```
static unsigned long int next = 1;

int rand(void)   /* RAND_MAX assumed to be 32767 */
{
      next = next * 1103515245 + 12345;
      return (unsigned int)(next/65536) % 32768;
}

void srand(unsigned int seed)
{
      next = seed;
}
```

7.10.3 Memory management functions

The order and contiguity of storage allocated by successive calls to the **calloc**, **malloc**, and **realloc** functions is unspecified. The pointer returned if the allocation succeeds is suitably aligned so that it may be assigned to a pointer to any type of object and then used to access such an object or an array of such objects in the space allocated (until the space is explicitly freed or reallocated). Each such allocation shall yield a pointer to an object disjoint from any other object. The pointer returned points to the start (lowest byte address) of the allocated space. If the space cannot be allocated, a null pointer is returned. If the size of the space requested is zero, the behavior is implementation-defined; the value returned shall be either a null pointer or a unique pointer. The value of a pointer that refers to freed space is indeterminate.

7.10.3.1 The `calloc` function

Synopsis

```
#include <stdlib.h>
void *calloc(size_t nmemb, size_t size);
```

Description

The **calloc** function allocates space for an array of **nmemb** objects, each of whose size is **size**. The space is initialized to all bits zero.[127]

Returns

The **calloc** function returns either a null pointer or a pointer to the allocated space.

7.10.3.2 The `free` function

Synopsis

```
#include <stdlib.h>
void free(void *ptr);
```

Description

The **free** function causes the space pointed to by **ptr** to be deallocated, that is, made available for further allocation. If **ptr** is a null pointer, no action occurs. Otherwise, if the argument does not match a pointer earlier returned by the **calloc**, **malloc**, or **realloc** function, or if the space has been deallocated by a call to **free** or **realloc**, the behavior is undefined.

127 Note that this need not be the same as the representation of floating-point zero or a null pointer constant.