ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ГОУ ВПО НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
им. Р.Е. АЛЕКСЕЕВА

Кафедра "Вычислительные системы и технологии"

ПРОГРАММИРОВАНИЕ

Отчёт

по лабораторной работе № 1 (3 семестр)

Шифрование.

Выполнил студент группы 21-ИВТз
Халеев Алексей Андреевич
(Фамилия Имя Отчество студента)
_____ «27» ноября 2022 г.
(личная подпись) (дата)
Провел старший преподаватель кафедры
«Вычислительные системы и технологии»
(должность, кафедра)
Мартынов Дмитрий Сергеевич
(Фамилия Имя Отчество преподавателя) .
_____«___» _____ 20___ г.
(личная подпись) (дата)

Нижний Новгород 2022

Цели работы:

1. Изучить принцип работы линейно-конгруэнтного генератора псевдослучайных чисел.

2. Исследовать механизм XoR преобразования данных.

3. Научится использовать побитовые (XoR) операции для шифрования текста.

Задание:

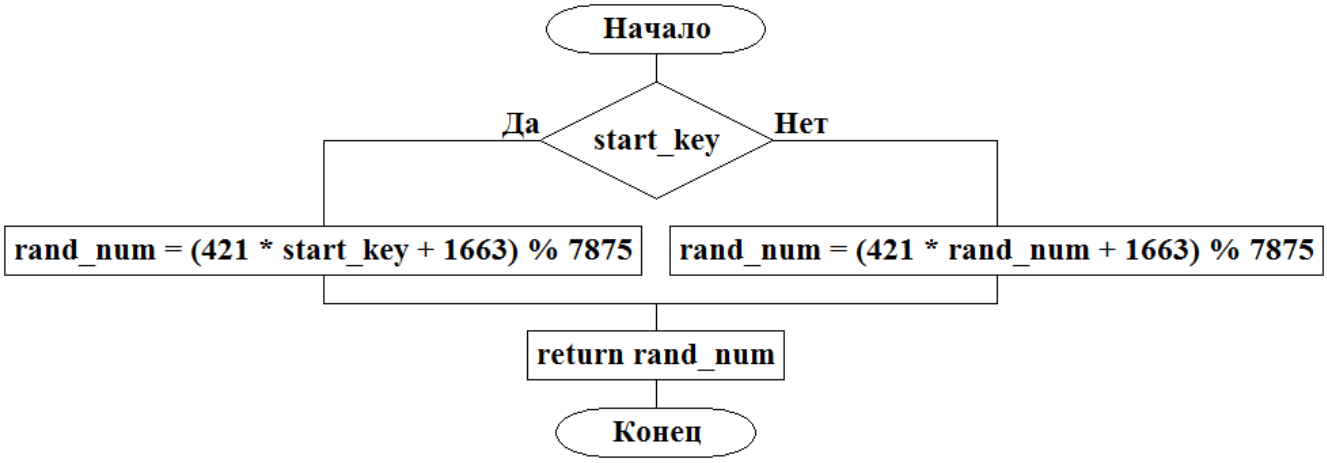
Составить алгоритм и написать программу на языке C++. Программа компилируется и запускается под управлением ОС Linux.

Разработанная программа должна содержать встроенную справочную информацию, описывающую правила использования и цель назначения. Аргументы запуска программа должна обрабатывать согласно рекомендациям POSIX.

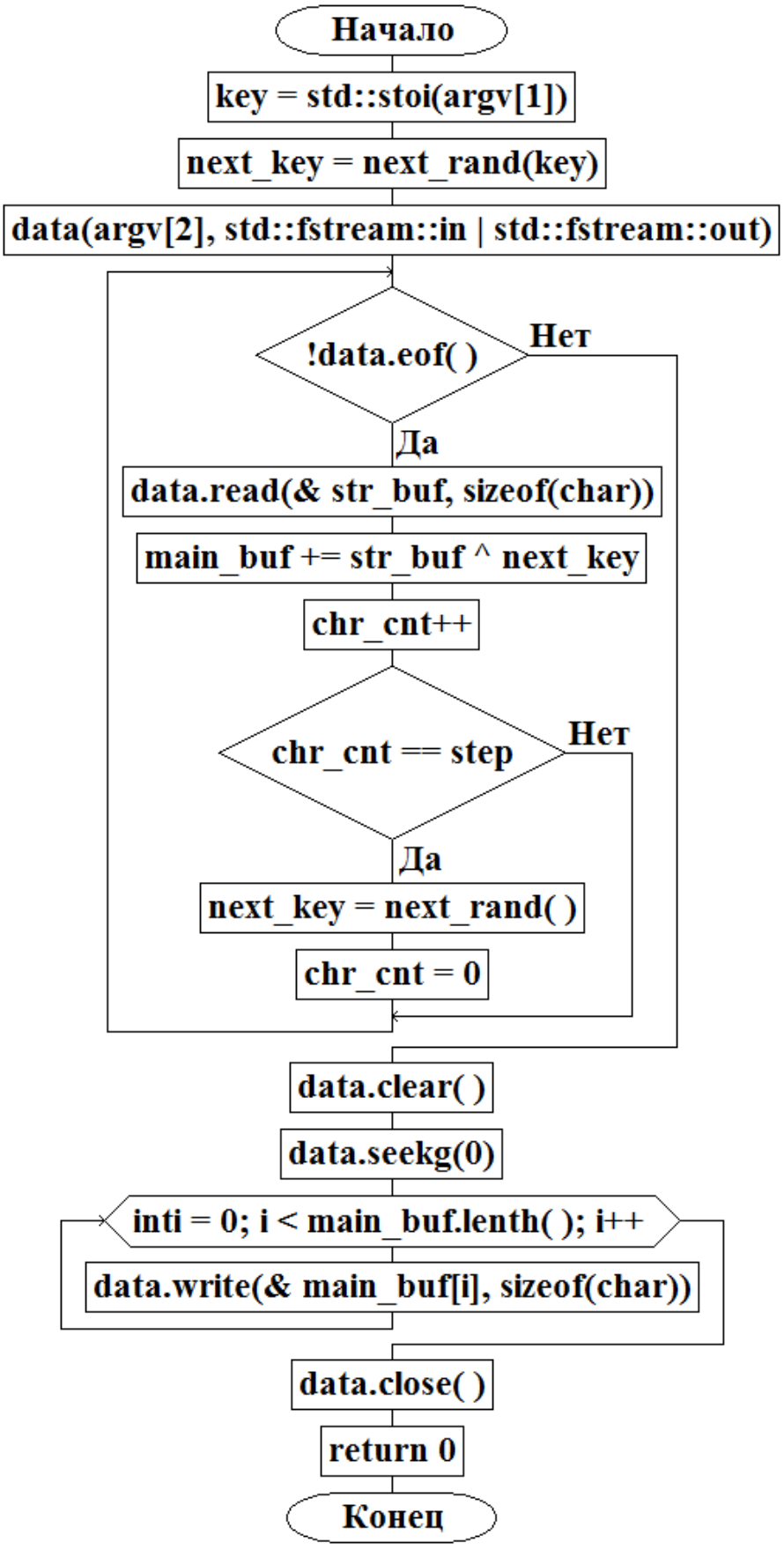
Назначение программы:

Программа предназначена для организации шифрования/дешифрования текста. Шифрование производится за счет XoR преобразования символов, ключ для которого генерируется по принципу линейно-конгруэнтного генератора псевдослучайных чисел. Шаг генерации 64 байта. Начальное значение передается в качестве параметра командной строки.

Блок-схема NextRand()



Блок схема main()



Листинг NextRand()

```
int NextRand(int start_key = 0) {
   /*****************************************************************************************************************
   * Цель: Возврат следующего случайного числа
   * Исходные данные:
   *      start_key   - начальный ключ ЛКГ генератора
   *      rand_num    - следующее случайное число, вычисляемое по принципу ЛКГ
   * Результат: функция возвращает случайное число
   * Вызываемые модули: -
   * Описание алгоритма:
   * 1) Передаваемый в функцию аргумент имеет значение по умолчанию, равное 0. Таким образом функция не требует
   * явного указания стартового ключа, однако его стоит задать, для упрощения задачи дешифрования, а также получения
   * последовательности при вызове без аргументов.
   * 2) Переменная rand_num является статической - ее значение не теряется при выходе из области видимости функции,
   * это необходимо для получения псевдо-случайной последовательности.
   * 3) При каждом вызове производится расчет следующего члена ПСП по формуле Д.Г. Лемера. Константы удовлетворяют
   * условиям получения "хорошей" последовательности, для данной функции период равен 7875.
   * Дата: 01 / 06 / 2022 Версия 1.01
   * Автор: Студент НГТУ ИРИТ, Халеев А. А. гр. 21-ИВТз
   * Исправления: нет
   *****************************************************************************************************************/
    static int rand_num{0};
    if (start_key) {
        rand_num = (421 * start_key + 1663) % 7875;
    } else
        rand_num = (421 * rand_num + 1663) % 7875;
    return rand_num;
}
```

Листинг main()

```
int main(int argc, char **argv) {
    try {
        switch (argc) {
            case 2: {   // получен 1 пользовательский аргумент + argc (помним, что argc всегда получаем первым)
                /* Если полученный аргумент соответствует запуску в режиме справки*/
                if (std::string(argv[1]) == "-h" || std::string(argv[1]) == "--help") {
                    std::string help_screen[] = {
                            "\nLab1_3 is text encoding/decoding app\n",
                            "Use the same start key to decode your encoded text\n",
                            "Be care of your path - it's need to be correct\n\n",
                            "Correct arguments for run this app:\n\n",
                            "--help or -h    : help mode\n",
                            "[N] [file_name] : encryption/decryption\n",
                            "    N          – start LKG key;\n",
                            "file_name      – file name or full path to save/read the text file;\n\n"
                    };
                    for (auto &i: help_screen) {
                        std::cout << i;
                    }
                } else
                    throw SyntaxException("First argument is wrong. Among other things, check your keyboard layout", 1,
                                          argc);
                break;
            } break;
            case 3: { // получено 2 пользовательских аргумента + argc
                int key;
                try {
                    key = std::stoi(argv[1]); // попытка привести к целому типу полученного аргумента
                } catch (std::exception &stoi_err) {
                    throw SyntaxException("The specified number of lines must be an integer", 2, argc);
                }
                if (argv[1] != std::to_string(key)) {
                    throw SyntaxException(
                            "It is not possible to use a floating point number to specify the start LKG number\n"
                            "You should also use only digits for an integer", 3, argc);
                } else {
                    int next_key = NextRand(key); // инициализируем стартовый ключ для XoR преобразования
                    char str_buf;                 // буфер для считанных символов
                    std::string main_buf;         // буфер-накопитель
                    std::size_t chr_cnt{0};       // счетчик считанных символов

                    /* Открытие файла */
                    std::fstream data(argv[2], std::fstream::in | std::fstream::out);
                    if (data.is_open()) {
                        while (!data.eof()) {                   // Пока не достигнут конец файла
                            data.read(&str_buf, sizeof(char));  // Считать очередной символ в str_buf
                            main_buf += str_buf ^ next_key;     // XoR преобразование символа с добавлением в буфер-накопитель
                            chr_cnt++;
                            if (chr_cnt == step) {
                                next_key = NextRand();          // генерируем следующий ключ
                                chr_cnt = 0;                    // обнуляем счетчик
                            }
                        }
                        data.clear();
                        data.seekg(0);
                        for (auto &elem: main_buf) {
                            data.write(&elem, sizeof(char));    // запись преобразованного текста
                        }
                        /* Закрытие файла */
                        data.close();
                        if (data.is_open() != 0) {
                            throw SyntaxException("Unable to close specified file", 4, argc);
                        }
                    } else {
                        throw SyntaxException(
                                "Unable to open or create specified file. Check that the specified file exists",
                                5, argc);                       // Аварийное завершение программы с выводом справки
                    }
                }
            } break;
            default:
            {
                throw SyntaxException(
                        "Wrong number of arguments", 6, argc);
            }
        }
```

```
    } catch (SyntaxException &ex)   // Если поймали собственное исключение (переданы некорректные аргументы)
    {
        ex.description(); // Вызываем сообщение, соответствующее ошибке с помощью метода .description()
        return 1;             // Завершение программы с кодом 1
    }
    return 0;
}
```

Вывод

В ходе выполнения лабораторной работы:

- изучен принцип работы линейно-конгруэнтного генератора случайных чисел.

- исследован механизм XoR преобразования.

- разработана программа, использующая побитовые(XoR) операции для шифрования текста.