



Linux Ubuntu 26.04 Manual Pages on command 'lcong48_r.3'

\$ man lcong48_r.3

drand48_r(3) Library Functions Manual drand48_r(3)

NAME

drand48_r, erand48_r, lrand48_r, nrand48_r, mrand48_r, jrand48_r, srand48_r, seed48_r,
lcong48_r - generate uniformly distributed pseudo-random numbers reentrantly

LIBRARY

Standard C library (libc, -lc)

SYNOPSIS

```
#include <stdlib.h>

int drand48_r(struct drand48_data *restrict buffer,
             double *restrict result);

int erand48_r(unsigned short xsubi[3],
             struct drand48_data *restrict buffer,
             double *restrict result);

int lrand48_r(struct drand48_data *restrict buffer,
             long *restrict result);

int nrand48_r(unsigned short xsubi[3],
             struct drand48_data *restrict buffer,
             long *restrict result);

int mrand48_r(struct drand48_data *restrict buffer,
             long *restrict result);

int jrand48_r(unsigned short xsubi[3],
             struct drand48_data *restrict buffer,
             long *restrict result);
```

```
int srand48_r(long int seedval, struct drand48_data *buffer);

int seed48_r(unsigned short seed16v[3], struct drand48_data *buffer);

int lcong48_r(unsigned short param[7], struct drand48_data *buffer);
```

Feature Test Macro Requirements for glibc (see `feature_test_macros(7)`):

All functions shown above:

```
/* glibc >= 2.19: */ _DEFAULT_SOURCE

|| /* glibc <= 2.19: */ _SVID_SOURCE || _BSD_SOURCE
```

DESCRIPTION

These functions are the reentrant analogs of the functions described in `drand48(3)`. Instead of modifying the global random generator state, they use the supplied data buffer. Before the first use, this struct must be initialized, for example, by filling it with `zeroes`, or by calling one of the functions `srand48_r()`, `seed48_r()`, or `lcong48_r()`.

RETURN VALUE

The return value is 0.

ATTRIBUTES

For an explanation of the terms used in this section, see `attributes(7)`.

??			
? Interface	? Attribute	? Value	?
??			
? <code>drand48_r()</code> , <code>erand48_r()</code> , <code>lrand48_r()</code> ,	? Thread safety ? MT-Safe race:buffer ?		
? <code>nrand48_r()</code> , <code>mrnd48_r()</code> , <code>jrand48_r()</code> ,	?	?	?
? <code>srand48_r()</code> , <code>seed48_r()</code> , <code>lcong48_r()</code>	?	?	?
??			

STANDARDS

GNU.

SEE ALSO

`drand48(3)`, `rand(3)`, `random(3)`