Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'lsearch.3p' command

$ man lsearch.3p

LSEARCH(3P) POSIX Programmer's Manual LSEARCH(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

lsearch, lfind ? linear search and update

SYNOPSIS

```
#include <search.h>
void *lsearch(const void *key, void *base, size_t *nelp, size_t width,
    int (*compar)(const void *, const void *));
void *lfind(const void *key, const void *base, size_t *nelp,
    size_t width, int (*compar)(const void *, const void *));
```

DESCRIPTION

The lsearch() function shall linearly search the table and return a pointer into the table for the matching entry. If the entry does not occur, it shall be added at the end of the table. The key argument points to the entry to be sought in the table. The base argument points to the first element in the table. The width argument is the size of an element in bytes. The nelp argument points to an integer containing the current number of elements in the table. The integer to which nelp points shall be incremented if the entry is added to the table. The

compar argument points to a comparison function which the application shall supply (for example, strcmp()). It is called with two arguments that point to the elements being compared. The application shall ensure that the function returns 0 if the elements are equal, and non-zero otherwise.

The lfind() function shall be equivalent to lsearch(), except that if the entry is not found, it is not added to the table. Instead, a null pointer is returned.

RETURN VALUE

If the searched for entry is found, both lsearch() and lfind() shall return a pointer to it. Otherwise, lfind() shall return a null pointer and lsearch() shall return a pointer to the newly added element.

Both functions shall return a null pointer in case of error.

ERRORS

No errors are defined.

The following sections are informative.

EXAMPLES

Storing Strings in a Table

This fragment reads in less than or equal to TABSIZE strings of length less than or equal to ELSIZE and stores them in a table, eliminating duplicates.

```
#include <stdio.h>
#include <string.h>
#include <search.h>
#define TABSIZE 50
#define ELSIZE 120
...
    char line[ELSIZE], tab[TABSIZE][ELSIZE];
    size_t nel = 0;
    ...
    while (fgets(line, ELSIZE, stdin) != NULL && nel < TABSIZE)
        (void) lsearch(line, tab, &nel,
            ELSIZE, (int (*)(const void *, const void *)) strcmp);
```

```
...
```

Finding a Matching Entry

The following example finds any line that reads "Thisisatest.".

```
#include <search.h>
#include <string.h>
...
char line[ELSIZE], tab[TABSIZE][ELSIZE];
size_t nel = 0;
char *findline;
void *entry;
findline = "This is a test.\n";
entry = lfind(findline, tab, &nel, ELSIZE, (
    int (*)(const void *, const void *)) strcmp);
```

APPLICATION USAGE

The comparison function need not compare every byte, so arbitrary data may be contained in the elements in addition to the values being com? pared.

Undefined results can occur if there is not enough room in the table to add a new item.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

hcreate(), tdelete()

The Base Definitions volume of POSIX.1?2017, <search.h>

The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at http://www.opengroup.org/unix/online.html .

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.ker?nel.org/doc/man-pages/reporting_bugs.html .