std::poisson_distribution

Defined in header <random>

```
template< class IntType = int >
class poisson_distribution;
```
(since C++11)

Produces random non-negative integer values i, distributed according to discrete probability function:

$$P(i|\mu) = \frac{e^{-\mu}\mu^i}{i!}$$

The value obtained is the probability of exactly i occurrences of a random event if the expected, *mean* number of its occurrence under the same conditions (on the same time/space interval) is μ.

`std::poisson_distribution` satisfies *RandomNumberDistribution*

Template parameters

IntType - The result type generated by the generator. The effect is undefined if this is not one of `short`, `int`, `long`, `long long`, `unsigned short`, `unsigned int`, `unsigned long`, or `unsigned long long`.

Member types

Member type	Definition
`result_type`	`IntType`
`param_type`	the type of the parameter set, see *RandomNumberDistribution*.

Member functions

(constructor)	constructs new distribution (public member function)
reset	resets the internal state of the distribution (public member function)

Generation

operator()	generates the next random number in the distribution (public member function)

Characteristics

mean	returns the *mean* distribution parameter (mean number of occurrences of the event) (public member function)
param	gets or sets the distribution parameter object (public member function)
min	returns the minimum potentially generated value (public member function)
max	returns the maximum potentially generated value (public member function)

Non-member functions

operator== **operator!=**	compares two distribution objects (function)
operator<< **operator>>**	performs stream input and output on pseudo-random number distribution (function template)

Example

Run this code

```
#include <iostream>
#include <iomanip>
#include <string>
#include <map>
#include <random>
int main()
{
    std::random_device rd;
    std::mt19937 gen(rd());

    // if an event occurs 4 times a minute on average
    // how often is it that it occurs n times in one minute?
    std::poisson_distribution<> d(4);

    std::map<int, int> hist;
    for(int n=0; n<10000; ++n) {
        ++hist[d(gen)];
    }
    for(auto p : hist) {
        std::cout << p.first <<
                ' ' << std::string(p.second/100, '*') << '\n';
    }
}
```

Output:

```
0 *
1 *******
2 **************
3 ******************
4 ******************
5 **************
6 **********
7 *****
8 **
9 *
10
11
12
13
```

External links

Weisstein, Eric W. "Poisson Distribution." (http://mathworld.wolfram.com/PoissonDistribution.html) From MathWorld--A Wolfram Web Resource.
