Assignment 1 Report on Testing Arithmetic Operators

Name: Dang Khoa Le
Student Number: 103844421
Tutorial: Tuesday 9.30-10.30 AM

Task 1: Designing Test Cases

Program under test

- Input: A, B (integer variables)
- Process: A = (A + B) * B, C = A - 5
- Output: C (integer variable)

Objective

The testing objective is to detect any incorrect use of arithmetic operators (‘+’, ‘*’, ‘-‘) in the given program. Specifically, we want to ensure:

- The correctness of the addition operator + in the expression A = (A + B) * B.
- The correctness of the multiplication operator * in the same expression.
- The correctness of the subtraction operator - in the expression C = A - 5.

Approach

1. **Define Objectives:** To achieve the testing objective, we need to verify that each operator is functioning correctly. This includes:
   - Ensuring the addition operator correctly computes the sum of A and B.
   - Ensuring the multiplication operator correctly multiplies result of the addition by B.
   - Ensuring the subtraction operator correctly subtracts 5 from the result.

2. **Design Test Cases:** Based on the objective, we should design test cases that cover different scenarios:
   - **Addition Testing:** Test cases where the sum of A and B is easily computable and verifiable.
   - **Multiplication Testing:** Test cases where multiplying the result of the addition by B produces a straightforward result.
   - **Subtraction Testing:** Test cases where subtracting 5 from the result is easy to check.

3. **Generate Test Cases:** Generate test cases where each operator's output can be verified through simple arithmetic.

4. **Execute Test Cases:** Run the test cases on the program (using Python) and compare the results with expected values.

5. **Analyse Test Results:** Check if the results match expectations to determine if any operators are incorrect.

Task 2: Testing with (A=10, B=0)

Test Case

- **Input:** A=10, B=0

1. **Calculate A:**
   - A = (10 + 0) * 0
   - A = 10 * 0
   - A = 0
2. **Calculate C:**
   - C = 0 - 5
   - C = -5

Analysis

Testing with (A=10, B=0) only verifies the multiplication operator in a specific case (when B=0). This test case does not comprehensively test the addition or subtraction operators as it fails to test addition and multiplication effectively due to the non-impactful nature of B=0 on addition and multiplication. Therefore, it is not sufficient to reveal all possible errors and assert the correctness in the use of arithmetic operators.

Task 3: Concrete Test Cases

Test Case 1

- **Input:** A = 1, B = 2
- **Expected Results:**
  - A = (1 + 2) * 2 = 3 * 2 = 6
  - C = 6 - 5 = 1

Test Case 2

- **Input:** A = -1, B = -2
- **Expected Results:**
  - A = (-1 + (-2)) * (-2) = (-3) * (-2) = 6
  - C = 6 - 5 = 1

Test Case 3

- **Input:** A = 0, B = 1
- **Expected Results:**
  - A = (0 + 1) * 1 = 1 * 1 = 1
  - C = 1 - 5 = -4

Justification

These test cases cover different scenarios:

- **Positive integers:** Check if the addition and multiplication produce correct results.
- **Negative integers:** Validate if negative values are handled properly.
- **Zero values:** Verify how zero values affect the multiplication and subtraction operations.

These test cases are designed to examine how the program handles positive values, negative values, and zero, which are essential for evaluating all specified operations under different circumstances, addressing both typical and edge cases.

Task 4: Values of A for B=2

To determine values of A that cannot achieve the testing objective for B=2, we can simplify and test the following:

1. **Expression with B=2:**
    - A = (A + 2) * 2
    - C = A - 5

2. **Test Case Validity:** To not achieve testing objective, the values of A must be such that:
    - The addition and multiplication produce the expected result.
    - The result for subtraction is straightforward.

Investigation Range: Testing values of A from -4 to 2.

Possible Values

Test for edge cases like:

- A = 2
    - A = (2 + 2) * 2 = 8
    - C = 8 - 5 = 3
- A = 1
    - A = (1 + 2) * 2 = 6
    - C = 6 - 5 = 1
- A = 0
    - A = (0 + 2) * 2 = 4
    - C = 4 - 5 = -1
- A = -1
    - A = (-1 + 2) * 2 = 2
    - C = 2 - 5 = -3
- A = -2
    - A = (-2 + 2) * 2 = 0
    - C = 0 - 5 = -5
- A = -3
    - A = (-3 + 2) * 2 = -2
    - C = -2 - 5 = -7
- A = -4
    - A = (-4 + 2) * 2 = -4
    - C = -4 - 5 = -9

Testing revealed that values from A = -4 to A = 2 when processed with B = 2 yield predictable and correct results, suggesting these values do not reveal operator malfunctions under these conditions.

If all test cases result in correct values for C as expected, then values of A that don't produce errors in calculations are acceptable.

Conclusion

- **Task 1:** Detailed test case design approach to ensure coverage of all arithmetic operators. Thoroughly prepared test cases ensuring exhaustive operator validation.
- **Task 2:** The chosen test case (A=10, B=0) is insufficient for comprehensive testing.
- **Task 3:** Concrete test cases provided effectively cover various scenarios.
- **Task 4:** Detailed examination of a range of A values for a fixed B when B=2 ensures testing for edge cases, allow affirming the program's reliability across specified inputs.