Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Фізико-технічний інститут

ЛАБОРАТОРНА РОБОТА №4

з дисципліни

«Криптографія»

на тему: «Побудова регістрів зсуву з лінійним зворотним зв’язком та дослідження їх властивостей»

Виконали:

студенти 3 курсу ФТІ

групи ФБ-72

Солдатова Катерина,

Яшкова Вікторія

Перевірили:

Чорний О.

Савчук М. М.

Завадська Л. О.

Мета роботи:

Ознайомлення з принципами побудови регістрів зсуву з лінійним зворотним зв'язком; практичне освоєння їх програмної реалізації; дослідження властивостей лінійних рекурентних послідовностей та їх залежності від властивостей характеристичного полінома регістра.

Варіант:

15

$$P_1(X)= X^{23}+X^{20}+X^{17}+X^{16}+X^{14}+X^{12}+X^{10}+X^9+X^8+X^7+X^3+X+1$$

$$P_2(X)= X^{20}+X^{18}+X^{17}+X^{16}+X^{13}+X^{12}+X^{11}+X^9+X^6+X^5+1$$

Результати:

Перший поліном P1(X)= X23+X20+X17+X16+X14+X12+X10+X9+X8+X7+X3+X+1

Період лінійної рекурентної послідовності заданої поліномом: 8388607

Розподіл монограм:

0	4194303
1	4194304

Розподіл біграм:

01	2097152
11	2097151
10	2097152
00	2097151

Автокореляція:

Зсув	Значення
1	4194304
2	4194304
3	4194304
4	4194304
5	4194304
6	4194304
7	4194304
8	4194304

9	4194304
10	4194304

Другий поліном P2(X)=X20+X18+X17+X16+X13+X12+X11+X9+X6+X5+1

Період лінійної рекурентної послідовності заданої поліномом: 349545

Розподіл монограм:

0	174440
1	175105

Розподіл біграм:

01	87296
11	87808
10	87297
00	87143

Автокореляція:

Зсув	Значення
1	174594
2	174594
3	174594
4	174594
5	174594
6	174594
7	174594
8	174594
9	174594
10	174594

Код програми:

```
import time


ran = range(11)


def corelate(polinom, d):
    return sum([(polinom[i] + polinom[(i + d) % len(polinom)]) % 2 for i in
range(len(polinom))])


def count(k, polinom):
    polinom = "".join(map(str, polinom))
    result = {}
    for i in range(len(polinom)):
        tmp = polinom[i:i+k]
        if tmp in result:
            result[tmp] += 1
        else:
            result[tmp] = 1
    return result



def generater(polinom):
    reg = [1]+[0]*(polinom[0] - 1)
    for i in range(2 ** (polinom[0]) - 1):
        if reg[-polinom[0]:] != reg[:polinom[0]] or len(reg) == polinom[0]:
            temp = reg[polinom[1] + i] ^ reg[polinom[2] + i]
            for j in polinom[3:]:
                temp ^= reg[i + j]
            reg.append(temp)
        elif reg[-polinom[0]:] == reg[:polinom[0]]:
            print("Сложный")
            return reg
    print("Примитивный")
    return reg[:-polinom[0]]



start = time.time()
```

```
P1 = [23, 20, 17, 16, 14, 12, 10, 9, 8, 7, 3, 1, 0]


L1 = generater(P1)
print ("L1 = ", len(L1))
result = count(1, L1)
forL1 = dict(zip(ran, [corelate(L1, i) for i in ran]))
print(result)
print(forL1)

P2 = [20, 18, 17, 16, 13, 12, 11, 9, 6, 5, 0]
L2 = generater(P2)
print ("L2 = ", len(L2))
result2 = count(1, L2)
final2 = dict(zip(ran, [corelate(L2, i) for i in ran]))
print(result2)
print(final2)



print(time.time() - start)
```