Appropriate use of 
\textit{traits::to\_int\_type} and \textit{traits::not\_eof} 
throughout the Iostreams chapter

The following changes need to be made, throughout the chapter 27 (Input/Output library) in order to reflect the correct use of the \texttt{ios\_traits (27.4.2)} member functions \textit{to\_int\_type} and \textit{not\_eof}.

27.5.2.2.3 Get area [\texttt{lib.streambuf.pub.get}] p 27-29

\texttt{int\_type sbumpc();}

It should say:

\textbf{Returns:} If the input sequence read position is not available, returns \texttt{uflow()}. Otherwise, returns \texttt{traits::to\_int\_type(*gptr())} and increments the next pointer for the input sequence.

Box 107 should be removed.

\texttt{int\_type sgetc();}

It should say:

\textbf{Returns:} If the input sequence read position is not available, returns \texttt{underflow()}. Otherwise, returns \texttt{traits::to\_int\_type(*gptr())}.

Box 108 should be removed.

27.5.2.4.3 Get area [\texttt{lib.streambuf.virt.get}] p 27-33

\texttt{int\_type uflow();}

It should say:

\textbf{Returns:} If the output sequence write position is not available, returns \texttt{overflow( traits::to\_int\_type(c) )}. Otherwise, stores \texttt{c} at the next pointer for the output sequence, increments the pointer, and returns \texttt{traits::to\_int\_type(c)}.

Box 109 should be removed.
**Requires:** The constraints are the same as for `underflow()`, except that the result character is transferred from the pending sequence to the backup sequence, and the pending sequence may not be empty before the transfer.

**Default behavior:** Calls `underflow()`. If `underflow()` returns `traits::eof()`, returns `traits::eof()`. Otherwise, returns the value of `traits::to_int_type(*gptr())` and increment the value of the next pointer for the input sequence.

**Returns:** `traits::eof()` to indicate failure.

Box 112 should be removed.

**27.5.2.2.4 Putback [lib.streambuf.pub.pback]** p 27-30

```cpp
int_type sputbackc(char_type c);
```

It should say:

**Returns:** If the input sequence putback position is not available, or if `!traits::eq_char_type(c, gptr())[-1]`, returns `pbackfail(traits::to_int_type(c))`. Otherwise decrements the next pointer for the input sequence and returns `traits::to_int_type(*gptr())`.

Editorial note: If Doc 96-0036R1=N0854R1 is approved by the committee, `!traits::eq_char_type(c, gptr())[-1]`, should be replaced by `!traits::eq(c, gptr())[-1]`.

```cpp
int_type sungetc();
```

It should say:

**Returns:** If the input sequence putback position is not available, returns `pbackfail()`. Otherwise, decrements the next pointer for the input sequence and returns `traits::to_int_type(*gptr())`.

**27.5.2.4.3 Get area [lib.streambuf.virt.get]** p 27-33

```cpp
int_type underflow();
```

Change the **Returns:** clause to:

**Returns:** `traits::to_int_type(c)`, where `c` is the first character of the pending sequence, without moving the input sequence position past it. If the pending sequence is null then the function returns `traits::eof()` to indicate failure.

```cpp
int_type overflow(int_type c = traits::eof());
```

Change footnote 243 to:

Typically, `overflow` returns `c` to indicate success, except when `traits::is_eof(c)` returns true, in which case it returns `traits::not_eof(c)`.

Editorial note: If Doc 96-0036R1=N0854R1 is approved by the committee, `traits::is_eof(c)`, should be replaced by `traits::eq_int_type(c,traits::eof())`.

**27.7.1.3 Overriden virtual functions [lib.stringbuf.virtuals]** p 27-60

```cpp
int_type underflow();
```
Returns: If the input sequence has a read position available, returns traits::to_int_type(*gptr()). Otherwise, returns traits::eof().

Box 130 should be removed.

int_type pbackfail(int_type c = traits::eof());

Change the third paragraph to:

- If traits::is_eof( c ) returns true and if the input sequence has a putback position available, assigns gptr() -1 to gptr().
  Returns: traits::not_eof( c ).

Editorial note: If Doc 96-0036R1=N0854R1 is approved by the committee, traits::is_eof( c ), should be replaced by traits::eq_int_type(c , traits::eof()).

27.8.1.4 Overridden virtual functions [lib.filebuf.virtuals] p 27-68

int_type pbackfail(int_type c = traits::eof());

It should say:

Effects: Puts back the character designated by c to the input sequence, if possible, in one of three ways:

- If traits::is_eof( c ) returns false and if the function makes a putback position available and if traits::eq_char_type( to_char_type( c ),gptr()[ -1 ] ) returns true, decrements the next pointer for the input sequence, gptr().
  Returns: c.

- If traits::is_eof( c ) returns false and if the function makes a putback position available, and if the function is permitted to assign to the putback position, decrements the next pointer for the input sequence, and stores c there.
  Returns: c.

- If traits::is_eof( c ) returns true and if either the input sequence has a putback position available or the function makes a putback position available, decrements the next pointer for the input sequence, gptr().
  Returns: traits::not_eof( c ).

Returns: traits::eof() to indicate failure.

Notes: “is kept unchanged”

Editorial note: If Doc 96-0036R1=N0854R1 is approved by the committee, traits::is_eof( c ), should be replaced by traits::eq_int_type(c , traits::eof()) and traits::eq_char_type(to_char_type(c) , gptr()[ -1 ]), should be replaced by traits::eq(to_char_type(c) , gptr()[ -1 ]).

D.6.1.3 strstreambuf overridden virtual functions [depr.strstreambuf.virtuals] p D-7

int_type pbackfail(int_type c = EOF);

It should say:
Puts back the character designated by c to the input sequence, if possible, in one of three ways:

- If c != EOF, if the input sequence has a putback position available, and if (char)c == gnext[-1], assigns gnext -1 to gnext.
  Returns: c.

- If c != EOF, if the input sequence has a putback position available, and if strmode & constant is zero, assigns c to *--gnext.
  Returns: c.

- if c == EOF and if the input sequence has a putback position available, assigns gnext -1 to gnext.
  Returns a value other than EOF.

**Returns:** EOF to indicate failure.
**Notes:** If the function can succeed in more than one of these ways, it is unspecified which way is chosen.
The function can alter the number of putback positions available as a result of any call.

Box 148 should be removed.