Proposal for C23
WG14 N2842

Title:	Normal and subnormal classification
Author, affiliation:	C FP group
Date:	2021-10-13
Proposal category:	Technical
Reference:	N2596

Vincent Lefevre sent email to CFP pointing out that the term "normal" is used in the definitions of **`FP_NORMAL`** and **`isnormal()`** but is not defined.

"Normal" might be presumed to be equivalent to "normalized" (5.2.4.2.2). For IEC 60559 binary interchange formats (including the common binary32 and binary64), C normalized floating-point numbers do correspond to normal values as defined in IEC 60559.

However, C (5.2.4.2.2) accommodates a variety of floating-point types, including ones with double-double style formats. These can contain values with extra precision (beyond the precision of the type) and extremely large-magnitude values with less precision than the precision of the type.

Also, a decimal floating-point type can have a cohort of representations, only one (at most) of which is normalized. The representations though distinguishable by a program compare equal to each other and all should be classified as normal (and are so classified in IEC 60559) if their value is in the normal range.

The second suggested change below specifies that the "normal" category for classification includes all numbers in the range of normalized floating-point numbers in the model (5.2.4.2.2). Thus "normal" can apply to numbers with extra precision (as in double-double formats) and to unnormalized numbers that are equal to normalized numbers (as in decimal formats). It does not apply to extremely large-magnitude numbers with less precision than the precision of the type and which hence might undermine error analysis.

In the same vein, the second suggested change specifies that the "subnormal" category includes all nonzero numbers whose magnitude is too small for a normal number, including any such numbers (added by the implementation) that are not subnormal floating-point numbers in the model.

The second suggested change clarifies how classification is done. It does not affect the C17 classification of model numbers or of values for IEC 60559 (binary) formats. However, the clarification does introduce new requirements, so could affect some implementations. For example, an unnormalized representation whose value is in

the normal range would have to be classified as “normal”, not as an implementation-defined category for unnormalized.

The first suggested change just reflects the fact that an implementation may add categories that are none of NaN, infinite, normal, subnormal, and zero (which is not new with this proposal).

Suggested changes:

In 7.12 #12, change:

> represent ~~the~~ mutually exclusive kinds of floating-point values

In 7.12.3, before the first paragraph, insert:

> [0] Floating-point values can be classified as NaN, infinite, normal, subnormal, or zero, or into other implementation-defined categories. Numbers whose magnitude is at least $b^{e_{\min}-1}$ (the minimum magnitude of normalized floating-point numbers in the type) and at most $(1 - b^{-p})b^{e_{\max}}$ (the maximum magnitude of normalized floating-point numbers in the type), where b, p, $e_{\min}$, and $e_{\max}$ are as in 5.2.4.2.2, are classified as normal. Nonzero numbers whose magnitude is less than $b^{e_{\min}-1}$ are classified as subnormal.