Порождение простейших псевдослучайных объектов (набор упражнений по программированию)

31 марта 2007 г.

Довольно часто в программах нужно создавать какие-то случайные величины: целые числа, числа с правающей запятой, булевские значения, массивы, строки, и т. д. В этом тексте перечислены функции для создания наиболее часто встречающихся случайных объектов.

Все эти функции легко реализовать через стандартную функцию `rand()` на языке `C` (либо через какой-нибудь другой генератор случайных целых чисел). Такая реализация является хорошим упражнением для начинающих программистов.

Через $[a, b)_{\mathbb{Z}}$ будем обозначать множество $[a, b) \cap \mathbb{Z}$, т. е. множество всех таких целых k, что $a \leqslant k < b$.

Будем описывать функции на языке `C`. Напомним, что стандартная функция `rand()` возвращает псевдослучайное целое число от `0` до `RAND_MAX`. Константа `RAND_MAX` зависит от реализации и обычно равна константе `INT_MAX` из `limits.h`. На 32-разрядных процессорах это $2^{31} - 1$.

Термин «псевдослучайное» здесь употреблён потому, что на самом деле значения функции `rand()` вычисляются по некоторому алгоритму и образуют периодическую последовательность (с очень большим периодом). Для случайной инициализации этой последовательности можно использовать таймер:

```
#include "time.h"
...
srand((int)time(NULL));
```

Можно считать, что `rand()` возвращает все возможные значения с одинаковой вероятностью.

Далее, ради краткости, вместо «псевдослучайное число» будем говорить «случайное число».

Порождение случайного целого числа

`int random_01()`
: возвращает случайное число из множества $\{0, 1\}$, причём значения 0 и 1 получаются с одинаковой вероятностью.

`int random_n(int n)`
: для заданного числа n ($n > 0$) возвращает случайное целое число из полуинтервала $[0, n)_{\mathbb{Z}}$, причём все возможные значения получаются с одинаковой вероятностью.

`int random_int(int a, int b)`
возвращает случайное целое число из полуинтервала $[a, b)_{\mathbb{Z}}$, где $a < b$, причём все возможные значения получаются с одинаковой вероятностью.

Порождение случайного числа типа double

`double random_decim()`
возвращает случайное число из множества $\{0, 0.1, \ldots, 0.9\}$, причём все возможные значения получаются с одинаковой вероятностью.

`double random_double()`
возвращает случайное число типа `double` из полуинтервала $[0, 1)$. Число возможных значений равно `RAND_MAX+1`, они равномерно распределены от 0 до 1 и получаются с одинаковой вероятностью.

`double random_double_ab()`
возвращает случайное число типа `double` из полуинтервала $[a, b)$. Число возможных значений равно `RAND_MAX+1`, они равномерно распределены от 0 до 1 и получаются с одинаковой вероятностью.

Порождение случайной булевской величины

Для булевских величин используем обозначения из C++: `bool`, `true`, `false`. В чистом C вместо этого нужно писать `int`, `1`, `0` либо использовать макроопределения `BOOL`, `TRUE`, `FALSE`.

`bool random_bool_fiftyfifty()`
возвращает значение типа `bool`, причём вероятность значения `true` равна $1/2$.

`bool random_bool_percent(int x)`
возвращает значение типа `bool`, причём вероятность значения `true` равна `x` процентов. Параметр `x` должен быть целым числом от 0 до 100.

`bool random_bool(double p)`
возвращает значение типа `bool`, причём вероятность значения `true` равна `p`. Параметр `p` должен принадлежать сегменту $[0, 1]$.

Порождение случайной буквы

`char random_char(char abc_begin, size_t abc_size)`
возвращает случайную букву из алфавита размера `abc_size`, начинающегося в таблице ASCII с символа `abc_begin`. Буквы получаются с одинаковой вероятностью.

`char random_char_from_abc(const char* abc, size_t abc_size)`
возвращает случайный элемент из массива символов `abc`, размер которого равен `abc_size`.

Заполнение массива случайными значениями

`void fill_random(int* a, size_t n, size_t m)`
заполняет массив `a` размера `n` случайными значениями из полуинтервала $[0, m)_{\mathbb{Z}}$.

`void fill_random_ab(int *ar, size_t n, int a, int b)`
заполняет массив `ar` случайными значениями из полуинтервала $[a, b)_{\mathbb{Z}}$.

Заполнение массива различными случайными значениями

Особенность следующих функций состоит в том, что они заполняют массивы **попарно различными** значениями.

`void fill_random_distinct(int *a, size_t n, size_t m)`
заполняет массив `a` размера `n` различными случайными значениями из полуинтервала $[0, m)_{\mathbb{Z}}$. Предполагается, что $0 \leqslant n \leqslant m$. Генератор случайных чисел (функция `rand()`) вызывается `n` раз.

`void fill_random_permut(int *a, size_t n)`
записывает в массив `a` размера `n` случайную перестановку чисел $0, 1, \ldots, m-1$. Генератор случайных чисел (функция `rand()`) вызывается `n` раз.

Создание случайной строки

Следующие функции предлагается писать на языке программирования, в котором есть готовый тип «строка» (например, на `C++`).

`string random_string(size_t n, char abc_begin, size_t abc_size)`
создаёт строку длины `n`, элементы которой являются случайными символами из фрагмента таблицы `ASCII` размера `abc_size`, начинающегося с символа `abc_begin`.

`string random_string_distinct(size_t n, char abc_begin, size_t abc_size)`
создаёт строку длины `n`, элементы которой являются случайными различными символами из фрагмента таблицы `ASCII` размера `abc_size`, начинающегося с символа `abc_begin`. Генератор случайных чисел (функция `rand()`) вызывается `n` раз.