NAME

mem_buy − (re)allocate storage space on the heap, optionally failing hard

SYNOPSIS

```
#include "mjsu.h"

VOID *mem_buy(VOID *old, UINT nbytes, BOOL force);
```

DESCRIPTION

mem_buy() attempts to allocate a cell of heap memory large enough to store an object of size *nbytes*.

If *old* is not NULL, it assumed to be the address of an already-allocated cell. In this case the cell is resized according to *nbytes*.

If the allocation request cannot be satisfied and *force* is YES, the calling process is terminated by the following call:

```
error("out of memory");
```

RETURNS

If successful, **mem_buy()** returns a pointer to the allocated memory, which is guaranteed to be aligned at a storage boundary suitable for a C datum of any type.

Otherwise, NULL (if anything) is returned.

EXAMPLE

To convert a linked-list into a NULL-terminated vector:

```
struct linkedlist
    {
    struct linkedlist *next;
    BAG *this;
    } *list, *tmp;
BAG *vec = NULL;

for (i = 0, vec = mem_buy(NULL, (i+1) * sizeof(*BAG), YES);
    list;
    ++i, vec = mem_buy(vec, (i+1) * sizeof(BAG), YES))
    {
    vec[i] = list->this;
    tmp = list;
    list = list->next;
    mem_free(tmp);
    }
vec[i] = NULL;
```

SEE ALSO

mem_free(), **error(3)**, **mjsu(7)**.

malloc(), **realloc()** and **free()**, as defined by ANSI X3.159-1989.

AVAILABILITY

mem_buy() is written in C, conforming to ANSI X3.159-1989 (hosted program environment).