Министерство образования и науки РФ
Федеральное государственное автономное образовательное учреждение высшего образования
«Омский государственный технический университет»

Факультет (институт) *Информационных технологий и компьютерных систем*

Кафедра *Прикладная математика и фундаментальная информатика*

Расчетно-графическая работа

по дисциплине ***Алгоритмизация и программирование***

на тему Разработка программы «Алгоритм RSA»

Пояснительная записка

Шифр проекта 020-РГР-02.03.02-№ 14-ПЗ

Студента Курпенова Куата Ибраимовича
фамилия, имя, отчество полностью

Курс *1* Группа ФИТ-212

Направление (специальность) ***02.03.02***
Фундаментальная информатика и информационные технологии
код, наименование

Руководитель ***ст. преподаватель***
ученая степень, звание

Федотова И.В.
фамилия, инициалы

Выполнил ***02.06.2022.***
дата, подпись студента

Работа защищена с количеством баллов	

04.06.2022
дата, подпись руководителя

Омск 2022

СОДЕРЖАНИЕ

ЗАДАНИЕ

Разработать схему алгоритма, написать и отладить программу с возможностью взаимодействия, используя объектно-ориентированное программирование. Объектно-ориентированное программирование (ООП) – методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

Идеологически ООП – подход к программированию как к моделированию информационных объектов, решающий на новом уровне основную задачу структурного программирования: структурирование информации с точки зрения управляемости, что существенно улучшает управляемость самим процессом моделирования, что, в свою очередь, особенно важно при реализации крупных проектов.

Управляемость для иерархических систем предполагает минимизацию избыточности данных (аналогичную нормализации) и их целостность, поэтому созданное удобно управляемым – будет и удобно пониматься. Таким образом, через тактическую задачу управляемости решается стратегическая задача – транслировать понимание задачи программистом в наиболее удобную для дальнейшего использования форму.

ФОРМУЛИРОВКА ЗАДАЧИ И АЛГОРИТМЫ ПРОГРАММЫ

1. Реализовать генерацию публичного ключа
2. Реализовать генерацию приватного ключа
3. Реализовать шифрование сообщения
4. Реализовать дешифрование сообщения

ОБЩАЯ СХЕМА АЛГОРИТМА

Расчётно-графическая работа объединяет следующие задачи:

- Вывод графических элементов в окно игры, а также взаимодействие игрока с ними с помощью клавиш;
- Движение объектов, то есть изменение их координат в окне;
- Прекращение игры в момент закрытия игрового окна;
- Демонстрация наглядного прогресса игрока (увеличение длины змеи);

ТЕКСТ ПРОГРАММЫ НА C++

Файл “RSA.h”

```
#pragma once


#include <cmath>


class RSA {
public:
    RSA(long long, long long);
    ~RSA();


    long long* generate_private();
    long long* generate_public();


    long long encrypt(long long, long long*&);
    long long decrypt(long long, long long*&);


private:
    long long p_;
    long long q_;
    long long n_;
    long long euler_;


    long long k_;
    long long public_exp_;
    long long private_exp_;


    long long* public_key_;
    long long* private_key_;
};
```

```
inline RSA::RSA(long long p, long long q) {
    p_ = p;
    q_ = q;

    n_ = p_ * q_;

    euler_ = (p_ - 1) * (q_ - 1);

    k_ = 2;
    public_exp_ = 3;
    private_exp_ = (1 + (k_ * euler_)) / public_exp_;
}

inline RSA::~RSA() {
    delete private_key_;
    delete public_key_;
}

inline long long* RSA::generate_private() {
    private_key_ = new long long[2];
    private_key_[0] = private_exp_;
    private_key_[1] = n_;

    return private_key_;
}

inline long long* RSA::generate_public() {
    public_key_ = new long long[2];
    public_key_[0] = public_exp_;
    public_key_[1] = n_;

    return public_key_;
```

```
}

inline long long RSA::encrypt(long long message, long long*& public_key) {
    long long e = public_key[0];
    long long n = public_key[1];

    return static_cast<long long>(std::pow(message, e)) % n;
}

inline long long RSA::decrypt(long long code, long long*& private_key) {
    long long d = private_key[0];
    long long n = private_key[1];

    return static_cast<long long>(std::pow(code, d)) % n;
}
```

Файл “main.cpp”

```
#include <iostream>

#include "RSA.h"

int main() {
    long long p;
    long long q;

    std::cout << "[>] Enter two simple numbers: ";
    std::cin >> p >> q;

    RSA rsa(p, q);

    long long* private_key = rsa.generate_private();
    long long* public_key = rsa.generate_public();
```

```
    std::cout << "[+] Private key: {" << private_key[0] << ", " << \
        private_key[1] << "}" << std::endl;
    std::cout << "[+] Public key: {" << public_key[0] << ", " << \
        public_key[1] << "}" << std::endl;

    long long message;
    std::cout << "[>] Enter message: ";
    std::cin >> message;

    long long encrypted = rsa.encrypt(message, public_key);
    long long decrypted = rsa.decrypt(encrypted, private_key);

    std::cout << "[+] Encrypted message: " << encrypted << std::endl;
    std::cout << "[+] Decrypted message: " << decrypted << std::endl;
}
```

РАЗРАБОТКА ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ

```
tux@tux-computer ~/D/O/2/A/C/Code (main)> g++ main.cpp
tux@tux-computer ~/D/O/2/A/C/Code (main)> ./a.out
[>] Enter two simple numbers: 13 11
[+] Private key: {80, 143}
[+] Public key: {3, 143}
[>] Enter message: 1
[+] Encrypted message: 1
[+] Decrypted message: 1
tux@tux-computer ~/D/O/2/A/C/Code (main)>
```

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1) Wikipedia RSA
2) Habr RSA
3) Tutorials Point RSA