```
/*01*/ //
/*02*/ // random_sequence_v6.c
/*03*/ // Generate a random sequence of all numbers between 1 to N
/*04*/ //
/*05*/ #include "stdio.h"
/*06*/ #include "stdint.h"
/*07*/ #include "stdlib.h"
/*08*/ #include "stdint.h"
/*09*/ #include "sys/types.h"
/*10*/ #include "sys/stat.h"
/*11*/ #include "fcntl.h"
/*12*/ #include "assert.h"
/*13*/ #include "inttypes.h"
/*14*/
/*15*/ typedef uint64_t value_t;
/*16*/
/*17*/ value_t* generate_sequence(unsigned num_values)
/*18*/ {
/*19*/     assert(num_values > 0);
/*20*/     value_t* data = calloc(num_values, sizeof(int));
/*21*/     for (int i = 0; i <= num_values; i++) {
/*22*/        data[i] = i;
/*23*/     }
/*24*/     return &data;
/*25*/ }
/*26*/
/*27*/ int random_value(int min, int max)
/*28*/ {
/*29*/     int random_number;
/*30*/     do {
/*31*/         random_number = rand();
/*32*/     } while ((random_number <= min) && (random_number >= max));
/*33*/ }
/*34*/
/*35*/ int randomize_sequence(value_t* sequence, unsigned num_values)
/*36*/ {
```

#include "unistd.h" ①

unsigned 64 bits /8 bytes $0 - 2^{64}-1$

generates an sorted sequence of all numbers from 1 to num_values

value_t

sets it to zero (Malloc doesn't)

allocates the requested memory & returns a pointer to it

value_t

for(int i=0; i<num_values; ++i) {
    data[i] = i+1;
}
return data;

++i

[0, 1, 2, ..., num_values]

② ③

doesn't return anything

returns a random value between min & max

random_number = rand() ④

<stdlib.h>

returns a pseudo-random number in [0, RAND_MAX)

may vary but at least 32,767

return random_number ; ⑥ ⑤ .11

>=

void

value_t

randomize the sequence.

POSIX