EXP00-C. Use parentheses for precedence of operation

C programmers commonly make errors regarding the precedence rules of C operators because of the unintuitive low-precedence levels of &, |, ^, <<, and >>. Mistakes regarding precedence rules can be avoided by the suitable use of parentheses. Using parentheses defensively reduces errors and, if not taken to excess, makes the code more readable.

Subclause 6.5 of the C Standard defines the precedence of operation by the order of the subclauses.

Noncompliant Code Example

The intent of the expression in this noncompliant code example is to test the least significant bit of x:

```
x & 1 == 0
```

Because of operator precedence rules, the expression is parsed as

```
x & (1 == 0)
```

which evaluates to

```
(x & 0)
```

and then to 0.

Compliant Solution

In this compliant solution, parentheses are used to ensure the expression evaluates as expected:

```
(x & 1) == 0
```

Exceptions

EXP00-C-EX1: Mathematical expressions that follow algebraic order do not require parentheses. For instance, in the expression

```
x + y * z
```

the multiplication is performed before the addition by mathematical convention. Consequently, parentheses to enforce the algebraic order would be redundant:

```
x + (y * z)
```

Risk Assessment

Mistakes regarding precedence rules may cause an expression to be evaluated in an unintended way, which can lead to unexpected and abnormal program behavior.

Recommendation	Severity	Likelihood	Remediation Cost	Priority	Level
EXP00-C	Low	Probable	Medium	**P4**	**L3**

Automated Detection

Tool	Version	Checker	Description
Axivion Bauhaus Suite	6.9.0	**CertC-EXP00**	Fully implemented
CodeSonar	5.2p0	**LANG.STRUCT.PARENS**	Missing Parentheses
ECLAIR	1.2	**CC2.EXP00**	Fully implemented

Klocwork	2018	**MISRA.EXPR.PARENS.2012**	
LDRA tool suite	9.7.1	**361 S, 49 S**	Fully implemented
Parasoft C/C++test	10.4.2	**CERT_C-EXP00-a**	Use parenthesis to clarify expression order if operators with precedence lower than arithmetic are used
Polyspace Bug Finder	R2019b	CERT C: Rec. EXP00-C	Checks for possible unintended evaluation of expression because of operator precedence rules (rec. fully covered)
PRQA QA-C	9.7	**3389** **3390** **3391** **3392** **3393** **3394** **3395** **3396** **3397** **3398** **3399** **3400**	Fully implemented
PVS-Studio	6.23	**V502**, **V593**, **V634**, **V648**	
SonarQube C/C++ Plugin	3.11	**S864**	

Related Vulnerabilities

Search for vulnerabilities resulting from the violation of this rule on the CERT website.

Related Guidelines

SEI CERT C++ Coding Standard	VOID EXP00-CPP. Use parentheses for precedence of operation
ISO/IEC TR 24772:2013	Operator Precedence/Order of Evaluation [JCW]
MISRA C:2012	Rule 12.1 (advisory)

Bibliography

[Dowd 2006]	Chapter 6, "C Language Issues" ("Precedence," pp. 287–288)
[Kernighan 1988]	
[NASA-GB-1740.13]	Section 6.4.3, "C Language"

← ↑ →