NAME

drand48, **erand48**, **lrand48**, **nrand48**, **mrand48**, **jrand48**, **srand48**, **seed48**, **lcong48** - pseudo random number generators and initialization routines

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

#include <stdlib.h>

double
drand48(*void*);

double
erand48(*unsigned short xseed[3]*);

long
lrand48(*void*);

long
nrand48(*unsigned short xseed[3]*);

long
mrand48(*void*);

long
jrand48(*unsigned short xseed[3]*);

void
srand48(*long seed*);

*unsigned short **
seed48(*unsigned short xseed[3]*);

void
lcong48(*unsigned short p[7]*);

DESCRIPTION

The functions described in this manual page are not cryptographically secure. Cryptographic applications should use arc4random(3) instead.

The **rand48**() family of functions generates pseudo-random numbers using a linear congruential algorithm working on integers 48 bits in size. The particular formula employed is r(n+1) = (a * r(n) + c) mod m where the default values are for the multiplicand a = 0x5deece66d = 25214903917 and the addend c = 0xb = 11. The modulo is always fixed at m = 2 ** 48. r(n) is called the seed of the random number generator.

For all the six generator routines described next, the first computational step is to perform a single iteration of the algorithm.

The **drand48**() and **erand48**() functions return values of type double. The full 48 bits of r(n+1) are loaded into the mantissa of the returned value, with the exponent set such that the values produced lie in the interval [0.0, 1.0).

The **lrand48**() and **nrand48**() functions return values of type long in the range [0, 2**31-1]. The high-order (31) bits of r(n+1) are loaded into the lower bits of the returned value, with the topmost (sign) bit set to zero.

The **mrand48**() and **jrand48**() functions return values of type long in the range [-2**31, 2**31-1]. The high-order (32) bits of r(n+1) are loaded into the returned value.

The **drand48**(), **lrand48**(), and **mrand48**() functions use an internal buffer to store r(n). For these functions the initial value of r(0) = 0x1234abcd330e = 20017429951246.

On the other hand, **erand48**(), **nrand48**(), and **jrand48**() use a user-supplied buffer to store the seed r(n), which consists of an array of 3 shorts, where the zeroth member holds the least significant bits.

All functions share the same multiplicand and addend.

The **srand48**() function is used to initialize the internal buffer r(n) of **drand48**(), **lrand48**(), and **mrand48**() such that the 32 bits of the seed value are copied into the upper 32 bits of r(n), with the lower 16 bits of r(n) arbitrarily being set to 0x330e. Additionally, the constant multiplicand and addend of the algorithm are reset to the default values given above.

The **seed48**() function also initializes the internal buffer r(n) of **drand48**(), **lrand48**(), and **mrand48**(), but here all 48 bits of the seed can be specified in an array of 3 shorts, where the zeroth member specifies the lowest bits. Again, the constant multiplicand and addend of the algorithm are reset to the default values given above. The **seed48**() function returns a pointer to an array of 3 shorts which contains the old seed. This array is statically allocated, thus its contents are lost after each new call to **seed48**().

Finally, **lcong48**() allows full control over the multiplicand and addend used in **drand48**(), **erand48**(),

lrand48(), **nrand48**(), **mrand48**(), and **jrand48**(), and the seed used in **drand48**(), **lrand48**(), and **mrand48**(). An array of 7 shorts is passed as argument; the first three shorts are used to initialize the seed; the second three are used to initialize the multiplicand; and the last short is used to initialize the addend. It is thus not possible to use values greater than 0xffff as the addend.

Note that all three methods of seeding the random number generator always also set the multiplicand and addend for any of the six generator calls.

SEE ALSO

arc4random(3), rand(3), random(3)

AUTHORS

Martin Birgmeier