

## Test Case 1: Large Multiplication

- **Test Case ID:** TC001
  - **Test Description:** Verify multiplication of two large positive numbers.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input 12345 .
    2. Press × .
    3. Input 6789 .
    4. Press = .
  - **Expected Result:** The result displayed is 83810205 .
- 

## Test Case 2: Subtraction with Negative Result

- **Test Case ID:** TC002
  - **Test Description:** Verify subtraction resulting in a negative number.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input 5 .
    2. Press - .
    3. Input 8 .
    4. Press = .
  - **Expected Result:** The result displayed is -3 .
- 

## Test Case 3: Multiplication with Negative Second Operand

- **Test Case ID:** TC003
  - **Test Description:** Verify behavior when the second operand in multiplication is negative.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input 6 .
    2. Press × .
    3. Input -3 .
    4. Press = .
  - **Expected Result:** Usually,  $6 \times -3$  would give -18 . However, here the operator automatically switches to subtraction, and the result incorrectly displays 3 (since  $6 - 3 = 3$  ).
- 

## Test Case 4: Division by Zero

- **Test Case ID:** TC004
  - **Test Description:** Verify behavior when dividing a number by zero.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input 9 .
    2. Press ÷ .
    3. Input 0 .
    4. Press = .
  - **Expected Result:** The result displayed is Infinity .
-

---

## Test Case 5: Modulo with Large Positive Operands

- **Test Case ID:** TC005
  - **Test Description:** Verify modulo operation with larger positive integers.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input 123 .
    2. Press % .
    3. Input 45 .
    4. Press = .
  - **Expected Result:** The result displayed is 33 .
- 

## Test Case 6: Consecutive Operators

- **Test Case ID:** TC006
  - **Test Description:** Verify behavior when two operators are pressed consecutively.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input 8 .
    2. Press + .
    3. Press × .
    4. Input 3 .
    5. Press = .
  - **Expected Result:** The second operator ( × ) overrides the first ( + ), and the result displayed is 24 ( 8 × 3 ).
- 

## Test Case 7: Operator Directly Followed by Equal to with One Operand

- **Test Case ID:** TC007
  - **Test Description:** Verify behavior when an operator is followed directly by = without providing a second operand.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input 7 .
    2. Press + .
    3. Press = .
  - **Expected Result:**

The calculator performs 7 + 7 and displays 14 . This behavior applies similarly for other operators like × , ÷ , % , etc.
- 

## Test Case 8: Operator Directly Followed by Equal to with Multiple Operands

- **Test Case ID:** TC008
- **Test Description:** Verify behavior when an operator is followed directly by = in a case with multiple operands and operators.
- **Preconditions:** Calculator is reset.
- **Test Steps:**

1. Input 5 .
2. Press + .
3. Input 5 .
4. Press + .
5. Input 5 .
6. Press - .
7. Press = .

- **Expected Result:**

The calculator treats  $5 + 5 + 5$  as 15 (intermediate result) and then subtracts this result from itself ( $15 - 15$ ), displaying 0 .

---

## Test Case 9: Decimal Multiplication with Negative First Operand

- **Test Case ID:** TC009
  - **Test Description:** Verify multiplication involving a negative decimal number as the first operand.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input -5.5 .
    2. Press  $\times$  .
    3. Input 2.2 .
    4. Press = .
  - **Expected Result:** The result displayed is -12.1 .
- 

## Test Case 10: Negative First Operand with Division

- **Test Case ID:** TC010
  - **Test Description:** Verify behavior when the first operand is negative and operation is division.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input -10 .
    2. Press  $\div$  .
    3. Input 2 .
    4. Press = .
  - **Expected Result:** The result displayed is -5 .
- 

## Test Case 11: Left to Right Execution with 2 Operators

- **Test Case ID:** TC011
- **Test Description:** Verify left-to-right execution when there are 2 or fewer arithmetic operators.
- **Preconditions:** Calculator is reset.
- **Test Steps:**
  1. Input 10 .
  2. Press + .
  3. Input 5 .
  4. Press  $\times$  .
  5. Input 2 .

6. Press `=` .

- **Expected Result:** The result displayed is `30` . The calculator evaluates from left to right:
    - First,  $10 + 5 = 15$  , then
    - $15 \times 2 = 30$  .
- 

## Test Case 12: BODMAS with 3 Operators

- **Test Case ID:** TC012
  - **Test Description:** Verify BODMAS is applied correctly when there are more than 2 arithmetic operators.
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input `2` .
    2. Press `+` .
    3. Input `6` .
    4. Press `×` .
    5. Input `5` .
    6. Press `/` .
    7. Input `5` .
    8. Press `=` .
  - **Expected Result:**

The result displayed is `6.4` .

    - BODMAS is applied for the initial operations ( $6 \times 5 = 30$  and  $2 + 30 = 32$  ), and the last operator is executed separately ( $32 / 5 = 6.4$  ).
- 

## Test Case 13: Invalid Modulo Operation Resulting in NaN

- **Test Case ID:** TC013
  - **Test Description:** Verify behavior when performing invalid modulo operations like `0 % 0 =` or `5 % 5 % =` , resulting in `NaN` .
  - **Preconditions:** Calculator is reset.
  - **Test Steps:**
    1. Input `5` .
    2. Press `%` .
    3. Input `5` .
    4. Press `%` .
    5. Press `=` .
  - **Expected Result:** The result displayed is `NaN` (Not a Number).
-