Laboratorium nr 7-8

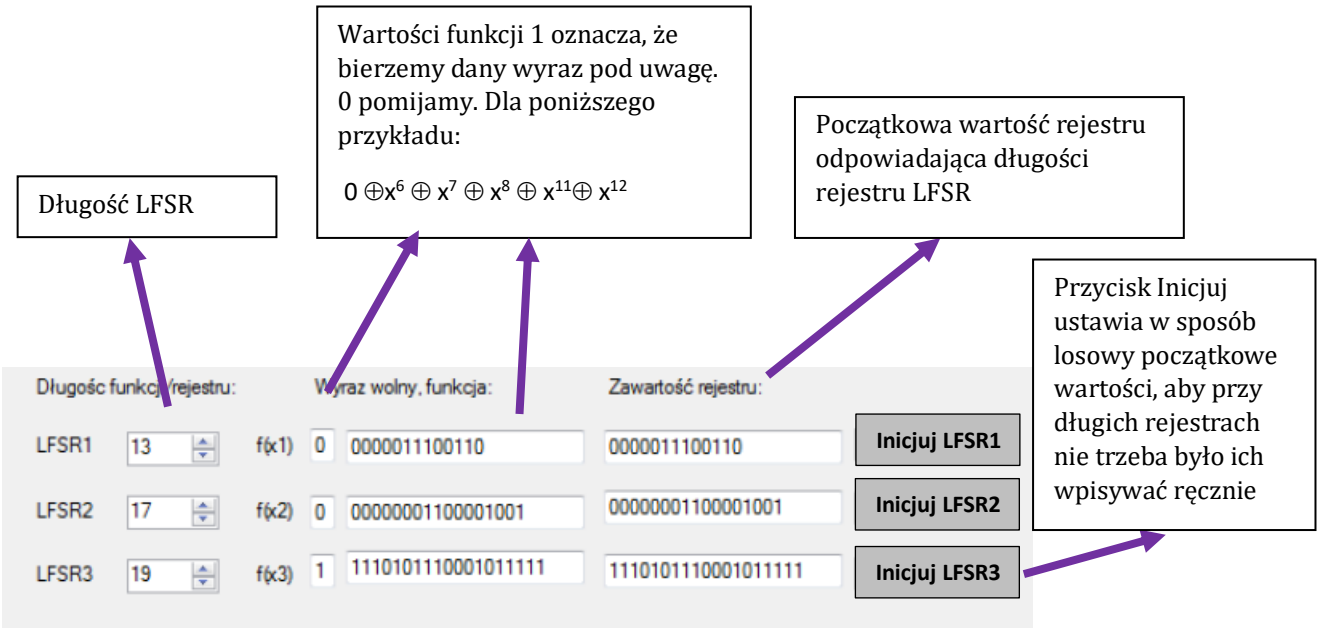
Generowanie strumieni klucza przy użyciu szyfratorów strumieniowych

Czas wykonania: **do terminu laboratorium nr 9**
Liczba punktów: **6 + 4 + 2 dodatkowe**
PRK: T-L-5

1. Opis laboratorium

Celem laboratorium jest stworzenie oraz przetestowanie aplikacji służącej do badania addytywnych binarnych szyfratorów strumieniowych opartych o liniowe rejestry przesuwające LFSR.

Przykładowy sposób budowy fragmentu interfejsu w zakresie LFSR:



Opis testów (wyciąg z wykładu nr 4):

<u>Test „pokerowy" (Poker test)</u>

1) Podzielić badany ciąg na 4-bitowe paczki obejmujące 4 kolejne bity (paczek tych jest 5000). Zliczyć częstości f (i) pojawiania się każdej z możliwych 16 sekwencji 4-bitowych ($0 \le i \le 15$).
2) Obliczyć wielkość X: $X = (16/5000)\ (\Sigma(f(i))^2)-5000$.
3) Wynik testu jest pomyślny wtedy, gdy $2.16 < X < 46.17$.

Test „długich podciągów identycznych ciągów" (Long runs test)

1) Jeżeli w badanym ciągu istnieje co najmniej jeden podciąg o długości > 26 bitów zawierający same bity o wartości 0 lub same bity o wartości 1, to wynik testu jest negatywny.

Test „podciągów identycznych ciągów" (Runs test)

1) Zliczyć wszystkie podciągi składające się tylko z bitów o wartości 0, albo tylko z bitów o wartości 1. Podzielić je na sześć grup:
   - pierwszą - zawierającą podciągi o długości 1 bita;
   - drugą - zawierającą podciągi o długości 2 bitów;
   - ....
   - szóstą - podciągi o długości 6 i więcej bitów.
2) Jeżeli liczebność którejkolwiek z sześciu grup podciągów nie mieści się w zakresie podanym w poniższej tablicy, to wynik testu jest negatywny.

1	***2315-2685***
2	***1114-1386***
3	***527-723***
4	***240-384***
5	***103-209***
6+	***103-209***

2. Materiały

1. Wykład nr 4
2. A. Menezes, P. van Oorschot, and S. Vanstone, Handbook of Applied Cryptography, Chapter 6 Stream Ciphers, http://cacr.uwaterloo.ca/hac/about/chap6.pdf
3. Shift-Register Stream Ciphers, http://www.quadibloc.com/crypto/co040801.htm
4. Ryszard Tanaś, Wykład na temat Szyfrowanie strumieniowe i generatory ciągów pseudolosowych, http://zon8.physd.amu.edu.pl/~tanas/krypt08.pdf
5. Bruce Schneier, Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C, http://friedo.szm.com/krypto/AC/ch17/17-06.html
6. NIST SP 800-22, https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf

Dotyczące implementacji testu w zadaniu dodatkowym:

7. https://docs.microsoft.com/en-us/archive/msdn-magazine/2013/government-special-issue/test-run-implementing-the-national-institute-of-standards-and-technology-tests-of-randomness-using-csharp
8. https://github.com/DSC-SPIDAL/csharp/blob/master/SalsaTPL/Salsa.CoreTPL/SpecialFunction.cs [C#]
9. http://accord-framework.net/docs/html/T_Accord_Math_Gamma.htm [C#]
10. https://docs.scipy.org/doc/scipy/reference/special.html [Python]

3. Zadania do wykonania

Napisz aplikacje umożlwiającą generowanie strumieni klucza oraz ich testowania zgodnie z poniższymi zadaniami. Aplikacja może posiadać interfejs graficzny lub działać w trybie tekstowym.

1) **Zadanie 7.1** (2 pkt do terminu lab 7) – zaimplementuj liniowy rejestr przesuwający.
2) **Zadanie 7.2** (2 pkt) – używając implementacji liniowego rejestru przesuwającego z zadania 7.1 zaimplementuj 3 różne warianty generatora liczb losowych:
   i) Generator Geffe'go,
   ii) Stop-and-Go,
   iii) Shrinking Generator.

   Aplikacja musi pozwalać na ustalanie długości wykorzystywanych rejestrów LFSR oraz współczynników określających liniową funkcję w pętli sprzężenia zwrotnego każdego z rejestrów; parametrami pełniącymi rolę klucza kryptograficznego są początkowe stany użytych w generatorze strumienia klucza rejestrów LFSR (patrz przykład na rysunku na stronie 1).
3) **Zadanie 8.1** (2 pkt) – Zaimplementuj następujące testy losowości wygenerowanych ciągów binarnych zgodnie z specyfikacją FIPS 140-2 tj.:
   i) (2 pkt do terminu lab 8), test pokerowy (*Poker test),*
   ii) test długich podciągów identycznych ciągów (*Long runs test*),
   iii) test podciągów identycznych ciągów (*Runs test*).
   iv) (opcjonalne, **+2 pkt**) zaimplementuj test „Frequency Test within a Block" z specyfikacji NIST SP 800-22 [6] – należy użyć gotowej implementacji funkcji **igamc** wymienionej w opisie testu.
4) **Zadanie 8.2** (2 pkt) – Przetestuj każdy z generatorów z użyciem powyższych testów z użyciem ciągu o długości 20000 bitów. Czy zaimplementowane przez ciebie generatory spełniają wymagania z testów? Jak długość LSFR wpływa na wyniki testów?