How to get random numbers

How to get random numbers

```
/*
 * This stock answer explains how to generate random numbers.  To see how
 * it really works, you can save this answer to a file.  If you are
 * using the olc_answers program, hit "s" and enter a filename.
 * Then compile that file using the command:
 *
 *  cc filename.c
 *
 * and try running it by typing
 *
 *  ./a.out
 *
 * NOTE:  On the Sun workstations, you should use the rand() and srand()
 * function calls, instead of random() and srandom().
 *
 * You might also find more information on alternative ways of getting
 * random numbers by looking at
 *
 * a. chapter 7 of the Numerical Recipes book
 * b. the NAG library manual.
 *
 *
 */

main()
{
    double a_number;

    /*
     * The simplest way to get a random number is just to call the
     * function 'random()'.  It returns a random number between
     * 1 and 2**31 - 1.  For example:
     */

    a_number = (float) random();
    printf("A big random number is %lf.\n", a_number);

    /*
     * To get a random number between 0 and 1, you would use this:
     *
     *       double number;
     *       number = (float) random() / (float) 0x7fffffff;
     *
     * Note that the constant 0x7fffffff is equal to (2**31)-1, which is the
     * maximum value of the random number generator.
     */

    a_number = (float) random() / (float) 0x7fffffff;
    printf("A random number between 0 and 1 is %lf,\n", a_number);

    /*
     * However, when used as above, the program will get
     * the same random numbers every time it is run.  Sometimes
     * this is good, sometimes not.  For example, in Monte Carlo
     * simulations a set of identical "random" numbers is useful
     * for debugging, but bad for getting real data.
     *
 * To change the set of numbers generated, use 'srandom' to
 * set an initial state.  The number that you use to set this
 * state is called a "seed".  Note that identical seeds will
 * generate identical sequences of random numbers.  A possible
 * seed is the number of seconds since Jan 1, 1970, GMT, the
 * value given by time or the process id (from 'getpid').
 * Both are used here.  This 'srandom' call only needs
 * to be done once per program.
 */

srandom(time(0) * getpid());

/*
 * Now get and print a "real" random number.
 */

a_number = (float) random() / (float) 0x7fffffff;
printf("But a more random number between 0 and 1 is %lf\n", a_number);

/*
 * So, if you wanted a random number between 0 and 10, you would take the
 * number you got above and multiply it by 10, and round to the nearest
 * integer (or whatever).
 */

a_number = 10.0 * (float) random() / (float) 0x7fffffff;
```

```
    printf("But a more random number between 0 and 10 is %lf\n", a_number);
}
```