**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский Авиационный Институт»**

**(Национальный Исследовательский Университет)**

**Институт: №8 «Информационные технологии   
и прикладная математика»   
Кафедра: 806 «Вычислительная математика   
и программирование»**

Лабораторная работа № 2  
по курсу «Криптография»

Группа: М8О-307Б-22

Студент: Е. С. Кострюков

Преподаватель: А. В. Борисов

Оценка:

Дата: 09.04.2025

Москва, 2025

**ОГЛАВЛЕНИЕ**

[**Задание** 3](#_Toc195119522)

[**Теория** 3](#_Toc195119523)

[**Ход лабораторной работы** 4](#_Toc195119524)

[**Выводы** 8](#_Toc195119525)

**Работа с хэш-функцией и факторизацией**

# **Задание**

Строку в которой записано своё ФИО подать на вход в качестве аргумента хеш-функции ГОСТ Р 34.11-2012 (Стрибог). Младшие 8 бит выхода интерпретировать как число, которое в дальнейшем будет номером варианта от 0 до 255. В отчёт включить снимок экрана с выбором номера варианта, а также описать шаги решения задачи. Задача: разложить каждое из чисел `a` и `b` на нетривиальные сомножители. Ниже представлены 256 вариантов.

# **Теория**

**Современный российский стандарт хеширования данных**

В 2012 году был утверждён ГОСТ Р 34.11-2012, известный как алгоритм "Стрибог", который стал новым этапом в развитии отечественных криптографических технологий. Этот стандарт разрабатывался в рамках государственной программы по созданию защищённых алгоритмов обработки информации и представляет собой усовершенствованную хеш-функцию для обеспечения цифровой безопасности.

**Основные принципы работы**

Криптографическая хеш-функция преобразует информацию произвольного объёма в уникальную цифровую сигнатуру фиксированного размера. Ключевыми требованиями к таким алгоритмам являются:

- Минимизация вероятности коллизий;

- Устойчивость к криптоаналитическим атакам;

- Высокая скорость обработки данных;

"Стрибог" реализует эти принципы через комплекс математических операций, включая побитовые преобразования, модульную арифметику и циклические сдвиги.

**Эволюция стандартов**

Новый алгоритм пришёл на смену устаревшему ГОСТ Р 34.11-94, устранив его уязвимости и повысив криптостойкость. Вместе со стандартом электронной подписи ГОСТ Р 34.12-2015 он образует современную систему защиты информации, соответствующую требованиям российского законодательства.

**Области применения**

1. Системы электронной подписи.

2. Защита персональных данных.

3. Обеспечение государственной тайны.

4. Финансовые операции и платежные системы.

**Преимущества перед аналогами**

- Оптимизирован для российских правовых требований;

- Устойчив к современным видам атак (включая rainbow-таблицы);

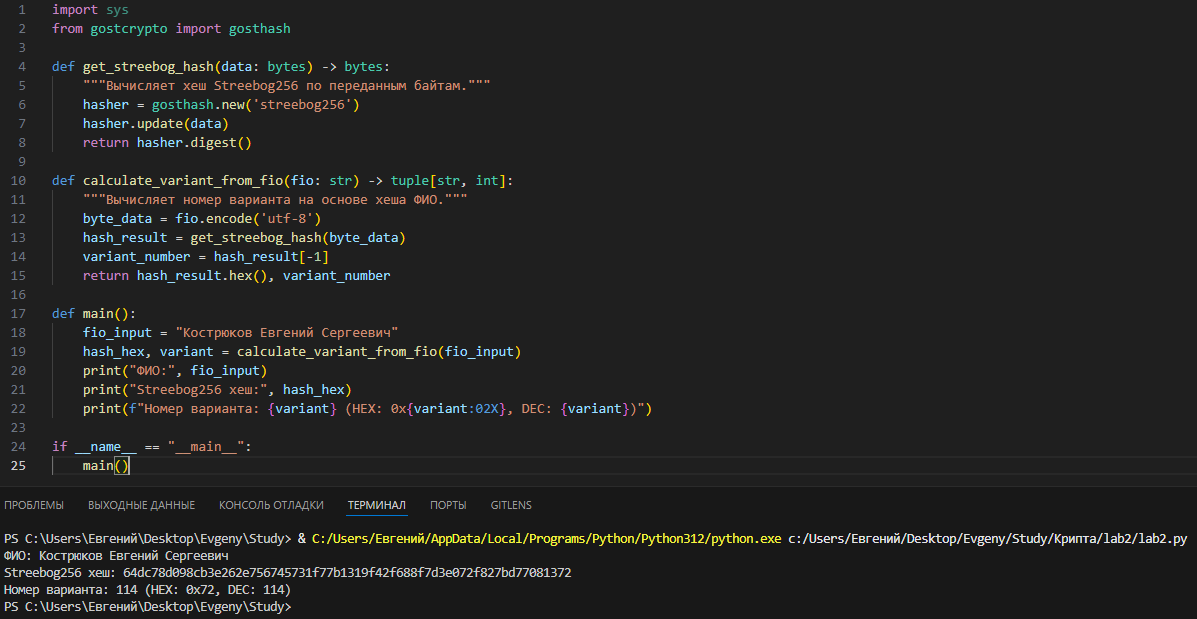
- Поддерживает различные режимы хеширования;

- Интегрирован в национальную систему стандартов;

Данный алгоритм стал обязательным компонентом защищённых информационных систем в государственном и коммерческом секторах, подтвердив свою эффективность на практике. Его разработка отражает стратегический курс России на технологическую независимость в области криптографии.

# **Ход лабораторной работы**

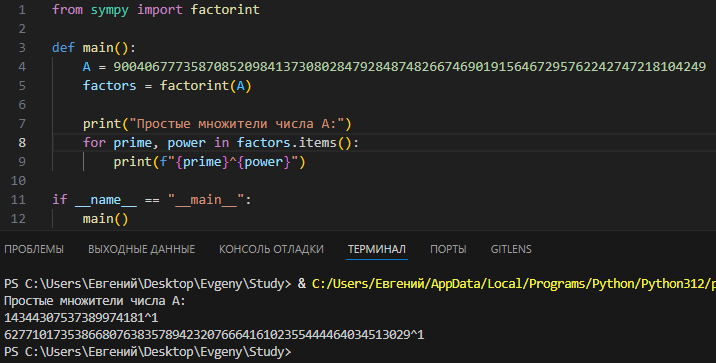
Для определения своего варианта я использовал Python и библиотеку gostcrypto, с помощью которых написал скрипт. В результате выполнения программы получил номер варианта — 114.

****

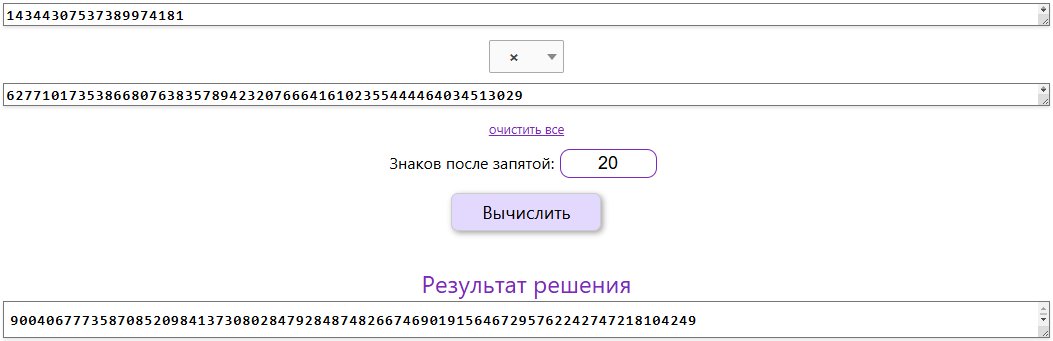
Вариант 114:

a[114]=90040677735870852098413730802847928487482667469019156467295762242747218104249

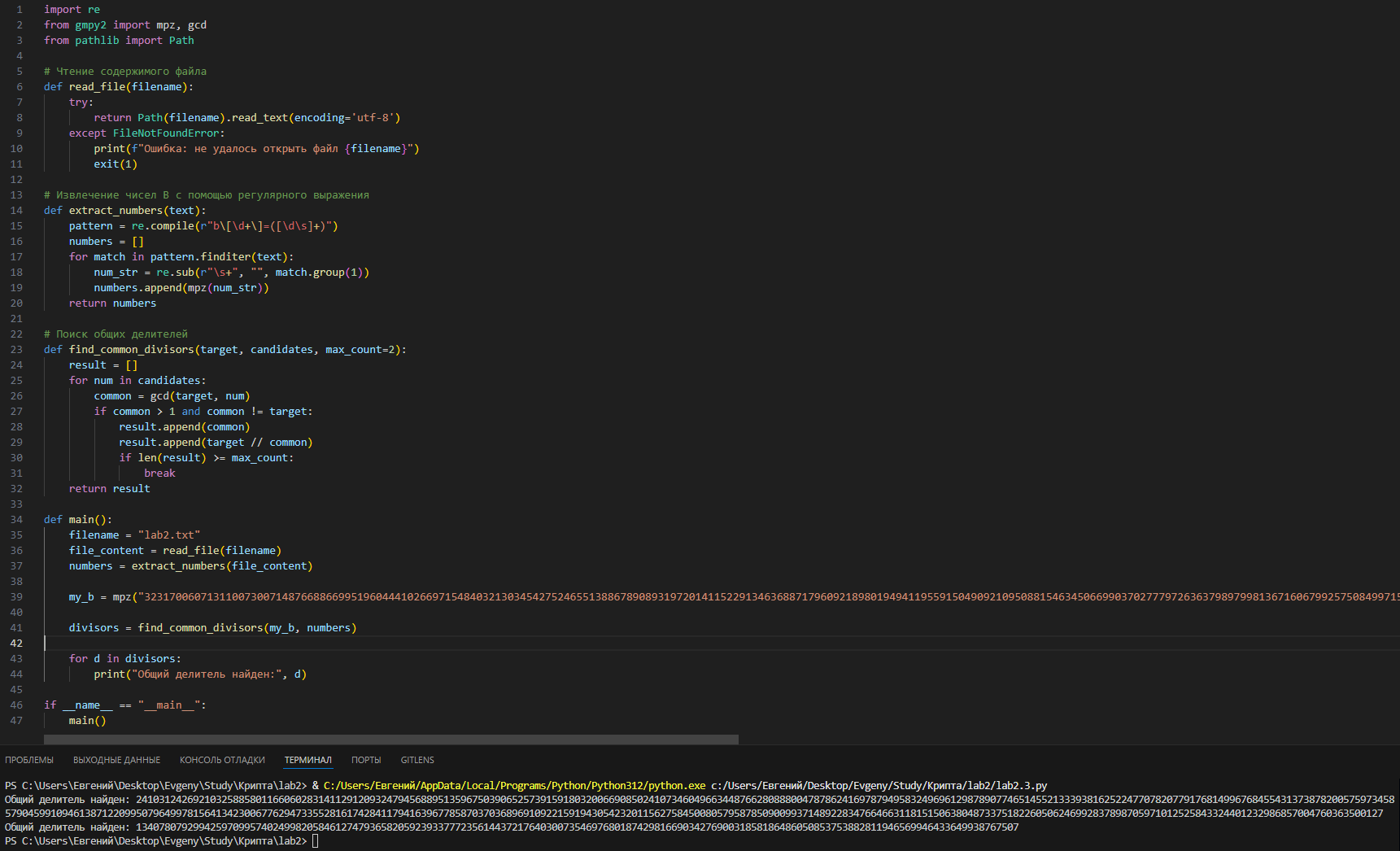
b[114]=32317006071311007300714876688669951960444102669715484032130345427524655138867890893197201411522913463688717960921898019494119559150490921095088154634506699037027779726363798979981367160679925750849971537538870855219307801448348596921540627795219927951379004452444719570738042378337465944112665294993327737106405440544513705275406544717522899690555069561839963232186804956513836192717374145392828808695477033015558202787064995031407079312795125272223392253120970835275756415864155319445535167341795525293598484555252724818812724051787201347021168925285903218700413299571102718214809898535950307826378982148851017973389



Найденные делители: {14344307537389974181: 1, 6277101735386680763835789423207666416102355444464034513029: 1}



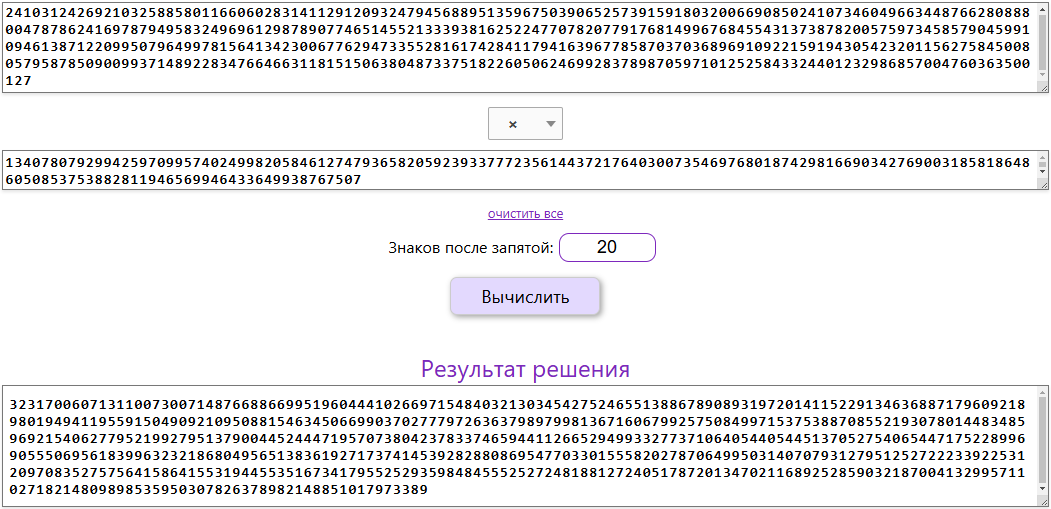
Во втором случае мне удалось выполнить разложение числа на множители с использованием языка Python и библиотеки gmpy2.  
Я реализовал регулярное выражение, которое извлекает все числа B из текстового файла и сохраняет их в массив.  
Затем с помощью алгоритма поиска наибольшего общего делителя (НОД) я сравнивал своё число с каждым из извлечённых значений.  
Если находился общий делитель, отличный от самого числа, я делил исходное число на этот делитель и таким образом находил оба множителя.  
В результате программа успешно определила множители числа путём нахождения общих делителей с другими элементами файла.



Вывод в консоль

Общий делитель найден: 2410312426921032588580116606028314112912093247945688951359675039065257391591803200669085024107346049663448766280888004787862416978794958324969612987890774651455213339381625224770782077917681499676845543137387820057597345857904599109461387122099507964997815641342300677629473355281617428411794163967785870370368969109221591943054232011562758450080579587850900993714892283476646631181515063804873375182260506246992837898705971012525843324401232986857004760363500127

Общий делитель найден: 13407807929942597099574024998205846127479365820592393377723561443721764030073546976801874298166903427690031858186486050853753882811946569946433649938767507



# **Выводы**

В ходе лабораторной работы я ознакомился с отечественным криптографическим стандартом ГОСТ Стриборг, изучил различные методы факторизации целых чисел, а также детально разобрал принципы работы алгоритма RSA, безопасность которого основывается на сложности разложения больших полупростых чисел на множители. Кроме того, при решении поставленных задач мне пришлось проявить творческий подход и нестандартное мышление для нахождения эффективного способа вычисления делителей заданного числа Б.