **Season Bought**:
    - Must be one of: "Winter", "Spring", "Summer", or "Fall".
6. **Product Category**:
    - Must be either "Unknown" or "Electronics".
7. **Rating**:
    - Must be an integer between 1 and 10.

---

**Output Requirements**

1. **Error Messages**:
    - Displayed for invalid username, age, or rating inputs.

2. **Discounts**:
   Discounts are determined by user and product attributes:
   - **New Users**: No discounts, regardless of other factors.
   - **Returning Users**:
     - **Summer**: Gold members buying Electronics receive a 15% discount.
     - **Spring**: Bronze members buying Electronics receive a 10% discount.
     - **Winter**: Gold members buying Electronics receive a 25% discount.
     - **Fall**: Silver members buying Electronics receive a 15% discount.
   - No discount is given for users with "Unknown" product categories or combinations not specified above.

**Testing Guidelines**

1. **Equivalence Partitioning**:
   - Divide inputs into valid and invalid partitions.
   - Test one invalid partition at a time to isolate defects.
2. **Boundary Value Analysis**:
   - Test edge cases for numerical inputs like age and rating.
3. **Cause-Effect Testing**:
   - Map input conditions to expected outcomes for thorough validation of program logic.