Министерство цифрового развития, связи и массовых коммуникаций

Российской Федерации

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

(МТУСИ)

Лабораторная работа №3 по теме:

«ЭЛЕКТРОННО-ЦИФРОВАЯ ПОДПИСЬ И ПРИЕМЫ ХЕШИРОВАНИЯ»

Вариант №1

Выполнил студент группы БФИ2202

Сидорук Данил

**Цель работы**

Овладеть практическими навыками закрытия информации электронно-цифровой подписью и приемами хеширования, рассмотрение хеширования методом контрольным сумм и методом наложения кодов — гаммированием.

**Ход работы**

1. Ознакомиться с теоретической частью данной работы.

2. Составить программу шифрования методом контрольных сумм.

3. Составить программу шифрования методов хеширования с применением гаммирования.

4. Составить отчет по проделанной работе.

5. Защитить работу.

**Постановка задачи**

Составить программу шифрования методом контрольных сумм и методом хеширования с применением гаммирования.

Пусть a = 17, b = 11, c = MaxVal + 1 = 256, = 172. Вычислить контрольные суммы для нескольких сообщений методом контрольных сумм и методом хеширования с применением гаммирования:

а) P = „0123456789“

б) P = „9876543210“

в) P = „1000005“

г) P = „1500000“

**Листинг программы**

*mainwindow.cpp*

#include "mainwindow.hpp"  
  
#include "hashing.hpp"  
  
MainWindowForm::MainWindowForm(QWidget \*parent) : QMainWindow(parent) {  
 ui.setupUi(this);  
  
 QObject::connect(ui.hashButton, &QPushButton::clicked, this, [this]() {  
 auto hash = hashSum(ui.textEdit->text(), {.max = ui.cEdit->text().toLongLong() - 1 });  
 ui.hashLabel->setText(QString::number(hash));  
 });  
 QObject::connect(ui.gammaButton, &QPushButton::clicked, this, [this]() {  
 auto hash = gammaSum(ui.textEdit->text(), {  
 .a = ui.aEdit->text().toLongLong(),  
 .b = ui.bEdit->text().toLongLong(),  
 .c = ui.cEdit->text().toLongLong(),  
 .t = ui.tEdit->text().toLongLong()  
 });  
 ui.gammaLabel->setText(QString::number(hash));  
 });  
}

*mainwindow.hpp*

#pragma once  
  
#include "ui\_mainwindow.h"  
  
class MainWindowForm : public QMainWindow {  
 **Q\_OBJECT**public:  
 explicit MainWindowForm(QWidget \*parent = nullptr);  
  
private:  
 Ui::MainWindow ui;  
};

*hashing.cpp*

#include "hashing.hpp"  
  
#include <QString>  
  
long long hashSum(const QString &text, HashParameters params) {  
 long long K{};  
 for (const auto c : text) {  
 K += c.unicode();  
 }  
 return (K <= params.max ? K : K % (params.max + 1));  
}  
  
long long gammaSum(const QString &text, GammaParameters params) {  
 long long K = params.t;  
 long long previousT = params.t;  
 for (const auto c: text) {  
 auto X = c.unicode();  
 auto T = (params.a \* previousT + params.b) % params.c;  
 auto Y = (X ^ T);  
 K = (K + Y) % params.c;  
 previousT = T;  
 }  
 return K;  
}

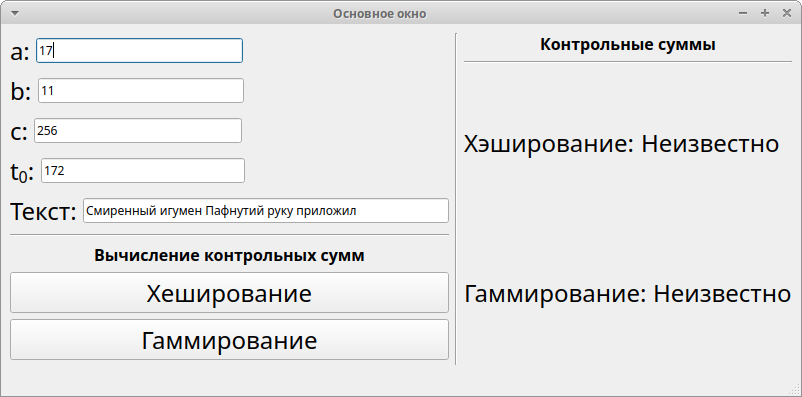
*hashing.hpp*

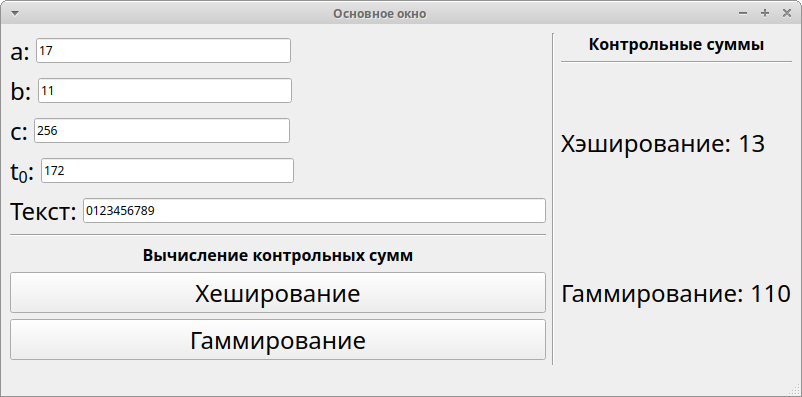
#pragma once  
  
class QString;  
  
struct HashParameters {  
 long long max;  
};  
  
struct GammaParameters {  
 long long a;  
 long long b;  
 long long c;  
 long long t;  
};  
  
long long hashSum(const QString &text, HashParameters params);  
  
long long gammaSum(const QString &text, GammaParameters params);

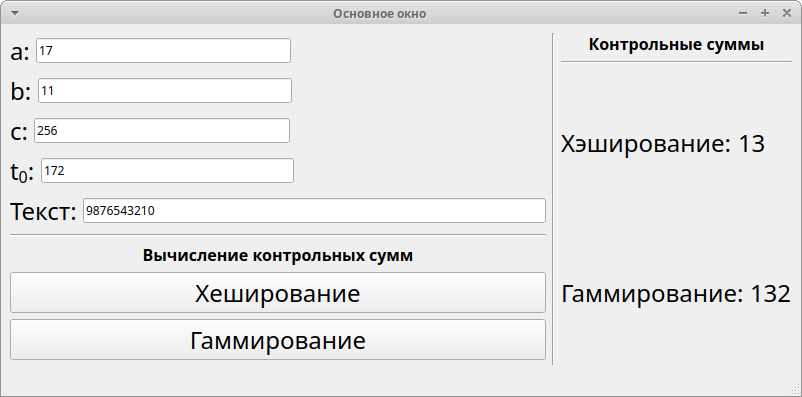
*main.cpp*

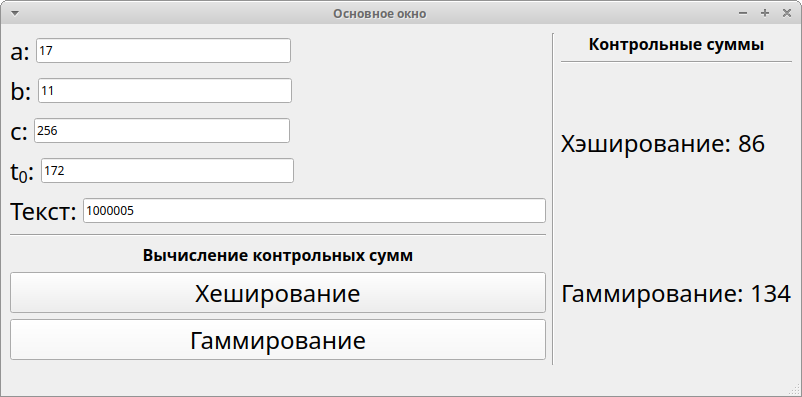
#include <QApplication>  
  
#include "forms/mainwindow.hpp"  
  
int main(int argc, char \*argv[]) {  
 QApplication app(argc, argv);  
 QApplication::setOrganizationName("eoanermine");  
 QApplication::setApplicationName("passwords\_strength");  
  
 MainWindowForm mainWindow;  
 mainWindow.show();  
 return QApplication::exec();  
}

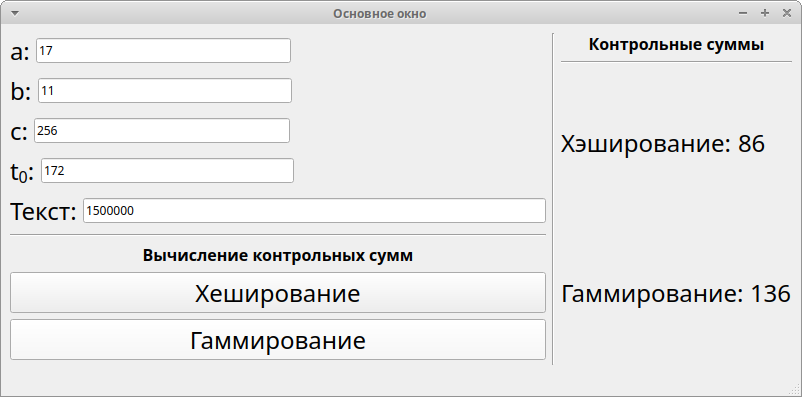
**Результат выполнения программы**

Рисунок 1 - основное окно программы

Рисунок 2 - результат выполнения программы для T = “0123456789”

Рисунок 3 - результат выполнения программы для T = “0123456789”

Рисунок 4 - результат выполнения программы для T = “1000005”

Рисунок 5 - результат выполнения программы для T = “1500000”