

Міністерство освіти і науки України

Сумський державний університет

Кафедра

Прикладної математики та моделювання складних систем

Звіт з лабораторної роботи №1

Дисципліна

Криптографія

Студентка:

Пороскун Олена Олегівна

Викладач:

Козлова Ірина Іванівна

Суми, Сумська область

2021

Лабораторна робота №1

Тема: Симетричні крипторетворення

Мета: навчитися застосувати вивчені методи шифрування для зашифровування та розшифровування повідомлень, скласти алгоритм та програму розв'язування наступних задач.

Зміст задачі:

1. Зшифрувати повідомлення «(Своє повне ім'я і прізвище)» методом простої підстановки, алфавіт український або російський. Визначити розмірність простору ключів n_k , $H(k)$, t_6 , l_0 , якщо потужність криптоаналітичної системи $y = 10$ вар/с. Розшифруйте повідомлення і перевірте однозначність процедури за шифрування-розшифрування.

Розв'язування задачі

Розмір простору ключів

$$n_k = N_{\text{кл}} = m! = 33! \approx 8,66 \cdot 10^{36} \approx 2^{122} (\text{де } m - \text{основа алфавіту}).$$

Ентропія джерела ключів

$$H(k) = - \sum P_i \log_2 P_i = N_{\text{кл}} \frac{1}{N} \log_2 N_{\text{кл}} = \log_2 2^{122} = 122$$

Безпечний час

$$t_6 = \frac{N_{\text{кл}}}{\gamma_k} P_k, \text{ де } P_i \text{ ймовірність, з якою має бути здійснений криптоаналіз.}$$

$$t_6 = \frac{8,66 \cdot 10^{36}}{10 \cdot 3,1 \cdot 10^7} = 2,79 \cdot 10^{28} (\text{років})$$

Відстань єдності для шифру

$$l_0 = \frac{H(k)}{r} = \frac{122}{0,5} = 244$$

Програмний код:

```
#include <string.h>
#include <Windows.h>
#include <iostream>

using namespace std;

int main(){

setlocale (LC_CTYPE, "ukr");

int size_mes, size_ABC;
char encrypted_mes[20];
char decrypted_mes[20];

char message[] = "олена_пороскун";
cout << message << " - повідомлення \n\n";

char ABC[] = "абвгдеєжзіїйклмнoprстуфхцчшъюя_";
cout << ABC << " – вхідний алфавіт\n";

char ABC1[] = "ъюя_абвгдеєжзіїйклмнoprстуфхцчш";
cout << ABC1 << " - вихідний алфавіт\n\n";

size_mes = sizeof(message);
size_ABC = sizeof(ABC);

for (int i = 0; i < size_mes; i++) {
for (int j = 0; j < size_ABC; j++) {
if (ABC[j] == message[i]) {
    encrypted_mes[i] = ABC1[j];
}
}
}
cout << encrypted_mes << " - зашифроване повідомлення \n";

for (int i = 0; i < size_mes; i++) {
for (int j = 0; j < size_ABC; j++) {
if (ABC1[j] == encrypted_mes[i]) {
    decrypted_mes[i] = ABC[j];
}
}
}
cout << decrypted_mes << " - розшифроване повідомлення\n";
return 0;
}
```

Прінт-скрін виконання програми:

```
Lab_1_Task1.cpp
1 #include <iostream>
2 #include <Windows.h>
3 #include <iostream>
4
5 using namespace std;
6
7 int main(){
8     setlocale (LC_CTYPE, "ukr");
9
10    int size_mes, size_ABC;
11    char encrypted_mes[20];
12    char decrypted_mes[20];
13
14    char message[] = "олена_поросятун";
15    cout << message << " - повідомлення\n\n";
16
17    char ABC[] = "абвгдеежзиїйклмнопрстуфхцшишъя_";
18    cout << ABC << " - вхідний алфавіт\n";
19
20    char ABC1[] = "ъю_абвгдеежзиїйклмнопрстуфхцшиш";
21    cout << ABC1 << " - вихідний алфавіт\n\n";
22
23    size_mes = sizeof(message);
24    size_ABC = sizeof(ABC);
25
26
27    for (int i = 0; i < size_mes; i++) {
28        for (int j = 0; j < size_ABC; j++) {
29            if (ABC[j] == message[i]) {
30                encrypted_mes[i] = ABC1[j];
31            }
32        }
33    }
34    cout << encrypted_mes << " - зашифроване повідомлення\n";
35
36    for (int i = 0; i < size_mes; i++) {
37        for (int j = 0; j < size_ABC; j++) {
38            if (ABC1[j] == encrypted_mes[i]) {
39                decrypted_mes[i] = ABC[j];
40            }
41        }
42    }
43    cout << decrypted_mes << " - розшифроване повідомлення\n";
44
45}
```

олена_поросятун - повідомлення
абвгдеежзиїйклмнопрстуфхцшишъя_ - вхідний алфавіт
ъю_абвгдеежзиїйклмнопрстуфхцшиш - вихідний алфавіт
кібіньшлкмкнній - зашифроване повідомлення
олена_поросятун - розшифроване повідомлення

Process exited after 0.04302 seconds with return value 0
Для продовження нажмите будь-яку клавішу . . .

олена_поросятун - повідомлення
абвгдеежзиїйклмнопрстуфхцшишъя_ - вхідний алфавіт
ъю_абвгдеежзиїйклмнопрстуфхцшиш - вихідний алфавіт
кібіньшлкмкнній - зашифроване повідомлення
олена_поросятун - розшифроване повідомлення

Process exited after 0.04302 seconds with return value 0
Для продовження нажмите будь-яку клавішу . . .

Зміст задачі:

2. Зашифрувати повідомлення з попереднього завдання, використовуючи шифр Віженера. Визначити розмірність простору ключів n_k , $H(k)$, t_6 , l_0 , якщо потужність крипто аналітичної системи $y = 10$ вар/с. Розшифруйте повідомлення і перевірте однозначність процедури за шифрування-розшифрування.

Ключ «небосхил», ${}^1k = 8$

Розв'язування задачі

Розмір простору ключів

$n_k = N_{\text{кл}} = m^{l_k} = 33^8 \approx 1,4 \cdot 10^{12} \approx 2^{40}$ (де m - основа алфавіту, l_k - розмір ключа (довжина)).

Ентропія джерела ключів

$$H(k) = - \sum_{i=1}^m \frac{1}{m} \log_2 \frac{1}{m} = \log_2 2^{40} = 40$$

Безпечний час

$$t_6 = \frac{N_{\text{кл}}}{\gamma_k} P_k, \text{ при } P_k = 1.$$

$$t_6 = \frac{1,4 \cdot 10^{12}}{10} = 1,4 \cdot 10^{11}(\text{с})$$

Відстань єдності для шифру

$$l_0 = \frac{H(k)}{d} = \frac{40}{0,5} = 80 \text{ бітів або } 16 \text{ пятибітних символів (букв укр. алфавіту)}$$

Програмний код:

```
#include <string.h>
#include <Windows.h>
#include <iostream>

using namespace std;

int main(){

    setlocale (LC_CTYPE, "ukr");

    int size_mes, size_ABC, size_key1;
    char encrypted_mes[20], decrypted_mes[20];

    char message[] = "олена_пороскун";
    cout << message << " - повідомлення\n";

    char key[] = "небосхил";
    cout << "\n" << key << " - ключ \n";

    char ABC[] = "абвгдеїжзїйклмнопрстуфх҃чшшїюя_";
    cout << "\n" << ABC << " - вхідний алфавіт\n\n";

    size_mes = sizeof(message)-1;
    size_ABC = sizeof(ABC)-1;

    // -----
    // символи масиву M1
    int M1[size_mes];
    for (int i = 0; i < size_mes; i++) {
        for (int j = 0; j < size_ABC; j++) {
            if (ABC[j] == message[i]) {
                M1[i] = j;
            }
        }
    }

    for (int i = 0; i < size_mes; i++) {
        printf("%d ", M1[i]);
    }
    printf(" - повідомлення(M1)\n");

    // -----
    // символи масиву ключа Г1
```

```

char key1[] = "небосхилнебосх";
cout << "\n" << key1 << " - ключ з повторюваннями \n\n";
size_key1 = sizeof(key1)-1;

int G1[size_key1];
for (int i = 0; i < size_key1; i++) {
    for (int j = 0; j < size_ABC; j++) {
        if (ABC[j] == key1[i]) {
            G1[i] = j;
        }
    }
}

for (int i = 0; i < size_mes; i++) {
    printf("%d ", G1[i]);
}
printf(" - ключ з повторюваннями(G1) \n\n");

// -----
// символи криптограми C1
int C1[size_key1];
for (int i = 0; i < size_key1; i++) {
    C1[i] = (M1[i] + G1[i]) % size_ABC;
}

for (int i = 0; i < size_mes; i++) {
    printf("%d ", C1[i]);
}
printf(" - зашифроване повідомлення (C1) \n\n");

// зашифроване повідомлення
for (int i = 0; i < size_mes; i++) {
    for (int j = 0; j < size_ABC; j++) {
        if (C1[i] == j) {
            encrypted_mes[i] = ABC[j];
        }
    }
}

for (int i = 0; i < size_mes; i++) {
    printf("%c", encrypted_mes[i]);
}
printf(" - зашифроване повідомлення \n\n\n");

// -----

```

```

// розшифроване повідомлення M2
int M2[size_key1];
for (int i = 0; i < size_key1; i++) {
    M2[i] = (C1[i] - G1[i]) % size_ABC;
    while (M2[i] < 0) {
        M2[i] = M2[i] + size_ABC;
    }
}

for (int i = 0; i < size_mes; i++) {
    printf("%d ", M2[i]);
}
printf(" - розшифроване повідомлення(M2) \n\n");

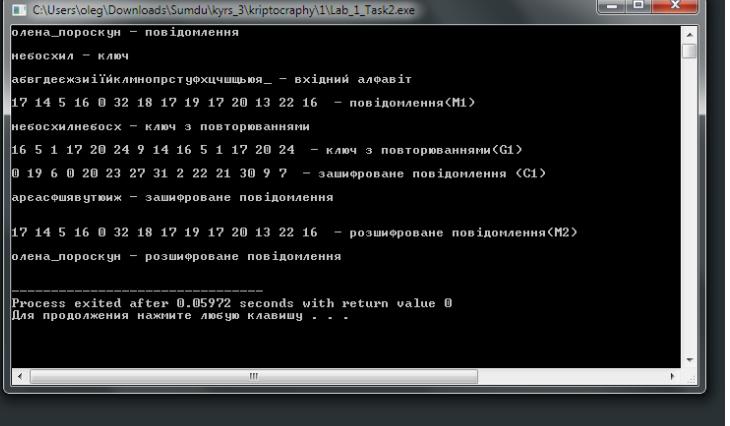
for (int i = 0; i < size_mes; i++) {
    for (int j = 0; j < size_ABC; j++) {
        if (M2[i] == j) {
            decrypted_mes[i] = ABC[j];
        }
    }
}

for (int i = 0; i < size_mes; i++) {
    printf("%c", decrypted_mes[i]);
}
printf(" - розшифроване повідомлення \n\n");

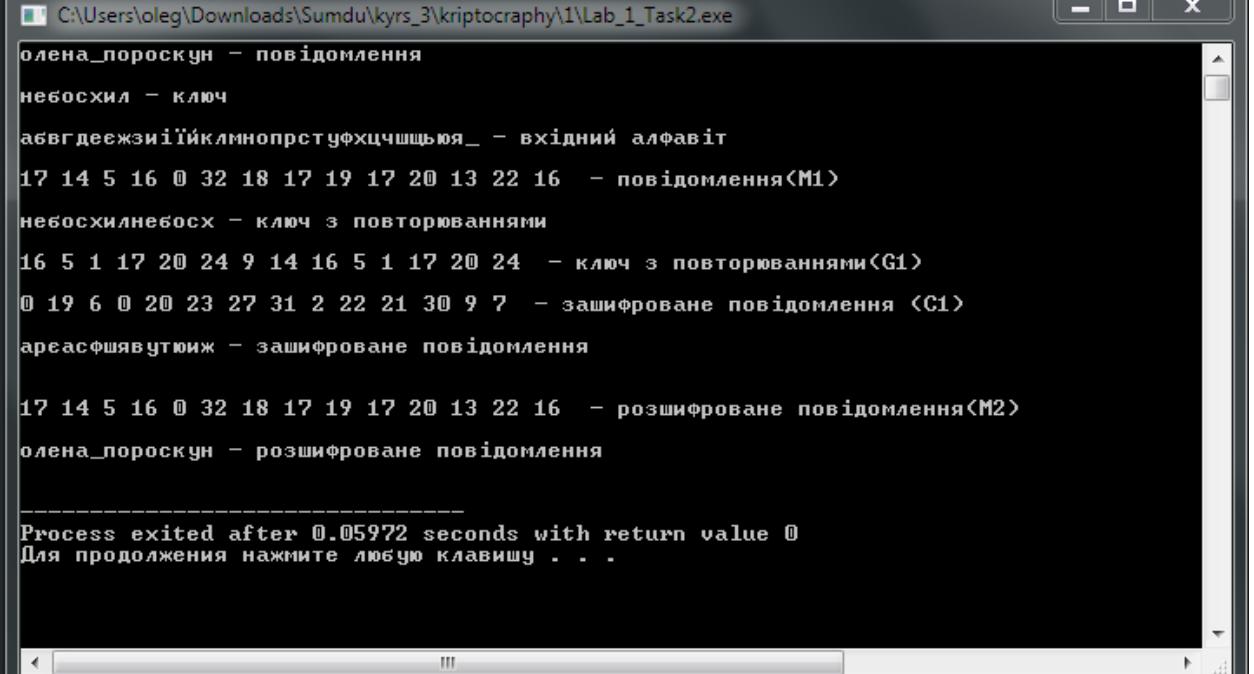
return 0;
}

```

Прінт-скрін програми:



```
Lab_1_Task1.cpp | Lab_1_Task2.cpp
1 #include <string.h>
2 #include <windows.h>
3 #include <iostream>
4
5 using namespace std;
6
7 int main(){
8     setlocale (LC_CTYPE, "ukr");
9
10    int size_mes, size_ABC, size_key1;
11    char encrypted_mes[20], decrypted_mes[20];
12
13    char message[] = "олена_пороскун";
14    cout << message << " - повідомлення\n";
15
16    char key[] = "небосхил";
17    cout << "\n" << key << " - ключ \n";
18
19    char ABC[] = "абвгдеежзиїйклмнопрстуфхцчшъю_";
20    cout << "\n" << ABC << " - вхідний алфавіт\n";
21
22    size_mes = sizeof(message)-1;
23    size_ABC = sizeof(ABC)-1;
24
25    // -----
26
27    // символи масиву M1
28    int M1[size_mes];
29
30    for (int i = 0; i < size_mes; i++) {
31        for (int j = 0; j < size_ABC; j++) {
32            if (ABC[j] == message[i]) {
33                M1[i] = j;
34            }
35        }
36    }
37
38    // -----
39
40    // вхідний алфавіт
41    cout << "абвгдеежзиїйклмнопрстуфхцчшъю_ - вхідний алфавіт
42    олена_пороскун - повідомлення
43    небосхил - ключ
44    небосхилнебосх - ключ з повторюваннями
45
46    17 14 5 16 0 32 18 17 19 17 20 13 22 16 - повідомлення(M1)
47    0 19 6 0 20 23 27 31 2 22 21 30 9 7 - зашифроване повідомлення (C1)
48
49    ареасфяшувтюиж - зашифроване повідомлення
50
51    17 14 5 16 0 32 18 17 19 17 20 13 22 16 - розшифроване повідомлення(M2)
52    олена_пороскун - розшифроване повідомлення
53
54    -----
55
```



```
C:\Users\oleg\Downloads\Sumdu\kyrs_3\kriptography\1\Lab_1_Task2.exe
олена_пороскун - повідомлення
небосхил - ключ
абвгдеежзиїйклмнопрстуфхцчшъю_ - вхідний алфавіт
17 14 5 16 0 32 18 17 19 17 20 13 22 16 - повідомлення(M1)
небосхилнебосх - ключ з повторюваннями
16 5 1 17 20 24 9 14 16 5 1 17 20 24 - ключ з повторюваннями(G1)
0 19 6 0 20 23 27 31 2 22 21 30 9 7 - зашифроване повідомлення (C1)
ареасфяшувтюиж - зашифроване повідомлення

17 14 5 16 0 32 18 17 19 17 20 13 22 16 - розшифроване повідомлення(M2)
олена_пороскун - розшифроване повідомлення

-----
Process exited after 0.05972 seconds with return value 0
Для продовження нажміть будь-яку клавішу . . .
```

Зміст задачі:

3. Зашифруйте повідомлення «2, 7, 9, D, A, 3, 8, D, C, 1, A», яке подано в шістнадцятковій системі числення, потоковим методом, використовуючи ключ

$$\{0110, 1110, 1010, 1000, 1011, 1001, 1111, 0011, 1100, 1000, 1101\}.$$

Знайдіть повну множину ключів і безпечний час такої крипtosистеми, якщо символи ключа з'являються рівномірно і незалежно, $y = 10$ вар/с. Розшифруйте повідомлення.

Розв'язування задачі

Розмір простору ключів

$n_k = N_{\text{кл}} = m^{l_k} = 16^{11} \approx 2^{44} \approx 10^{13,2}$ (де m - основа алфавіту, l_k - розмір ключа (довжина)).

Ентропія мови повідомлення

$$H_0 = \log_2 m = \log_2 16 = \log_2 2^4 = 4 \text{ біти},$$

де $H_0(M)$ - ентропія мови, де всі символи рівномовірні і незалені.

Безпечний час

$$t_6 = \frac{N_{\text{кл}}}{\gamma_k} P_k, \text{ при } P_k = 1.$$

$$t_6 = \frac{10^{13,2}}{10 \cdot 3,10 \cdot 10^7} = 51\,125,5869 \text{ років}$$

Програмний код:

```
#include <string.h>
#include <Windows.h>
#include <iostream>

using namespace std;

int main(){

    setlocale (LC_CTYPE, "ukr");

    int size_message, size_ABC, size_key;
    char encrypted_mes[20], decrypted_mes[20];

    // повідомлення "2, 7, 9, D, A, 3, 8, D, C, 1, A",
    char message[] = "279DA38DC1A";
    cout << message << " - повідомлення\n";
    size_message = sizeof(message) - 1;
    //cout << size_message << " - size_message \n";

    // ключ - {0110, 1110, 1010, 1000, 1011, 1001, 1111, 0011, 1100, 1000, 1101}.
    char key[11][5] = {"0110", "1110", "1010", "1000", "1011", "1001", "1111", "0011", "1100",
    "1000", "1101"} ;
    size_key = sizeof( key ) / sizeof( key[0] );
    //cout << size_key << " - size_key \n";

    for (int i = 0; i < size_key; i++) {
        printf("%s ", key[i]);
    }
    printf(" - ключ(key) \n\n\n");

    char ABC1[16][5] = {"0000", "0001", "0010", "0011", "0100", "0101", "0110", "0111",
        "1000", "1001", "1010", "1011", "1100", "1101", "1110", "1111"};

    char ABC2[] = "0123456789ABCDEF";

    size_ABC = sizeof( ABC1 ) / sizeof( ABC1[0] );
    //cout << size_ABC << " - size_ABC \n";

    for (int i = 0; i < size_ABC; i++) {
        printf("%s ", ABC1[i]);
    }
    printf(" - ABC1 \n\n");

    for (int i = 0; i < size_ABC; i++) {
        printf("%c ", ABC2[i]);
    }
```

```

}

printf(" - ABC2 \n\n");

// ----

// ключ в 16-вій системі
char key1[11];
for (int i = 0; i < size_key; i++){
    for (int j = 0; j < size_ABC; j++){
        if (ABC1[j][0] == key[i][0] && ABC1[j][1] == key[i][1] && ABC1[j][2] ==
key[i][2] && ABC1[j][3] == key[i][3]) {
            key1[i] = ABC2[j];
        }
    }
}

for (int i = 0; i < size_key; i++) {
    printf("%c ", key1[i]);
}
printf(" - ключ в 16-вій системі (key1) \n\n");

printf("_____\n\n");

// ----

// символи масиву M1
int M1[size_message];
for (int i = 0; i < size_message; i++) {
    for (int j = 0; j < size_ABC; j++) {
        if (ABC2[j] == message[i]) {
            M1[i] = j;
        }
    }
}

for (int i = 0; i < size_message; i++) {
    printf("%d ", M1[i]);
}
printf(" - повідомлення(M1)\n\n");

// ----

// символи масиву ключа Г1

```

```

int G1[size_key];
for (int i = 0; i < size_message; i++) {
    for (int j = 0; j < size_ABC; j++) {
        if (ABC2[j] == key1[i]) {
            G1[i] = j;
        }
    }
}

for (int i = 0; i < size_message; i++) {
    printf("%d ", G1[i]);
}
printf(" - ключ(G1) \n");

printf(" _____\n\n");
n");

// -----
// символи криптограми C1
int C1[size_key];
for (int i = 0; i < size_key; i++) {
    C1[i] = (M1[i] + G1[i]) % size_ABC;
}

for (int i = 0; i < size_message; i++) {
    printf("%d ", C1[i]);
}
printf(" - зашифроване повідомлення (C1) \n\n");

// зашифроване повідомлення

for (int i = 0; i < size_message; i++) {
    for (int j = 0; j < size_ABC; j++) {
        if (C1[i] == j) {
            encrypted_mes[i] = ABC2[j];
        }
    }
}

for (int i = 0; i < size_message; i++) {
    printf("%c ", encrypted_mes[i]);
}
printf(" - зашифроване повідомлення в 16-вій системі \n\n");

```

```

printf("_____\n\n");
n");

// ----

// розшифроване повідомлення M2
int M2[size_key];
for (int i = 0; i < size_key; i++) {
    M2[i] = (C1[i] - G1[i]) % size_ABC;
    while (M2[i] < 0){
        M2[i] = M2[i] + size_ABC;
    }
}

for (int i = 0; i < size_message; i++) {
    printf("%d ", M2[i]);
}
printf(" - розшифроване повідомлення(M2) \n\n");

// розшифроване повідомлення

for (int i = 0; i < size_message; i++) {
    for (int j = 0; j < size_ABC; j++) {
        if (M2[i] == j) {
            decrypted_mes[i] = ABC2[j];
        }
    }
}

for (int i = 0; i < size_message; i++) {
    printf("%c ", decrypted_mes[i]);
}
printf(" - розшифроване повідомлення в 16-вій системі \n\n");

// 16 система
/*
0000 0
0001 1
0010 2
0011 3

0100 4
0101 5
0110 6
0111 7

```

```

1000 8
1001 9
1010 A
1011 B

1100 C
1101 D
1110 E
1111 F
*/
return 0;
}

```

Прінт-скрін програми:

```

Lab_1_Task1.cpp Lab_1_Task2.cpp Lab_1_Task3.cpp
1 #include <iostream>
2 #include <Windows.h>
3 #include <iostream>
4
5 using namespace std;
6
7 int main(){
8
9     setlocale (LC_CTYPE, "ukr");
10
11    int size_message, size_ABC, size_key;
12    char encrypted_mes[20], decrypted_mes[20];
13
14    // повідомлення "2, 7, 9, D, A, 3, 8, D, C, 1, A",
15    char message[] = "279DA38DC1A";
16    cout << message << " - повідомлення\n";
17    size_message = sizeof(message) - 1;
18    //cout << size_message << " - size_message \n";
19
20    // ключ - {0110, 1110, 1010, 1000, 1011, 1001
21    char key[11][5] = {"0110", "1110", "1010", "1000", "1011",
22    size_key = sizeof( key ) / sizeof( key[0] );
23    //cout << size_key << " - size_key \n";
24
25    for ( int i = 0; i < size_key; i++ )
26        printf("%s ", key[i]);
27
28    printf(" - ключ(key) \n\n\n");
29
30    char ABC1[16][5] = {"0000", "0001", "0010", "0011", "0100", "0101", "0110", "0111",
31    ...                                "1000", "1001", "1010", "1011", "1100", "1101", "1110", "1111"};
32
33    char ABC2[] = "0123456789ABCDEF";

```

279DA38DC1A - повідомлення
0110 1110 1010 1000 1011 1001 1111 0011 1100 1000 1101 - ключ(key)
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111 - ABC1
0 1 2 3 4 5 6 7 8 9 A B C D E F - ABC2
6 E A 8 B 9 F 3 C 8 D - ключ в 16-вій системі (key1)

2 7 9 13 10 3 8 13 12 1 10 - повідомлення(M1)
6 14 10 8 11 9 15 3 12 8 13 - ключ(G1)

8 5 3 5 5 12 7 0 8 9 7 - зашифроване повідомлення (C1)
8 5 3 5 5 C 7 0 8 9 7 - зашифроване повідомлення в 16-вій системі

2 7 9 13 10 3 8 13 12 1 10 - розшифроване повідомлення(M2)
2 7 9 D A 3 8 D C 1 A - розшифроване повідомлення в 16-вій системі

Process exited after 0.1208 seconds with return value 0
Для продовження нажміть будь-яку клавішу . . .

279DA38DC1A - повідомлення
0110 1110 1010 1000 1011 1001 1111 0011 1100 1000 1101 - ключ(key)
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111 - ABC1
0 1 2 3 4 5 6 7 8 9 A B C D E F - ABC2
6 E A 8 B 9 F 3 C 8 D - ключ в 16-вій системі (key1)

2 7 9 13 10 3 8 13 12 1 10 - повідомлення(M1)
6 14 10 8 11 9 15 3 12 8 13 - ключ(G1)

8 5 3 5 5 12 7 0 8 9 7 - зашифроване повідомлення (C1)
8 5 3 5 5 C 7 0 8 9 7 - зашифроване повідомлення в 16-вій системі

2 7 9 13 10 3 8 13 12 1 10 - розшифроване повідомлення(M2)
2 7 9 D A 3 8 D C 1 A - розшифроване повідомлення в 16-вій системі

Process exited after 0.1208 seconds with return value 0
Для продовження нажміть будь-яку клавішу . . .

Контрольне питання

9. Дати визначення блокового шифру.

Блочними називаються шифри, в яких логічною одиницею шифрування є деякий блок відкритого тексту, після перетворення якого виходить блок шифрованого тексту такої ж довжини. Зазвичай використовуються блоки довжиною 64 біта (128).

Нехай блочний шифр оперує з блоками довжини n . Всього таких блоків 2^{2n} . Оборотних перетворень блоків довжини n у блоки такого ж розміру всього $(2^{2n})!$. Якщо n невелике, то такий шифр еквівалентний шифру підстановки на відповідному алфавіті і нестійкий до частотного аналізу. Чем більше n , тим менш ефективна статистична атака. Виписати таблицю перетворення блоків довжини 64 біта не є можливим.

Блоковий шифр зазвичай складається з простих перетворень над відкритим текстом, що виконуються в певній послідовності деяку кількість разів. Ці перетворення дають змогу усунути або істотно зменшити статистичну складову інформації та залежності відкритого тексту, тобто підвищити його ентропію до такого значення, коли між тим, що існує на вході криптографічного алгоритму, та тим, що отримано на виході, не спостерігається зв'язку. У більшості випадків використані операції та перетворення повинні мати обернені до себе

$$E^{-1}(E(M)) = M.$$

У такому випадку виконання ряду операцій та перетворень, що мають обернені, над відкритим текстом теж матимуть зворотну операцію, що буде набором обернених перетворень, застосованих у зворотному порядку

$$E_3^{-1} \left(E_2^{-1} \left(E_1^{-1} \left(E_3 \left(E_2 \left(E_1(M) \right) \right) \right) \right) \right) = M$$

Висновок

Виконуючи лабораторну роботу, навчилися застосовувати метод простої підстановки, шифр Віженера та потоковий метод в 16-вій системі числення на практиці, складати алгоритм та програму розв'язування цих задач, а також обчислювати розмір простору ключів, ентропію джерела ключів/мови повідомлення, безпечний час, відстань єдності для шифру крипtosистеми.

Додаток

Лабораторна робота № 1

Тема: СИМЕТРИЧНІ КРИПТОПЕРЕТВОРЕННЯ

Загальні відомості

Задача 1.

Зашифрувати повідомлення «Фамилія ім'я» методом простої підстановки, алфавіт російський. Визначити розмірність простору ключів n_k , ентропію $H(k)$, безпечний час t_b та відстань єдності l_b , якщо потужність криptoаналітичної системи $\gamma = 10^7$ вар/с. Розшифрувати повідомлення і пересвідчитися в однозначності процедури зашифрування.

Ключ:

Вх:	а	б	в	г	д	е	ж	з	и	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ
Вих:	б	в	г	д	е	ж	з	и	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ
Вх:	ъ	ы	ь	э	ю	я																			
Вих:	ы	ь	э	ю	я	а																			

Розв'язок задачі:

Використовуючи ключ підстановки, зашифруємо повідомлення «Фамилія ім'я». Зашифроване повідомлення має вигляд:

$C = \text{«хбнкмк_акн_»}$.

Знаходимо розмір простору ключів:

$$N_{k,i} = m! = 32! = 2,6 \cdot 10^{35} \approx 2^{117},$$

де m – основа алфавіту.

Знаходимо ентропію джерела ключів:

$$H(k) = -\sum P_i \log_2 P_i = N_{k,i} \frac{1}{N} \log_2 N_{k,i} = \log_2 2^{117} = 117.$$

Далі визначаємо безпечний час:

$$t_b = \frac{N_{k,i}}{\gamma k} P_K,$$

де P_K – імовірність, з якою має бути здійснений криptoаналіз.

У результаті маємо:

$$t_b = \frac{2,6 \cdot 10^{35}}{10^7 \cdot 3,1 \cdot 10^7} = 10^{21} \text{ років.}$$

Отже шифр може володіти дуже високою стійкістю, але у зв'язку з тим, що природні мови володіють значною надмірністю, пов'язаною з нерівномірністю появи букв у мові і залежністю букв між собою, існує ефективний метод частотного криptoаналізу. Суть аналізу: набирається об'єм не менше за 2000 символів, далі будується гістограма частот появи символів у криптограмі.

Відстань єдності для шифру:

$$l = \frac{H(k)}{r} = \frac{117}{0,5} = 234 \approx 47 \text{ літер.}$$

Далі визначаємо безпечний час:

$$t_0 = \frac{N_{\text{кз}}}{\gamma k} P_k; \quad \text{при } P_k = 1.$$

В результаті маємо

$$t_0 = (3,6 \cdot 10^{16}) / 10^{13} = 3,6 \cdot 10^3 = 58 \text{ хв. 20 с.}$$

$I_0 = \frac{H(K)}{d} = \frac{55}{0,5} = 110$ бітів або 22 п'ятибітних символів (букв російсько-го алфавіту).

Задача 3.

Зашифруйте повідомлення «2,7,9, D, A, 3,8, D, C, 1, A» яке подано в шістнадцятковій системі числення, потоковим методом, використовуючи ключ $K=\{7,9, A, 3,8,4, C, B, 1, 5, 4\}$. Знайдіть повну безліч ключів і безпечний час такої крипто-системи, якщо символи ключа з'являються рівномірно і незалежно.

Пронумеруємо символи повідомлення згідно з таблицею 1.3 ($m=16$):

Таблиця 1.3

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Розв'язок задачі:

$$C_i = (M_i + \Gamma_i) \pmod{16};$$

$$M_i = (C_i - \Gamma_i) \pmod{16}.$$

Записуємо наше повідомлення:

$$M_i = 2, 7, 9, 10, 3, 8, 13, 12, 1, 10.$$

Знаходимо Γ_i , перетворивши ключ у цифрову послідовність

$$\Gamma_i = 7, 9, 10, 3, 8, 4, 12, 11, 1, 5, 4.$$

Визначаємо символи криптограми Сi:

$$C_i = (M_i + \Gamma_i) \pmod{16}.$$

В результаті маємо

$$C_i = 9, 0, 3, 0, 2, 7, 4, 8, 13, 6, 14 = 9, 0, 3, 0, 2, 7, 4, 8, D, 6, E.$$

Знаходимо розмір простору ключів:

$$N_k = m^{L_k} = (16)^{11} = 2^{44} = 10^{13,2}.$$

Знаходимо ентропію мови повідомлення за формулою:

$H_0 = \log m = \log 2^4 = 4$ біти, де $H_0(M)$ – ентропія мови, де всі символи рівно ймовірні і незалежні.

Далі визначаємо безпечний час:

$$t_0 = \frac{N_{\text{кз}}}{\gamma k} P_k; \quad \text{при } P_k = 1,$$

$$t_0 = \frac{10^{10.6}}{10^7 \cdot 3.1 \cdot 10^7} = 5 * 10^{-2} \text{ років.}$$

Задача 4.

Визначити основні показники обчислювально стійких систем, реалізованих з використанням: ГОСТ 28147-89, DES та RIJNDAEL.

Розв'язок задачі:

У криптоалгоритмах, що розглядаються, довжини ключів та число ключів мають такі значення.

$$\text{ГОСТ-28147-89} \quad l_k = 256 \text{ біт} \quad n_k = 2^{256}$$

$$\text{DES} \quad l_k = 56 \text{ біт} \quad n_k = 2^{56}$$

$$\text{RIJNDAEL} \quad l_k = 128 \text{ біт} \quad n_k = 2^{128}$$

Основними показниками, за якими може бути оцінена стійкість криптоалгоритмів, є такі: t_0 (безпечний час), $N_{\text{кг}}$ (кількість ключів), $H(K)$ (ентропія джерела ключів), l_0 (відстань єдиності).

$$N_{\text{кг}}(\Gamma) = 2^{256} \text{ ключів},$$

$$N_{\text{кг}}(\text{D}) = 2^{56} \text{ ключів},$$

$$N_{\text{кг}}(\text{R}) = 2^{128} \text{ ключів}.$$

Вважатимемо, що ключі з'являються рівновірно і незалежно, визначимо ентропію джерела ключів $H(K)$:

$$H(K) = - \sum_{i=1}^{N_k} P(K_i) \log P(K_i) = - \sum_{i=1}^{N_k} \frac{1}{N_{\text{кг}}} \log N_{\text{кг}}^{-1}.$$

$$H(K) = 256 \text{ біт}$$

$$H(K) = 56 \text{ біт}$$

$$H(K) = 128 \text{ біт}$$

Якщо

$$P(K_i) = \frac{1}{N_{\text{кг}}} = N_{\text{кг}}^{-1},$$

тоді

$$t_0^{\text{DES}} = \frac{N_{\text{кг}}^{\text{DES}}}{\gamma \cdot K} \cdot 1 = \frac{2^{56}}{10^9 \text{ операцій/с} \cdot 3 \cdot 10^7} = \frac{10^{16.8}}{3 \cdot 10^{16}} \approx 2.1 \text{ (роки)},$$

де γ – продуктивність криптоаналітичної системи;

K – кількість секунд у році.

Таким чином, для здійснення криптоаналізу методом грубої сили необхідно приблизно 2 роки.

Знайдемо t_6 для випадку диференціального або лінійного криптоаналізу, для яких $N_{\text{вар}} = 2^{47}$. В результаті маємо, що

$$t_6 = \frac{2^{47}}{\gamma \cdot K} = \frac{10^{44.1}}{3 \cdot 10^{16}} = 3 \cdot 10^{-3} \text{ (років)} \approx 1,2 \text{ дні.}$$

Вважатимемо, що в системі передаються повідомлення англійським текстом, надмірність якого 0,45.

Відстань єдності:

$$l_0^{\text{DES}} = \frac{H(K)}{d \cdot \log m} = \frac{H(K)}{d \cdot \log 2} = \frac{56}{0.45 \cdot 1} = 124 \text{ (біт).}$$

Зі значення l_0 робимо висновок, що для того, щоб успішно здійснити криптоаналіз, треба перехопити не менш ніж 124 біт символів криптограми, тоді, затративши t_6 , ми можемо успішно розв'язати задачу криптоаналізу.

Визначте основні параметри стійкості ГОСТ 28147-89 та RIJNDAEL самостійно.

Завдання:

Використовуючи будь-яку мову програмування на ваш вибір, скласти алгоритм та програму розв'язання наступних задач. Бути готовим прокоментувати роботу програми та відповісти на контрольні запитання. Результат виконання лабораторної роботи викласти у звіті, який повинен містити: титульну сторінку, номер та тему лабораторної роботи, мету, зміст задачі, програмний код та прінт-скрін виконання програми, відповідь на контрольне запитання (відповідно номеру в журналі академгрупи), висновок.

1. Зашифрувати повідомлення «(Своє повне ім'я і прізвище)» методом простої підстановки, алфавіт український або російський. Визначити розмірність простору ключів n_k , $H(k)$, t_6 і l_0 , якщо потужність криптоаналітичної системи $u = 10$ вар/с. Розшифруйте повідомлення і перевірте однозначність процедури зашифрування-розшифрування.

2. Зашифрувати повідомлення з попереднього завдання, використовуючи шифр Віжнера. Визначити розмірність простору ключів n_k , $H(k)$, t_6 і l_0 , якщо

потужність криптоаналітичної системи $u = 10$ вар/с. Розшифруйте повідомлення і перевірте однозначність процедури зашифрування-розшифрування.

Ключ: "небосхил", $\mathbf{k} = ^8$.

3. Зашифруйте повідомлення «2,7,9, D, A, 3, 8, D, C, 1, A», яке подано в шістнадцятковій системі числення, потоковим методом, використовуючи ключ $\{0110,1110,1010,1000,1011,1001,1111,0011,1100,1000,1101\}$.

Знайдіть повну множину ключів і безпечний час такої крипtosистеми, якщо символи ключа з'являються рівномовірно і незалежно, $u = 10$ вар/с. Розшифруйте повідомлення.

Контрольні запитання:

1. В чому сутність алгоритму потокового шифрування.
2. Якими властивостями володіють безумовно стійкі криптоалгоритми?
3. Дайте визначення обчислювально стійких систем. Що розуміють під

відстанню єдності для безумовно і обчислюально стійких крипtosистем?

4. Дайте визначення надмірності.
5. У чому відмінність безумовно стійких і обчислюально стійких крип-tosistem?
6. Що розуміють під структурною скритністю?
7. Які основні показники оцінки криптостійкості?
8. Що собою являють афінні перетворення?
9. Дати визначення блокового шифру.
10. В чому сутність складового шифру.
11. В чому сутність шифру монопідстановки.
12. Визначіть кількість ключів, які можуть бути використані в шифрі монопідстановки.
13. Який основний недолік шифру монопідстановки, якщо він використо-vується для зашифтування повідомлення українською мовою.
14. Дайте визначення підстановки та назвіть її властивості.

Міністерство освіти і науки України

Сумський державний університет

Кафедра

Прикладної математики та моделювання складних систем

Звіт з лабораторної роботи № 2

Дисципліна

Криптографія

Студентка:

Пороскун Олена Олегівна

Викладач:

Козлова Ірина Іванівна

Суми, Сумська область

2021

Лабораторна робота №2

Тема: Несиметричні крипторетворення. Алгоритм RSA.

Мета: навчитися будувати пару (E_k, D_k) для RSA крипто алгоритму, знаходити модуль перетворення та значення функції Ейлера.

Зміст задачі:

- Побудувати пару (E_k, D_k) для RSA криптоалгоритму, якщо $P = P_n, Q = Q_n$ (значення P і Q дивись у табл. 2.1)

Таблиця 2.1 – Значення P і Q для задачі 1

n	1	2	3	4	5	6	7	8	9	10
P_n	29	11	11	11	23	7	29	17	19	19
Q_n	11	17	23	13	17	23	7	7	7	11

$n = k \bmod 11$, де k – номер за списком.

Якщо $k = 11$, то $n = (k + 1) \bmod 11$.

Програмний код(C++):

```
#include <string.h>
#include <Windows.h>
#include <iostream>
#include <math.h>

#define MAS_SIZE 50

using namespace std;

int r[MAS_SIZE];
int num[MAS_SIZE];

// НСД(найбільший спільний дільник) двох чисел
int NSD(int a, int b){
    if (b == 0) {
        return a;
    }
    int r = a % b;
    return NSD(b, r);
}

// функція, що допомагає рекурентно знайти значення масиву a, для знаходження y = Dк
int recur_A(int a[], int r[], int mu){
    int ans = 0;
    cout << "\nЗнаходимо рекурентно значення a" << mu - 1 << "\n";
    for (int i = 0; i < mu; i++){
        if (i == 0){
            a[0] = r[0];
        }
        if (i == 1){
            a[i] = r[0]*r[1] + 1;
        }
        if (i > 1){
            a[i] = a[i-1] * r[i] + a[i-2];
        }
        cout << a[i] << " - a" << i << "\n";
    }
    ans = a[mu-1];
    return ans;
}

int main(){
```

```

setlocale (LC_CTYPE, "ukr");

int N, P, Q;
int phi, E, nsd;

cout << "Введіть P: ";
cin >> P;
cout << "Введіть Q: ";
cin >> Q;
cout << "P: " << P << "\n";
cout << "Q: " << Q << "\n";

N = P*Q;
cout << "Модуль перетворення N: " << N << "\n";

phi = (P-1)*(Q-1);
cout << "Функція Ейлера phi(N): " << phi << "\n\n";

int key = 0;
while (key != 1){
    cout << "Виберіть випадково ключ Ek(так щоб він був взаємопростий з ф.
Ейлера): ";
    cin >> E;
    nsd = NSD(phi, E);
    cout << "\nПеревірка НСД (phi(N), Ek): " << nsd << "\n\n";
    key = nsd;
    if (nsd == 1){
        cout << "Діафантове рівняння: ";
        cout << phi << "x + " << E << "y = " << nsd << "\n";
        //printf("%dx + %dy = %d\n", phi, E, nsd);
    }
    else{
        cout << "НСД (phi(N), Ek) не дорівнює 1. \n\n";
    }
}

int phi1 = phi, E1 = E;
for(int i = 0; i < MAS_SIZE; i++){
    r[i] = phi1 / E1;
    num[i] = phi1 % E1;

    phi1 = E1;
    E1 = num[i];

    if (E1 == 1){

```

```

        break;
    }
}

cout << "\n";

int m = 0;
for (int i = 0; i < MAS_SIZE; i++){
    if (r[i] != 0){
        m++;
    }
}

cout << "Подамо a/b = phi(N)/Ek у вигляді ланцювого дробу:";

for (int i = 0; i < m; i++){
    cout << "\n" << r[i] << " - r" << i;
    //cout << "\n" << num[i] << " - num " << i;
}

cout << "\nТаким чином, Mu = " << m << "\n\n";
cout << "Розрахуємо значення y = Dk\n";      // y = (-1)^m * a_(m-1);

int a[m];
int A = recur_A(a, r, m);
cout << "Значення a" << m - 1 << " = " << A;

int y = pow(-1, m)*A;
y = y % phi;
while (y < 0){
    y = y + phi;
}
cout << "\n\n" << "y = Dk = y mod(phi(N)) = " << y << "\n\n";

cout << "Перевірка\n\n";
int multEkDk = E*y;
printf("Вираз (Ek * Dk) mod(phi(N)) має дорівнювати 1. \n", E, y, phi);
printf("Цей вираз дорівнює (%d) mod(%d) = 1. \n\n", multEkDk, phi);
if ((multEkDk % phi) == 1){
    printf("Отже, (Ek, Dk) = (%d, %d) складає RSA ключову пару.\n\n", E, y);
}
else{
    printf("Отже, (Ek, Dk) = (%d, %d) не складає RSA ключову пару.\n\n", E, y);
}

return 0;
}

```

Прінт-скрін виконання програми:

The screenshot shows the code for Lab_2_Task1.cpp on the left and its execution output in a terminal window on the right.

Code (Lab_2_Task1.cpp):

```
1 //include <string.h>
2 #include <Windows.h>
3 #include <iostream>
4 #include <math.h>
5
6 #define MAS_SIZE 50
7
8 using namespace std;
9
10
11 int r[MAS_SIZE];
12 int num[MAS_SIZE];
13
14 // НСД(наайдільший спільний дільник) двох чисел
15 int NSD(int a, int b){
16     if (b == 0) {
17         return a;
18     }
19     int r = a % b;
20     return NSD(b, r);
21 }
22
23 // функція, що допомагає рекурентно знайти значення масиву a, для знаходження у = Dk
24 int recur_A(int a[], int r[], int mu){
25     int ans = 0;
26     cout << "Пізнаходимо рекурентно значення a" << mu - 1 << "\n";
27     for (int i = 0; i < mu; i++){
28         if (i == 0){
29             a[0] = r[0];
30         }
31         if (i == 1){
32             a[i] = r[0]*r[1] + 1;
33         }
34         if (i > 1){
35             a[i] = a[i-1] * r[i] + a[i-2];
36         }
37         cout << a[i] << " - a" << i << "\n";
38     }
39 }
```

Terminal Output:

```
C:\Users\oleg\Downloads\Sumdu\kyrs_3\kriptography\2\Lab_2_Task1.exe
Введіть P: 19
Введіть Q: 7
P: 19
Q: 7
Модуль перетворення N: 133
Функція Ейлера phi(N): 108
Виберіть випадково ключ Ek<так щоб він був взаємопростий з ф. Ейлера>: 5
Перевірка НСД <phi(N), Ek>: 1
Діафантове рівняння: 108x + 5y = 1
Подамо a/b = phi(N)/Ek у вигляді ланцювого дробу:
21 - r0
1 - r1
1 - r2
Таким чином, Mu = 3
Розраховано значення у = Dk
Знаходимо рекурентно значення a2
21 - a0
22 - a1
43 - a2
Значення a2 = 43
у = Dk = у mod<phi(N)> = 65
Перевірка
Вираз <Ek * Dk> mod<phi(N)> має дорівнювати 1.
Цей вираз дорівнює <325> mod<108> = 1.
Отже, <Ek, Dk> = <5, 65> складає RSA ключову пару.

Process exited after 4.246 seconds with return value 0
Для продовження нажмите будь-яку клавішу . . .
```

The screenshot shows the execution output of Lab_2_Task1.exe in a terminal window.

Terminal Output:

```
C:\Users\oleg\Downloads\Sumdu\kyrs_3\kriptography\2\Lab_2_Task1.exe
Введіть P: 19
Введіть Q: 7
P: 19
Q: 7
Модуль перетворення N: 133
Функція Ейлера phi(N): 108
Виберіть випадково ключ Ek<так щоб він був взаємопростий з ф. Ейлера>: 5
Перевірка НСД <phi(N), Ek>: 1
Діафантове рівняння: 108x + 5y = 1
Подамо a/b = phi(N)/Ek у вигляді ланцювого дробу:
21 - r0
1 - r1
1 - r2
Таким чином, Mu = 3
Розрахуємо значення у = Dk
Знаходимо рекурентно значення a2
21 - a0
22 - a1
43 - a2
Значення a2 = 43
у = Dk = у mod<phi(N)> = 65
Перевірка
Вираз <Ek * Dk> mod<phi(N)> має дорівнювати 1.
Цей вираз дорівнює <325> mod<108> = 1.
Отже, <Ek, Dk> = <5, 65> складає RSA ключову пару.

Process exited after 5.698 seconds with return value 0
Для продовження нажмите будь-яку клавішу . . .
```

Зміст задачі:

2. Розв'язати порівняння $E_k \cdot D_k \equiv 1 \pmod{\phi(N)}$, якщо $N = N_n$ (значення N дивись у табл. 2.2)

Таблиця 2.2 – Значення P і Q для задачі 2

n	1	2	3	4	5	6	7	8	9	10
N_n	187	203	297	189	351	209	133	391	143	145

$n = k \pmod{11}$, де k – номер за списком.

Якщо $k = 11$, то $n = (k + 1) \pmod{11}$.

Програмний код(C++):

```
#include <string.h>
#include <Windows.h>
#include <iostream>
#include <math.h>

#define MAS_SIZE 50

using namespace std;

int r[MAS_SIZE];
int num[MAS_SIZE];

// НСД(найбільший спільний дільник) двох чисел
int NSD(int a, int b){
    if (b == 0) {
        return a;
    }
    int r = a % b;
    return NSD(b, r);
}

// функція, що допомагає рекурентно знайти значення масиву a, для знаходження y = Dк
int recur_A(int a[], int r[], int mu){
    int ans = 0;
    cout << "\nЗнаходимо рекурентно значення a" << mu - 1 << "\n";
    for (int i = 0; i < mu; i++){
        ans = r[i] * a[i];
    }
    return ans;
}
```

```
    if (i == 0){  
        a[0] = r[0];  
    }  
  
    if (i == 1){  
        a[i] = r[0]*r[1] + 1;  
    }  
  
    if (i > 1){  
        a[i] = a[i-1] * r[i] + a[i-2];  
    }  
  
    cout << a[i] << " - a" << i << "\n";  
}  
  
ans = a[mu-1];  
  
return ans;  
}
```

```
int main(){  
  
    setlocale (LC_CTYPE, "ukr");  
  
    int N, P, Q;  
  
    int phi, E, nsd;  
  
  
    cout << "Введіть P: ";  
    cin >> P;  
  
    cout << "Введіть Q: ";  
    cin >> Q;  
  
    cout << "P: " << P << "\n";
```

```

cout << "Q: " << Q << "\n";

//N = P*Q;

cout << "Введіть модуль перетворення N: ";

cin >> N;

cout << "N: " << N << "\n";

phi = (P-1)*(Q-1);

cout << "Функція Ейлера phi(N): " << phi << "\n\n";

int key = 0;

while (key != 1){

    cout << "Виберіть випадково ключ Ek(так щоб він був взаємопростий з ф.
Ейлера): ";

    cin >> E;

    nsd = NSD(phi, E);

    cout << "\nПеревірка НСД (phi(N), Ek): " << nsd << "\n\n";

    key = nsd;

    if (nsd == 1){

        cout << "Діафантове рівняння: ";

        cout << phi << "x + " << E << "y = " << nsd << "\n";

        //printf("%dx + %dy = %d\n", phi, E, nsd);

    }

    else{

        cout << "НСД (phi(N), Ek) не дорівнює 1. \n\n";

    }

}

```

```

int phi1 = phi, E1 = E;

for(int i = 0; i < MAS_SIZE; i++){
    r[i] = phi1 / E1;
    num[i] = phi1 % E1;

    phi1 = E1;
    E1 = num[i];

    if (E1 == 1){
        break;
    }

    cout << "\n";
}

int m = 0;

for (int i = 0; i < MAS_SIZE; i++){
    if (r[i] != 0){

        m++;
    }
}

cout << "Подамо a/b = phi(N)/Ek у вигляді ланцюового дробу:";

for (int i = 0; i < m; i++){

    cout << "\n" << r[i] << " - r" << i;
    //cout << "\n" << num[i] << " - num " << i;
}

cout << "\nТаким чином, Mu = " << m << "\n\n";

cout << "Розрахуємо значення y = Dk\n";      // y = (-1)^m * a_(m-1);

```

```

int a[m];
int A = recur_A(a, r, m);
cout << "Значення a" << m - 1 << " = " << A;

int y = pow(-1, m)*A;
y = y % phi;
while (y < 0){
    y = y + phi;
}
cout << "\n\n" << "y = Dk = y mod(phi(N)) = " << y << "\n\n";

cout << "Перевірка\n\n";
int multEkDk = E*y;
printf("Вираз (Ek * Dk) mod(phi(N)) має дорівнювати 1. \n", E, y, phi);
printf("Цей вираз дорівнює (%d) mod(%d) = 1. \n\n", multEkDk, phi);
if ((multEkDk % phi) == 1){
    printf("Отже, (Ek, Dk) = (%d, %d) складає RSA ключову пару.\n\n", E, y);
}
else{
    printf("Отже, (Ek, Dk) = (%d, %d) не складає RSA ключову пару.\n\n", E, y);
}

return 0;
}

```

Прінт-скрін виконання програми:

```
Lab_2_Task1.cpp Lab_2_Task2.cpp
1 #include <string.h>
2 #include <Windows.h>
3 #include <iostream>
4 #include <math.h>
5
6 #define MAS_SIZE 50
7
8 using namespace std;
9
10
11 int r[MAS_SIZE];
12 int num[MAS_SIZE];
13
14 // НСД(найбільший спільний дільник) двох чисел
15 int NSD(int a, int b){
16     if (b == 0) {
17         return a;
18     }
19     int r = a % b;
20     return NSD(b, r);
21 }
22
23 // функція, що допомагає рекурентно знайти значення
24 int recur_A(int a[], int r[], int mu){
25     int ans = 0;
26     cout << "\nЗнаходимо рекурентно значення a" <<
27     for (int i = 0; i < mu; i++){
28         if (i == 0){
29             a[0] = r[0];
30         }
31         if (i == 1){
32             a[i] = r[0]*r[1] + 1;
```

Введіть P: 19
Введіть Q: 7
P: 19
Q: 7
Введіть модуль перетворення N: 143
Функція Ейлера phi(N): 108
Виберіть випадково ключ Ek(так щоб він був взаємопростий з ф. Ейлера): 5
Перевірка НСД(phi(N), Ek): 1
Діамантове рівняння: 108x + 5y = 1
Подамо a/b = phi(N)/Ek у вигляді ланцюгового дробу:
21 - x0
1 - x1
1 - x2
Таким чином, Mu = 3
Розрахуємо значення y = Dk
Знаходимо рекурентно значення a2
21 - a0
22 - a1
43 - a2
Значення a2 = 43
y = Dk = y mod(phi(N)) = 65
Перевірка
Вираз <Ek * Dk> mod(phi(N)) має дорівнювати 1.
Цей вираз дорівнює 325 mod 108 = 1.
Отже, <Ek, Dk> = <5, 65> складає RSA ключову пару.

Process exited after 26.32 seconds with return value 0
Для продовження нажміть будь-яку клавішу . . .

```
C:\Users\oleg\Downloads\Sumdu\kyrs_3\criptography\2\Lab_2_Task2.exe
Введіть P: 19
Введіть Q: 7
P: 19
Q: 7
Введіть модуль перетворення N:
```

```
C:\Users\oleg\Downloads\Sumdu\kyrs_3\criptography\2\Lab_2_Task2.exe
Введіть P: 19
Введіть Q: 7
P: 19
Q: 7
Введіть модуль перетворення N: 143
N: 143
Функція Ейлера phi(N): 108
Виберіть випадково ключ Ek(так щоб він був взаємопростий з ф. Ейлера):
```

```
C:\Users\oleg\Downloads\Sumdu\kyrs_3\kriptography\2\Lab_2_Task2.exe
Введіть P: 19
Введіть Q: 7
P: 19
Q: 7
Введіть модуль перетворення N: 143
N: 143
Функція Ейлера phi(N): 108
Виберіть випадково ключ Ek(так щоб він був взаємопростий з ф. Ейлера): 5
Перевірка НСД (phi(N), Ek): 1
Діафантове рівняння: 108x + 5y = 1
Подамо a/b = phi(N)/Ek у вигляді ланцюгового дробу:
21 - r0
1 - r1
1 - r2
Таким чином, Mi = 3
Розрахуємо значення y = Dk
Знаходимо рекурентно значення a2
21 - a0
22 - a1
43 - a2
Значення a2 = 43
y = Dk = y mod(phi(N)) = 65
Перевірка
Вираз (Ek * Dk) mod(phi(N)) має дорівнювати 1.
Цей вираз дорівнює (325) mod(108) = 1.
Отже, (Ek, Dk) = (5, 65) складає RSA ключову пару.

-----
Process exited after 26.32 seconds with return value 0
Для продовження нажмите будь-яку клавішу . . . ■
```

Контрольне питання

9. Порівняйте складність різних методів RSA крипто перетворень.

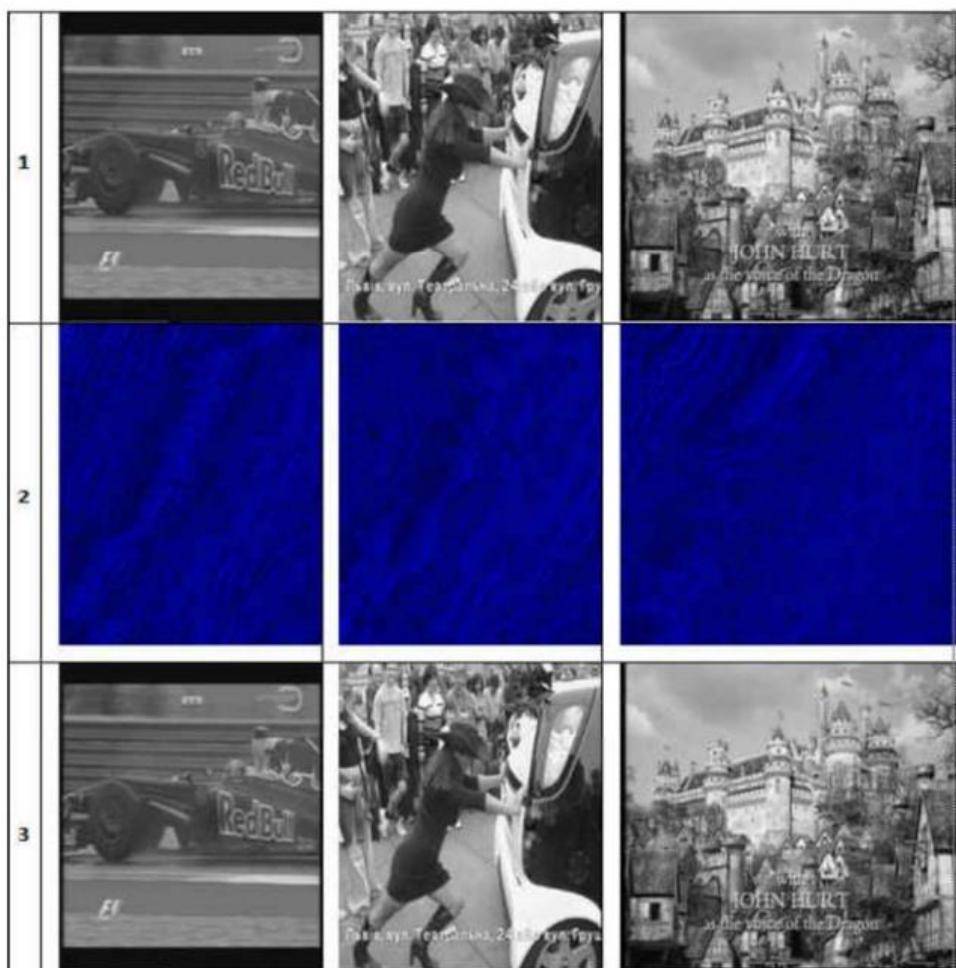
Опис модифікацій алгоритму RSA

Шифрування і дешифрування по одному рядку матриці зображення [1]. Нехай P, Q – пара довільних простих чисел і $N = P \cdot Q$. Шифрування відбувається поелементно з використанням такого перетворення елементів матриці зображення C :

1. Випадково вибирають натуральне число $e < \phi(N)$ і знаходять таке натуральне d , що виконується конгруенція $ed \equiv 1 \pmod{\phi(N)}$.
2. Будують число $A = c_{ij} + Q + P + i + j - d$.
3. Зашифрованим значенням інтенсивності i -го піксела, $i = 1, 2, \dots, m$, m – кількість елементів у рядку, вибирають число $B \equiv A^e \pmod{N}$.

Дешифрування проводять в порядку, протилежному до шифрування після отримання числа $B^d \equiv (A^e)^d \pmod{N}$, виконанням протилежних операцій до змісту пунктів 3), 2), 1). Результати наведено на рис. 1.

Національний лісотехнічний університет України



Rис. 1.

- 1) початкові зображення;
- 2) зашифровані зображення;
- 3) дешифровані зображення

Шифрування і дешифрування по одному рядку матриці з додатковим зашумленням

Нехай P, Q – пара довільних простих чисел і $N = P \cdot Q$. Шифрування відбувається поелементно з використанням такого перетворення елементів матриці зображення C :

1. Випадково вибирають натуральне число $e < \phi(N)$ і знаходять таке натуральне d , що виконується конгруенція $ed \equiv 1 \pmod{\phi(N)}$.
2. Будують число $A = c_{ij} + Q + P + i + j - d$.
3. Зашифрованим значенням інтенсивності i -го піксела, $i = 1, 2, \dots, m$, m – кількість елементів у рядку, вибирають число $C \equiv A^e \pmod{N} + f(i, j)$. Дешифрування проводять в порядку, протилежному до шифрування після отримання числа $(C - f(i, j))^d \equiv (A^e)^d \pmod{N}$, виконанням протилежних операції до змісту пунктів 3), 2), 1).

Результати наведено на рис. 2. Для шифрування вибрали такі функції:

$$f(i, j) = i^2, \quad f(i, j) = i \cdot j, \quad f(i, j) = j^2.$$

Науковий вісник НЛТУ України. – 2012. – Вип. 22.6

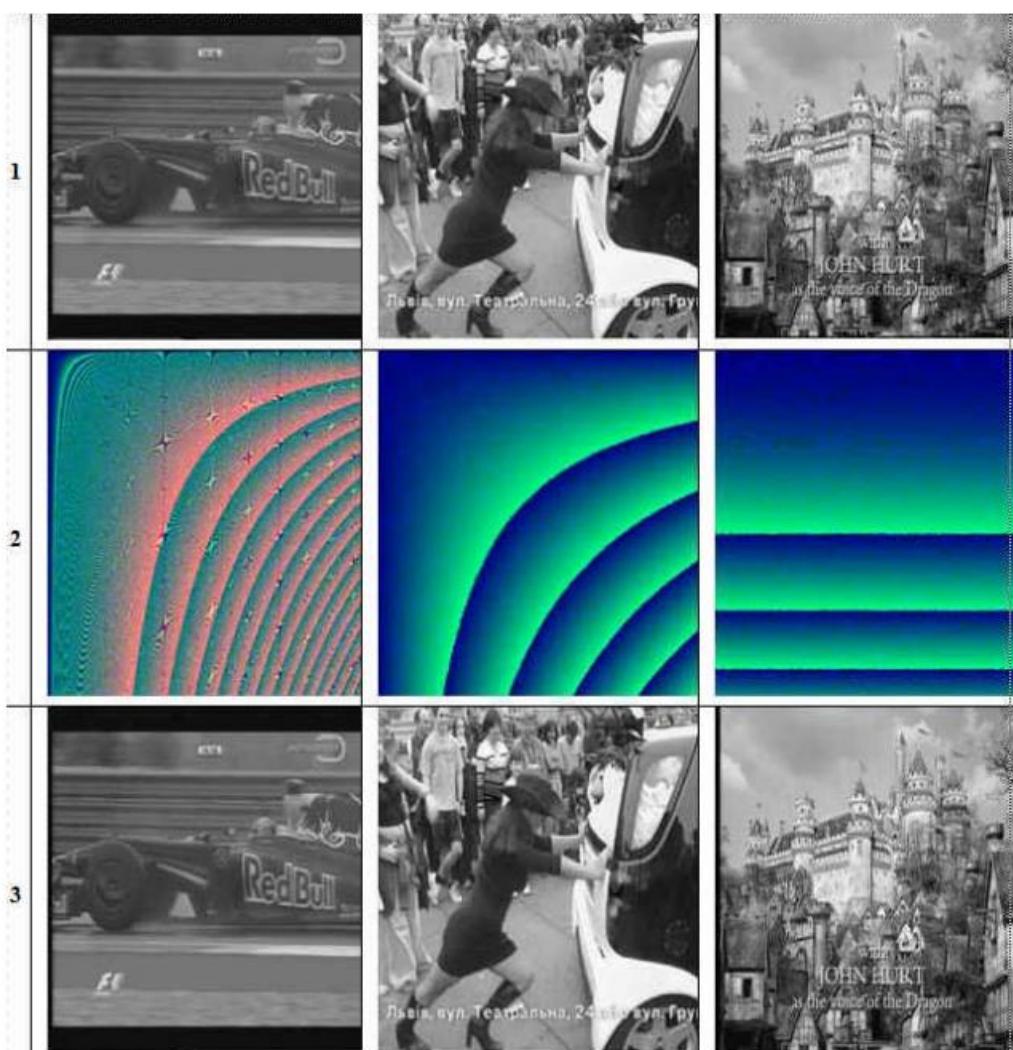


Рис. 2.

- 1) початкові зображення;
- 2) зашифровані зображення;
- 3) дешифровані зображення

З порівняння рис. 1, 2) і рис. 2, 2) видно, що шифрування з додатковим зашумленням відрізняється від шифрування без додаткового зашумлення. Контури в обох зашифрованих зображеннях відсутні. Початкові і дешифровані зображення тільки незначно відрізняються рівнем яскравості. Функції додаткової зашумленості $f(i, j)$ можуть бути довільними цілозначними функціями і додатково, до створюваної алгоритмом RSA зашумленості, підвищують криптографічну стійкість вказаних модифікацій.

Висновки:

1. Запропоновані модифікації шифрування призначені для шифрування зображень в градаціях сірого кольору і ґрунтуються на використанні ідей базового алгоритму RSA. Їх також можна використати і стосовно кольорових зображень.
2. Стійкість до несанкціонованого дешифрування, запропонованими потоковою модифікацією, забезпечується алгоритмом RSA.
3. Модифіковані методи шифрування побудовані так, що за малих значень ключа також можна досягти якісного шифрування, але за умови вірного підбору параметрів шифрування. При цьому досягається висока швидкість роботи алгоритму.

Отже, порівнюючи дві дані модифікації алгоритму RSA, можна сказати про те, що більш складна модифікація – та, у якій використовується шифрування і дешифрування по одному рядку матриці з додатковим зашумленням. Функції додаткової зашумленості підвищують криптографічну стійкість модифікацій.

Висновок

Виконуючи лабораторну роботу, навчилися будувати пару (E_k, D_k) для RSA крипто алгоритму, знаходити модуль перетворення та значення функції Ейлера на практиці, складати програму розв'язування цих задач.

Список використаної літератури

1. Ю. О. Борзов, А. М. Ковальчук, Д. Д. Пелешко Модифікація алгоритму RSA: шифрування та дешифрування за одним рядком матриці зображення // Науковий вісник НЛТУ України . 2012. №6. URL: <https://cyberleninka.ru/article/n/modifikatsiya-algoritmu-rsa-shifruvannya-ta-deshifruvannya-za-odnim-ryadkom-matrtsi-zobrazhennya>

Додаток

Лабораторна робота № 2

Тема: НЕСИМЕТРИЧНІ КРИПТОПЕРЕТВОРЕННЯ: АЛГОРИТМ RSA

Загальні відомості

Задача 1.

Нехай $P=11$, $Q=7$, $E_k=37$. Побудуйте ключову пару (E_k, D_k) для RSA-перетворення.

Розв'язок задачі:

Модуль перетворення має значення

$$N = P \cdot Q = 11 \cdot 7 = 77.$$

Розраховуємо значення функцій Ойлера

$$\varphi(N) = (P-1)(Q-1) = 10 \cdot 6 = 60.$$

Для знаходження D_k ключа розв'яжемо порівняння

$$E_k \cdot D_k = 1 (\text{mod } \varphi(N_j)).$$

Подамо це порівняння у вигляді (1.58)

$$\varphi(N_j) \cdot x + E_k \cdot y = 1.$$

Підставивши значення $\varphi(N_j)$ та E_k , маємо

$$60 \cdot x + 37 \cdot y = 1.$$

Подамо a/b у вигляді ланцюгового дробу

$$\frac{a}{b} = \frac{\varphi(N_j)}{E_k} = \frac{60}{37};$$

$$60/37 = 1 + 23/37; 37/23 = 1 + 14/23; 23/14 = 1 + 9/14; 14/9 = 1 + 5/9; 9/5 = 1 + 4/5;$$

$$5/4 = 1 + 1/4; 4/1 = 4 + 0.$$

Це означає, що $\mu = 6$.

Тоді значення $y = D_k$ можна знайти з виразу

$$y = (-1)^\mu a_{\mu-1} = (-1)^6 a_5 = a_5.$$

Підрахуємо коефіцієнти, a_0, a_1, a_2, a_3, a_4 та a_5 .

$$a_0 = r_0 = 1;$$

$$a_1 = r_0 r_1 + 1 = 1 \cdot 1 + 1 = 2;$$

$$a_2 = a_1 r_2 + a_0 = 2 \cdot 1 + 1 = 3;$$

$$a_3 = a_2 r_3 + a_1 = 3 \cdot 1 + 2 = 5;$$

$$a_4 = a_3 r_4 + a_2 = 5 \cdot 1 + 3 = 8;$$

$$a_5 = a_4 r_5 + a_3 = 8 \cdot 1 + 5 = 13.$$

Визначимо ключ розшифрування

$$y = D_k = (-1)^6 \cdot 13 = 13;$$

$$(E_k, D_k) = (37, 13).$$

Перевірку здійснюємо підстановкою значень E_k та D_k в основне порівняння
 $37 \cdot 13 \equiv 1 \pmod{60}$.

Таким чином ($E_k = 37$ та $D_k = 13$) є ключовою парою RSA-перетворення.

Задача 2.

Нехай $P=11$ і $Q=7$. Побудувати пару (E_k, D_k) для RSA-перетворення, обґрунтuvавши та вибравши один із випадкових ключів.

Розв'язок задачі:

Знаходимо модуль перетворення та значення функції Ойлера $\phi(N)$

Модуль перетворення має значення

$$N = P \cdot Q = 11 \cdot 7 = 77.$$

Розраховуємо значення функції Ойлера

$$\phi(N) = (P-1)(Q-1) = 10 \cdot 6 = 60.$$

Порівняння виду

$$E_k \cdot D_k = 1 \pmod{\phi(N)}$$

запишемо у вигляді Діафантового рівняння

$$\phi(N) \cdot x + E_k \cdot y = 1.$$

Вибравши випадково $E_k = 17$ ключ, взаємопростий з функцією Ойлера, тобто $(E_k, \phi(N)) = 1$, маємо

$$60 \cdot x + 17 \cdot y = 1.$$

Тепер подамо a/b у вигляді ланцюгового дробу:

$$\frac{a}{b} = \frac{\phi(N)}{E_k} = \frac{60}{17};$$

$$60/17 = 3 + 9/17; 17/9 = 1 + 8/9; 9/8 = 1 + 1/8; 8/1 = 8 + 0.$$

Таким чином

$$\mu = 3.$$

Розраховуємо значення $y - D_k$

$$y = (-1)^\mu a_{\mu-1} = (-1)^3 a_2 = -a_2.$$

Знаходимо рекурентно значення a_2 :

$$a_0 = r_0 = 3;$$

$$a_1 = r_0 r_1 + 1 = 3 \cdot 1 + 1 = 4;$$

$$a_2 = a_1 r_2 + a_0 = 4 \cdot 1 + 3 = 7.$$

Далі знаходимо

$$y = D_k = (-7) \bmod 60 = 53.$$

Отже пара

$(E_k, D_k) = (17, 53)$ складає RSA ключову пару.

Перевірка:

Підставивши значення ключів $E_k = 17$ та $D_k = 53$, маємо
 $17 \cdot 53 \equiv 1 \pmod{60}$.

Задача 3.

Нехай $P=11$ і $Q=7$. Виберемо $E_k=9$ як випадковий ключ і побудуємо пару (E_k, D_k) для RSA ключів.

Розв'язок задачі:

Знаходимо модуль RSA перетворення та значення функції Ойлера:

Модуль перетворення має значення

$$N = P \cdot Q = 11 \cdot 7 = 77.$$

Розраховуємо значення функції Ойлера

$$\varphi(N) = (P-1)(Q-1) = 10 \cdot 6 = 60.$$

Ключ D_k знайдемо із порівняння

$$E_k \cdot D_k = 1 \pmod{\varphi(N_j)},$$

далі зведемо вищепередоване порівняння до Діафантового рівняння

$$\varphi(N_j) \cdot x + E_k \cdot y = 1$$

$$60 \cdot x + 9 \cdot y = 1.$$

Знайдемо розклад ланцюгового дробу:

$$\frac{a}{b} = \frac{\varphi(N_j)}{E_k} = \frac{60}{9};$$

$$60/9 = 6 + 6/9; 9/6 = 1 + 3/6; 6/3 = 2 + 0.$$

Оскільки НСД $(E, \varphi(N)) \neq 1$, то розв'язку для пари ключів (E_k, D_k) немає.

2.4 Задачі для самостійного розв'язку

Задача 1. Побудувати пару (E_k, D_k) для RSA криптоалгоритму, якщо $P = P_n, Q = Q_n$ (значення P і Q дивись у табл. 2.1).

Таблиця 2.1 – Значення P і Q для задачі 1

n	1	2	3	4	5	6	7	8	9	10
P_n	29	11	11	11	23	7	29	17	19	19
Q_n	11	17	23	13	17	23	7	7	7	11

$n = k \bmod 11$, де k – номер за списком.

Якщо $k = 11$, то $n = (k + 1) \bmod 11$.

Задача 2. Розв'язати порівняння $E_k \cdot D_k \equiv 1 \pmod{\phi(N)}$, якщо $N = N_n$ (значення N дивись у табл. 2.2).

Таблиця 2.2 – Значення P і Q для задачі 2

n	1	2	3	4	5	6	7	8	9	10
N_n	187	203	297	189	351	209	133	391	143	145

$n = k \bmod 11$, де k – номер за списком.

Якщо $k = 11$, то $n = (k + 1) \bmod 11$.

Завдання:

Використовуючи будь-яку мову програмування на ваш вибір, скласти алгоритм та програму розв'язання задач для самостійного розв'язку. Бути готовим прокоментувати роботу програми та відповісти на контрольні запитання. Результат виконання лабораторної роботи викласти у звіті, який повинен містити: титульну сторінку, номер та тему лабораторної роботи, мету, зміст задачі, програмний код та прінт-скрін виконання програми, відповідь на контрольне запитання (відповідно номеру в журналі академгрупи), висновок.

Контрольні запитання

1. Дайте визначення асиметричного криптоперетворення.
2. Як взаємопов'язані ключі в асиметричній RSA крипtosистемі?
3. Поясніть основні спiввiдношення зашифрування та розшифрування в RSA системi.
4. Якi вимоги висуваються до модуля криптоперетворення в RSA системi?
5. Якi вимоги висуваються до простих чисел, що входять спiвмножниками в модуль преретворення?
6. Якi параметри RSA перетворення є конфiденцiйними, а якi вiдкритими, та чому?
7. Якими властивостями володiють сильнi простi числа?
8. Назвiть основнi методи криптоаналiзу RSA криптоперетворень.
9. Порiвняйте складнiсть riзних методiв RSA криптоперетворень.
10. Який метод RSA криптоаналiзу має найменшу складнiсть?
11. Якi ключi є ключами-блiзнюками? Чи можна iх використовувати?
12. Назвiть основнi переваги та недолiki RSA криптоперетворення.
13. Якi вимоги висуваються до довжин простих чисел модулiв перетворення?

Міністерство освіти і науки України

Сумський державний університет

Кафедра

Прикладної математики та моделювання складних систем

Звіт з лабораторної роботи № 3

Дисципліна

Криптографія

Студентка:

Пороскун Олена Олегівна

Викладач:

Козлова Ірина Іванівна

Суми, Сумська область

2021

Лабораторна робота №3

Тема: Криптографя на еліптичних кривих

Мета: навчитися перевіряти належність точок еліптичній кривій, складати точки еліптичної кривої, знаходити елементи поля GF(p).

Зміст задачі:

1. Нехай є ЕК з рівняння: $y^2 = (x^3 + x + 1) \text{mod } 23, a = 1, b = 1, p = 23$. Перевірити, чи належать точки, що наведені в табл. 3.2, еліптичній кривій.

Таблиця 3.2 – Значення P_n

N	1	2	3	4	5	6	7	8	9	10	11	12
P_n	(3,10)	(3,13)	(4,0)	(5,4)	(5,19)	(6,4)	(6,19)	(7,11)	(7,2)	(9,7)	(9,16)	(17,3)
N	13	14	15	16								
P_n	(17,20)	(18,20)	(19,5)	(13,16)								

Якщо $n - k \text{ (mod } 17)$, k – номер у журналі.

Якщо $k=17$, то $k=k+1$.

Програмний код(C++):

```
#include <iostream>
#include <math.h>

using namespace std;

int main() {

    setlocale (LC_CTYPE, "ukr");

    int Pn[2];

    cout << "Введіть значення Pn: ";
    cin >> Pn[0] >> Pn[1];
    cout << "Значення Pn = (" << Pn[0] << ", " << Pn[1] << ")" << endl;

    int x = Pn[0], y = Pn[1];
    //cout << "Значення Pn: x = " << x << ", y = " << y << endl;

    int a = 1, b = 1, p = 23;

    int f1, f2;
    f1 = pow(y, 2);
    f2 = (pow(x,3) + x + 1);

    f2 = f2 % p;
    while (f2 < 0){
        f2 = f2 + p;
    }

    //  $y^2 = (x^3 + x + 1) \text{ mod } p$  - рівняння еліптичної кривої
    cout << "Підставивши значення Pn у рівняння еліптичної кривої, маємо:" << endl;
    cout << f1 << " = " << f2 << endl;

    if (f1 == f2){
        cout << "Отже, точка Pn = (" << x << ", " << y << ") належить еліптичній кривій.";
    }
    else{
        cout << "Отже, точка Pn = (" << x << ", " << y << ") не належить еліптичній кривій.";
    }

    return 0;
}
```

Прінт-скрін виконання програми:

```
Lab_3_Task1.cpp
1 #include <iostream>
2 #include <math.h>
3
4 using namespace std;
5
6 int main() {
7     setlocale (LC_CTYPE, "ukr");
8
9     int Pn[2];
10
11    cout << "Введіть значення Pn: ";
12    cin >> Pn[0] >> Pn[1];
13    cout << "Значення Pn = (" << Pn[0] << ", " << Pn[1] << ")";
14
15    int x = Pn[0], y = Pn[1];
16    //cout << "Значення Pn: x = " << x << ", y = " << y << endl;
17
18    int a = 1, b = 1, p = 23;
19
20    int f1, f2;
21    f1 = pow(y, 2);
22    f2 = (pow(x,3) + x + 1);
23
24    f2 = f2 % p;
25    while (f2 < 0){
26        f2 = f2 + p;
27    }
28 }
```

D:\Alenka\Sumdu\Cryptography\Practice\3\Lab_3_Task1.exe

Введіть значення Pn: 7 2

Значення Pn = (7, 2)

Підставивши значення Pn у рівняння еліптичної кривої, маємо:

4 = 6

Отже, точка Pn = (7, 2) не належить еліптичній кривій.

Process exited after 15.57 seconds with return value 0

Для продовження нажмите будь-яку клавішу . . .

D:\Alenka\Sumdu\Cryptography\Practice\3\Lab_3_Task1.exe

Введіть значення Pn: ■

D:\Alenka\Sumdu\Cryptography\Practice\3\Lab_3_Task1.exe

Введіть значення Pn: 7 2

Значення Pn = (7, 2)

Підставивши значення Pn у рівняння еліптичної кривої, маємо:

4 = 6

Отже, точка Pn = (7, 2) не належить еліптичній кривій.

Process exited after 15.57 seconds with return value 0

Для продовження нажмите будь-яку клавішу . . .

2. Знайти $2P_n$ (P_n з попередньої задачі).

Програмний код(C++):

```
#include <iostream>
#include <math.h>

using namespace std;

int main() {

    setlocale (LC_CTYPE, "ukr");

    int Pn[2];

    cout << "Введіть значення Pn: ";
    cin >> Pn[0] >> Pn[1];
    cout << "Значення Pn = (" << Pn[0] << ", " << Pn[1] << ")"<< endl;

    int x = Pn[0], y = Pn[1];
    //cout << "Значення Pn: x = " << x << ", y = " << y << endl;

    int a = 1, b = 1, p = 23;

    int lambda, x2, y2;
    // x1 = x2, y1 = y2, 2Pn = Pn + Pn = (x, y) + (x, y) = 2(x, y)
    //lambda = ((y2 - y1) / (x2 - x1)) % p;
    lambda = 0;

    // x2 = (lambda^2 - 2*x)mod p або x3 = (lambda^2 - x1 - x2) mod p
    x2 = (pow(lambda, 2) - 2*x);
    x2 = x2 % p;
    while (x2 < 0){
        x2 = x2 + p;
    }

    //y2 = (lambda*(x - x2) - y) mod p або y3 = (lambda*(x1 - x3) - y1) mod p
    y2 = (lambda*(x - x2) - y) % p;
    while (y2 < 0){
        y2 = y2 + p;
    }

    cout << "Отже, 2Pn = Pn + Pn = (" << x2 << ", " << y2 << ")"<< endl;

    return 0;
}
```

Прінт-скрін виконання програми:

```
Lab_3_Task1.cpp Lab_3_Task2.cpp
1 #include <iostream>
2 #include <math.h>
3
4 using namespace std;
5
6 int main() {
7
8     setlocale (LC_CTYPE, "ukr");
9
10    int Pn[2];
11
12    cout << "Введіть значення Pn: ";
13    cin >> Pn[0] >> Pn[1];
14    cout << "Значення Pn = (" << Pn[0] << ", "
15
16    int x = Pn[0], y = Pn[1];
17    //cout << "Значення Pn: x = " << x << ", y =
18
19    int a = 1, b = 1, p = 23;
20
21    int lambda, x2, y2;
22    // x1 = x2, y1 = y2, 2Pn = Pn + Pn = (x, y) - (x, y) = λ(x, y)
23    //Lambda = ((y2 - y1) / (x2 - x1)) % p;
24    lambda = 0;
```

D:\Alenka\Sumdu\Cryptography\Practice\3\Lab_3_Task2.exe

Введіть значення Pn: 7 2
Значення Pn = (7, 2)
Отже, 2Pn = Pn + Pn = (9, 21)

Process exited after 20.69 seconds with return value 0
Для продовження нажмите будь-яку клавішу . . .

D:\Alenka\Sumdu\Cryptography\Practice\3\Lab_3_Task2.exe

Введіть значення Pn:

D:\Alenka\Sumdu\Cryptography\Practice\3\Lab_3_Task2.exe

Введіть значення Pn: 7 2
Значення Pn = (7, 2)
Отже, 2Pn = Pn + Pn = (9, 21)

Process exited after 20.69 seconds with return value 0
Для продовження нажмите будь-яку клавішу . . .

3. Знайти $P_n + Q$, якщо $Q = 2P_n$ (P_n з попередньої задачі).

Програмний код(C++):

```
#include <iostream>
#include <math.h>

using namespace std;

// перетворення від'ємної остачі
int absRemain(int rem, int d){
    while(rem < 0){
        rem = rem + d;
    }
    return rem;
}

int main() {
    setlocale (LC_CTYPE, "ukr");

    int Pn[2];

    cout << "Введіть значення Pn: ";
    cin >> Pn[0] >> Pn[1];
    cout << "Значення Pn = (" << Pn[0] << ", " << Pn[1] << ")" << endl;

    int x1 = Pn[0], y1 = Pn[1];

    int a = 1, b = 1, p = 23;

    int lambda1, x2, y2;
    // x1 = x2 = x, y1 = y2 = y, 2Pn = Pn + Pn = (x, y) + (x, y) = 2(x, y)
    // lambda = ((y2 - y1) / (x2 - x1)) % p;
    lambda1 = 0;

    // x2 = (lambda^2 - 2*x)mod p або x3 = (lambda^2 - x1 - x2) mod p
    x2 = (pow(lambda1, 2) - 2*x1);
    x2 = x2 % p;
    x2 = absRemain(x2, p);

    // y2 = (lambda*(x - x2) - y) mod p або y3 = (lambda*(x1 - x3) - y1) mod p
    y2 = (lambda1*(x1 - x2) - y1) % p;
    y2 = absRemain(y2, p);

    cout << "Q = 2Pn = Pn + Pn = (" << x2 << ", " << y2 << ")" << endl << endl;
```

```

// Pn + Q = (x3, y3)
// lambda = ((y2 - y1) / (x2 - x1)) % p;

//cout << "x1 = " << x1 << ", y1 = " << y1 << endl;
//cout << "x2 = " << x2 << ", y2 = " << y2 << endl;

int lambda2, x3, y3;

lambda2 = ((y2 - y1) / (x2 - x1)); // % p;
//cout << "lambda2 = " << lambda2 << endl

int rem = ((y2 - y1) % (x2 - x1));
//cout << "rem = " << rem << endl;

int Z = 0;
if (rem != 0) {

    cout << "Знаходимо в полі G(" << p << ") обернений елемент Z, розв'язавши
порівняння:" << endl;
    cout << abs(x2 - x1) << " * Z" << " = 1 mod " << p << endl;

    int key = (abs(x2 - x1) * Z);
    key = key % p;
    while (key != 1) {
        Z++;
        key = (abs((x2 - x1)) * Z) % p;
    }

    cout << "Це порівняння має розв'язок при Z = " << Z << ", тому " << endl;
    lambda2 = (abs(y2 - y1) * Z) % p;
}

else {
    lambda2 = lambda2 % p;
}

lambda2 = absRemain(lambda2, p);
cout << "lambda = " << lambda2 << endl << endl;

x3 = (pow(lambda2, 2) - x1 - x2);
x3 = x3 % p;
x3 = absRemain(x3, p);

y3 = (lambda2*(x1 - x3) - y1) % p;
y3 = absRemain(y3, p);

```

```
cout << "x3 = " << x3 << endl;
cout << "y3 = " << y3 << endl << endl;

cout << "Отже, Pn + Q = Pn + 2Pn = (" ;
cout << x1 << ", " << y1 << ") + (" << x2 << ", " << y2 << ") = ";
cout << "P3 = (" << x3 << ", " << y3 << ")" << endl;

return 0;
}
```

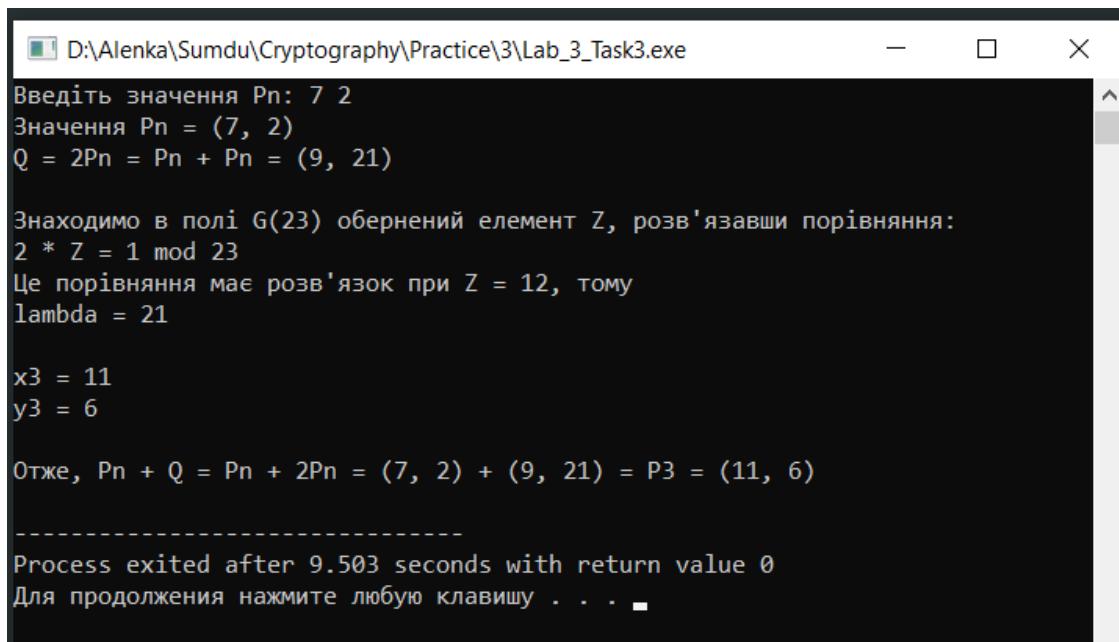
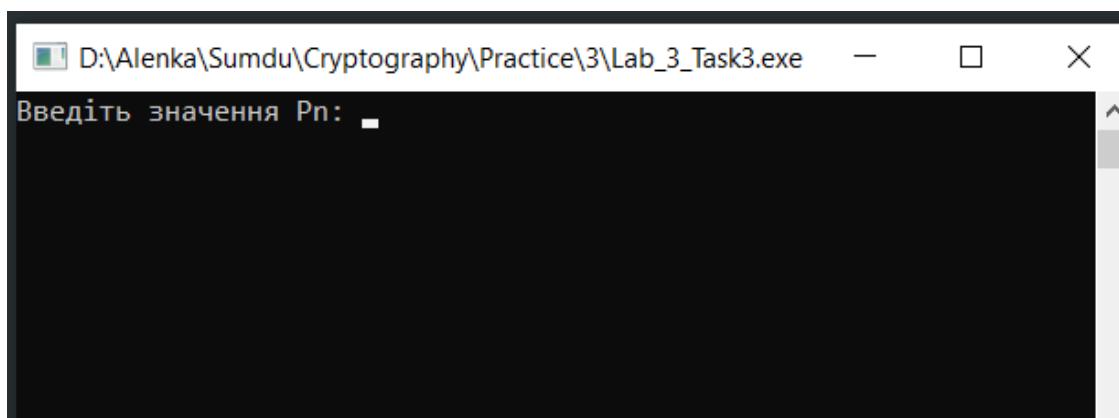
Прінт-скрін виконання програми:

```
Lab_3_Task1.cpp Lab_3_Task2.cpp Lab_3_Task3.cpp
1 #include <iostream>
2 #include <math.h>
3
4 using namespace std;
5
6 // переворення від'ємної остачі
7 int absRemain(int rem, int d){
8     while(rem < 0){
9         rem = rem + d;
10    }
11    return rem;
12 }
13
14
15 int main() {
16     setlocale (LC_CTYPE, "ukr");
17
18     int Pn[2];
19
20     cout << "Введіть значення Pn: ";
21     cin >> Pn[0] >> Pn[1];
22     cout << "Значення Pn = (" << Pn[0] << ", "
23
24     int x1 = Pn[0], y1 = Pn[1];
25
26 }
```

D:\Alenka\Sumdu\Cryptography\Practice\3\Lab_3_Task3.exe

Введіть значення Pn: 7 2
Значення Pn = (7, 2)
 $Q = 2Pn = Pn + Pn = (9, 21)$
Знаходимо в полі G(23) обернений елемент Z, розв'язавши порівняння:
 $2 * Z = 1 \text{ mod } 23$
Це порівняння має розв'язок при $Z = 12$, тому
 $\lambda = 21$
 $x3 = 11$
 $y3 = 6$
Отже, $Pn + Q = Pn + 2Pn = (7, 2) + (9, 21) = P3 = (11, 6)$

Process exited after 9.503 seconds with return value 0
Для продовження нажмите будь-яку клавишу . . .



Контрольне питання

9. Запишіть елементи поля GF(7).

$$GF(p) = \{0, 1, \dots, p-1\}$$

$$GF(7) = \{0, 1, 2, 3, 4, 5, 6\}$$

Висновок

Виконуючи лабораторну роботу, навчилися перевіряти належність точок еліптичній кривій, складати точки еліптичної кривої та складати програму розв'язування цих задач, а також знаходити елементи поля GF(p).

Додаток

Лабораторна робота № 3

Тема: КРИПТОГРАФІЯ НА ЕЛІПТИЧНИХ КРИВИХ

Загальні відомості

Задача 1.

Еліптична крива має вигляд $y^2 = (x^3 + x + 1) \pmod{23}$, причому $a = 1$, $b = 1$, $P = 23$.

1. Перевірити, що точки $P_1 = (3, 10)$, $P_2 = (9, 7)$ належать кривій та знайти $P_1 + P_2 = P_3$.

Підставивши значення P_1 та P_2 у рівняння еліптичної кривої бачимо, що вони належать кривій (ліва та права частини порівнюються).

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \pmod{P} = \frac{7 - 10}{9 - 3} \pmod{23} = -\frac{3}{6} \pmod{23} = -\frac{1}{2} \pmod{23} = \frac{22}{2} \pmod{23} = 11$$

Знаходимо

$$x_3 = (121 - 3 - 9)m23 = 109(m23) = 17$$

$$y_3 = (11(3 - 17) - 10)m23 = (-11 \cdot 14 - 10)m23 = -164(m23) = 20$$

Отже $P_1 + P_2 = (3, 10) + (9, 7) = (17, 20)$.

Задача 2.

Скласти точки P_1 і P_2 . $P_1 = (12, 19)$, $P_2 = (5, 4)$. Якщо еліптична крива має вигляд $y^2 = x^3 + x + 1 \pmod{23}$, тобто $a = 1$, $b = 1$, $P = 23$.

Розв'язок задачі:

$$\lambda = \frac{4 - 19}{5 - 12} = \frac{-15}{-7} \pmod{23} = \frac{15}{7} \pmod{23}.$$

Знаходимо в полі $G(23)$ обернений елемент Z , розв'язавши порівняння $7 \cdot Z \equiv 1 \pmod{23}$.

Це порівняння має розв'язок при $Z = 10$, тому

$$\lambda = 15 \cdot 10 \pmod{23} = 12.$$

$$x_3 = 12^2 - 12 - 5 = 144 - 12 - 5 \pmod{23} = 127 \pmod{23} = 12 \pmod{23}.$$

$$y_3 = 12 \cdot (12 - 12) - 19 \pmod{23} = 4 \pmod{23}.$$

Таким чином:

$$P_1 + P_2 = (x_1, y_1) + (x_2, y_2) = P_3 = (x_3, y_3) = (12, 4).$$

$$P_3 = (12, 4).$$

Задача 3.

Знайти точку, яка дорівнює $3 \cdot P_1$. $P_1 = (5, 4)$. Еліптична крива має вигляд

$$y^2 = x^3 + x + 1 \pmod{23}$$
, тобто $a=1$, $b=1$.

Розв'язок задачі:

Спочатку подвоїмо точку P_1 , тобто знайдемо $P_2 = 2 \cdot P_1$.

Знайдемо λ , використовуючи (3.13)

$$\lambda = \frac{3 \cdot x_1^2 + a}{2 \cdot y_1} \pmod{23} = \frac{3 \cdot 25 + 1}{2 \cdot 4} \pmod{23} = \frac{76}{2 \cdot 4} \pmod{23} = \frac{19}{2} \pmod{23}.$$

Знайдемо зворотний елемент

$$2 \cdot Z \equiv 1 \pmod{23}; Z = 12.$$

$$\text{Отже } \lambda = 19 \cdot 12 \pmod{23} = 21.$$

Використовуючи (3.10) – (3.11) та враховуючи, що $x_1=x_2$, знайдемо координати x_2 і y_2 точки $P_2(x_2, y_2)$

$$x_2 = (21^2 - 2 \cdot 5) \pmod{23} = (441 - 10) \pmod{23} = 431 \pmod{23} = 17;$$

$$y_2 = (21(5 - 17) - 4) \pmod{23} = -256 \pmod{23} = 20.$$

$$\text{Отже: } P_2 = 2 \cdot P_1 = 2 \cdot (5, 4) = (17, 20).$$

Знайдемо λ , використовуючи (1.87)

$$\lambda = \frac{20 - 4}{17 - 5} = \frac{16}{12} \pmod{23} = \frac{4}{3} \pmod{23}.$$

Знайдемо зворотний елемент

$$3 \cdot Z \equiv 1 \pmod{23}; Z = 8.$$

$$\text{Отже } \lambda = 4 \cdot 8 \pmod{23} = 32 \pmod{23} = 9.$$

Використовуючи (3.10) і (3.11), знайдемо x_3 та y_3

$$x_3 = (9^2 - 5 - 17) \pmod{23} = (81 - 22) \pmod{23} = 13;$$

$$y_3 = (9(5 - 13) - 4) \pmod{23} = (9 \cdot (-8) - 4) \pmod{23} = -76 \pmod{23} = 16.$$

$$\text{Отже: } 3P_1 = 3(5, 4) = (13, 16).$$

Задача 4.

Нехай є ЕК з рівнянням: $y^2 = (x^3 + x + 1) \pmod{23}$, $a=1$, $b=1$, $p=23$.

Перевірити, чи належать такі точки ЕК:

$$(1, 7), (1, 16).$$

Розв'язок задачі:

$$7^2 \equiv (1+1+1) \pmod{23}, \text{ точка } (1,7) \text{ належить ЕК};$$

$$16^2 \equiv (1+1+1) \pmod{23}, \text{ точка } (1,16) \text{ належить ЕК}.$$

Задача 5.

Знайти порядок базової точки $(17,20)$. Якщо рівняння еліптичної кривої має вигляд $y^2 = x^3 + x + 1 \pmod{23}$.

Розв'язок задачі:

Для визначення порядку точки $(17,20)$ на еліптичній кривій

$$y^2 = x^3 + x + 1 \pmod{23}$$

знайдемо таке значення d , щоб $d^*(17,20) \pmod{23} = 0$.

Для цього підставимо в останнє порівняння $d = 1, 2, 3, \dots, k$, та визначимо значення скалярного добутку.

$$\lambda = \frac{3 \cdot 289 + 1}{40} \pmod{23} = \frac{868}{40} \pmod{23} = 1;$$

$$x_3 = (\lambda^2 - 2x_1) \pmod{p} = (1 - 34) \pmod{23} = 13;$$

$$y_3 = (\lambda(x_1 - x_3) - y) \pmod{p} = (1 \cdot 4 - 20) \pmod{23} = 7.$$

При $d = 2$ маємо

$$Q_1 = (13, 7).$$

При $d = 3$

$$P_1 + Q_1 = Q_2;$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \pmod{p} = \frac{7 - 20}{13 - 17} \pmod{23} = \frac{-13}{-4} \pmod{23} = 9;$$

$$x_3 = (\lambda^2 - x_1 - x_2) \pmod{p} = (81 - 17 - 13) \pmod{23} = 5;$$

$$y_3 = (\lambda(x_1 - x_3) - y_1) \pmod{p} = (9 \cdot 2 - 20) \pmod{23} = 19;$$

$$Q_2 = (5, 19).$$

При $d = 4$ маємо:

$$P_1 + Q_2 = Q_3;$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \pmod{p} = \frac{19 - 20}{5 - 17} \pmod{23} = \frac{-1}{-12} \pmod{23} = 2;$$

$$x_3 = (\lambda^2 - x_1 - x_2) \pmod{p} = (4 - 17 - 5) \pmod{23} = 5;$$

$$y_3 = (\lambda(x_1 - x_3) - y_1) \pmod{p} = (2 \cdot 12 - 20) \pmod{23} = 4;$$

$$Q_3 = (5, 4).$$

При $d = 5$

$$P_1 + Q_3 = Q_4$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \pmod{p} = \frac{4 - 20}{5 - 17} \pmod{23} = \frac{-16}{-12} \pmod{23} = 9;$$

$$x_3 = (\lambda^2 - x_1 - x_2) \pmod{p} = (81 - 17 - 5) \pmod{23} = 13;$$

$$y_3 = (\lambda(x_1 - x_3) - y_1) \pmod{p} = (9 \cdot 4 - 20) \pmod{23} = 16;$$

$$Q_4 = (13, 16).$$

При $d = 6$

$$P_I + Q_4 - Q_5$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \pmod{p} = \frac{16 - 20}{13 - 17} \pmod{23} = \frac{-4}{-4} \pmod{23} = 1;$$

$$x_3 = (\lambda^2 - x_1 - x_2) \pmod{p} = (1 - 17 - 13) \pmod{23} = 17;$$

$$y_3 = (\lambda(x_1 - x_3) - y_1) \pmod{p} = (1 \cdot 0 - 20) \pmod{23} = 3;$$

$$Q_5 = (17, 3).$$

При $d = 7$

$$P_I + Q_5 - Q_6$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \pmod{p} = \frac{3 - 20}{17 - 17} \pmod{23} = \infty;$$

$$Q_6 = (0, 0).$$

Таким чином порядок p точки $G = (17, 25)$ на еліптичній кривій дорівнює 7.

Задача 6.

Побудувати $GF(2^m)$, $m=4$, $\alpha = x$, $f(x) = x^4 + x + 1$.

Розв'язок задачі:

Поле $GF(2^4)$ – може бути задано 16 поліномами не вище третього ступеня, наприклад, поліномами над полем $GF(2)$.

$$\begin{aligned}0, x, x^2, x^3, x+1, x^2+1, x^3+1, x^2+x, x^2+x+1, \\x^3+x, x^3+x+1, x^3+x^2, x^3+x^2+x, \\x^3+x^2+x+1, 1\end{aligned}$$

Елементи поля можуть бути задані у двійковому вигляді, тоді, наприклад,
 $x^3+x^2+1 \Rightarrow 1101$
 $x^3+1 \Rightarrow 1001$.

Операція множення елементів поля виконується в полі $GF(P)$.

Наприклад:

$$\begin{aligned}(1101) \cdot (1001) \Leftrightarrow (x^3 + x^2 + 1) \cdot (x^3 + 1) \pmod{f(x), 2} = \\= (x^6 + x^5 + x^2 + 1) \pmod{f(x), 2}.\end{aligned}$$

Зведемо цей поліном за модулем $f(x) = x^4 +$

$$\begin{array}{r} x^6 + x^5 + x^2 + 1 \\ x^6 + x^3 + x^2 \\ \hline x^5 + x^3 + 1 \\ x^5 + x^2 \\ \hline x^3 + x^2 + x + 1 \end{array}$$

Отже: $(x^3 + x^2 + 1)(x^3 + 1) \pmod{(x^4 + x + 1), 2} = x^3 + x^2 + x + 1 \Rightarrow 1111$.

В табл. 3.1 наведено елементи α^j для $j \in \overline{0, 7}$, якщо $\alpha = x$.

Таблиця 3.1 – Значення α^j

j	0	1	2	3	4	5	6	7
α^j	1	x	x^2	x^3	$x+1$	x^2+x	x^3+x^2	x^3+x^2+x+1

Елементи поля можна отримати як $\alpha^j = \alpha^j \pmod{f(x), 2}, j \in \overline{0, n}$.

Задача 7.

Нехай задано поле $GF(2^m)$, $m=4$, $\alpha = x$, $f(x) = x^4 + x + 1$.

Знайти $P_3 = 2P_1 + P_2$, якщо $P_1 = (0111, 0010)$, $P_2 = (1100, 1001)$.

Розв'язок задачі:

Знайдемо $2P_1$:

$$\lambda = 0111 + \frac{0010}{0111}.$$

Далі знайдемо зворотний елемент для числа 0111, яке подамо поліномом $x^2 + x + 1$.

Враховуючи, що:

$$g^{p-2} = g^{-1};$$

$$(x^2 + x + 1)^{2^4 - 2} = (x^2 + x + 1)^{14} = ((x^2 + x + 1)^2)^2 \cdot ((x^2 + x + 1)^2)^2 \cdot ((x^2 + x + 1)^2)^2 \cdot (x^2 + x + 1)^2;$$

$$(x^2 + x + 1)^2 = x^4 + 2x^3 + 3x^2 + 2x + 1 \pmod{f(x), 2} = x^2 + x;$$

$$(x^2 + x)^2 = x^4 + 2x^3 + x^2 \pmod{f(x), 2} = x^2 + x + 1;$$

$$\begin{aligned} (x^2 + x + 1)^{14} &= (x^2 + x + 1)(x^2 + x + 1)(x^2 + x + 1)(x^2 + x) = (x^2 + x)(x^2 + x)(x^2 + x + 1) = \\ &= (x^2 + x + 1)(x^2 + x + 1) = x^2 + x; \end{aligned}$$

$$\lambda = x^2 + x + 1 + x(x^2 + x) = x^2 + x + 1 + x^3 + x^2 \pmod{f(x), 2} = x^3 + x + 1;$$

$$x_3 = (x^3 + x + 1)^2 + x^3 + x + 1 + x + 1 = x^3 + 1 + x^3 + 2x + 2 \pmod{f(x), 2} = 1;$$

$$(x^3 + x + 1)^2 = x^6 + 2x^4 + 2x^3 + x^2 + 2x + 1 \pmod{f(x), 2} = x^3 + 1;$$

$$y_3 = (x^3 + x + 1)^2 + (x^3 + x + 1 + 1) \cdot 1 = x^2 + x + x^3 + x \pmod{f(x), 2} = x^3 + x^2;$$

Отже

$$2P_1 = (1, x^3 + x^2).$$

Знайдемо $2P_1 + P_2$:

Завдання:

Використовуючи будь-яку мову програмування на ваш вибір, скласти алгоритм та програму розв'язання задач для самостійного розв'язку. Бути готовим прокоментувати роботу програми та відповісти на контрольні запитання. Результат виконання лабораторної роботи викласти у звіті, який повинен містити: титульну сторінку, номер та тему лабораторної роботи, мету, зміст задачі, програмний код та принт-скрін виконання програми, відповідь на контрольне запитання (відповідно номеру в журналі академгрупи), висновок.

Контрольні запитання

1. Дати визначення базової точки ЕК.
2. Запишіть та поясніть формулу складання точок еліптичної кривої над простим полем.
3. Запишіть та поясніть формулу подвоєння точок еліптичної кривої над простим полем.
4. Запишіть та поясніть формулу складання точок еліптичної кривої над розширеним полем.
5. Запишіть та поясніть формулу подвоєння точок еліптичної кривої над розширеним полем.
6. Яким вимогам відповідає примітивний поліном $f(x)$?
7. Скільки елементів у полі $GF(2^m)$, якщо $m = 4, 9, 81, 160$?
8. Скільки елементів у полі $GF(P)$, якщо $P = 17, 31, 47, 89, 107, 257$?
9. Запишіть елементи поля $GF(7)$.
10. Запишіть елементи поля $GF(2^3)$.
11. Запишіть та поясніть рівняння еліптичної кривої над простим полем $GF(P)$.
12. Запишіть та поясніть рівняння еліптичної кривої над розширеним полем $GF(2^m)$.
13. Запишіть та поясніть рівняння еліптичної кривої в проективному вигляді.
14. Що забезпечує використання проективного базису?

Міністерство освіти і науки України

Сумський державний університет

Кафедра

Прикладної математики та моделювання складних систем

Звіт з лабораторної роботи № 4

Дисципліна

Криптографія

Студентка:

Пороскун Олена Олегівна

Викладач:

Козлова Ірина Іванівна

Суми, Сумська область

2021

Лабораторна робота №4

Тема: Аналіз алгоритмів формування псевдовипадкових послідовностей

Мета: отримати знання про генерацію псевдовипадкових послідовностей та про базові статистичні ймовірнісні тести(частотний(монобітний) тест, тест двох бітових серій, тест Поккера, тест серій(загальний), автокореляційний тест), навчитися застосовувати ці тести.

Зміст задачі:

1. Федеральним стандартом США FIPS – 140 – 1, 140 – 2 використовуються 4 статистичні тести на випадковість. Замість вибору користувачами потрібних рівнів значущості задаються реальні границі. Довжина бітової послідовності 20000 бітів. Проведіть аналіз цього стандарту.

Монобітний тест. Кількість n_0 та n_1 , $9654 < n_i < 10346$.

Тест Поккера. Статистичний параметр χ_3 обчислюється для $m=4$; тест виконується, якщо $1,03 < \chi_3 < 57,4$.

Тест серій. Тести проходять, якщо кожний із 12 номерів B_i та G_i , $1 \leq i \leq 6$, знаходиться в інтервалі згідно з табл. 4.1.

Таблиця 4.1 – Значення інтервалів

Довжина серії	Відповідний інтервал
1	2267 – 2733
2	1079 – 1421
3	502 – 748
4	223 – 402
5	90 – 223
6	90 – 223

Тест довжин серій. Цей тест проходить, якщо не існує ніяких серій довжиною 34 або більше. Для високозахищених застосувань FIPS – 140 – 1 зобов'язує, щоб 4 тести виконувалися при кожній ініціалізації генератора випадкових бітів. В FIPS-140-2 максимальна довжина серії збільшена до 36 бітів.

2. Необхідно розробити такі процедури мовою C++ або іншою мовою:

а) генерація псевдовипадкової послідовності довільної довжини з використанням лінійного рекурентного реєстру, закон генерації якої визначається відповідно до примітивних поліномів, наведених в табл. 4.2;

Таблиця 4.2 – Примітивні поліноми

№ п/п	1	2	3	4	5
1	$x^{31} + x^3 + 1$	$x^{61} + x + 1$	$x^{95} + x^{11} + 1$	$x^{127} + x^{63} + 1$	$x^{31} + x^3 + 1$
2	$x^6 + x + 1$	$x^7 + x + 1$	$x^9 + x^4 + 1$	$x^{10} + x^3 + 1$	$x^5 + x^2 + 1$
№ п/п	6	7	8	9	10
1	$x^{100} + x^{15} + 1$	$x^{63} + x^5 + 1$	$x^{41} + x^3 + 1$	$x^{237} + x^{12} + 1$	$x^{52} + x^{21} + 1$
2	$x^9 + x^5 + 1$	$x^7 + x^3 + 1$	$x^{10} + x^7 + 1$	$x^6 + x^5 + 1$	$x^5 + x^3 + 1$

- б) реалізація монобітного тесту згідно з вищесказаною теорією для довільної довжини послідовності;
- в) реалізація тесту Поккера згідно з вищесказаною теорією для довільної довжини послідовності;
- г) реалізація тесту серій згідно з вищесказаною теорією для довільної довжини послідовності;
- д) реалізація тесту довгих серій згідно з вищесказаною теорією для довільної довжини послідовності.

3. Розв'язати задачі 2 (а-д) з використанням ЕОМ при довжині послідовності $l = 2 \cdot 10^4$ бітів.

4. Розв'яжіть задачу 1 пункту 1.10.2, якщо псевдовипадкова послідовність довжиною $n=128$ бітів побудована шляхом чотириразового повторення такої 32-бітової послідовності:

$$S=\{0110\ 0100\ 0111\ 1010\ 1100\ 1000\ 1111\ 0101\}.$$

Відповідь: ($\chi_1=0,5$; $\chi_2=2,295$; $\chi_3=1,048$; $\chi_4=5,99$; $\chi_5=0,18$).

Програмний код(C++):

Прінт-скрін виконання програми:

Контрольне питання

9. Дайте визначення лінійного конгруентного генератора.

Лінійний конгруентний генератор - це алгоритм, який дає послідовність псевдовипадкових чисел, обчислені за допомогою розривного кусочно-лінійного рівняння.

Цей метод являє собою один з найстаріших і найбільш відомих алгоритмів генератора псевдовипадкових чисел. Теорія, що лежить в їх основі, відносно проста для розуміння, і вони легко реалізуються і швидкі, особливо на комп'ютерному обладнанні, яке може забезпечити модульну арифметику шляхом усічення бітів пам'яті.

Метод лінійного порівняння (конгруентний спосіб)

Ціличисельні константи, що визначає генератор:

m модуль порівняння $m > 0,$

(натуральне число, відносно якого
обчислюють остатчу від ділення)

a множник $0 \leq a < m,$

c приріст $0 \leq c < m,$

X_0 початкове число $0 \leq X_0 < m.$

Послідовність випадкових чисел $\{X\}$ отримується за допомогою ітерацій:

$$X_{n+1} = (aX_n + c) \bmod m.$$

1. Якщо m, a, c і X_0 є цілими, то буде отримана послідовність цілих чисел з діапазону $0 \leq X_n < m.$
2. При $c = 0$, найбільший період складе $m/4$ при $a = 3+8j$ або $a = 5+8j$ і непарному початковому числі. Якщо $c = 0$, генератор часто називають мультиплікативним конгруентним генератором.
3. Якщо c непарне, а $a = 1+4j$, то період можна збільшити до числа $m = 2^n$

Висновок

Виконуючи лабораторну роботу, отримали знання про генерацію псевдовипадкових послідовностей та про базові статистичні ймовірнісні тести(частотний(монобітний) тест, тест двох бітових серій, тест Поккера, тест серій(загальний), автокореляційний тест).

Додаток

Лабораторна робота № 4

Тема: АНАЛІЗ АЛГОРИТМІВ ФОРМУВАННЯ ПСЕВДОВИПАДКОВИХ ПОСЛІДОВНОСТЕЙ

Базові статистичні ймовірнісні тести

Як базові рекомендується використовувати п'ять тестів:

- частотний (монобітний) тест;
- тест двох бітових серій;
- тест Поккера;
- тест серій (загальний);
- автокореляційний тест.

Монобітний тест.

Метою монобітного тесту є перевірка того, чи є число двійкових символів “0” та “1” в послідовності $a = a_0, a_1, \dots, a_n$ приблизно таким, як у випадкової послідовності. Якщо n_0 є число символів “0” в послідовності, а n_1 – символів “1”, то параметр ПВП

$$\chi_1 = \frac{(n_0 - n_1)^2}{n}$$

підпорядковується χ^2 розподілу з одним ступенем свободи (якщо $n > 10$).

Тест двобітових серій.

Метою тесту двобітових серій є перевірка того, чи є число з'явлень серій “00” – n_{00} , “01” – n_{01} , “10” – n_{10} , “11” – n_{11} такою, як і у випадковій послідовності. Параметр ПВП

$$\chi_2 = \frac{4}{n-1} \left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2 \right) - \frac{2}{n} \left(n_0^2 + n_1^2 \right) + 1$$

підпорядковується χ^2 розподілу з двома ступенями свободи (якщо $n \geq 21$).

Тест Поккера.

Нехай m – додатне ціле число, таке що

$$\left\lfloor \frac{n}{m} \right\rfloor \geq 5 \cdot (2^m); \quad k = \left\lfloor \frac{n}{m} \right\rfloor.$$

Розділимо послідовність Y на k не перекриваючих частин, кожна довжиною m , нехай і буде число з'явлення послідовності довжиною m . Тест Поккера дозволяє визначити, чи дійсно послідовності довжиною m кожна приблизно з'являються стільки ж разів, скільки очікується у випадковій послідовності. Параметр

$$\chi_3 = \frac{2^m}{k} \left(\sum_{i=1}^{2^m} n_i^2 \right) - k$$

підпорядковується χ^2 розподілу з $2^m - 1$ ступенями свободи.

Зазначимо, що тест Поккера є узагальненням частотного тесту – при $m = 1$ співпадає з частотним.

Тест серій.

Тест серій дозволяє визначити, чи дійсно число нулів або одиниць (серії) різної довжини в послідовності Y такі ж як і у випадковій послідовності. Бажане число інтервалів довжиною i у випадковій послідовності n є

$$l_i = (n - i + 3)/2^{i+2}.$$

Нехай k буде рівним найбільшому цілому числу i , для якого $l_i \geq 5$. Нехай також B та G буде числом блоків та інтервалів відповідно довжиною i в Y для кожного i , $1 \leq i \leq k$. Тоді параметр

$$\chi_4 = \sum_{i=1}^k \frac{(B_i - l_i)^2}{l_i} + \sum_{i=1}^k \frac{(G_i - l_i)^2}{l_i}$$

підпорядковується χ^2 розподілу з $2k - 2$ ступенями свободи.

Автокореляційний тест.

Метою автокореляційного тесту є перевірка ступеня зв'язку між Y_v і її зсувами. Нехай d фіксоване ціле число, $1 \leq d \leq \lfloor n/2 \rfloor$. Число бітів у Y послідовності дорівнює

$$R(d) = \sum_{i=0}^{n-d-1} s_i \oplus s_{i+d}.$$

Статистика параметра $\chi_5 = 2 \left(R(d) - \frac{n-d}{2} \right) / \sqrt{n-d}$ приблизно підпорядковується $N(0,1)$ нормальному розподілу, якщо $n - d \geq 10$. Автокореляційний метод має бути двостороннім, щоб розглядати його як для малих значень $R(d)$, так і для великих.

4.3 Приклади розв'язку задач

Задача 1.

Розглянемо послідовність Y довжиною $n=160$, яку сформуємо із чотирьох 40 бітових послідовностей Y_0 .

$$Y_0 = 11100\ 01100\ 01000\ 10100\ 11101\ 11100\ 10010\ 01001.$$

1. Частотний (монобітний) тест

$$n_0 = 21 \cdot 4 = 84, \quad n_1 = 19 \cdot 4 = 76.$$

$$\chi_1 = \frac{(84 - 76)^2}{160} = 0,4.$$

2. Двобітовий тест

$$n_{00} = 44; \quad n_{01} = 40; \quad n_{10} = 40; \quad n_{11} = 35.$$

$$\chi_2 = \frac{4}{159} (44^2 + 40^2 + 40^2 + 35^2) - \frac{2}{160} (84^2 + 76^2) + 1 = 0,62.$$

3. Тест Поккера

Нехай $m=3$ і $k=53$. Блоки 000, 001, 010, 011, 100, 101, 110, 111 з'являються відповідно 5, 10, 6, 4, 12, 3, 6 та 7 разів. Значення параметра

$$\chi_3 = \frac{2^3}{53} (5^2 + 10^2 + 6^2 + 4^2 + 12^2 + 3^2 + 6^2 + 7^2) - 53 = 9,6415.$$

4. Тест серій

$$l_i = (n - i + 3) / 2^{i+2};$$

$$l_1 = (160 - 1 + 3) / 2^3 = 20,25;$$

$$l_2 = (160 - 2 + 3) / 2^4 = 10,0625;$$

$$l_3 = (160 - 3 + 3) / 2^5 = 5.$$

Є 25, 4 та 5 блоків довжиною 1, 2, 3 відповідно, 8, 20 та 12 інтервалів довжиною 1, 2 та 3 відповідно. Параметр χ_4 приймає значення $\chi_4 = 31,7913$.

5. Автокореляційний тест.

Якщо $d=8$, то $R(8)=100$.

Значення статистичного параметра

$$\chi_5 = \left(2 \left(100 - \frac{160 - 8}{2} \right) \right) / \sqrt{160 - 8} = 3,8933.$$

Для рівня значущості $\alpha = 0,05$ порогові значення $\chi_1, \chi_2, \chi_3, \chi_4$ та χ_5 дорівнюють 3.8415, 5.9915, 14.0671, 9.4877, 1.96 відповідно.

Бачимо, що послідовність проходить частотний (монобітний) тест, двобітовий тест та тест Поккера, але не проходить тест серій та автокореляційний тест.

Задачі для самостійного розв'язання

1. Федеральним стандартом США FIPS – 140 – 1, 140 – 2 використовуються 4 статистичні тести на випадковість. Замість вибору користувачами потрібних рівнів значущості задаються реальні граници. Довжина бітової послідовності 20000 бітів. Проведіть аналіз цього стандарту.

Монообітний тест. Кількість n_0 та n_1 , $9654 < n_i < 10346$.

Тест Поккера. Статистичний параметр χ_3 обчислюється для $m=4$; тест виконується, якщо $1,03 < \chi_3 < 57,4$.

Тест серій. Тести проходять, якщо кожний із 12 номерів B_i та G_i , $1 \leq i \leq 6$, знаходиться в інтервалі згідно з табл. 4.1.

Контрольні запитання та завдання

1. Поясніть алгоритм функціонування генератора псевдовипадкових послідовностей.
2. Назвіть основні показники оцінки властивостей генератора псевдовипадкових послідовностей.
3. Визначте період повторення лінійної рекурентної послідовності, якщо $m=\{7, 10, 12, 19, 31, 63, 89, 127, 257, 52\}$.
4. Визначте структурну скритність лінійної рекурентної послідовності періоду $L = 2^m - 1$.
5. Визначте значення безпечної часу, якщо в генератор псевдовипадкової послідовності може бути введено N_k ($10^{20}, 10^{25}, 10^{30}, 10^{35}, 10^{40}, 10^{50}, 10^{70}, 10^{80}, 10^{90}, 10^{100}, 10^{127}$) ключів.
6. Визначте структурну скритність лінійної рекурентної послідовності періоду $L = 2^m - 1$, $m=20, 25, 30, 35, 40, 61, 63, 85, 89, 127, 257, 507$.
7. Поясніть сутність:
 - монобітного тесту;
 - тесту двобітових серій;
 - тесту Поккера;
 - тесту серій;
 - автокореляційного тесту.
8. Поясніть вихідний код процедури реалізації:
 - монобітного тесту;
 - тесту двобітових серій;
 - тесту Поккера;
 - тесту серій;
 - автокореляційного тесту.
9. Дайте визначення лінійного конгруентного генератора.
10. Які вимоги висуваються до коефіцієнтів a та b рекурентного співвідношення, що визначає алгоритм функціонування лінійного конгруентного генератора.
11. За яких умов лінійний конгруентний генератор забезпечує максимальний період формувальної послідовності.
12. Зробіть пропозиції щодо реалізації та виберіть параметри конгруентного генератора, що забезпечує максимальний період послідовності.
13. Зробіть пропозиції щодо реалізації та виберіть параметри конгруентного генератора, що забезпечує максимальний період послідовності.