4. Return Statement

4.1 Basic Return Usage

The `return` statement serves two purposes:

1. **Return control** to the calling function
2. **Return a value** (optional, depending on function type)

```
// Function that returns a value
int get_maximum(int a, int b) {
    if (a > b) {
        return a;
    } else {
        return b;
    }
}

// Function that returns without a value
void print_status(int score) {
    if (score < 0) {
        printf("Invalid score!\n");
        return;  // Early exit
    }

    if (score >= 60) {
        printf("Passed!\n");
    } else {
        printf("Failed!\n");
    }
    // Implicit return at end of void function
}
```

4.2 Multiple Return Statements

A function can have multiple return statements, but only one will execute:

```
char determine_grade(int score) {
    if (score >= 90) {
        return 'A';
    }
    if (score >= 80) {
        return 'B';
    }
    if (score >= 70) {
        return 'C';
    }
    if (score >= 60) {
        return 'D';
    }
    return 'F';  // Default case
}
```

4.3 Returning Different Data Types

```
#include <stdio.h>

// Return integer
int get_absolute(int num) {
    if (num < 0) {
        return -num;
    } else {
        return num;
    }
}

// Return float
float celsius_to_fahrenheit(float celsius) {
    return (celsius * 9.0 / 5.0) + 32.0;
}

// Return character
char get_letter_grade(float percentage) {
    if (percentage >= 85.0) return 'A';
    if (percentage >= 75.0) return 'B';
    if (percentage >= 65.0) return 'C';
    if (percentage >= 50.0) return 'D';
```

```
    return 'F';
}

int main() {
    printf("Absolute value of -15: %d\n", get_absolute(-15));
    printf("25°C in Fahrenheit: %.1f°F\n", celsius_to_fahrenheit(25.0));
    printf("Grade for 78.5%%: %c\n", get_letter_grade(78.5));
    return 0;
}
```

Revision #1
Created 2025-09-06 10:17:39 UTC by DS
Updated 2025-09-06 10:19:12 UTC by DS