

```
1 #include <lnv3/lnv3.h>
2 #include <nttl/randomPrime.h>
3 #include <nttl/gcd.h>
4 #include <nttl/inverse.h>
5
6 #define P 739
7
8 bool TestGenerator(int g);
9
10 void main()
11 {
12     for( int g = 2; g < P; g++ )
13     {
14         bool valid = TestGenerator( g );
15         if( valid == true )
16         {
17             cout << "Generator found, g = " << g << endl;
18             return;
19         }
20         else
21             cout << "g = " << g << " failed as a generator" << endl;
22     }
23 }
24
25 bool TestGenerator(int g)
26 {
27     bool present[ P ];
28     memset( present, false, sizeof( bool ) * P );
29
30     for( int t = 1; t < P; t++ )
31     {
32         ln p = ln( g ).FastExp( t, P );
33         int v = p.GetDigit( 0 );
34
35         present[ v ] = true;
36     }
37
38     for( int n = 1; n < P; n++ )
39     {
40         if( present[ n ] == false )
41             return false;
42     }
43
44     return true;
45 }
46
```