the random numbers more random. To plant the seed, you use the `srand()` function.

The `srand` function is used to help kick off the computer's random-number machine in a more *random* manner. Here's the format:

```
void srand((unsigned)seed)
```

The `seed` value is an unsigned integer value or variable, ranging from 0 up to 65,000-something. It's that value the compiler uses to help *seed* the random-number-generation equipment located in the bowels of your PC.

You must include the following line at the beginning of your source code to make the `srand()` function behave:

```
#include <stdlib.h>
```

Because the `rand()` function already requires this line, you have no need to specify it twice (unless you're just seeding the random-number generator out of some perverse horticultural lust).

- The `(unsigned)` deal is used to ensure that the number `srand()` uses is of the unsigned type (not negative). It's known as *type casting*.
- Using the value `1` (one) to seed the random-number generator causes the compiler to start over, by using the same, uninspirational numbers you witness when `srand()` isn't used. Avoid doing that, if possible.

Randoming up the RANDOM program

Now comes the time for some really random numbers. The following source code is for RANDOM2.C, a mild modification to the original program. This time, a new function is added, `seedrnd()`, which lets you reset the random-number generator and produce more random numbers:

```
#include <stdio.h>
#include <stdlib.h>

int rnd(void);
void seedrnd(void);

int main()
{
    int x;

    seedrnd();
    puts("Behold! 100 Random Numbers!");
```