

五月花

时间在哪里，成就就



问题少年学校



RSS订阅

联系我们



请扫描二维码联系客服
webmaster@csdn.net
400-660-0108
QQ客服 客服论坛

关于 招聘 广告服务 百度
©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

法和梅森旋转法

阅读数：5905

转【随机
2015年05月29日

一般我们用到
下一个随机数。
一个公用函数，

常用的快速随机
等。Windows
内一个数字只出

算法呢？伪随机算法意思是假如知道第一个随机种子和随机算法的话就可以推算出机函数的第一个随机种子，然后将随机函数返回的值作为下一个种子，随机函数是子，所以说是随机的。很多公司都有自己的一套随机算法。

法。平均分布的伪随机算法都是周期性的，在一个周期内各个数字的分布概率相/indow C运行库的随机序列周期是65536，Linux的C运行库是2^31，在一个周期当前的a[n]，下一个随机数从这里开始计算产生。

但是线性同余的周期太短了，后来出现了梅森旋转算法：请参考wiki，梅森旋转算法各项指标非常优秀，现在一般使用的是产生器有2^199 37-1长的周期，而且在1-623维均匀分布的（映射到一维直线均匀分布，二维平面内均匀分布，三维空间内均匀分布...）。

不过上面这两种算法还都不适合用于加密，因为攻击者如果收集了足够长的一段序列，就可以从数学上猜解出整个序列。可以用于加密的随机数生成算法有RSA随机数生成算法等。而产生高斯分布的随机数产生算法，据我所知有box-mueller算法。

一、线性同余法

古老的LCG(linear congruential generator)代表了最好的伪随机数产生器算法。主要原因是容易理解，容易实现，而且速度快。这种算法数学上基于X(n+1) = (a * X(n) + c) % m这样的公式，其中：
模m, m > 0
系数a, 0 < a < m
增量c, 0 <= c < m
原始值(种子) 0 <= X(0) < m
其中参数c, m, a比较敏感，或者说直接影响了伪随机数产生的质量。

一般而言，高LCG的m是2的指数次幂(一般2^32或者2^64)，因为这样取模操作截断最右的32或64位就可以了。

多数编译器的库中使用了该理论实现其伪随机数发生器rand()。下面是部分编译器使用的各个参数值：

Source	m	a	c	rand() / Random(L)的种子位
Numerical Recipes	2^32	1664525	1013904223	
Borland C/C++	2^32	22695477	1	位30..16 in rand(), 30..0 in lrand()
glibc (used by GCC)	2^32	1103515245	12345	位30..0
ANSI C: Watcom, Digital Mars, CodeWarrior, IBM VisualAge C/C++	2^32	1103515245	12345	位30..16
Borland Delphi, Virtual Pascal	2^32	134775813	1	位63..32 of (seed * L)
Microsoft Visual/Quick C/C++	2^32	214013	2531011	位30..16
Apple CarbonLib	2^31-1	16807	0	见Park-Miller随机数发生器

LCG不能用于随机数生成，因为LCG有一些严重的缺陷。另外一个问题就是，输出序列的分布一般不均匀，有些场合LCG有缺陷。



拟，不能用于加密应用。有些点最多位于 $m1/n$ 超平面上(Marsaglia定理)，这是由于产生的相继 $X(n)$ 值的关联所致。 n 远小于整体。最大周期 bn 。游戏控制台用的小整数，使用高位可以胜任。

LCG的一种实现

[cpp]

问题少年学校

```
1. // ran
2. // Cop
3. //
4. // Thi
5. // it
6. // pub
7. // Lic
8. //
9. // but
10. // MER
11. // GNU
12. //
13. // You
14. // alo
15. //
16.
17. #ifndef
18. #define
19. #includ
20.
21. class r
22. {
23. public:
24. expli
25. {
26.     if (0 == seed) seed = std::time(0);
27.     randomize();
28. }
29. void reset(unsigned long s = 0)
30. {
31.     seed = s;
32.     if (0 == seed) seed = std::time(0);
33.     randomize();
34. }
35. unsigned long rand()
36. {
37.     //returns a random integer in the range [0, -1UL)
38.     randomize();
39.     return seed;
40. }
41. double real()
42. {
43.     //returns a random real number in the range [0.0, 1.0)
44.     randomize();
45.     return double(seed) / -1UL;
46. }
47. private:
48.     unsigned long seed;
49.     void randomize() { seed = 1103515245UL * seed + 12345UL; }
50. };
51.
52. class rand_help
53. {
54.     static random r;
55. public:
56.     rand_help() {}
57.     void operator()(unsigned long s) { r.reset(s); }
58.     unsigned long operator()() const { return r.rand(); }
59.     double operator()(double) { return r.real(); }
60. };
61. random rand_help:: r;
62.
63. extern void srand(unsigned long ns = 0) { rand_help()(ns); } //reset seed
64. extern unsigned long irand() { return rand_help()(); } //negative numbers disallowed
65. extern double drand() { return rand_help()(1.0); } //for real numbers
66.
67. #endif // RANDOM_HPP_
68.
69. //以上随机数产生器产生的随机数比rand()产生的随机数更加随机(可以用数学方法检验),
70. //范围更大(一目了然), 速度更快(测试一下便知, 稍加修改我还可以让它再快一些, 如果有必要).
```

联系我们



请扫描二维码联系客服

webmaster@csdn.net

400-660-0108

QQ客服 客服论坛

关于 招聘 广告服务 百度
©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

ite it and/or modify
lic License as
version 3 of the

nd warranty of
SE. See the
s.

General Public License
gnu.org/licenses/>.

/*假设随机数均匀分布为理想分布, 粗略估计随机性*/
#include <iostream>
#include <vector>

```
#include <iom...
#include "rand...
int main()
{
    srand(time(0));

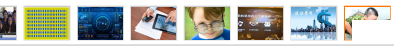
    //SZ分别取值2
    const size_t SZ = 2;
    std::vector<uint32_t> v1(SZ);
    std::vector<uint32_t> v2(SZ);
    for(size_t i = 0; i < SZ; ++i)
    {
        ++v1[rand() % v1.size()];
        ++v2[rand() % v2.size()];
    }

    for(size_t i = 0; i < SZ; ++i)
    {
        ++v3[v1[i]];
        ++v4[v2[i]];
    }

    //假设 [ 0, SSZ) 是理想分布
    //最理想的显示
    //0:表示间断, 1:表示连续
    const size_t SSZ = 10000;
    std::cout.fill(' ');
    for(size_t i = 0; i < SSZ; ++i)
    {
        std::cout << "other: " << std::setw(SSZ) << v3.size() - v3[0] - v3[1] - v3[2] - v3[3] - v3[4] << "\n";
        for(size_t i = 0; i < SSZ; ++i)
        {
            std::cout << i << ": " << std::setw(SSZ) << v4[i] << " ";
            if(i % 10 == 9) std::cout << "\n";
        }
        std::cout << "other: " << std::setw(SSZ) << v4.size() - v4[0] - v4[1] - v4[2] - v4[3] - v4[4] << "\n";
        system("pause");
    }
}
```



问题少年学校



联系我们



请扫描二维码联系客服
webmaster@csdn.net
400-660-0108
QQ客服 客服论坛

关于 招聘 广告服务 百度
©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

之间每个都应该是1

们只做粗略模糊统计，因为没有必要那么精确)
: 0, other: 0
重复

```
//做速度测试
#include <iostream>
#include <ctime>
#include <cstdlib>
#include <windows.h>
#include "random.hpp"

int main()
{
    const size_t SZ = 1 << 27;

    ...

    std::cout << "generating random numbers in progress\n";
    std::cout << SZ << " random numbers generated.\n";
    std::cout << "random used: ";
    Sleep(1000);
    std::clock_t time = clock();
    for(size_t i = 0; i < SZ; ++i)
    {
        irand();
        std::cout << clock() - time << "ms, ";
        if(i % 10 == 9) std::cout << "\n";
    }

    std::cout << "rand() used: ";
    Sleep(1000);
    std::srand(std::time(0));
    std::clock_t t = clock();
    for(size_t i = 0; i < SZ; ++i)
    {
        std::rand();
        std::cout << clock() - t << "ms\n";
    }

    std::system("PAUSE");
    return 0;
}
```

二、梅森旋转法

问题少年学校

ster算法是个不错的选择。Mersenne twister产生随机数的质量几乎超过任何LCG。不过一

imura (西村)于1997年开发的伪随机数产生器，基于有限二进制字段上的矩阵线性再生。可

缺陷。Mersenne twister这个名字来自周期长度通常取Mersenne质数这样一个事实。常见


er MT19937-64。

一般的应用来说，足够大了，序列关联比较小，能通过很多随机性测试。

pbpblog.com/Chipset/archive/2009/01/19/72330.html

- 梅森旋转算法 (Mersenne twister) 是一个伪随机数发生算法。由松本真和西村拓士[1]在1997年开发，基于有限二进制字段上的矩阵线性递归F₂。可以快速产生高质量的伪随

联系我们



请扫描二维码联系客服

✉ webmaster@csdn.net

☎ 400-660-0108

👤 QQ客服 🗣 客服论坛

梅森旋转和NAG数在SAS

Matlab, GMP和GSL的默认伪随机数产生器。从C++11开始，C++也可以使用这种算法。在Boost C++, Glib 5个PRNG中的一个：另一个是产生器仅仅为保证旧程序的兼容性，梅森旋转被描述为“更加可靠”。梅森旋转用。

最为广泛

关于 招聘 广告服务 百度

©1999-2018 CSDN版权所有

京ICP证09002463号

经营性网站备案信息

主32位整数序列。具有以下优点：

随机数产生器在一个长周期中不能保证生成随机数的质量。[2]

乱数生成器法中是最快的(当时)[3]

一种实现版本如7

```

1 //*****
2 // This is a slightly modified version of Equamen mersenne twister.
3 //
4 // Copyright (C) 2009 Chipset
5 //
6 // This program is free software: you can redistribute it and/or modify
7 // it under the terms of the GNU Affero General Public License as
8 // published by the Free Software Foundation, either version 3 of the
9 // License, or (at your option) any later version.
10 //
11 // but WITHOUT ANY WARRANTY; without even the implied warranty of
12 // MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
13 // GNU Affero General Public License for more details.
14 //
15 // You should have received a copy of the GNU Affero General Public License
16 // along with this program. If not, see <http://www.gnu.org/licenses/>.
17 //*****
18
19 // Original Coyright (c) 1997 - 2002, Makoto Matsumoto and Takuji Nishimura
20 //
21 // Functions for MT19937, with initialization improved 2002/2/10.
22 // Coded by Takuji Nishimura and Makoto Matsumoto.
23 // This is a faster version by taking Shawn Cokus's optimization,
24 // Matthe Bellew's simplification, Isaku Wada's real version.
25 // C++ version by Lyell Haynes (Equamen)
26 //
27 // Redistribution and use in source and binary forms, with or without
28 // modification, are permitted provided that the following conditions
29 // are met:
30 //
31 // 1. Redistributions of source code must retain the above copyright
32 // notice, this list of conditions and the following disclaimer.
33 //
34 // 2. Redistributions in binary form must reproduce the above copyright
35 // notice, this list of conditions and the following disclaimer in the
36 // documentation and/or other materials provided with the distribution.
37 //
38 // 3. The names of its contributors may not be used to endorse or promote
39 // products derived from this software without specific prior written
40 // permission.
41 //
42 // THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
43 // "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
44 // LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
45 // A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
46 // CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
47 // EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,

```

```

48 // PROCURE
49 // PROFITS
50 // LIABILITY
51 // NEGLIGENCE
52 // SOFTWARE
53 //
54
55 #ifndef mtrand
56 #define mtrand
57
58 #include <
59
60 class mtrand
61 {
62 public:
63     mtrand()
64
65     explicit mtrand(int seed)
66
67     mtrand(int seed, int n)
68     {
69         int i =
70         int k =
71         init();
72         for(; i
73     {
74         sta
75         sta
76         ++
77         ++
78         if(i
79     {
80         state[0] = state[N - 1];
81         i = 1;
82     }
83     if(j >= key_length)
84         j = 0;
85 }
86
87 for(k = N - 1; k; --k)
88 {
89     state[i] = (state[i] ^ ((state[i - 1] ^ (state[i - 1] >> 30)) * 1664525UL)) + init_key[j] + j; // non linear
90     state[i] &= 4294967295UL; // for WORDSIZE > 32 machines
91     ++i;
92     if(i >= N)
93     {
94         state[0] = state[N - 1];
95         i = 1;
96     }
97 }
98
99 state[0] = 2147483648UL; // MSB is 1; assuring non-zero initial array
100 }
101
102 void reset(size_t rs)
103 {
104     init(rs);
105     next_state();
106 }
107
108 size_t rand()
109 {
110     size_t y;
111     if(0 == --left)
112         next_state();
113     y = *next++;
114     // Tempering
115     y ^= (y >> 11);
116     y ^= (y << 7) & 0x9d2c5680UL;
117     y ^= (y << 15) & 0xefc60000UL;
118     y ^= (y >> 18);
119     return y;
120 }
121
122 double real() { return (double)rand() / -1UL; }
123
124 // generates a random number on [0,1) with 53-bit resolution
125 double res53()

```

问题少年学校

联系我们

请扫描二维码联系客服

webmaster@csdn.net

400-660-0108

QQ客服 客服论坛

关于 招聘 广告服务 百度

©1999-2018 CSDN版权所有

京ICP证09002463号

经营性网站备案信息

网络110报警服务

中国互联网举报中心

北京互联网违法和不良信息举报中心

state[0] = state[N - 1];

i = 1;

if(j >= key_length)

j = 0;

state[i] = (state[i] ^ ((state[i - 1] ^ (state[i - 1] >> 30)) * 1664525UL)) + init_key[j] + j; // non linear

state[i] &= 4294967295UL; // for WORDSIZE > 32 machines

++i;

if(i >= N)

{

state[0] = state[N - 1];

i = 1;

}

state[0] = 2147483648UL; // MSB is 1; assuring non-zero initial array

void reset(size_t rs)

{

init(rs);

next_state();

}

size_t rand()

{

size_t y;

if(0 == --left)

next_state();

y = *next++;

// Tempering

y ^= (y >> 11);

y ^= (y << 7) & 0x9d2c5680UL;

y ^= (y << 15) & 0xefc60000UL;

y ^= (y >> 18);

return y;

}

double real() { return (double)rand() / -1UL; }

// generates a random number on [0,1) with 53-bit resolution

double res53()

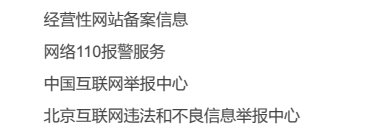
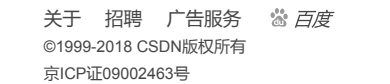
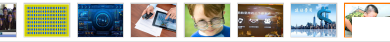
```

126 {
127     size_t
128     return 0.0;
129 }
130
131 private:
132 void init
133 {
134     state
135     for(int i = 0; i < N; ++i)
136     {
137         state[i] = (state[i - 1] >> 30) + j);
138         // See Knuth TAOCP Vol 2, 2nd Ed, p.106 for multiplier.
139         // 联系我们
140         //
141         //
142         state[i] = (state[i] * multiplier) >> shift;
143     }
144 }
145
146 void next
147 {
148     size_t
149     int i;
150
151     for(i = 0; i < N; ++i)
152     {
153         *p
154         for(i = 0; i < N; ++i)
155         {
156             *p =
157             left = iv;
158             next = state;
159         }
160
161     size_t mixbits(size_t u, size_t v) const
162     {
163         return (u & 2147483648UL) | (v & 2147483647UL);
164     }
165
166     size_t twist(size_t u, size_t v) const
167     {
168         return ((mixbits(u, v) >> 1) ^ (v & 1UL ? 2567483615UL : 0UL));
169     }
170
171     static const int N = 624, M = 397;
172     size_t state[N];
173     size_t left;
174     size_t* next;
175 };
176
177 class mtrand_help
178 {
179     static mtrand random r;
180 public:
181     mtrand_help() {}
182     void operator()(size_t s) { r.reset(s); }
183     size_t operator()() const { return r.rand(); }
184     double operator()(double) { return r.real(); }
185 };
186 mtrand random mtrand_help::r;
187
188 extern void mtsrand(size_t s) { mtrand_help()(s); }
189 extern size_t mtirand() { return mtrand_help()(); }
190 extern double mtdrand() { return mtrand_help()(1.0); }
191
192 #endif // mtrand_HPP_
193

```



问题少年学校



```

1 //*****
2 // This is a slightly modified version of Equamen mersenne twister.
3 //
4 // Copyright (C) 2009 Chipset
5 //
6 // This program is free software: you can redistribute it and/or modify
7 // it under the terms of the GNU Affero General Public License as

```



```

8 // published
9 // License,
10 //
11 // but WIT
12 // MERCHA
13 // GNU Aff
14 //
15 // You sho
16 // along wi
17 //*****
18
19 // Original
20 //
21 // Function:
22 // Coded by
23 // This is a
24 // Matthe B
25 // C++ ver:
26 //
27 // Redistrib
28 // modificat
29 // are met:
30 //
31 // 1. Redist
32 // notice,
33 //
34 // 2. Redist
35 // notice,
36 // docum
37 //
38 // 3. The na
39 // products dervea from this software without specific prior written
40 // permission.
41 //
42 // THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
43 // "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
44 // LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
45 // A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
46 // CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
47 // EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
48 // PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
49 // PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
50 // LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
51 // NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
52 // SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
53 //
54
55 #ifndef mtrandonm_HPP_
56 #define mtrandonm_HPP_
57
58 #include <stddef.h>
59
60 class mtrandonm
61 {
62 public:
63     mtrandonm() : left(1) { init(); }
64
65     explicit mtrandonm(size_t seed) : left(1) { init(seed); }
66
67     mtrandonm(size_t* init_key, int key_length) : left(1)
68     {
69         int i = 1, j = 0;
70         int k = N > key_length ? N : key_length;
71         init();
72         for(; k; --k)
73         {
74             state[i] = (state[i] ^ ((state[i - 1] ^ (state[i - 1] >> 30)) * 1664525UL)) + init_key[j] + j; // non linear
75             state[i] &= 4294967295UL; // for WORDSIZE > 32 machines
76             ++i;
77             ++j;
78             if(i >= N)
79             {
80                 state[0] = state[N - 1];
81                 i = 1;
82             }
83             if(j >= key_length)
84                 j = 0;
85         }

```

问题少年学校

联系我们

请扫描二维码联系客服

webmaster@csdn.net

400-660-0108

QQ客服 客服论坛

关于 招聘 广告服务 百度

©1999-2018 CSDN版权所有

京ICP证09002463号

经营性网站备案信息

网络110报警服务

中国互联网举报中心

北京互联网违法和不良信息举报中心

sion 3 of the

ied warranty of
JRPOSE. See the

eneral Public License
org/licenses/>.

and Takuji Nishimura

02/2/10.

ization,
on.

n or without
conditions

e copyright
mer.

bove copyright
mer in the
the distribution.

ndorse or promote

```

86
87 for(k
88 {
89     sta >> 30)) * 1566083941UL)) - i; // non linear
90     sta 32 machines
91     ++
92     if(i
93     {
94
95
96     }
97 }
98
99 state[ 联系我们 non-zero initial array
100 }
101
102 void res
103 {
104     init(r
105     next_
106 }
107
108 size_t ra
109 {
110     size_
111     if(0 =
112         ne
113         y =
114         // Te
115         y ^=
116         y ^=
117         y ^= (y << 15) & 0xercb0000UL;
118         y ^= (y >> 18);
119     return y;
120 }
121
122 double real() { return (double)rand() / -1UL; }
123
124 // generates a random number on [0,1) with 53-bit resolution
125 double res53()
126 {
127     size_t a = rand() >> 5, b = rand() >> 6;
128     return (a * 67108864.0 + b) / 9007199254740992.0;
129 }
130
131 private:
132 void init(size_t seed = 19650218UL)
133 {
134     state[0] = seed & 4294967295UL;
135     for(int j = 1; j < N; ++j)
136     {
137         state[j] = (1812433253UL * (state[j - 1] ^ (state[j - 1] >> 30)) + j);
138         // See Knuth TAOCP Vol2. 3rd Ed. P.106 for multiplier.
139         // In the previous versions, MSBs of the seed affect
140         // only MSBs of the array state[].
141         // 2002/01/09 modified by Makoto Matsumoto
142         state[j] &= 4294967295UL; // for >32 bit machines
143     }
144 }
145
146 void next_state()
147 {
148     size_t* p = state;
149     int i;
150
151     for(i = N - M + 1; --i; ++p)
152         *p = (p[M] ^ twist(p[0], p[1]));
153
154     for(i = M; --i; ++p)
155         *p = (p[M - N] ^ twist(p[0], p[1]));
156     *p = p[M - N] ^ twist(p[0], state[0]);
157     left = N;
158     next = state;
159 }
160
161 size_t mixbits(size_t u, size_t v) const
162 {
163     return ((u & 2147483648ULL) | (v & 2147483647ULL));

```




```
164 }
165
166 size_t tv
167 {
168     return
169 }
170
171 static co
172 size_t st
173 size_t le
174 size_t*
175 };
176
177 class mtrai
178 {
179     static mtr
180 public:
181     mtrand_f
182     void oper
183     size_t op
184     double op
185 };
186 mtrandom
187
188 extern voic
189 extern size
190 extern dou
191
192 #endif // n
193
```



问题少年学校



联系我们



请扫描二维码联系客服
✉ webmaster@csdn.net
☎ 400-660-0108
🗣 QQ客服 🗣 客服论坛

关于 招聘 广告服务 百度

©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

文章标签： C++ 随机数 生成算法 线性同余法 梅森旋转法

个人分类： 算法

查看更多>>

想对作者说点什么？ 我来说两句

用线性同余法生成“伪”随机数

线性同余方法（LCG）是个产生伪随机数的方法。它是根据递归公式： 其中是产生器设定的常数。LCG的周期最大为，但大部分情况都会少于M。要令LCG达到最大周期，应符合以...

memray 2013-05-15 21:41:39 阅读数：27311

rand()函数实现原理：线性同余法

关于“随机数”的产生有许多算法，但无论如何，都不可能产生真正的随机数，因为电脑程序是个确定状态转换机，一种输入必定产生一种确定的输出。但要实现“不可预知”还是可以做到的，只需有“不可预知”的...

chienchia 2014-11-21 20:14:45 阅读数：5991

肾虚怎么办?快男调理妙招分享!!!

尤芙生物 · 顶新

使用线性同余法生成伪随机数/序列（C++实现）

计算机上可以用物理方法来产生随机数，但价格昂贵，不能重复，使用不便。另一种方法是用数学递推公式产生，这样产生的序列与真正的随机数序列不同，所以称为伪随机数或伪随机序列，只要方法和参数选择合适，所产生的...

liyuefeilong

线性同余法

现在的随机函数发生器都是采用线性同余法来生成随机数的。其原理如下：
b对模m同余。所以，如果两个整数a，b用m除，所得的余数相同，则称a，b对模m同余。记作： $a \equiv b \pmod m$ 。这样描述的：设m是一个给定的正整数，如果两个整数a，b用m除，所得的余数相同，则称a，b对模m同余。记作： $a \equiv b \pmod m$ 。

jicheng687



随机数生成器

1、随机数生成器（Random Number Generator，RNG）是产生随机数的“器”（器存在软件与硬件之分），真正的随机数生成器产生的随机数具有随机性、不可预测性、均匀性等特性。...

Apollon_krj



线性同余法

线性同余法求伪随机数

t6_17 2010-02-27 11:14:00

联系我们



请扫描二维码联系客服

webmaster@csdn.net

400-660-0108

QQ客服 客服论坛

喝水都会胖

牧凡·顶新

关于 招聘 广告服务 百度
©1999-2018 CSDN版权所有
京ICP证09002463号

RNG分析：

古老的LCG(linear congruential generator)公式： $X(n+1) = (aX(n) + b) \pmod m$ 。随机数产生器算法。主要原因是容易理解，容易实现，而且速度快。LCG 算法数学上基于公式： $X(n+1) = (aX(n) + b) \pmod m$ 。0, $m > 0$, $m \% a \neq 0$ 首先，说明一下取随机数一般会用rand函数，取ti...

flyoxs 2010-02-27 11:14:00 阅读数：4528

线性同余法

2013年01月27日 581B 下载



梅森旋转算法

梅森旋转算法（Mersenne twister）是一个伪随机数发生算法。由松本真和西村拓士[1]在1997年开发，基于有限二进制字段上的矩阵线性递归。可以快速产生高质量的伪随机数，修正了古典随机...

xiaogou56a 2014-04-06 19:59:23 阅读数：9504

伪随机数生成——梅森旋转（Mersenne Twister/MT）算法笔记

前言 最近在看吴军博士的《数学之美》一书，把很多之前没注意到，没用到，甚至不知道怎么用的数学知识和实际问题联系了起来，感觉打开了新世界的大门一样。这本书很多知识点还有技术都是点到为止，并没有深入...

tick_tock97 2017-11-28 20:11:51 阅读数：1355

线性反馈移位寄存器与梅森旋转算法

今天主要是来研究梅森旋转算法，它是用来产生伪随机数的，实际上产生伪随机数的方法有很多种，比如线性同余法，平方取中法等等。但是这些方法产生的随机数质量往往不是很高，而今天介绍的梅森旋转算法可以产生高质量...

ACdreamers 2015-03-26 21:54:33 阅读数：14134

梅森旋转随机算法

梅森旋转随机算法，C++和Java代码。 C++：#ifndef _MersenneTwister_H_ #define _MersenneTwister_H_ #include #incl...

xiadasong007 2014-01-27 20:10:31 阅读数：6109

梅森旋转算法--伪随机数（加密、身份信息ID号）

http://blog.csdn.net/ACdreamers/article/details/44656743 今天主要是来研究梅森旋转算法，它是用来产生伪随机数的，实际上产生伪随机数的方法有很多...

Touch_Dream 2017-04-01 22:56:20 阅读数：1114

扫地机器人

618狂欢科沃斯直

梅森旋转法

直接贴代码了：

shuimu12345



RSA算法之

本文的目的在于

及4096位。由于

qmickecs

梅森旋转随

梅森旋转算法实现

chinabhl

梅森旋转算

Mersenne Twiste

具 Ir...

q270274978

联系我们

请扫描二维码联系客服

webmaster@csdn.net

400-660-0108

QQ客服 客服论坛

关于 招聘 广告服务 百度

©1999-2018 CSDN版权所有

京ICP证09002463号

经营性网站备案信息

网络110报警服务

中国互联网举报中心

北京互联网违法和不良信息举报中心

#define N (624) // ...

RSA加密程序。RSA中的密钥长度指的是公钥的长度，目前主流的公钥长度为1024、2048以

***** // This is a ...

的随机函数不能满足需求，找了个梅森旋转算法。因为使用lua，找了个c写的为lua实现的工

浅谈随机数发生器

我们平时所使用的无论什么编程语言都会提供一个随机数函数，而且它是伪随机数（Pseudo Random Number），它是由算法计算得出的，是可以预测的，也就是说当随机种子相同时，对于同一个随机函数，...

nash_ 2013-12-19 01:51:28 阅读数：16144

免费云主机试用一年

云主机免费推荐吗

百度广告



产生伪随机数两种常用算法

我们讲的随机数其实暗指伪随机数。不少朋友可能想到C语言的rand()，可惜这个函数产生的随机数随机性非常差，而且速度很慢，相信几乎不能胜任一般的应用。古老的LCG(linear congruent...

mergerly 2015-06-24 11:29:09 阅读数：11630

生成伪随机数的算法—线性同余法

现在的随机函数发生器大都采用的是线性同余法。同余的概念是这样描述的：设m是一个给定的正整数，如果两个整数a，b用m除，所得的余数相同，则称a，b对模m同余。所谓线性同余法（又叫混合合同...

myself_helper 2013-03-19 23:03:46 阅读数：5785

线性同余法产生随机数

使用线性同余法产生均匀分布的随机数 说明：程序主要是参考一本很老的书《数字信号处理的C语言程序集》，大连理工大学的殷福亮，宋爱军编写，很不错，大家可以看一下，一些基础的信号处理方法上面都有，以后有程...

OpticalSoliton 2014-12-20 10:02:43 阅读数：1445

线性同余算法以及java中随机数的实现分析

线性同余算法(伪随机数算法)

lizhi20091225 2013-08-20 23:07:33 阅读数：1493

伪随机生成之线性同余算法

【参考博文】伪随机数生成器
如果能够通过统计方法验证其随机性，那么就可以认为是随机的。

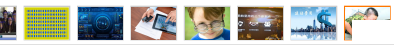
ych1995612

扫地机器人

618狂欢科沃斯直降



问题少年学校



随机算法

我们讲的随机数生成器，其实就是一种算法。古老的LCG（线性同余法）就是其中一种。

hoxily 2018-06-08

C++随机数生成器

下面的代码来自C++标准库。

ClamReason

mt19937 随机数生成器

一下内容整理自网络，仅供参考。相信几乎不能胜任任何工作。

Real_Myth

联系我们



请扫描二维码联系客服
webmaster@csdn.net
400-660-0108
QQ客服 客服论坛

关于 招聘 广告服务 百度
©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

通过某种数学公式或者**算法**产生的数值序列。虽然在数学意义上伪随机数是不随机的，但是...



rand(), 可惜这个函数产生的随机数随机性非常差，而且速度很慢，相信几乎不能胜任一般的应

or #include <iostream> #include <i...

不少朋友可能想到C语言的rand(), 可惜这个函数产生的随机数随机性非常差，而且速度很慢，

关于同余与模运算的总结

123456789*987654321 = (A: 121932631112635266 B: 121932621...

chocolate_22 2011-05-31 21:55:00 阅读量：4704

同余定理

同余运算及其基本性质 100除以7的余数是2，意思就是说把100个东西七个七个分成一组的话最后还剩2个。余数有一个严格的定义：假如被除数是a，除数是b（假设它们均为正整数），那么我们总能...

CodeHarvest 2017-04-21 14:36:40 阅读量：2968

做网站需要多少钱

一个网站建立到运营需要多少钱

百度广告



随机数生成算法，每一次生成都不一样

项目中要用到一个**随机数生成算法**，但是每一次生成间隔时间很短，还要保证每一次生成的随机数都不相同，于是就想到了利用当前时间的毫秒作为随机数种子来生成。#include #include // f...

chenmoo0821 2016-02-25 09:41:58 阅读量：1392

C语言之实现随机数产生算法

随机数，也就是在不同的时刻产生不同的数值。在UNIX操作系统和window的操作系统上，我们知道有一个函数rand，它就是用来产生随机数的函数API接口，那么它的原理如何实现？如果约定a1=f(s...

morixinguan 2016-02-20 15:50:51 阅读量：8020

一个生成伪随机数的超级算法【转】

最近浏览“程序员论坛”时发现不少好帖，增长了不少知识，现拿其中一则为例与大家共同分享心得。 某人提出一个问题：怎样才能生成一亿个不重复的随机数？ 问题表述起来很简单，似乎只要弄明白...

lianyumook2010 2018-01-16 23:47:06 阅读量：104

随机数算法

一、随机数概述在密码技术中，随机序列是非常重要的，比如密钥产生、数字签名、身份认证和众多的密码学协议等都要用到随机序列。所以产生高质量的随机数序列对信息的安全性具有十分重要的作用。随机数分为真随机数...

北京互联网违法和不良信息举报中心



是的，没错

元素顺序随机分布，范围为0 - n-1，且元素不能重复。解决思路：1)、声明一个数组N



简介 随机数在概率论和统计学中，指在一系列可能的取值中，每一个值出现的概率相等。在数学意义上，随机数是指一个数列元素之间近似独立的随机数，一般使用伪随机数发生器产生的伪随机数。伪随机数发生器是一个**算法**，产生的随机数。

Xminyang

问题少年学校

伪随机生成

定义：伪随机数：在数学意义上伪随机数是不随机的，但是如果能够通过统计检验，可以当成真随机数使用。**算法**：伪随机数生成器。

orchidzouqr

联系我们

50万码农评

不背单词和语法，

线性同余法

1、**线性同余法**： $x_{n+1} = (ax_n + c) \bmod m$

fengying2016

关于 招聘 广告服务 百度

【专题】线性同余法

定义：a,b是整数，m是正整数，且a,b,m互质。称形如 $x_{n+1} = (ax_n + b) \bmod m$ 的递推公式为**线性同余法**。当a=1,b=0时，称为**线性同余法**。当a=1,b=1时，称为**线性同余法**。当a=1,b=0时，称为**线性同余法**。当a=1,b=1时，称为**线性同余法**。

mmy1996

经营性网站备案信息

想是通过

然在数学意义上伪随机数是不随机的，但是如果能够通过统计检验，可以当成真随机数使用。

联系我们

请扫描二维码联系客服

webmaster@csdn.net

400-660-0108

QQ客服 客服论坛

线性同余法

想是通过

想是通过

然在数学意义上伪随机数是不随机的，但是如果能够通过统计检验，可以当成真随机数使用。

j2me random类(线性同余法)

前几天,在做J2ME软件时,用到了RANDOM类的nextInt(int arg0)方法,感觉不怎么好,所以自己利用线性同余法重新写了一个,希望有用!...

下载 2018年05月08日 00:00

Math.random() 线性同余 伪随机数

同余方程：对于一组整数Z，Z里的每一个数都除以同一个数m，得到的余数可以为0，1，2，...m-1,共m种。我们就以余数的大小作为标准将Z分为m类。每一类都有相同的余数 在每一类下的任意两...

u011916334

2016-07-29 13:46:46

阅读数：717

均匀分布随机数的生成算法简介

均匀分布随机数，线性同余法，反馈位寄存器法，组合随机数发生器

lidbcode

2017-11-17 14:43:22

阅读数：7681

一个网站建立到运营需要多少钱

做网站需要多少钱

百度广告



混合同余法产生随机数和M序列产生方法（算例及matlab程序）

2009年10月07日 83KB 下载



伪随机数生成算法（1）线性同余法

线性同余随机数生成器介绍：古老的LCG(linear congruential generator)代表了最好最朴素的伪随机数产生器**算法**。主要原因是容易理解，容易实现，而且速度快。 ...

feipeixuan

2014-04-29 09:05:22

阅读数：2265

没有更多推荐了，[返回首页](#)

个人资料

friendbkf

https://blog.csdn.net/friendbkf/article/details/46237635

14/16



原创
109

粉丝
15

等级： 博客 5

积分： 2933

勋章：



问题少年学校





心理抑郁测试



最新文章

win7 64位Ultimate
iewer 7.2的方法

晶体三极管工作原

64位Linux环境 编译32位汇编程序（外链C
库函数）

返回给定值在有序数组中的插入位置

【剑指offer】复杂链表的复制

联系我们



请扫描二维码联系客服
✉ webmaster@csdn.net
☎ 400-660-0108
🗣 QQ客服 🗣 客服论坛

关于 招聘 广告服务  百度

©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

个人分类

C++学习笔记40篇

Qt学习笔记10篇

字符编码5篇

MySql1篇

C++模板与泛型2篇

展开

归档

2016年11月1篇

2016年10月2篇

2016年2月1篇

2016年1月1篇

2015年12月25篇

展开

热门文章

vs2013的代码 自动提示的问题
阅读量：9996

linux下配置Qt5 开发环境
阅读量：7556

QtCreator支持C++11的设置方法
阅读量：6975

二叉树的层次遍历+每一层单行输出
阅读量：6716

安装及设置MASM32 SDK
阅读量：6531

https://blog.csdn.net/friendbkf/article/details/46237635

15/16

最新评论

八皇后问题的分析
u011732358 : good job

LS和DV路由协议的
Lee_feiyue : 受教了!

八皇后问题的分析
qq_37804582 : [quote]谢谢博主[/quote]

八皇后问题的分析
acrelshrl : 写得挺清楚

C++重载 箭头运算
zhouyue777111 : 讲得



问题少年学校



联系我们



请扫描二维码联系客服

 webmaster@csdn.net

 400-660-0108

 QQ客服  客服论坛

关于 招聘 广告服务  百度

©1999-2018 CSDN版权所有
京ICP证09002463号

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

https://blog.csdn.net/friendbkf/article/details/46237635

16/16