# CWE Detail – CWE-484

## Description

The product omits a break statement within a switch or similar construct, causing code associated with multiple conditions to execute. This can cause problems when the programmer only intended to execute code associated with one condition.

## Extended Description

This can lead to critical code executing in situations where it should not.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Alter Execution Logic — Notes: This weakness can cause unintended logic to be executed and other unexpected application behavior.

## Potential Mitigations

**•** Implementation: Omitting a break statement so that one may fall through is often indistinguishable from an error, and therefore should be avoided. If you need to use fall-through capabilities, make sure that you have clearly documented this within the switch statement, and ensure that you have examined all the logical possibilities. (Effectiveness: N/A)

**•** Implementation: The functionality of omitting a break statement could be clarified with an if statement. This method is much safer. (Effectiveness: N/A)

## Applicable Platforms

**•** C (Class: None, Prevalence: Undetermined)

**•** C++ (Class: None, Prevalence: Undetermined)

**•** Java (Class: None, Prevalence: Undetermined)

**•** C# (Class: None, Prevalence: Undetermined)

**•** PHP (Class: None, Prevalence: Undetermined)

## Demonstrative Examples

**•** Both examples do not use a break statement after each case, which leads to unintended fall-through behavior. For example, calling "printMessage(10)" will result in the text "OctoberNovemberDecember is a great month" being printed.