# EECS 348

# Arithmetic Expression Evaluator

## Test Case

**Version 3.0**

# Revision History

| Date | Version | Description | Author |
|---|---|---|---|
| 11/28/2023 | 1.0 | Create document | Marcos Lepage |
| 11/30/2023 | 2.0 | Edited Document | Dylan O'Brien |
| 12/3/2023 | 3.0 | Updated Test Cases | Marcos Lepage, Andrew Ward |
| | | | |

# Table of Contents

# Test Case

## 1. Purpose

The purpose of the Test Case Specification document is to ensure that all the mathematical operations for the calculator including addition, subtraction, multiplication, division, modulo, and exponentiation work as expected and produce the correct results. Additionally, the document serves to validate any advanced functions that involve any uncommon conditions to ensure the calculator's functionality meets the user's expectations.

## 2. Test case identifier

Refer to the table below for details.

## 3. Test item

Refer to the table below for details.

## 4. Input specifications

Refer to the table below for details.

## 5. Output specifications

| Test Case ID | Test Case Description | Test Data | Expected Results | Actual Results | Pass/ Fail |
| --- | --- | --- | --- | --- | --- |
| TC01 | Test the addition of two numeric constants. | 3+4 | 7 | 7 | Pass |
| TC02 | Test the subtraction of an expression with parentheses, ensuring valid precedence. | 8-(5-2) | 5 | 5 | Pass |
| TC03 | Test multiplication and division operators are applied from left to right. | 10*2/5 | 4 | 4 | Pass |
| TC04 | Test the "^" operator calculates the exponentiation of 2 raised to the power of 3. | 2^3 | 8 | 8 | Pass |
| TC05 | Test the combination of multiple operators and parentheses ensuring correct precedence. | 4*(3+2)%7-1 | 5 | 5 | Pass |
| TC06 | Test that multiple sets of extraneous parentheses do not affect the validity of the expression. | (((2+3)))+(((1+2))) | 8 | 8 | Pass |

| | | | | | |
|---|---|---|---|---|---|
| TC07 | Test the combination of various operators with multiple sets of extraneous parentheses do not change the order or operations of the expression. (User's Manual) | ((5*2)-((3/1)+((4%3)))) | 6 | 6 | Pass |
| TC08 | Test the use of nested parentheses and exponentiation, ensuring the correct order of operations. (User's Manual). | (((2^(1+1))+((3-1)^2))/((4/2)%3)) | 4 | 4 | Pass |
| TC09 | Test both extraneous parentheses and necessary parentheses to clarify the order of operations. | (((((5-3)))*(((2+1)))+((2*3))) | 12 | 12 | Pass |
| TC10 | Test the use of extraneous parentheses for clarity, ensuring the validity of the expression remains. | ((9+6))/((3*1)/(((2+2)))-1) | -60 | -60 | Pass |
| TC11 | Test the combination of unary "+" and "-" operators with other various operators. (User's Manual) | +(-2)*(-3)-((-4)/(+5)) | 6.8 | 6.8 | Pass |
| TC12 | Test unary negation and addition operators used within parentheses. | -(+1)+(+2) | 1 | 1 | Pass |
| TC13 | Test nested unary negations and additions, ensuring some values are negated and others added. | -(-(-3))+(-4)+(+5) | -2 | -2 | Pass |
| TC14 | Test unary "+" and "-" operators with exponentiation to calculate a fractional result. | +2^(-3) | 0.125 | 0.125 | Pass |
| TC15 | Test the combination of unary operators with parentheses and arithmetic operations. | -(+2)*(+3)-(-4)/(-5) | -6.8 | -6.8 | Pass |
| TC16 | This expression has unmatched opening and closed parentheses, making it invalid. | 2*(4+3-1 | Error | [ERROR]: Open parenthesis is not closed. | Pass |
| TC17 | This expression has unmatched opening and | 2*4+3-1) | Error | [ERROR]: Closed | Pass |

| | | | | | |
|---|---|---|---|---|---|
| | closed parentheses, making it invalid. | | | parenthesis is never opened. | |
| TC18 | The “*” operator lacks operands on the left, making the expression invalid. | *5+2 | Error | [ERROR]: Operator left-hand is empty. | Pass |
| TC19 | Division by zero is undefined in mathematics, so this expression is invalid. | 4/0 | Error | [ERROR]: Divide by zero. | Pass |
| TC20 | The expression lacks an operator between 5 and (2+3), making it invalid. | 5(2+3) | Error | [ERROR]: Operands not separated by operator. | Pass |
| TC21 | The “&” character is not a valid arithmetic operator, so this expression is invalid in the context of arithmetic operators. | 7&3 | Error | [ERROR]: Unrecognized symbols in expression. | Pass |
| TC22 | The parentheses are not properly matched, with one closing parenthesis missing, making the expression invalid. | (((3+4)-2)+(1) | Error | [ERROR]: Open parenthesis is not closed. | Pass |
| TC23 | This expression attempts to divide by zero, which is mathematically undefined, resulting in an invalid expression. | ((5+2)/(3*0)) | Error | [ERROR]: Divide by zero. | Pass |
| TC24 | The expression contains an operator “-“ without a valid operand on its left, making it invalid. | ((2-)1+3) | Error | [ERROR]: Operands not separated by operator. | Pass |
| TC25 | The expression is missing an operand after the “-“ operator, making it invalid. | ((4*2)+(-)) | Error | [ERROR]: Operator right-hand is empty. | Pass |
| TC26 | The “@” character is not a valid arithmetic operator in this context, causing the expression to be invalid. | ((7*3)@2) | Error | [ERROR]: Unrecognized symbols in expression. | Pass |
| TC27 | The trailing parenthesis group is empty making the expression invalid. | 123 + () | Error | [ERROR]: Expression group has no content. | Pass |
| TC28 | Unary operators that have no right-hand operand are invalid. | 5*2+ | Error | [ERROR]: Operator right-hand is empty. | Pass |

| | | | | | |
|---|---|---|---|---|---|
| TC29 | The program should handle symbol overlap between “**” and “*” normally. | 25 *** 3 | Error | [ERROR]: Operator right-hand is empty. | Pass |
| TC30 | Very large evaluations should be handled in a meaningful way. | 1000 ^ 1000 | inf | inf | Pass |
| TC31 | Brackets, like parenthesis, should be valid grouping operators. | [12 + 3] * 3 | 45 | 45 | Pass |
| TC32 | Brackets and parenthesis should be able to coexist while still being discrete. | (2 * [7 + 3]) ^ 2 | 400 | 400 | Pass |
| TC33 | Both “**” and “^” should be valid symbols for the exponentiation operator. | (2 ** 3) ^ 2 | 64 | 64 | Pass |
| TC34 | Unary operators should be indefinitely chainable. | 123+-+-+2 | 125 | 125 | Pass |
| TC35 | Exponentiation by a fraction should be valid to allow for square rooting. | 4 ** 0.5 + 13 | 15 | 15 | Pass |
| TC36 | Performing the modulo operation on floats should yield non-truncated results. | 256.5 % 16 + 17 | 17.5 | 17.5 | Pass |
| TC37 | Test for both extraneous brackets and parenthesis at once. | [([[1+3]])*4]+3 | 19 | 19 | Pass |
| TC38 | Allow unary operator chaining on parenthesis group. | ---(3 * 2) + 5 | -1 | -1 | Pass |
| TC39 | Whitespace gaps shouldn’t influence expression evaluation. | ( 3 * 5 ) + 1 | 16 | 16 | Pass |
| TC40 | Non-space whitespace should not influence expression evaluation (note the 4-character gaps here are tabs). | (6 * 2) + 1 | 13 | 13 | Pass |

## 6. Environmental needs

### *6.1.1* *Hardware*

N/A

### *6.1.2* *Software*

N/A

### *6.1.3* *Other*

None

## 7. Special procedural requirements

N/A

## 8. Intercase dependencies

N/A