-- --

IEEE P1003.2 Draft 11.2 − September 1991

4.68 `uname` — Return system name

4.68.1 Synopsis

```
uname [−amnrsv]
```

4.68.2 Description

By default, the `uname` utility shall write the operating system name to standard output. When options are specified, symbols representing one or more system characteristics shall be written to the standard output. The format and contents of the symbols are implementation defined. On systems conforming to POSIX.1 {8}, the symbols written shall be those supported by the POSIX.1 {8} *uname* () function.

4.68.3 Options

The `uname` utility shall conform to the utility argument syntax guidelines described in 2.10.2.

The following options shall be supported by the implementation:

- `−a` Behave as though all of the options `−mnrsv` were specified.
- `−m` Write the name of the hardware type on which the system is running to standard output.
- `−n` Write the name of this node within an implementation-specified communications network.
- `−r` Write the current release level of the operating system implementation.
- `−s` Write the name of the implementation of the operating system.
- `−v` Write the current version level of this release of the operating system implementation.

If no options are specified, the `uname` utility shall write the operating system name, as if the `−s` option had been specified.

4.68.4 Operands

None.

4.68.5 External Influences

4.68.5.1 Standard Input

None.

4.68.5.2 Input Files

None.

4.68.5.3 Environment Variables

The following environment variables shall affect the execution of `uname`:

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_CTYPE This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments).

LC_MESSAGES This variable shall determine the language in which messages should be written.

4.68.5.4 Asynchronous Events

Default.

4.68.6 External Effects

4.68.6.1 Standard Output

By default, the output shall be a single line of the following form:

`"%s\n"`, *<sysname>*

If the `-a` option is specified, the output shall be a single line of the following form:

`"%s %s %s %s %s\n"`, *<sysname>*, *<nodename>*, *<release>*, *<version>*, *<machine>*

Additional implementation-defined symbols may be written; all such symbols shall be written at the end of the line of output before the `<newline>`.

If options are specified to select different combinations of the symbols, only those symbols shall be written, in the order shown above for the −a option. If a symbol is not selected for writing, its corresponding trailing <blank>s also shall not be written.

4.68.6.2 Standard Error

Used only for diagnostic messages.

4.68.6.3 Output Files

None.

4.68.7 Extended Description

None.

4.68.8 Exit Status

The uname utility shall exit with one of the following values:

0 The requested information was successfully written.

>0 An error occurred.

4.68.9 Consequences of Errors

Default.

4.68.10 Rationale. *(This subclause is not a part of P1003.2)*

Examples, Usage

The following command:

```
uname −sr
```

writes the operating system name and release level, separated by one or more <blank>s.

Note that any of the symbols could include embedded <space>s, which may affect parsing algorithms if multiple options are selected for output.

The node name is typically a name that the system uses to identify itself for inter-system communication addressing.

History of Decisions Made

It was suggested that this utility cannot be used portably, since the format of the symbols is implementation defined. The POSIX.1 {8} working group could not achieve consensus on defining these formats in the underlying *uname*() function and there is no expectation that POSIX.2 would be any more successful. In any event, some applications may still find this historical utility of value. For example, the symbols could be used for system log entries or for comparison with operator or user input.

4.69 `uniq` — Report or filter out repeated lines in a file

4.69.1 Synopsis

`uniq [−c|−d|−u][−f` *fields* `][−s` *chars* `][`*input_file* `[`*output_file* `]]`

Obsolescent Version:

`uniq [−c|−d|−u][−`*n*`][+`*m*`][`*input_file* `[`*output_file* `]]`

4.69.2 Description

The `uniq` utility shall read an input file comparing adjacent lines, and write one copy of each input line on the output. The second and succeeding copies of repeated adjacent input lines shall not be written.

Repeated lines in the input shall not be detected if they are not adjacent.

4.69.3 Options

The `uniq` utility shall conform to the utility argument syntax guidelines described in 2.10.2; the obsolescent version does not, as one of the options begins with `+` and the *−m* and *+n* options do not have option letters.

The following options shall be supported by the implementation:

`−c` Precede each output line with a count of the number of times the line occurred in the input.

`−d` Suppress the writing of lines that are not repeated in the input.

`−f` *fields* Ignore the first *fields* fields on each input line when doing comparisons, where *fields* shall be a positive decimal integer. A field is the maximal string matched by the basic regular expresssion:

```
[[:blank:]]*[^[:blank:]]*
```

If the *fields* option-argument specifies more fields than appear on an input line, a null string shall be used for comparison.