\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



## **FACULTY of**

## **CYBERNETICS AND**

## **INFORMATION SECURITY**

###### Department of

###### Cryptology and discrete mathematics

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Course: «Cryptographic techniques of information security»**

**Project №2**

on subject

**«Developing and testing the pseudorandom number generator»**

Made by:

Semkin Pete

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Moscow –– 2014

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Contents

[1) Description of the investigated pseudorandom number generator (PRNG)](#Part1)

[2) Description of the techniques to test sequences generated with the PRNG](#Part2)

[3) Analyses of the test results and conclusions about the cryptographic quality of the PRNG](#Part3)

1) Description of the investigated pseudorandom number generator

In this project the generator is based on the three linear feedback shift registers (LFSR):

1. LSFR-1 of length 27 (the significant bits are 1, 2, 5, 27),

2. LFSR-2 of length 26 (1, 2, 6, 25),

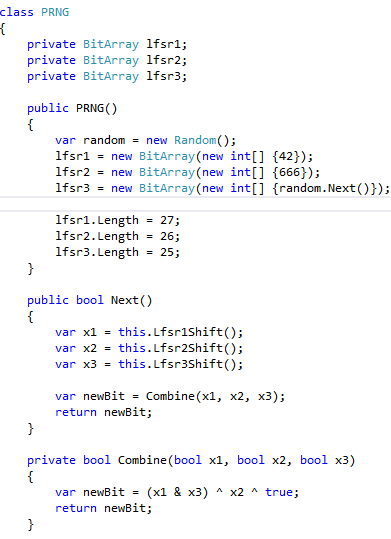
3. LFSR-3 of length 25 (3, 25).

The PRNG uses the following non-linear combining function:

*g* = *x*1 ⋅ *x*3 ⊕ *x*2 ⊕1,

where *x*1, *x*2, *x*3 are the bits we get after shifting LFSR-1, LFSR-2, LFSR-3.

Implementation on C#/.NET (excerpt):



2) Description of the techniques to test sequences generated with the pseudorandom number generator

In this project the test to investigate the quality of PRNG is the Test for the Longest Run of Ones in a Block). The description and the reference data are taken from the NIST specification (“A statistical test suite for random and rseudorandom number generators for cryptographic applications”).

This test does the following:

1. The PRNG is used to get a binary sequence.

2. The sequence is split to the blocks of length M.

3. In every block the longest run of ones is found.

4. All the longest runs are stored.

5. The frequencies of different values of the stored lengths are calculated.

6. Basing on the calculated frequencies and some known (from reference data) coefficients the so called P-value is computed with the help of the partial Gamma function. This P-value is actually the indicator of randomness of the sequence.

7. P-value is compared with the value 0.01. If P-value is not less than 0.01 then the sequence is considered to be random.

The test is conducted three times for the following lengths of blocks:

1. M = 8,

2. M = 128,

3. M = 10000.

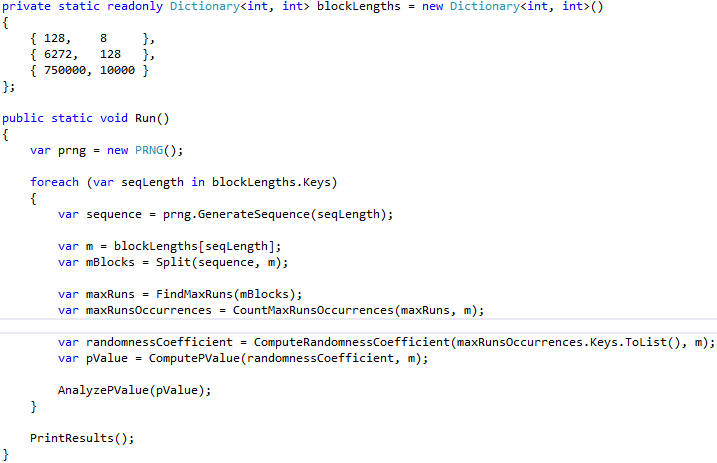
The corresponding lengths of the sequences are the following:

1. n = 128,

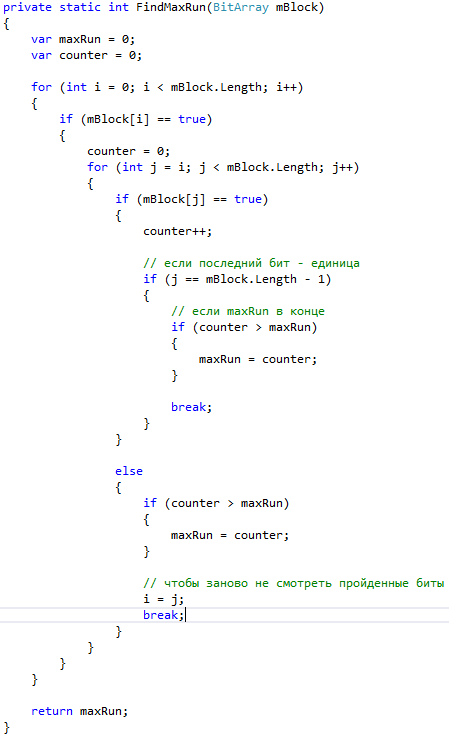
2. n = 6272,

3. n = 750000.

Implementation on C#/.NET (excerpt):

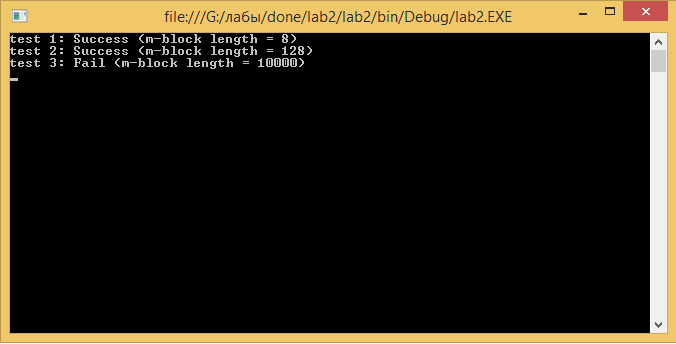


The method to find the longest run of ones in a block:



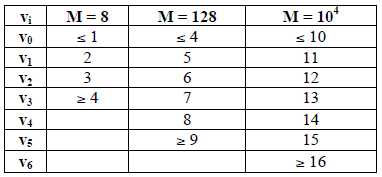
3) Analyses of the test results and conclusions about the cryptographic quality of the pseudorandom number generator

The test succeeded for two lengths of M-blocks: for 8 и 128 (failed for 10000).

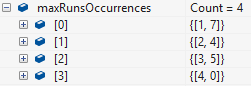


The result of the last test can be explained by the much bigger length of m-block than in two other tests. It seems that in this case the distribution of the values of the longest runs of ones is not good enough to consider the sequence to be random.

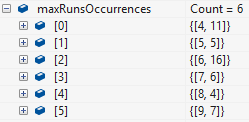
The investigated values of the longest runs of ones were taken from the following table (NIST):



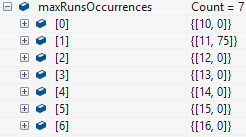
Length distribution for 8-blocks:



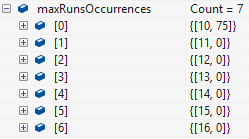
Length distribution for 128-blocks:



Length distribution for 10000-blocks:



Length distribution for 10000-blocks (another sequence):



As we can see, for the big length of M every M-block somehow has an equal longest run of ones. As a result, the P-value turns to be much less than 0.01 and the test fails. The conclusion is that the PRNG under research is not good enough.