-- --

IEEE P1003.2b Draft 2 − December 1991

4.24 `find` — Find files

⇒ **4.24.4 `find` Operands.** *Add the following primary in the proper sorted order:*

`−follow`	The primary always shall evaluate as true; it shall cause `find` to traverse directories by following symbolic links into other parts of the hierarchy. By default, `find` shall not follow symbolic links. In following symbolic links, `find` shall keep track of the directories visited so that it can detect infinite loops. (For example, a loop would occur if a symbolic link refers to a directory that is an ancestor.) When it detects a loop, `find` shall write a diagnostic message to standard error and shall either recover its position in the hierarchy in an implementation-defined manner or terminate.

⇒ **4.24.4 `find` Operands.** *In the* `−type` *c description, add the character* `l` *(ell) to represent a symbolic link.*

4.33 `ln` — Link files

⇒ **4.33.1 `ln` Synopsis.** *Modify the Synopsis to be:*

`ln [−f]` *source_file target_file*

`ln [−f]` *source_file . . . target_dir*

`ln −s` *source_file target_file*

`ln −s` *source_file . . . target_dir*

⇒ **4.33.2 `ln` Description.** *Modify the entire Description subclause as follows:*

The *ln* () utility creates either hard or symbolic links, depending on the presence of the `−s` option: hard links shall be created by default, symbolic links with `−s`.

4.33.2.1 Creating Hard Links

<current contents of 4.33.2 . . . >

If *source_file* is a symbolic link, `ln` shall follow the link and create a hard link at *destination* to the file referenced by *source_file* . If either *target_file* or *target_dir* is a symbolic link, the third or fourth synopsis forms (respectively) shall be assumed, even though `−s` was not specified; use of `−f` in these cases is undefined.

4.33.2.2 Creating Symbolic Links

In the third synopsis form, the `ln` utility shall create a symbolic link for the file specified by the *source_file* operand, at the *destination* path specified by the *target_file* operand. If *target_file* is a symbolic link to a nondirectory file, `ln` shall overwrite the symbolic link. This third synopsis form shall be assumed when there are two operands and the second is not an existing directory.

In the fourth synopsis form, the `ln` utility shall create a symbolic link for each file specified by a *source_file* operand, at a *destination* path in the existing directory named by *target_dir*. The corresponding destination path for each *source_file* shall be the concatenation of the target directory pathname, a slash character, and the last pathname component of the *source_file.* If *target_dir* is a symbolic link to a directory, `ln` shall follow the link and create a symbolic link to *source_file* in the directory *target_dir*.

For both the third and fourth synopsis forms:

— If the directory in which the destination is to reside does not support {POSIX 2_SYMLINKS }, results are undefined.

— The *source_file* operand can be any pathname and need not represent an existing or accessible file.