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** | | | | | |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | дата, подпись руководителя |  |  |  |

Омск 2022

**СОДЕРЖАНИЕ**

[ЗАДАНИЕ 3](#_Toc1)

[ФОРМУЛИРОВКА ЗАДАЧИ И АЛГОРИТМЫ ПРОГРАММЫ 4](#_Toc2)

[ОБЩАЯ СХЕМА АЛГОРИТМА 5](#_Toc3)

[ТЕКСТ ПРОГРАММЫ НА C++ 6](#_Toc4)

[РАЗРАБОТКА ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ 14](#_Toc5)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 15](#_Toc7)

**ЗАДАНИЕ**

Разработать схему алгоритма, написать и отладить программу с возможностью взаимодействия, используя объектно-ориентированное программирование. Объектно-ориентированное программирование (ООП) – [методология программирования](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), основанная на представлении программы в виде совокупности [объектов](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), каждый из которых является экземпляром определённого [класса](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), а классы образуют иерархию наследования.

Идеологически ООП – подход к программированию как к моделированию информационных объектов, решающий на новом уровне основную задачу [структурного программирования](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5): 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++**

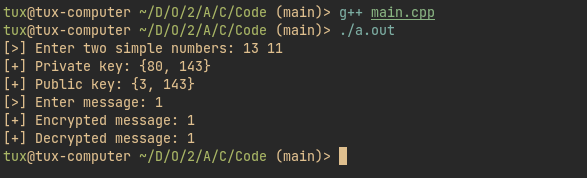
Файл “RSA.h”

|  |
| --- |
| #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 ПОЛЬЗОВАТЕЛ**Я**



**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. [Wikipedia RSA](https://ru.wikipedia.org/wiki/RSA)
2. [Habr RSA](https://habr.com/ru/post/534014/)
3. [Tutorials Point RSA](https://www.tutorialspoint.com/cplusplus-program-to-implement-the-rsa-algorithm)