Государственное образовательное учреждение высшего профессионального образования
«Московский Государственный Технический Университет имени Н. Э. Баумана»

ОТЧЕТ
По лабораторной работе №8
По курсу «Анализ алгоритмов»
Тема: **«Потоковые алгоритмы»**

Студент: Кононенко С. Д.
Группа: ИУ7-51

Москва, 2017

Постановка задачи

1. Реализовать потоковый алгоритм обработки данных

Теория

В данной лабораторной работе я реализовал потоковый алгоритм шифрования, а именно алгоритм симулирующий работу шифровальной машины Enigma
Поточный шифр - это симметричный шифр, в котором каждый символ открытого текста преобразуется в символ шифрованного текста в зависимости не только от используемого ключа, но и от его расположения в потоке открытого текста.

Принцип работы(кратко) Энигма – семейство роторных шифровальных машин. Шифрование данных происходит при помощи сложной комбинации шифра замены и шифра цезаря, технически осуществляемых прохождением тока по цепи от клавишной панели до панели индикаторов
Каждый набор роторов, позиция роторов и настрока коммутационной панели обеспечивает уникальный ключ шифрования, что в итоге гарантирует около 159e18 уникальных стартовых настроек машины.
За счет вращения роторов обеспечивается потоковость алгоритма, т.к. две одинаковых буквы идущие последовательно будут шифроваться различными символами.
Подробно ознакомиться с механизмом работы машины Enigma можно по сслыке https://habrahabr.ru/post/217331/

Алгоритм

```
typedef struct rotor_t
{
     int permutations[26];
     int start_pos;
     int cur_pos;
}rotor_t;

typedef struct reflector_t
{
    int permutations[26];
}reflector_t;

rotor_t rotor_I   =
{4,10,12,5,11,6,3,16,21,25,13,19,14,22,24,7,23,20,18,15,0,8,1,17,2,9, 17};
rotor_t rotor_II  =
{0,9,3,10,18,8,17,20,23,1,11,7,22,19,12,2,16,6,25,13,15,24,5,21,14,4, 5};
rotor_t rotor_III =
{1,3,5,7,9,11,2,15,17,19,23,21,25,13,24,4,8,22,6,0,10,12,20,18,16,14, 22};
reflector_t reflector_B =
{24,17,20,7,16,18,11,3,15,23,13,6,14,10,12,8,4,1,5,25,2,22,21,9,0,19};

static int rotate_rot(rotor_t *rot)
{
    rot->cur_pos++;
    (rot->cur_pos > 25) ? (rot->cur_pos = 0) : NOTHING;
    return (rot->cur_pos == rot->start_pos) ? 1 : 0;
}

static void rotate_rotors(void)
{
    if (rotate_rot(&rotor_I))
        if(rotate_rot(&rotor_II))
            rotate_rot(&rotor_III);
}
```

```
static char dec26(char c, char symb)
{
    c -= symb;
    c < 0 ? c+=26 : NOTHING;
    return c;
}

static int ind_of(int perm[26], int num)
{
    int i = 0;
    while (perm[i++] != num);
    return --i;
}
static char magic(char c)
{
    c -= 'a';
    c = (c + rotor_I.cur_pos) % 26;
    c = rotor_I.permutations[c];

    c = (c + dec26(rotor_II.cur_pos,rotor_I.cur_pos)) % 26;
    c = rotor_II.permutations[c];


    c = (c + dec26(rotor_III.cur_pos,rotor_II.cur_pos)) % 26;
    c = rotor_III.permutations[c];


    c = dec26(c, rotor_III.cur_pos);
    c = reflector_B.permutations[c];


    c = (c + rotor_III.cur_pos) % 26;
    c = ind_of(rotor_III.permutations, c);


    c = dec26(c,dec26(rotor_III.cur_pos,rotor_II.cur_pos));
    c = ind_of(rotor_II.permutations, c);


    c = dec26(c, dec26(rotor_II.cur_pos,rotor_I.cur_pos));
    c = ind_of(rotor_I.permutations, c);


    c = dec26(c, rotor_I.cur_pos);

    return c + 'a';
}

void encryption(void)
{
    printf("******Encription******\n"
           "Enter roters position(3 num[0,25] sep. by space : ");
    int p1, p2, p3;
    scanf("%d %d %d", &p1, &p2, &p3);
    rotor_I.cur_pos   = p3;
    rotor_II.cur_pos  = p2;
    rotor_III.cur_pos = p1;
    printf("Input your message here(/ to cancel) :\n");

    rotate_rotors();
    char c = getch();
    while (c != '/')
    {
```

```
        if (c >= 'a' && c <= 'z')
        {
            c = magic(c);
        }
        printf("%c", c);
        rotate_rotors();
        c = getch();
    }
    printf("\n");
}
```

Данный алгоритм реализует упрощенную модель машины Enigma, здесь отсутствуют: коммутационная панель, выбор роторов и их позиций относительно друг друга.

Заключения

В ходе выполнения лабораторной работы я более подробно изучил механизм работы шифровальнйо работы Enigma, а также прочих потоковых алгоритмов шифрования.