

STL: random_shuffle

Kontribusi: Taufik Abidin
Wednesday, 15 November 2006

C++ STL (Standard Template Library) merupakan ANSI/ISO standard dan telah menjadi bagian utuh dari setiap distribusi C++. Secara garis besar, STL terdiri dari 6 komponen, yaitu: containers, generic algorithms, iterators, function objects, adaptors, dan allocators.

Tulisan singkat ini membahas method `random_shuffle` yang merupakan bagian dari komponen generic algorithms. Untuk dapat menggunakan method ini, pernyataan `#include <algorithm>` harus disertakan. Method `random_shuffle` secara acak menempatkan elemen dalam interval `[first, last)`. Permutasi yang dihasilkan oleh `random_shuffle` adalah mengikuti distribusi uniform, yaitu probabilitas dari setiap N permutasi dari range berukuran N adalah $1/N$ [1]. Kompleksitas dari algoritma ini adalah $O(N)$ atau linear.

Program berikut memperlihatkan bagaimana method `random_shuffle` digunakan untuk mengacak elemen dari sebuah vector berukuran 10. `#include <iostream.h>`

```
#include <algorithm>
#include <vector>
#define TOTAL 10

using namespace std;

int main(char **argv, int argc){
    vector<int> number(TOTAL);
    for(int i=0; i<TOTAL; i++)
        number[i] = i;
    for(int i=0; i<TOTAL; i++){
        random_shuffle(number.begin(), number.end());
        copy(number.begin(), number.end(), ostream_iterator<int> (cout, " "));
        cout<<endl;
    }
    return EXIT_SUCCESS;
}
```

Pernyataan `vector<int> number(TOTAL)` mendeklarasi sebuah vector `number` bertipe `int` berukuran 10 (variabel `TOTAL` telah didefinisikan sebelumnya dengan nilai 10). Loop dibawahnya menginisialisasikan elemen dari vector `number`, mulai dari slot pertama vector, ditandai dengan pernyataan `number.begin()`, sampai dengan slot terakhir vector, yang ditandai dengan pernyataan `number.end()`.

Loop selanjutnya mengacak nilai dari vector `number` tersebut sebanyak 10 kali. Hasilnya kemudian ditampilkan ke output stream (misalnya layar monitor) menggunakan method `copy`. Argumen terakhir dari method `copy` adalah object bertipe `ostream_iterator`.

Pernyataan `copy(number.begin(), number.end(), ostream_iterator<int> (cout, " "))` dapat juga ditulis secara terpisah dalam dua baris sebagai berikut: `ostream_iterator<int> output (cout, " "); copy(number.begin(), number.end(), output);`

Baris pertama dari pernyataan di atas menginisialisasi sebuah objek dengan nama `output` yang bertipe `int`. Pada saat inisialisasi tersebut, dua parameter dikirim melalui constructor `ostream_iterator`. Parameter yang pertama adalah `cout` (object `c` output) dan parameter yang kedua adalah " " (spasi) yang digunakan sebagai karakter pemisah antara elemen yang akan ditampilkan. Kemudian, objek ini kita gunakan sebagai argumen ketiga dari method `copy`.

Bila program di atas dikompile (diasumsikan program tersebut disimpan dengan nama `rshuffle.cc`, dan dikompile di Linux sebagai berikut: `c++ -o rshuffle rshuffle.cc`) maka bila binary object `rshuffle` dijalankan di terminal (misalnya: `$./rshuffle` [enter]), maka tampilan yang muncul adalah sebagai berikut:

```
1 7 9 3 5 6 0 4 8 2
4 2 1 9 3 6 0 8 7 5
9 4 2 1 6 5 0 8 7 3
0 4 1 9 5 8 3 2 6 7
5 6 2 4 8 3 1 9 0 7
9 1 5 2 3 8 7 4 0 6
```

2 0 5 3 8 7 4 1 9 6
1 5 8 6 9 2 4 0 3 7
2 1 4 8 3 6 5 0 9 7
6 8 4 3 0 7 9 5 1 2

Versi kedua dari method `random_shuffle` mengizinkan programmer mengirim object `&rand` yang bertipe `RandomNumberGenerator` melalui parameter ketiga. Dengan method ini, `random seed` dapat dikontrol oleh programmer. Namun untuk menggunakan versi ini, programmer harus mengimplementasi class `RandomNumberGenerator` terlebih dahulu. Selamat mencoba!

[1] Musser, D.R. Derge, G.J, and Saini, A., *STL Tutorial and Reference Guide: C++ Programming with the Standard Template*, 2nd edition, Addison Wesley, New Jersey, March 2001.