**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
 РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФГБОУ ВО «БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ   
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**Кафедра** «Информатика и программное обеспечение»

**РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА**

Оценка показателей качества программного обеспечения

*по дисциплине* **«Метрология и качество программного обеспечения»**

Всего листов

Выполнил ст. гр. О-18-ПРИ-рпс-Б

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Лядов В. С.

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Руководитель:

\_\_\_\_\_к.т.н., доц. Азарченков А.А.

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Брянск 2021

**СОДЕРЖАНИЕ**

[введение 4](#_Toc93883465)

[1. Формирование и анализ требований 5](#_Toc93883466)

[1.1. Введение 5](#_Toc93883467)

[1.1.1. Цель 5](#_Toc93883468)

[1.1.2. Область применения 5](#_Toc93883469)

[1.1.3. Определения, термины и сокращения 5](#_Toc93883470)

[1.1.4. Обзор 6](#_Toc93883471)

[1.2. Общее описание 6](#_Toc93883472)

[1.2.1. Перспектива продукта 6](#_Toc93883473)

[1.2.2. Концепции операций 7](#_Toc93883474)

[1.2.3. Программные интерфейсы 7](#_Toc93883475)

[1.2.4. Ограничения памяти 8](#_Toc93883476)

[1.2.5. Операции 8](#_Toc93883477)

[2. Оценка сложности на основе лексических показателей 9](#_Toc93883478)

[2.1. Задание 9](#_Toc93883479)

[2.2. Ход работы 10](#_Toc93883480)

[2.2.1. Расчет структурных параметров ПС 10](#_Toc93883481)

[2.2.2. Расчет длины программы 10](#_Toc93883482)

[2.2.3. Расчет объема программного средства 10](#_Toc93883483)

[2.2.4. Расчет количества команд ассемблера 11](#_Toc93883484)

[2.2.5. Расчет календарного времени программирования 11](#_Toc93883485)

[2.2.6. Расчет начального количества ошибок 11](#_Toc93883486)

[2.2.7. Расчет начальной надежности ПС 11](#_Toc93883487)

[3. Оценка сложности проекта на основе процедурно-ориентированных метрик 13](#_Toc93883488)

[3.1. Задание 13](#_Toc93883489)

[3.2. Ход работы 13](#_Toc93883490)

[4. Оценка трудоемкости разработки на основе стоимостной модели COCOMO II 16](#_Toc93883491)

[4.1. Задание 16](#_Toc93883492)

[4.2. Ход работы 16](#_Toc93883493)

[5. Выбор и обоснование метрик исходного кода 18](#_Toc93883494)

[5.1. Оценка характеристик программ на основе объектно-ориентированных метрик. Метрика Мартина 18](#_Toc93883495)

[6. Оценка показателей внутреннего качества и выработка рекомендаций по улучшению показателей 20](#_Toc93883496)

[7. Документация к коду 21](#_Toc93883497)

[7.1. Класс SHA256 21](#_Toc93883498)

[7.1.1. Метод Encod 21](#_Toc93883499)

[7.1.2. Метод EncriptBlock 21](#_Toc93883500)

[7.1.3. Метод MakeEncriptKey 21](#_Toc93883501)

[7.1.4. Метод Decod 21](#_Toc93883502)

[7.1.5. Метод MakeDecriptKey 22](#_Toc93883503)

[7.1.6. Метод InvertKey 22](#_Toc93883504)

[7.1.7. Метод Inv 22](#_Toc93883505)

[7.1.8. Метод Mul 22](#_Toc93883506)

[8. Заключение 23](#_Toc93883507)

# введение

Целью работы является получение практических навыков в оценке показателей качества программного обеспечения и выработки рекомендаций по повышению показателей внутреннего качества.

# Формирование и анализ требований

## Введение

### Цель

Этот документ предоставляет все требования для программы «Шифрование на основе режима сцепления блоков шифра IDEA».

### Область применения

Этот документ охватывает требования к программному обеспечению «Реализация алгоритма ЭЦП на основе хэш-функции SHA256 и метода RSA». По данному документу будут делаться замечания относительно некоторых конкретных особенностей данной программы. Цель этого – направление процесса проектирования во время разработки приложения.

### Определения, термины и сокращения

| **Сокращение или термин** | **Определение** |
| --- | --- |
| Шифр | Какая-либо система преобразования текста с секретом (ключом) для обеспечения секретности передаваемой информации. |
| С - требование | Сводка требований к приложению, сформулированных в форме, понятной клиенту |
| D - требование | Сводка требований к приложению, сформулированных достаточно четко для использования программистами при проектировании и реализации. По возможности D-требования должны быть также понятны и клиенту |
| Ключ | Это секретная информация, используемая криптографическим алгоритмом при шифровании/дешифровании сообщений |
| Алгоритм | Набор инструкций, описывающих порядок действий исполнителя для достижения некоторого результата |
| Приложение | Прикладной компьютерный сервис, который обладает набором определенных функций и является одним из компонентов программного обеспечения. |

### Обзор

Части 1 и 2 предназначены преимущественно для заказчиков приложения, но также будут интересны инженерам-разработчикам, разрабатывающим или поддерживающим его.

## Общее описание

«Реализация алгоритма ЭЦП на основе хэш-функции SHA256 и метода RSA». планируется как приложение, выполняющее шифрование текста с использованием алгоритма SHA256 и метода RSA.

Оно должно быть интересно людям, задействованным в области криптографии. Оценка успеха приложения остается за пользователем.

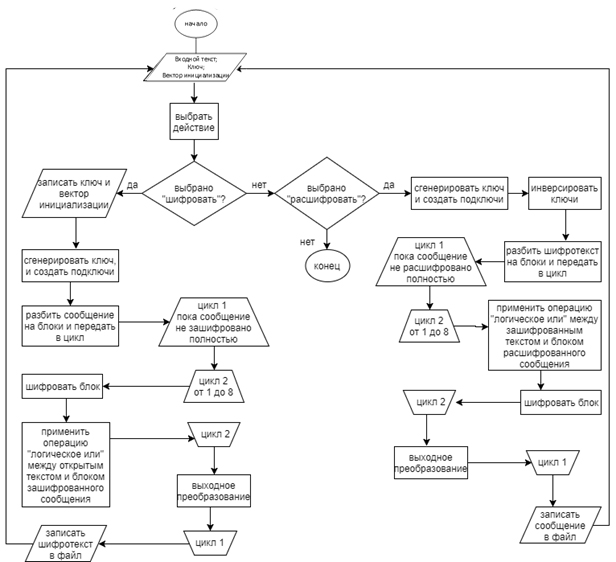
### Перспектива продукта

«Реализация алгоритма ЭЦП на основе хэш-функции SHA256 и метода RSA». должно удовлетворить нужды программистов в обладании большим влиянием на содержание программы с дополнительным программированием. Данное приложение должно быть интересно людям, задействованным в области криптографии.

Проект и документация программного продукта позволит легко расширять и изменять приложение.

### Концепции операций

Приложение осуществляет ввод текста, его отображение в бинарном виде, задание пользователем стартового ключа для дальнейшей генерации ключей. Также, генерация ключей с помощью алгоритма SHA-256. Отображение результата происходит на выводе шифртекста, а также блоков по 64 бита, которые получились в результате работы данного алгоритма.



Блок-схема

### Программные интерфейсы

Программа будет работать на ПК с операционной системой Windows и установленной на нем средой выполнения Node.js. Языком разработки будет JavaScript.

### Ограничения памяти

Для данного программного продукта потребуется не более 4ГБайт оперативной памяти и 50 Мбайт вспомогательного запоминающего устройства. Размер текста не более 231 символов.

### Операции

В программе будут находится поля ввода для сообщения, ключа. После введения сообщения и выбора действия, введенный текст будет сконвертирован в двоичную систему для последующей операции шифрования (XOR) с ключами. Ключ, который будет вводить пользователь, будет стартовым числом для генерации дальнейших ключей. После выбора действия, программа выведет ключи, сгенерированные алгоритмом. Будет осуществлено разбиение на блоки по 64 бита, для дальнейшей работы режима сцепления блоков шифра. Для наглядности все будет выведено в Textbox, для того чтобы пользователь понимал алгоритм работы данного шифра.

# Оценка сложности на основе лексических показателей

## Задание

Оценить, используя метрики Холстеда, сложность разработки программного средства «Реализация алгоритма ЭЦП на основе хэш-функции SHA256 и метода RSA».

## Ход работы

Для выполнения данного этапа работы необходимо оценить параметр (число входных и выходных параметров). Для данного ПС данный параметр равен 112.

### Расчет структурных параметров ПС

Принять рекомендацию Холстеда об оптимальном количестве входных переменных модуля (напомним, что, по его мнению, это число должно быть равно 8), находим число модулей ПС:

Поскольку условие выполняется, данное ПC является многоуровневым.

### Расчет длины программы

Так как , то .

Тогда длина модуля определится следующим образом:

1494 слова.

Отсюда полная длина программного средства будет равна:

### Расчет объема программного средства

Находим объем одного модуля:

,

тогда общий объем будет программы содержащей K модулей, определяется как кратное найденной величины количеству модулей:

.

### Расчет количества команд ассемблера

Расчет данной метрики производится по следующей формуле:

где – коэффициент пересчета Кнута на команды ассемблера (машинные команды).

### Расчет календарного времени программирования

Расчет данной метрики производится по следующей формуле:

где – количество программистов в бригаде разработчиков, – производительность (число отлаженных программ в день), устанавливаемая директивно (обычно задается в пределах 10 < v <30).

### Расчет начального количества ошибок

Расчет данной метрики производится по следующей формуле:

### Расчет начальной надежности ПС

Исходя из обобщения практического опыта, период отладки 𝜏 в пределах календарного времени разработки 𝑇𝑘 можно определить, используя неравенство:

Для данного ПС:

Отладка различных программных средств может занимать различное время в зависимости от сложности программы, однако то, что отладка занимает весьма солидную часть этапа проектирования программ, ни у кого не вызывает сомнения.

Вычислим время наработки программы на отказ (т.е работы ПС до появления первого сбоя):

Поскольку данный показатель зависит от периода отладки, вычислим значение данной метрики на граничных значениях периода отладки.

Для :

Для :

# Оценка сложности проекта на основе процедурно-ориентированных метрик

## Задание

Оценить, используя метод функциональных точек, сложность разработки программного средства «Реализация алгоритма ЭЦП на основе хэш-функции SHA256 и метода RSA».

## Ход работы

Произведем расчет количества функциональных точек для ПС, реализующего шифрование на основе режима сцепления блоков шифра SHA-256.

В программе необходимо реализовать шифрование текста с помощью трех генераторов псевдослучайных чисел. Подсчитаем количество функциональных точек за пять шагов.

1. Установление границ данного ПС не вызывает трудностей, так как оно является полностью локальным, и обмен данными с другими ПС в нем не предусматривается.
2. В ПС имеется один внутренний логический файл (ILF) для хранения информации необходимой для шифрования. Причем, данные могут храниться как в обычном файле, так и в таблице СУБД.

Число типов элементов записей (RET) для этого файла может равно единице, так как все данные хранятся в формате строки. Число типов элементов данных (DET) внутреннего логического файла будет равно восьми вне зависимости от формата представления данных. Таким образом, уровень сложности внутреннего логического файла – низкий.

Внешних интерфейсных файлов (EIF) ПС не имеет.

В ПС имеются четыре внешних ввода (EI): «Шифровать», «Расшифровать», «Вектор инициализации» и «Ключ» (Сдвиговый регистр).

Так как внешний ввод «Ключ» ссылается на одну строку (поле ввода текста пользователем), то уровень сложности этого ввода низкий.

Уровень сложности внешнего ввода «ЛКГ» также низкий, поскольку имеется пять DET (поле «x0», «число m», «число a», «Исходный текст», «число c»).

Уровень сложности внешнего ввода «Вектор инициализации» низкий, поскольку имеется четыре DET (последний ключ ЛКГ «key», «число m», «число a», «Исходный текст»).

Аналогично уровень сложности внешнего ввода «СР» также низкий, поскольку имеется три DET (поле «гамма BBS», результат XOR – «число k», «Исходный текст»).

В ПС отсутствуют внешние запросы.

Полученные данные сведем в табл. 9. и рассчитаем ненормированное количество функциональных точек UFPC.

Таблица 1

*Данные для расчета числа UFPC*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Характеристика** | **Уровень сложности** | | | **Итого** |
| **Низкий** | | |
| **Кол.** | **Ранг** | **Итог** |
| Внешние вводы | 2 | 3 | 6 | 6 |
| Внешние выводы | 2 | 4 | 8 | 8 |
| Внешние запросы | 0 | 3 | 0 | 0 |
| Внутренние логические файлы | 1 | 7 | 7 | 7 |
| Внешние логические файлы | 0 | 5 | 0 | 0 |
| Итого (UFPC) | | | | 21 |

Подсчитаем итоговую степень влияния (TDI) общих характеристик системы и нормирующий фактор (VAF).

В результате выясним, что для телефонного справочника важны следующие характеристики:

* Диалоговый ввод данных – 2
* Эффективность для конечного пользователя – 1
* Сложность обработки данных – 2
* Повторное использование – 3
* Распространенность – 5

Остальные характеристики либо не присутствуют, либо не имеют значения для данного ПС и поэтому имеют вес, равный 0.

Нормирующий фактор (VAF) определится как:

Таким образом, нормированное количество функциональных точек вычисляется по формуле:

В заключение, оценим количество строк исходного кода с использованием бэкфайер – метода исходя из того, что программу необходимо разработать с использованием языка программирования C#:

Таким образом, законченная программа будет содержать примерно строк исходного кода на языке программирования C#.

# Оценка трудоемкости разработки на основе стоимостной модели COCOMO II

## Задание

Оценить, используя стоимостную модель COCOMOI I, трудоемкость разработки программного средства шифрования с помощью генераторов псевдослучайных чисел «Реализация алгоритма ЭЦП на основе хэш-функции SHA256 и метода RSA».

## Ход работы

Выполним расчет трудоемкости и сроков реализации программного проекта с использованием модели COCOMO для ПС.

В качестве исходных данных о размере ПС используем оценку количества строк кода на языке программирования JavaScript, полученную с использованием метода функциональных точек и бэкфайер-метода, а именно, 824 строки.

Очевидно также, что по всем признакам (размеру, сложности, ограничениям и т.п.) данный проект следует отнести к типу распространенных.

По этим исходным данным с помощью формулы без учета влияния стоимостных факторов, т.е. при единичном коэффициенте нормирования трудозатрат, можно определить ненормированные номинальные трудозатраты:

а также по формуле ненормированную длительность проекта

Далее оценим стоимостные факторы данного проекта. Результаты такой оценки приведены в табл. 2.

Таблица 2

*Оценка стоимостных факторов ПС*

| **Фактор** | **Уровень** | **Коэффициент** |
| --- | --- | --- |
| RELY | Очень низкий | 0,75 |
| DATA | Низкий | 0,88 |
| CPLX | Очень низкий | 0,70 |
| TIME | Номинальный | 1,0 |
| STOR | Номинальный | 1,0 |
| VIRT | Низкий | 0,87 |
| TURN | Низкий | 0,87 |
| ACAP | Очень низкий | 1,46 |
| PCAP | Очень низкий | 1,42 |
| AEXP | Низкий | 1,13 |
| LEXP | Номинальный | 1,0 |
| VEXP | Низкий | 1,10 |
| TOOL | Высокий | 0,91 |
| MODP | Некоторое | 1,0 |
| SCED | Номинальный | 1,0 |
| Коэффициент нормирования трудозатрат | | 0,82 |

Определим нормированные трудозатраты на реализацию проекта:

а также нормированную длительность проекта:

Таким образом, для того чтобы разработать данную программу необходимо распределить трудозатраты эквивалентные 1,23 человеко-месяцам. Оптимальным планируемым сроком разработки данного программного проекта в предполагаемых условиях будет 2,7 месяца.

# Выбор и обоснование метрик исходного кода

Для оценки характеристик разработанного программного обеспечения произведем анализ с использованием объектно-ориентированных метрик – метрик Мартина. Данный выбор обоснован тем, что данная программа написана с использованием объектно-ориентированного стиля.

## Оценка характеристик программ на основе объектно-ориентированных метрик. Метрика Мартина

Проанализируем текст программы для оценки ее качества с помощью метрик Мартина, которые позволяют оценить меру сложности объектно- ориентированной программы на основе анализа детализации организационной структуры классов программы и их связности, так как все классы работают во взаимодействии.

Исходный код программы включает два класса, предназначенный для решения задачи.

Классы:

* Класс SHA256;
* Класс RSA;

Кроме указанного классов в программе используются обращения к библиотеке встроенных классов среды программирования, которые не входят в рассматриваемую категорию классов. Необходимо отметить, что классы, не входящие в рассматриваемую категорию, никак не зависят от данного класса. Следовательно, центростремительное сцепление классов в обеих категориях по теории Мартина Са = 0.

Определим центробежное сцепление классов для каждой категории. В первой категории шесть классов зависят от классов, вне данной категории. Во второй категории два класса зависят от классов, вне данной категории. Следовательно, .

Расчетная метрика нестабильности 𝐼 определяется следующим образом:

Исходя из полученных значений, следует, что категории класса анализируемой программы является максимально нестабильными, так как зависимость от классов других категорий носит преобладающий характер.

Определим меру абстрактности 𝐴 рассматриваемых категорий классов. В программе не присутствует абстрактных классов, следовательно, . Всего в первой категории семь классов, а во второй два. Следовательно, .

Исходя из значений метрики A, можно сделать вывод, что рассматриваемая категория класса является полностью конкретной.

Расстояние до главной последовательности определяется уравнением . В соответствии с теорией Мартина расстояние до главной последовательности определятся формулой

Категории расположены на прямой главной последовательности, что свидетельствует о наилучшей сбалансированности между абстрактностью и нестабильностью. Нормализованное расстояние до главной последовательности равно:

# Оценка показателей внутреннего качества и выработка рекомендаций по улучшению показателей

В целом качество разработанной программы можно считать достаточно высоким, так как рассматриваемая категория класса исходного кода находятся в пределах нормализованного расстояния до главной последовательности.

С целью улучшения показателей можно добавить больше абстрактных классов в категории классов и увеличить количество классов вне конкретной категории, которые зависят от классов внутри нее.

# Документация к коду

Данный программный продукт состоит из двух классов SHA256 и RSA.

## Класс SHA256

Данный класс осуществляет комплексную реализацию программы. Он содержит восемь методов: Encod, EncriptBlock, MakeEncriptKey, Decod, MakeDecriptKey, InvertKey, Inv, Mul.

### Метод Encod

Функция шифрования алгоритмом SHA256 режима сцепление блоков.

Входные параметры:

* Byte[ ] InputArrByte;
* Byte[ ] KeyArrByte.

### Метод EncriptBlock

Функция шифрования блока алгоритмом SHA256.

Входные параметры:

* Byte[ ] BufIn,
* UInt16[ ] Key

### Метод MakeEncriptKey

Функция получения ключей для шифрования.

Входные параметры:

* Byte[ ] KeyArrByte

### Метод Decod

Функция расшифровки алгоритмом SHA256 режима сцепление блоков

Входные параметры:

* Byte[ ] InputArrByte,
* Byte[ ] KeyArrByte,
* Byte[ ] ArrVec

### Метод MakeDecriptKey

Функция получения ключей для расшифровки.

Входные параметры:

* Byte[ ] KeyArrByte

### Метод InvertKey

Функция получения ключей для расшифрования из ключей шифровки/

Входные параметры:

* UInt16[ ] EncKey

### Метод Inv

Функция инвертирования числа

Входные параметры:

* UInt16 x

### Метод Mul

Функция перемножения двух чисел

Входные параметры:

* UInt16 x,
* UInt16 y

# Заключение

В данной расчетно-графической работе была произведена оценка показателей качества программного обеспечения, разработанного в рамках курсовой работы по предмету «Защита информации», а также были выработаны рекомендации по повышению показателей внутреннего качества кода.

В ходе работы были сформированы и проанализированы требования, предъявляемые к данному программному продукту, была оценена сложность разработки программы несколькими способами, а также была проведена оценка качества программы и составлена документация к коду.