Search: Go

Not logged in

Reference <random> uniform_int_distribution register log in

C++

- Information
- Tutorials
- Reference
- Articles
- Forum

Reference

C library:
Containers:
Input/Output:
Multi-threading:
Other:

- <algorithm>
- <bitset>
- <chrono>
- <codecvt>
- <complex>
- <exception>
- <functional>
- <initializer_list>
- <iterator>
- <limits>
- <locale>
- <memory>
- <new>
- <numeric>
- <random>
- <ratio>
- <regex>
- <stdexcept>
- <string>
- <system_error>
- <tuple>
- <typeindex>
- <typeinfo>
- <type_traits>
- <utility>
- <valarray>

<random>

distributions:

- bernoulli_distribution
- binomial_distribution
- cauchy_distribution
- chi_squared_distribution
- discrete_distribution
- exponential_distribution
- extreme_value_distribution
- fisher_f_distribution
- gamma_distribution
- geometric_distribution
- lognormal_distribution
- negative_binomial_distribution
- normal_distribution
- piecewise_constant_distribution
- piecewise_linear_distribution
- poisson_distribution
- student_t_distribution
- uniform_int_distribution
- uniform_real_distribution
- weibull_distribution

generators:

- default_random_engine
- discard_block_engine
- independent_bits_engine
- knuth_b
- linear_congruential_engine
- mersenne_twister_engine
- minstd_rand
- minstd_rand0
- mt19937
- mt19937_64
- random_device
- ranlux24
- ranlux24_base
- ranlux48
- ranlux48_base
- shuffle_order_engine
- subtract_with_carry_engine

other:

- generate_canonical
- seed_seq

uniform_int_distribution

uniform_int_distribution::(constructor)

member functions:

- uniform_int_distribution::a
- uniform_int_distribution::b
- uniform_int_distribution::max
- uniform_int_distribution::min

class template

std::uniform_int_distribution

<random>

```
template <class IntType = int> class uniform_int_distribution;
```

Uniform discrete distribution

Random number distribution that produces integer values according to a *uniform discrete distribution*, which is described by the following *probability mass function*:

$$P(i|a,b) = \frac{1}{b-a+1} \quad , \quad a \leq i \leq b$$

This distribution produces random integers in a range `[a,b]` where each possible value has an equal likelihood of being produced. This is the distribution function that appears on many trivial random processes (like the result of rolling a die).

The distribution parameters, a and b, are set on construction.

To produce a random value following this distribution, call its member function operator().

For a discrete distribution that can have different probabilities for each possible value, see discrete_distribution.

Template parameters

`IntType`
An integer type. Aliased as member type `result_type`.
By default, this is `int`.

Member types

The following aliases are member types of `uniform_int_distribution`:

member type	definition	notes
`result_type`	The first template parameter (`IntType`)	The type of the numbers generated (defaults to `int`)
`param_type`	*not specified*	The type returned by member param.

Member functions

(constructor)	Construct uniform discrete distribution (public member function)
operator()	Generate random number (public member function)
reset	Reset distribution (public member function)
param	Distribution parameters (public member function)
min	Minimum value (public member function)
max	Maximum value (public member function)

Distribution parameters:

a	Lower bound of range (public member function)
b	Upper bound of range (public member function)

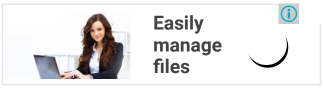
Non-member functions

operator<<	Insert into output stream (function template)
operator>>	Extract from input stream (function template)
relational operators	Relational operators (function template)

Example

```
// uniform_int_distribution
#include <iostream>
#include <random>

int main()
{
  const int nrolls = 10000; // number of experiments
  const int nstars = 95;    // maximum number of stars to distribute

  std::default_random_engine generator;
  std::uniform_int_distribution<int> distribution(0,9);

  int p[10]={};

  for (int i=0; i<nrolls; ++i) {
    int number = distribution(generator);
    ++p[number];
  }

  std::cout << "uniform_int_distribution (0,9):" << std::endl;
  for (int i=0; i<10; ++i)
    std::cout << i << ": " << std::string(p[i]*nstars/nrolls,'*') << std::endl;

  return 0;
}
```

Possible output:



```
uniform_int_distribution (0,9):
0: *********
1: *********
2: *********
3: *********
4: *********
5: *********
6: *********
7: *********
8: *********
9: *********
```

See also

uniform_real_distribution	Uniform real distribution (class template)
bernoulli_distribution	Bernoulli distribution (class)