IEEE Standards Interpretations for IEEE Std 1003.2™-1992 IEEE Standard for Information Technology--Portable Operating System Interfaces (POSIX®)--Part 2: Shell and Utilities

Interpretations are issued to explain and clarify the intent of a standard and **do not** constitute an alteration to the original standard. In addition, interpretations are not intended to supply consulting information. Permission is hereby granted to download and print one copy of this document. Individuals seeking permission to reproduce and/or distribute this document in its entirety or portions of this document must contact the IEEE Standards Department for the appropriate license. Use of the information contained in this document is at your own risk.

IEEE Standards Department, Copyrights and Permissions, 445 Hoes Lane, Piscataway, New Jersey 08855-1331, USA

Interpretation Request #150
Topic: getopt() **Relevant Clauses:** B.7.2

Contrary to historical practice and behavior that would be expected by utility users, the description of the getopt() function in POSIX.2 requires that an option with an option argument presented as a seperate argument to a utility that appear as the last two arguments to a utility be treated as an error instead of being recognized as an option with an option argument. The problem is on POSIX.2, P733, L502-508, where it says: (1) If the option was the last character in the string pointed to by an element of argv, then optarg contains the next element of argv, and optind shall be incremented by 2. If the resulting value of optind is not less than argc, this indicates a missing option argument, and getopt() shall return an error indication. (2) Otherwise, optarg points to the string following the option character in that element of argv, and optind shall be incremented by 1.

An example that shows the problem here is the following:

1. A utility (util) takes an option (a) that takes an option argument.
2. The utility can be invoked as: util -a arg # case 1 or util -aarg # case 2 and should get the same results (although strictly conforming applications should use the first form instead of the second form).
3. The utility invokes getopt() to parse its arguments with a call similar to: ret = getopt(argc, argv, “a:”); Following the rules stated in the description of getopt(), optind is initialized to 1 before getopt() is invoked the first time.

In case 2, getopt() will return ‘a’ with optind set to 2 and optarg pointing to “arg” in argv[1] (where argv[1] is “-aarg”). This is what everyone would expect. In case 1, however, the sequence of events is: 1. optind is initialized to 1 before optarg() is invoked

2. optarg is set to point to argv[2] (where argv[2] is “arg”) 3. optind is incremented by 2 (setting it to 3) 4. since optind (3) is not less than argc (3), getopt() returns an error indicating a missing option argument But, the option argument is not missing, it is just an off by one error in the specification that says that getopt() has to indicate that the option argument is missing.

To match historic and expected practice, the “not less than argc” on POSIX.2, P733, L504-505 would need to be changed to “greater than argc”. POSIX.2 implies that getopt() should be able to be used by applications that want to parse command line arguments in a way consistent with the Utility Syntax Guidelines in section 2.10.2. This is not possible for applications that have one or more options that take an option argument unless they also have one or more mandatory operands. Was this change to historic practice intended? Shouldn’t applications be able to use getopt() to parse command line arguments as long as the utility’s syntax adheres to the Utility Syntax Guidelines specifed on POSIX.2, P100-101, L3663-3724?

Interpretation Response
The standards states the requirements for getopt and conforming implementations must conform to this. However, concerns have been raised about this which are being referred to the sponsor. In particular the interpretations committee do not believe that this change from historical practice was intended. The interpretations committee beleive that on page 733, line 504, the words ‘not less than’ should be changed to ‘greater than’.

Rationale for Interpretation
None.