Document Number: X3J16/97-0093
WG21/N1131
Date: November 11, 1997
Project: Programming Language C++
Reply to: Pete Becker
petebecker@acm.org

# Library Glitches

The library portion of the current working paper contains a few errors where changes approved by the committee did not make it into the working paper. This document indicates what the working paper currently says and what the approved changes were.

## char_traits copy operations

Clause 21.1.2 [lib.char.traits.require], in Table 37, says that char_traits::move yields s+n, that is, a pointer to the next character after the moved string. The committee approved a return value of s, a pointer to the start of the moved string, for consistency with strcpy.

Clause 21.1.2 [lib.char.traits.require], in Table 37, says that char_traits::assign(s,n,c) returns s, a pointer to the start of the target string. This is correct.

Clause 21.1.2 [lib.char.traits.require], in Table 37, says that char_traits::copy yields s+n. This should also be s, again for consisency with C.

***Proposed resolution:*** change the 'yields' requirement for char_traits::move and char_traits::copy from 's+n' to 's'.

## wstreamoff

Clause 21.1.4.2 [lib.char.traits.specializations.wchar_t] uses `wstreamoff` in the definition of char_traits<wchar_t>, and mentions it again in paragraphs 1, 3, and 4. It is not used anywhere else in the working paper, and it is not needed. It has a checkered past of being removed and reinstated a number of times, but the most recent resolution was to remove it. This requires the following changes:

***Proposed resolution:***
In the definition of char_traits<wchar_t>, change the line

```
typedef wstreamoff off_type;
```

to

```
typedef streamoff off_type;
```

Remove it from paragraph 1.
Remove paragraph 3.

Change the opening phrase of paragraph 4 from

> The pairs of types `streampos` and `wstreampos`, and `streamoff` and `wstreamoff` may be different ...

to

> The types `streampos` and `wstreampos` may be different ...

## char_traits::get_state

Clause 21.1.2 [lib.char.traits.require], in Table 37, defines the member function char_traits::get_state. This member function is also mentioned in clause 21.1.4.1 [lib.char.traits.specialization.char] in the definition of char_traits<char>, and in clause 21.1.4.2 [lib.char.traits.specialization.wchar_t] in the definition of char_traits<wchar_t>. It is not mentioned anywhere else in the working paper. It was removed by vote of the committee.

***Proposed resolution:*** remove the row in Table 37 that requires char_traits::get_state and remove the two lines in char_traits<char> and char_traits<wchar_t> that define it.

## basic_string constructor (not a glitch, just an error)

In clause 21.3.1 [lib.string.cons], paragraph 7 says that the constructor `basic_string(const charT* s, size_type n,const Allocator& a = Allocator())` throws `out_of_range` if n == npos. Here n is the length of a string, and the correct exception is `length_error`.

***Proposed resolution:*** in clause 21.3.1/7, change `out_of_range` to `length_error`.