**Садов Д.В. ИВБО 7-14.**

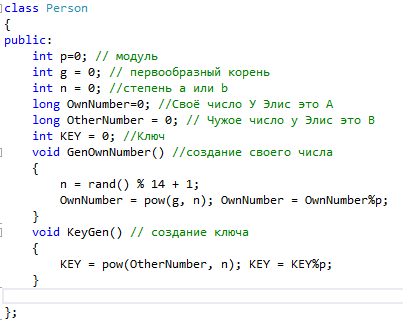
**Лабораторная работа №3** В данной лабораторной работе нужно было реализовать алгоритм Диффи-Хеллмана.

Реализована программа следующим образом:

1. Через файл общаются Элис и Боб.

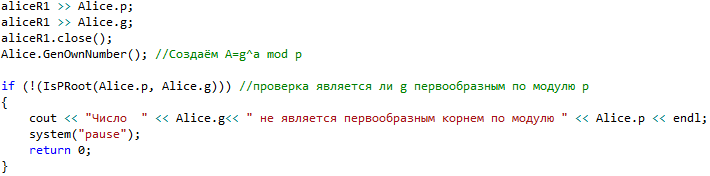
У каждого есть свой файл: Alice.txt и Bob.txt

Создается класс Person со следующими полями и методами и его экземпляры (Alice,Bob):

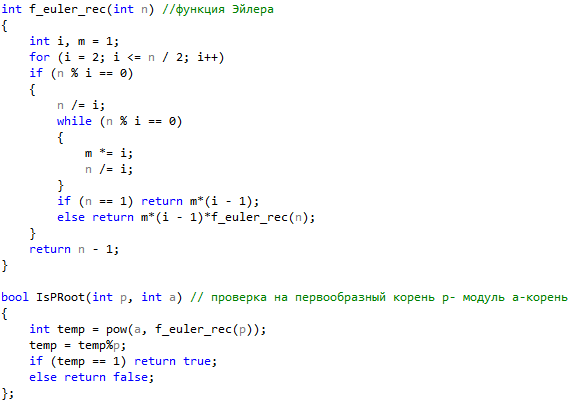


1. В файле Alice.txt лежат два значения – модуль и первообразный корень

Эти значения считываются в поля объекта Alice. На основе считанных данных Элис выбирает случайное приватное число (a) и по формуле A=g^a mod p получает свое число А, которое рассылает публично. Вычисляется это в методе GenOwnNumber.

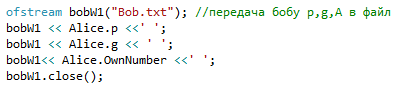


Также реализуется проверка является ли число g первообразным по модулю p. Делается это с помощью функций f\_euler\_rec (функции Эйлера) и IsPRoot

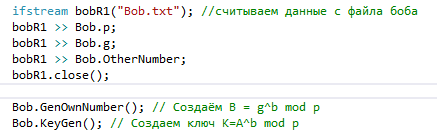


1. После того как число А посчитано в файл Боба (Bob.txt) записываются значения

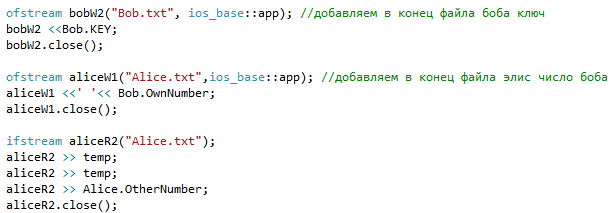
p,g,A



1. Боб используя полученные p,g заполняет свои поля объекта. Далее он выбирает свое рандомное приватное число (b) и генерирует число B по формуле B=g^b mod p.



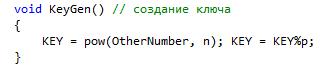
После того как число B сформировано, оно записывается в конец файла Боба (bob.txt) а также пересылается к Элис, где также записывается в конец файла (alice.txt). Также формируется общий ключ по формуле K=A^b mod p и записывается в конец файла.



1. Далее Элис берет публичный результат Боба (B) и вычисляет ключ по формуле

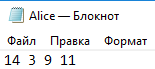
K=B^a mod p, где а - ее случайное приватное число. Ключ записывается в конец файла.

Реализовано это в методе KeyGen.

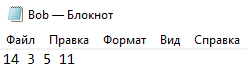


В итоге получаются 2 файла имеющие 1 общий секретный ключ.

14-p 3-g 9-A 11-Key



14-p 3-g 5-B 11-Key



**Листинг программы (файлы alice.txt , bob.txt есть на гитхабе)**

#include <iostream>

#include <fstream>

#include <time.h>

#include <stdlib.h>

using namespace std;

int f\_euler\_rec(int n) //функция Эйлера

{

int i, m = 1;

for (i = 2; i <= n / 2; i++)

if (n % i == 0)

{

n /= i;

while (n % i == 0)

{

m \*= i;

n /= i;

}

if (n == 1) return m\*(i - 1);

else return m\*(i - 1)\*f\_euler\_rec(n);

}

return n - 1;

}

bool IsPRoot(int p, int a) // проверка на первообразный корень p- модуль а-корень

{

int temp = pow(a, f\_euler\_rec(p));

temp = temp%p;

if (temp == 1) return true;

else return false;

};

class Person

{

public:

int p=0; // модуль

int g = 0; // первообразный корень

int n = 0; //степень a или b

long OwnNumber=0; //Своё число У Элис это А

long OtherNumber = 0; // Чужое число у Элис это B

int KEY = 0; //Ключ

void GenOwnNumber() //создание своего числа

{

n = rand() % 14 + 1;

OwnNumber = pow(g, n); OwnNumber = OwnNumber%p;

}

void KeyGen() // создание ключа

{

KEY = pow(OtherNumber, n); KEY = KEY%p;

}

};

int main(int args, const char \* argv[])

{ //в файле элис хранятся два значения p и g

setlocale(LC\_ALL, "rus");

srand(time(NULL));

int temp=0;

Person Alice,Bob; //человеки

ifstream aliceR1("Alice.txt"); // входной поток из исходного файла

aliceR1 >> Alice.p;

aliceR1 >> Alice.g;

aliceR1.close();

Alice.GenOwnNumber(); //Создаём А=g^a mod p

if (!(IsPRoot(Alice.p, Alice.g))) //проверка является ли g первообразным по модулю p

{

cout << "Число " << Alice.g<< " не является первообразным корнем по модулю " << Alice.p << endl;

system("pause");

return 0;

}

ofstream bobW1("Bob.txt"); //передача бобу p,g,A в файл

bobW1 << Alice.p <<' ';

bobW1 << Alice.g << ' ';

bobW1<< Alice.OwnNumber <<' ';

bobW1.close();

ifstream bobR1("Bob.txt"); //считываем данные с файла боба

bobR1 >> Bob.p;

bobR1 >> Bob.g;

bobR1 >> Bob.OtherNumber;

bobR1.close();

Bob.GenOwnNumber(); // Создаём B = g^b mod p

Bob.KeyGen(); // Создаем ключ K=A^b mod p

ofstream bobW2("Bob.txt", ios\_base::app); //добавляем в конец файла боба ключ

bobW2 <<Bob.KEY;

bobW2.close();

ofstream aliceW1("Alice.txt",ios\_base::app); //добавляем в конец файла элис число боба

aliceW1 <<' '<< Bob.OwnNumber;

aliceW1.close();

ifstream aliceR2("Alice.txt");

aliceR2 >> temp;

aliceR2 >> temp;

aliceR2 >> Alice.OtherNumber;

aliceR2.close();

Alice.KeyGen(); // Создаем ключ K=B^a mod p

ofstream aliceW2("Alice.txt", ios\_base::app); //добавляем в конец файла элис ключ

aliceW2 << ' ' << Alice.KEY;

aliceW2.close();

//файл Элис содержит p(модуль) g(первообразный корень) B(число собеседника) KEY(ключ для шифра)

//файл Боба содержит p(модуль) g(первообразный корень) A(число собеседника) KEY(ключ для шифра)

system("pause");

return 0;

}