Group 12

Calculator

Test Case

Version <1.4>

Revision History

Date	Version	Description	Author
04/December/24	1.1	Started making test cases	Maxfield Freeman
07/December/24	1.2	Continued making test cases	Maxfield Freeman
09/December/24	1.3	Validated Test Cases	Maren Proplesch
11/December/24	1.4	Finalized Document	Maren Proplesch

Table of Contents

Test Case

1. Purpose

This Test Case Specification document for the Calculator defines a test case for an item that should be tested. The provided test cases account for any scenario that a user may come across while using the calculator. These scenarios include all of the provided arithmetic functions, including modulus. As well as any expected errors that the user may encounter. Such expected errors include but are not limited to, dividing by zero, improper input, and unicode expressions. This documentation also provides a simple explanation as well and expected inputs and outputs for over 30 test cases.

2. Test case identifier

The test case identifier follows a simple pattern, beginning with “TC” which stands for test case and then a for further identification. While these numbers do not have a specific meaning, numbers that are close to each other represent similar test cases. Such as numbers TC-01 through TC-06 testing each of the implemented arithmetic expressions, and TC-17 through TC-19 testing the behavior on multi-digit inputs and how they react to positive and negative inputs

3. Test item

ID	Name	Purpose	Inputs	Expected Outputs	Observed Outputs	Pass/Fail Y/N
TC-01	Addition	Test the (+) operator	5+6+8	19	19	Y
TC-02	Space management	Test the (+) operator with spaces	5 + 6 + 8	19	19	Y
TC-03	Subtraction	Test the (-) operator	125 -13 -9	103	103	Y
TC-04	Multiplication	Test the (*) operator	120*12	1440	1440	Y
TC-05	Division	Test the (/) operator	187/5	37.4	37.4	Y
TC-06	Modular	Test the (%) operator	18%4	2	2	Y
TC-07	PEMDAS	The the order of operations	8+17/2*3-7	26.5	26.5	Y
TC-08	Exponent with large number	Test the (**) operator with large number	3**20	3486784401	3.48678e+09	Y

TC-09	Exponent	Test the (**) operator	2**6	64	64	Y
TC-10	Testing parenthesis	Test () in the order of operations	4*(8-2)	24	24	Y
TC-11	Testing multiple parenthesis	Test if multiple () works	(4-5)*(7%2)+((8))	7	7	Y
TC-12	So many parenthesis	Test 8 parentheses around a number	(((((((4)))))))	4	4	Y
TC-13	Exponent followed by multiplication	Test if the *** raises and error	2***5	Syntax Error	Syntax Error	Y
TC-14	negative Multiplication	Test 2 negative numbers multiplied	-2*-2	4	4	Y
TC-15	Positive single-digit constant	Tests if calculator returns positive single character inputs correctly	2	2	2	Y
TC-16	Negative single-digit constant	Tests if calculator returns negative single character inputs correctly	-2	-2	-2	Y
TC-17	Positive multi-digit constant	Tests if calculator returns positive multi character inputs correctly	22	22	22	Y
TC-18	Negative multi-digit constant	Tests if calculator returns negative multi character inputs correctly	-22	-22	-22	Y
TC-19	Multi-negative digit	Tests if calculator can handle multiple negative’s	------22	22	22	Y

TC-20	Multi-operator	Tests if calculator can handle multiple operators adjacent to each other	8+*2	Syntax Error	Syntax Error	Y
TC-21	Multi-operator	Tests if calculator can handle multiple operators adjacent to each other	2-+8	Syntax Error	Syntax Error	Y
TC-22	Unary Operation	Testing if the calculator can handle unary operations	-(+1)+(+2)	1	1	Y
TC-23	Multi-operation	Tests if calculator can handle multiple operators adjacent to each other	8*/2	Syntax Error	Syntax Error	Y
TC-24	Unary Operation	Tests if the calculator can handle unary operations	+1++2	Syntax Error	Syntax Error	Y
TC-25	operator in parenthesis	Test if program correctly runs parenthesis	8(*)2	Syntax Error	Syntax Error	Y
TC-26	Modulo edge case	Test if program can modulo by a bigger number	9 % 51	9	9	Y
TC-27	Modulo one	Test if the edge case of modulo 1 works	8 % 1	0	0	Y
TC-28	Division by 0	Test if division by 0 is handled gracefully	6/0	Error: Division by 0	Error: Division by zero	Y
TC-29	Invalid unicode inputs	Tests if calculator correctly returns an error if unicode	‘8+9’	Syntax Error	Syntax Error	Y

		characters like a 0 width space are included				
TC-30	Invalid multiplication	Tests if invalid character in multiplication	2 * A	Syntax Error	Syntax Error	Y
TC-31	Invalid divisor	Tests if invalid character in divisor	9 / B	Syntax Error	Syntax Error	Y
TC-32	Invalid Modulo	Tests if invalid character in modulo	A % 6	Syntax Error	Syntax Error	Y
TC-33	Invalid Modulo	Test if modulo integer is 0	2 % 0	Invalid Modulo	Error: Invalid Modulo by zero	Y
TC-34	Invalid Addition	Tests if invalid characters produce an error	A + B	Syntax Error	Syntax Error	Y
TC-35	Invalid Input	Tests if a string of valid and invalid inputs produce an error	A - 2 + 65 * B	Syntax Error	Syntax Error	Y
TC-36	Invalid Input	Tests if invalid character produces an output	A	Syntax Error	Syntax Error	Y
TC-37	Template expression	Test if template expression works	5+(6*(9**8))/2	129140168	1.2914e+08	Y
TC-38	Template expression without parentheses	Test if template expression works without parenthesis	5 + 6 * 9**8 / 2	129140168	1.2914e+08	Y
TC-39	Multiple divisors	Test to see if multiple divisors work correctly	6/4/2	0.75	0.75	Y
TC-40	Divisor with addition	Test to see if divisor with addition works correctly	6/4+2	3.5	3.5	Y

4. Input specifications

The users input must be similar to the following format: 5+(6*(9**8))/2

5. Output specifications

The output of the program will either be a number, such as 5, or an error message displaying why the equation is wrong.

6. Environmental needs

6.1.1 Other

The user must have a computer that is able to download the source code and a C++ compiler..