Nestable comments in Algol 68

by Jose E. Marchesi

Foreword

The following specification has been released under the auspices of the GNU Algol 68 Working Group, and has been scrutinized to ensure that

a. it is strictly upwards-compatible with Algol 68,

b. it is consistent with the philosophy and orthogonal framework of the language, and

c. it fills a clearly discernible gap in the expressive power of that language.

The source of this document can be found at `https://git.sr.ht/~jemarch/gnu68`.

The informal description of this proposal introduces the proposed new language features, providing a rationale and usage examples.

The formal definition of this proposal uses the existing formalism and conventions of the Revised Report, and it is expressed as modifications to the Report.

Finally, the implementation notes of this proposal describes a way in which the features added by this specification can be implemented. No implementer should feel committed to do things as described there; the same language facilities may well be implementable in other ways, more suitable to specific implementations.

1 Informal Description

Comments in Algol 68

The Algol 68 strict language defines several styles of comments, each style using a different delimiter symbol:

```
brief comment symbol                        ¢
bold comment symbol                         comment
style i comment symbol                      co
style ii comment symbol                     #
```

In all cases the same symbol marks the beginning and the end of the comment. For example, a bold comment starts with **comment** and it must be ended with another **comment**. In the sequel we will refer to this kind of delimiters as *symmetrical*.

We can think on several pragmatic reasons why a programmer may choose to use some particular style in some particular circumstance.

Styles featuring short delimiters will probably be favored for short comments that fit in a single line, whereas styles with longer delimiters will likely be used for "block comments" that span for more than one line. This is shown in the next example.

```
    comment
      Send a chunk of the data to the server.  Note how each chunk
      consists on a size encoded in printable hexadecimal digits
      followed by the contents.
    comment
    begin while bool done;
                   string some_content = cb (done);
                   not done
        do co Send chunk.  co
           fputs (socket, itoa (upb some_content, 16));
           fputs (socket, "'r'n");
           fputs (socket, some_content);
           fputs (socket, "'r'n")
```

```
      od
    end
```

Another reason for choosing some particular style of comment may be to nest comments. It is generally not possible to nest comments with symmetrical delimiters, but a given comment can be nested inside another comment that uses some other style. An application of this is shown in the following example:

```
comment
  begin while bool done;
                 string some_content = cb (done);
                 not done
         do co Send chunk.   co
            fputs (socket, itoa (upb some_content, 16));
            fputs (socket, "'r'n");
            fputs (socket, some_content);
            fputs (socket, "'r'n")
         od
    end
comment
```

This trick only works to some extent, however, since the maximum nesting level is in theory limited to the number of different styles available (most implementations provide up to three styles) and in practice it depends on the code being commented out and the comments it already contains. A programmer or a program (such as a text editor that provides automatic commenting facilities) need to examine the code being commented out in order to decide whether it is possible to comment it out, and the style of comment required to do so.

An additional problem derived from using symmetrical comment delimiters is that they make it necessary to inspect the whole source in order to determine, given a comment delimiter, to determine whether it begins or finishes a comment. For programmers this can be confusing and error prone, for programs such as text editors wanting to highlight a comment, or to skip over it, this is very inefficient.

Finally, stropping regimes based on reserved words are very likely to not provide bold styles for comments, in order to minimize the risk of collision with identifiers and avoid introducing grave restrictions in the contents of comments. For example, `co` is too short and too generic to be turned into a reserved word. On the other hand, turning `comment` into a reserved word denoting a symmetrical comment delimiter would make it not possible to write `comment` within a comment, which is silly to say the least. In these regimes, therefore, it will likely be only possible to use one or two comment styles, with the associated restrictions in nesting comments.

Nestable comments in Algol 68

The shortcomings described in the previous section motivate the addition of nestable comments to Algol 68.

Representations of nestable comments are defined for the *bold* and *brief* styles, both of which use asymmetric delimiters:

```
bold comment begin symbol                    note
bold comment end symbol                      eton
brief comment begin symbol                   {
brief comment end symbol                     }
```

The recommended representation for brief nestable comments, the curly braces { and }, has been chosen for several reasons.

On one hand curly braces are used in the Revised Report, articles, and other documents related to Algol 68 to denote *pragmatic remarks*, which sometimes are even nested; comments in programs often play the role of pragmatic remarks. On the other hand the ALGOL68-RS compiler supported comments delimited by curly braces, so that is a precedent. Finally, the curly braces are not worthy characters in standard Algol 68, so there is no possibility of lexical ambiguity.

2 Formal Description

New symbols have been invented for opening and closing comments:

```
9.4.1.f
  bold comment begin symbol                NOTE
  bold comment end symbol                  ETON
  brief comment begin symbol               {
  brief comment end symbol                 }
```

Syntax of pragments is expanded to also include nestable comments:

```
9.2.1

b) PRAGMENT{a} : ... ; STYLE nested comment{d}.
d) STYLE nested comment{b} :
     STYLE comment begin symbol{94h,-},
       STYLE nested comment contents{e} sequence,
       STYLE comment end symbol{94h,-}.
e) STYLE nested comment contents{d} :
     STYLE nested comment item{c} sequence option,
       STYLE nested comment{d} option.
f) STYLE nested comment item{e} :
     character glyph{814c} ; STYLE other nested comment item{d}.
```

3 Implementation Notes

The new nestable comments are introduced as an addition to the existing non-nestable comments. The lexical analyzer will thus this in mind while recognizing the beginning and end delimiters and maintain a stack of comment levels.

As mentioned above, implementations offering stropping regimes based on reserved words will probably not want to offer bold styles of nested comments but only the brief styles.