Title: Counter Directive
Author: Marcus Johnson
Company: Mad Scientist
Date: July 1st 2025
Document: N3636
Proposal Category: New Feature
Audience: Compiler authors, Library authors, application authors

Abstract:
Allow programmers to create named preprocessor counters

Rationale:
__COUNTER__ is a commonly supported extension, but it can only be used in one context per translation unit, by naming our counters we can have as many of them as we want in our translation units.

Syntax:
#counter NAME_OF_COUNTER <initial value as integer literal OR object like macro which expands to a positive integer literal>

Semantics:
NAME_OF_COUNTER is a object-like macro that hasn't been previously declared, it's value is expanded every time it's encountered to yield a positive integer literal, it's type is size_t

Example:

```
#include <stdio.h>

// Define counters for test cases and suites
#counter NUM_TEST_SUITES 0
#counter NUM_TEST_CASES 0

// Macro to declare a test case with a unique ID
#define ADD_TEST_CASE(name) \
    static const int name##_id = NUM_TEST_CASES; \
    void name() { printf("Running Test Case %s (ID %d)\n", #name, name##_id); }

// Macro to declare a test suite with a unique ID
#define ADD_TEST_SUITE(name) \
    static const int name##_id = NUM_TEST_SUITES; \
    void name() { printf("Running Test Suite %s (ID %d)\n", #name, name##_id); }
```

```
// Declare some test cases and suites
ADD_TEST_SUITE(unit_tests);
ADD_TEST_SUITE(integration_tests);
ADD_TEST_CASE(test_workflow);

int main() {
   unit_tests();
   integration_tests();
   test_workflow();
   return 0;
}
```