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 'hcreate.3p' command

$ man hcreate.3p

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

hcreate, hdestroy, hsearch ? manage hash search table

SYNOPSIS

```
#include <search.h>
int hcreate(size_t nel);
void hdestroy(void);
ENTRY *hsearch(ENTRY item, ACTION action);
```

DESCRIPTION

The hcreate(), hdestroy(), and hsearch() functions shall manage hash search tables.

The hcreate() function shall allocate sufficient space for the table, and the application shall ensure it is called before hsearch() is used.

The nel argument is an estimate of the maximum number of entries that the table shall contain. This number may be adjusted upward by the al? gorithm in order to obtain certain mathematically favorable circum? stances.

The hdestroy() function shall dispose of the search table, and may be

followed by another call to hcreate(). After the call to hdestroy(), the data can no longer be considered accessible.

The hsearch() function is a hash-table search routine. It shall return a pointer into a hash table indicating the location at which an entry can be found. The item argument is a structure of type ENTRY (defined in the <search.h> header) containing two pointers: item.key points to the comparison key (a char *), and item.data (a void *) points to any other data to be associated with that key. The comparison function used by hsearch() is strcmp(). The action argument is a member of an enu? meration type ACTION indicating the disposition of the entry if it can? not be found in the table. ENTER indicates that the item should be in? serted in the table at an appropriate point. FIND indicates that no en? try should be made. Unsuccessful resolution is indicated by the return of a null pointer.

These functions need not be thread-safe.

RETURN VALUE

The hcreate() function shall return 0 if it cannot allocate sufficient space for the table; otherwise, it shall return non-zero.

The hdestroy() function shall not return a value.

The hsearch() function shall return a null pointer if either the action is FIND and the item could not be found or the action is ENTER and the table is full.

ERRORS

The hcreate() and hsearch() functions may fail if:

ENOMEM Insufficient storage space is available.

The following sections are informative.

EXAMPLES

The following example reads in strings followed by two numbers and stores them in a hash table, discarding duplicates. It then reads in strings and finds the matching entry in the hash table and prints it out.

```
#include <stdio.h>
#include <search.h>
```

```
#include <string.h>
struct info {        /* This is the info stored in the table */
    int age, room;   /* other than the key. */
};
#define NUM_EMPL    5000    /* # of elements in search table. */
int main(void)
{
    char string_space[NUM_EMPL*20];   /* Space to store strings. */
    struct info info_space[NUM_EMPL]; /* Space to store employee info. */
    char *str_ptr = string_space;     /* Next space in string_space. */
    struct info *info_ptr = info_space;
                                      /* Next space in info_space. */
    ENTRY item;
    ENTRY *found_item; /* Name to look for in table. */
    char name_to_find[30];
    int i = 0;
    /* Create table; no error checking is performed. */
    (void) hcreate(NUM_EMPL);
    while (scanf("%s%d%d", str_ptr, &info_ptr->age,
            &info_ptr->room) != EOF && i++ < NUM_EMPL) {
        /* Put information in structure, and structure in item. */
        item.key = str_ptr;
        item.data = info_ptr;
        str_ptr += strlen(str_ptr) + 1;
        info_ptr++;
        /* Put item into table. */
        (void) hsearch(item, ENTER);
    }
    /* Access table. */
    item.key = name_to_find;
    while (scanf("%s", item.key) != EOF) {
        if ((found_item = hsearch(item, FIND)) != NULL) {
            /* If item is in the table. */
```

```
            (void)printf("found %s, age = %d, room = %d\n",
                found_item->key,
                ((struct info *)found_item->data)->age,
                ((struct info *)found_item->data)->room);
        } else
            (void)printf("no such employee %s\n", name_to_find);
    }
    return 0;
}
```

APPLICATION USAGE

The hcreate() and hsearch() functions may use malloc() to allocate space.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

bsearch(), lsearch(), malloc(), strcmp(), tdelete()

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