< Workshop 4 – Reflection >

Name : Ahram Lee

Email : alee239@myseneca.ca

Student ID : 133849232

Date: 2024-05-31

Bug 1: Infinite Recursion in ‘factorial’

**Original Line(s) Containing the Bug:**

return (n <= n) ? n \* factorial(n - 1) : 1;

**Corrected Line(s):**

return (n > 1) ? n \* factorial(n - 1) : 1;

**Explanation:**

The condition n <= n is always true, which leads to infinite recursion and eventually a stack overflow. The correct condition should be n > 1 to ensure that the recursion stops when n is 1.

**Technique Used to Find the Bug:**

Static analysis and reasoning about the logic of the factorial calculation. Understanding that factorial should return 1 when n is 1 or less helped identify the incorrect condition.

Bug 2: Division by Zero in ‘reduceFactorial’

**Original Line(s) Containing the Bug:**

return n / n;

**Corrected Line(s):**

return (n > 0) ? (n / n) : 1;

**Explanation:**

The original line n / n would result in a division by zero when n is 0. The corrected line ensures that if n is greater than 0, it returns 1 (since any non-zero number divided by itself is 1); otherwise, it returns 1 to avoid division by zero.

**Technique Used to Find the Bug:**

Static analysis and understanding of the potential for division by zero in arithmetic operations.

Bug 3: Incorrect Structure Modification in ‘computeFactorials’

**Original Line(s) Containing the Bug:**

void computeFactorials(struct FactorialResults results, int numFactorials)

**Corrected Line(s):**

void computeFactorials(struct FactorialResults\* results, int numFactorials)

**Explanation:**

Passing the results structure by value does not modify the original structure in main. By changing the parameter to a pointer (struct FactorialResults\*), the function now modifies the original structure.

**Technique Used to Find the Bug:**

Understanding of C pointers and memory management, along with debugging the behavior of the program when results was not updating as expected.

Bug 4: Incorrect Access of Structure Members in ‘computeFactorials’

**Original Line(s) Containing the Bug:**

results.results[i] = factorial(i);

**Corrected Line(s):**

results->results[i] = factorial(i);

**Explanation:**

Accessing members of a pointer to a structure requires the -> operator instead of the . operator. This correction ensures proper access to the members of the results structure.

**Technique Used to Find the Bug:**

Knowledge of C syntax for accessing members of structures through pointers.

Bug 5: Incorrect Format Specifier in printf

**Original Line(s) Containing the Bug:**

printf("%5d %12f\n", i, results.results[i]);

**Corrected Line(s):**

printf("%5d %12d\n", i, results.results[i]);

**Explanation:**

The format specifier %12f is for floating-point numbers, while results.results[i] is an integer. Changing %12f to %12d correctly formats the integer value.

**Technique Used to Find the Bug:**

Understanding of printf format specifiers and their appropriate use for different data types.