STR09-C. Don't assume numeric values for expressions with type plain character

For portable applications, use only the assignment = operator, the equality operators == and !=, and the unary & operator on plain-character-typed or plain-wide-character-typed expressions.

This practice is recommended because the C Standard requires only the digit characters (0–9) to have consecutive numerical values. Consequently, operations that rely on expected values for plain-character- or plain-wide-character-typed expressions can lead to unexpected behavior.

However, because of the requirement for digit characters, other operators can be used for them according to the following restrictions:

- The binary + operator may be used to add integer values 0 through 9 to '0'.
- The binary - operator may be used to subtract character 0.
- Relational operators <, <=, and >= can be used to check whether a character or wide character is a digit.

Character types should be chosen and used in accordance with STR04-C. Use plain char for characters in the basic character set.

Noncompliant Code Example

This noncompliant code example attempts to determine if the value of a character variable is between 'a' and 'c' inclusive. However, because the C Standard does not require the letter characters to be in consecutive or alphabetic order, the check might not work as expected.

```c
char ch = 'b';
if ((ch >= 'a') && (ch <= 'c')) {
    /* ... */
}
```

Compliant Solution

In this example, the specific check is enforced using compliant operations on character expressions:

```c
char ch = 't';
if (((ch == 'a') || (ch == 'b')) || (ch == 'c')) {
    /* ... */
}
```

Exceptions

STR09-C-EX1: Consecutive values for characters like a–z can be assumed on platforms where ASCII or Unicode is used. This recommendation is primarily concerned with platform portability, for example, if code is migrated from ASCII systems to non-ASCII systems.

Risk Assessment

<table>
<thead>
<tr>
<th>Rule</th>
<th>Severity</th>
<th>Likelihood</th>
<th>Remediation Cost</th>
<th>Priority</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>STR09-C</td>
<td>Low</td>
<td>Unlikely</td>
<td>Low</td>
<td>P3</td>
<td>L3</td>
</tr>
</tbody>
</table>

Automated Detection

<table>
<thead>
<tr>
<th>Tool</th>
<th>Version</th>
<th>Checker</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axivion Bauhaus Suite</td>
<td>6.9.0</td>
<td>CertC-STR09</td>
<td></td>
</tr>
<tr>
<td>LDRA tool suite</td>
<td>9.7.1</td>
<td>329 S</td>
<td>Fully implemented</td>
</tr>
<tr>
<td>Parasoft C /C++test</td>
<td>10.4.2</td>
<td>CERT_C-STR09-a</td>
<td>Expressions with type (plain) char and wchar_t shall not be used as operands to built-in operators other than =, ==, != and the unary &amp; operator</td>
</tr>
<tr>
<td>PRQA QA-C</td>
<td>9.7</td>
<td>2106</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2107</td>
<td></td>
</tr>
</tbody>
</table>
Related Guidelines

SEI CERT C++ Coding Standard: VOID STR07-CPP. Don't assume numeric values for expressions with type plain character

Bibliography

[Jones 2009] Section 5.2.1, "Character Sets"